Accelerating Image Analysis for Localization Microscopy with FPGAs

Frederik Grüll, Manfred Kirchgessner, Rainer Kaufmann, Michael Hausmann,
Kirchhoff Institute for Physics, Heidelberg University
Heidelberg, Germany

Udo Kebschull
IRI, Goethe University Frankfurt
Frankfurt, Germany

***Abstract*—Localization microscopy enhances the resolution of fluorescence light microscopy by about an order of magnitude. Single fluorescent molecules act as switchable markers. Their detected signals can be fitted with a two-dimensional Gaussian distribution and thus located with sub-pixel resolution. In this paper we propose that these fits can be done by calculating the center of mass instead of an iterative least-square fit without loosing precision. The simplification of the algorithm leads to an acceleration of more than a factor of 100 and enables an FPGA implementation with an additional performance boost by a factor of 225. Our findings allow the real-time processing of current and future image data rates in localization microscopy.**

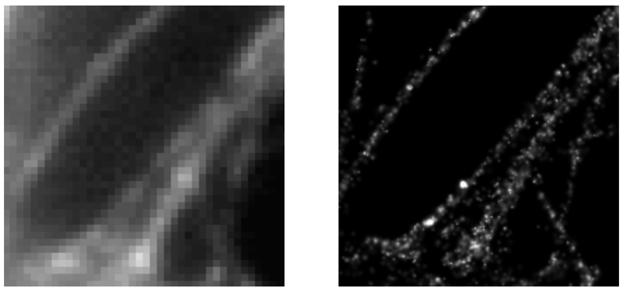
I. Introduction

The resolution of visible light microscopy is approximately half the wavelength of light. Structures smaller than $d \simeq \lambda/2$ cannot be resolved and appear as a single diffraction-limited spot. However, the location of a point-like object can still be retrieved more precisely from the image by determining the center of the spot.

Fluorophores are switched between two different spectral states (e. g. "bright" and "dark"). This is used for the optical isolation of the single molecule signals. Typically data stacks consisting of thousand frames are recorded. The positions of all single molecule signals are determined by fitting a model function to the spots and are plotted into a new image. The structural resolution provided in this image is now only dependent on the localization accuracy and the density of the detected molecules. Localization microscopy improves the resolution by about an order of magnitude and enables the non-destructive imaging of structures in living cells with visible light [1]–[4]. An example image can be seen in Fig. 1.

The airy disk of each spot is described by the two-dimensional point spread function (PSF) of the object and can be approximated by a Gaussian distribution. The center of the distribution is obtained by fitting. For maximum accuracy this has been a slow numerical process so far, hindering its widespread use. The data analysis of a 5 minute measurement with 2 000 frames can take up to several hours on standard PC hardware in practice.

To speed up analysis the *fluoroBancroft* algorithm has been developed recently as a non-iterative alternative [5]. It delivers an accuracy comparable to numerical Gaussian fitting for small ROIs. *fluoroBancroft* is model based and requires the width of the airy disk as an input parameter. The *MaLiang* method [6] is a port of iterative maximum-likelihood fitting to Graphic Processing Units (GPU) and accelerates data analysis on the hardware level. In [7] further algorithms have been compared. We argue that the center-of-mass method is capable of achieving the same accuracy as iterative solutions given the conditions that apply to localization microscopy. The simplification of the algorithm then unlocks the vast acceleration potential of a pipelined implementation on FPGAs which is presented in this paper.



(a) conventional wide-field fluorescence microscopy (b) localization microscopy image

Fig. 1. Localization microscopy improves the resolution of fluorescence microscopy by about an order of magnitude. The positions of fluorescent molecules are determined with sub-pixel accuracy and plotted into a new image. [4]

A. The Method of Maximum Likelihood

The airy disk in an image can be either fitted with the least-square method or the method of maximum likelihood. Our algorithm is based on the likelihood. It is defined as

$$L = \prod_{\gamma} f\left(x_{\gamma}\right) \tag{1}$$

with f being the fit function. The product runs over all detected photons in the region of interest (ROI) of the airy disk and has to be maximized by adjusting the fit parameters. The fit function of the airy disk is described as a scaled Gaussian distribution

$$f\left(\vec{x}\right) = \frac{Q}{2\pi\sigma_x\sigma_y} e^{-\frac{1}{2}\left(\left(\frac{x-\mu_x}{\sigma_x}\right)^2+\left(\frac{y-\mu_y}{\sigma_y}\right)^2\right)} + b \tag{2}$$

with fit parameters $\vec{\mu}$ (center of the distribution), σ_x, σ_y (width) and the total intensity Q. For a circular spot, the values of the width parameters σ_x and σ_y should be equal in both directions. The constant background b in the ROI can be measured and subtracted before fitting. With $b = 0$ the problem of maximizing the likelihood can now be solved analytically. We maximize $l = \ln L$ by differentiating l with respect to each fit parameter and get the Gaussian estimation [8], where i is the index of each pixel in the ROI.

$$\vec{\mu} = \sum_i \frac{q_i}{Q} \vec{x}_i \quad (3)$$

$$\sigma_x^2 = \sum_i (x_i - \mu_x)^2 \frac{q_i}{Q} \quad (4)$$

The total intensity Q cannot be obtained from the Gaussian estimation, but is easily calculated by summing over all pixels. The algorithm is already used for hardware-accelerated feature extraction in particle physics [9].

$$Q = \sum_i q_i \quad (5)$$

Having an analytical expression for the localization, we use error propagation to get the deviation of $\vec{\mu}$. The position variance of a photon in a pixel is $\Delta x^2 = 1/12$. The variance of the background is $\Delta N_B = \sigma_{N_B} = \sqrt{N_B}$ for Poisson noise.

$$\Delta\mu_x = \sqrt{\sum_\gamma \left(\frac{\partial\mu}{\partial x_\gamma}\Delta x_\gamma\right)^2 + \sum_i \left(\frac{\partial\mu}{\partial q_i}\Delta q_i\right)^2} \quad (6)$$

$$= \sqrt{\frac{1}{12Q} + \sum_i \left(\frac{x_i - \mu_x}{Q}\right)^2 (q_i + N_B)} \quad (7)$$

II. The Algorithm

Our approach to image analysis consists of three or four steps. First, the signal of a spot has to be found and must be centered in the ROI. The second step removes values at the periphery of the ROI. This suppresses the most noisy pixel values and improves the accuracy of the fit. An optional third step separates spots with overlapping ROIs if possible. The final step then calculates the position of the center, the confidence, and all other spot parameters.

A. Finding the spots

Our algorithm detects signals by comparing the background with each pixel value. The background intensity N_B can be measured and is known to follow a Poisson random distribution with width $\sigma_{N_B} = \sqrt{N_B}$. For complex structures like a biological cell the background is inhomogeneous, but changes only slowly over time (compared to the time scale of the blinking spots) for each pixel. We smooth the pixel values of every frame Img_t at time t exponentially over time and interpret the result as a map of the background Bg_t (8). The rise is limited to σ_{N_B} to prevent blinking signals from changing the background map too much. N defines the inverse smoothing factor and should be chosen much larger than the average number of frames a spot can fluoresce before it bleaches out.

$$N_{B,t} = N_{B,t-1} + \frac{1}{N}\left(\min\left(Img_t - N_{B,t-1}, \sigma_{N_{B,t-1}}\right)\right) \quad (8)$$

We require the maximum of an airy disk to top the background by more than $4\sigma_{N_B}$ and center a quadratic ROI around the local maximum. The background is then subtracted from the signal.

The size of the ROI can be chosen accordingly to the theoretical width σ_{theo}, which does only depend on the wavelength and the numerical aperture $N.A.$ of the setup.

$$\sigma_{\text{theo}} = \frac{1.22\,\lambda}{4\,N.A.\ln 2} \quad (9)$$

B. Zero suppression

The second summand in the error of $\vec{\mu}$ increases with the size of the ROI (6). The bigger the ROI, the less accurate the fit will be. Especially the corners of a quadratic ROI will contain few signal, but much noise for a circular airy disk.

After the average background N_B got subtracted, we therefore reduce all pixel values by $2\sigma_{N_B}$ for the calculation of $\vec{\mu}$ and set negative values to zero. This effectively creates a round ROI, as (3) ignores pixels with $q_i = 0$. For the calculation of $\sigma_{x,y}$ (4) this step is omitted.

C. Optional signal separation

If the density of fluorescent molecules in a biological sample is high in certain regions, the resulting spots will overlap in a significant number of cases. The localization would then be biased towards the second spot when calculating the center of mass. The least-square method would also struggle to converge close to the true position. Hence, we provide an optional method that scans the signal from center to the boundaries of the ROI and, after coming across a local minimum, sets all further pixel values in the current line to zero. A tolerance of σ_{N_B} ensures that the separation does not remove values too aggressively, as noise may also cause local minima. If too much intensity is removed the ROI will be discarded.

D. Feature extraction

The Gaussian estimator finally gives us the fit parameters μ and $\sigma_{x,y}$. As σ_{theo} is known in advance, it can be used to remove false positives from the result set.

III. Methods

We tested our algorithm with a Monte-Carlo Simulation. First, we generated an array containing the signal at a random position on a constant background and applied Poisson noise on the result. Then we searched for the maximum as described before and centered the ROI on it. For feature extraction our Gaussian estimator was used alongside with the fluoroBancroft algorithm. As a reference we also applied the least-square fit `lsqcurvfit` in Matlab 7.10.0 with fit function (2).

Fig. 2 shows a generated noisy signal before and after pre-processing. For each data point we simulated 2 000 of these arrays. We benchmarked the localization performance by comparing every measured location with the true value of $\vec{\mu}$ for different values of σ, Q and N_B. The signal-to-noise ratio (SNR) was defined as in [10]. $q_{\max}$ names the value of the brightest pixel in the center of each ROI after the background got subtracted (10).

$$SNR = \frac{q_{\max}}{\sqrt{N_B + \sigma^2_{q_{\max}}}} \tag{10}$$

A. Hardware Implementation

The implementation of the algorithm in hardware was carried out on a MaxWorkstation from Maxeler Technologies. The platform generates a pipelined version of the whole algorithm after its data flow graph has been described in the Java programming language [11]. This speeds up development compared to a description in VHDL or a similar hardware language significantly.

The bitstream of the design was built for a Xilinx Virtex-5 LX330 on an acceleration card in the MaxWorkstation. The input and output streams of the design are connected by PCIe streams to the memory and the CPU.

The algorithm is run on two cooperating kernels (Fig. 3). Background removal and signal finding operate on every recorded pixel. Both operations therefore form the pipeline of the first kernel, which consumes one pixel per clock cycle. The ROIs from the output of this kernel are then processed by the signal separator and the Gaussian estimator in the pipeline of the second kernel. A manager takes care of synchronizing both kernels.

When the first kernel finds a signal, it copies the corresponding ROI within one clock cycle to the elastic buffer that connects both kernels. The coordinates of the ROI and the frame number are also attached to each ROI for later reference. The second kernel then examines the ROI pixel by pixel, with one pixel per clock cycle. This is where zero suppression, signal separation and the actual feature extraction take place. The final result contains the position $\vec{\mu}$, the squared localization error $\Delta\mu^2_{x,y}$ and width $\sigma^2_{x,y}$, the total charge Q and the frame number. The results for each ROI are received by a C program on the CPU and stored in a file for visualization.

The translation of the background removal and signal finder to a hardware pipeline is achieved by scanning over each pixel of each frame. All pixel values are 16-bit integers of a 320 px×288 px frame, and one value is consumed per clock

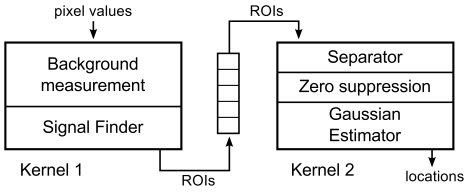


Fig. 3. The hardware design. Each pixel from the camera is fed through the first kernel, where the background gets measured and subtracted. If a signal is found, its ROI is sent to the second kernel. Here, the separator removes values from signals that touch the ROI. The estimator finally calculates the position data. Both kernels are decoupled by an elastic buffer.

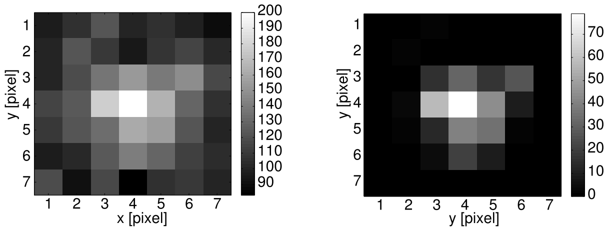
cycle. The signal separator and the estimator however do not run on every pixel, but only on pixels in a ROI that have passed the signal finder. A custom access pattern is used at this place to scan the ROI from the center to the margin (Fig. 4). The Gaussian estimator finally had to be rewritten to an accumulator-based version to enable its translation to a pipeline. For $\vec{\mu}$ and Q this is already the case and we can calculate these values by accumulating $q_i x_i$, $q_i y_i$ (3), and q_i (5) in a fixed-point accumulator. $\sigma_{x,y}$ was rewritten as shown in (11) and can now be calculated in the same run as $\vec{\mu}$ from accumulators of $q_i x_i^2$ or $q_i y_i^2$ respectively. A more lengthy, but similar transformation can be done for $\Delta\mu_{x,y}$. It requires additional accumulators.

$$\sigma_x^2 = \sum_i \frac{q_i}{Q} x_i^2 - \mu_x^2. \tag{11}$$

IV. Results

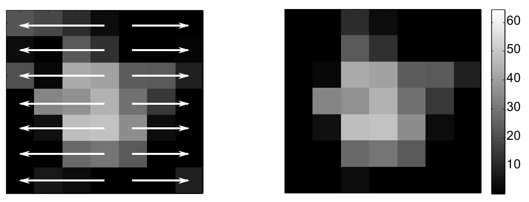
A. Accuracy

The errors of all algorithms were obtained in a Monte Carlo simulation and are shown in Fig. 5. We reach full accuracy compared to the numerical fits within a 5% margin, and even outperform by about 5% if the background is low. Before zero suppression was added we observed a dependency of the feature extraction step on the signal finder. Due to noise the pixel with the maximum value in the center of the ROI may not coincide with the true center. This would then lead to a bias. An implementation of the least-square fit algorithm that ignores small values like our approach let to an increase in fit failures and did not improve the localization accuracy.



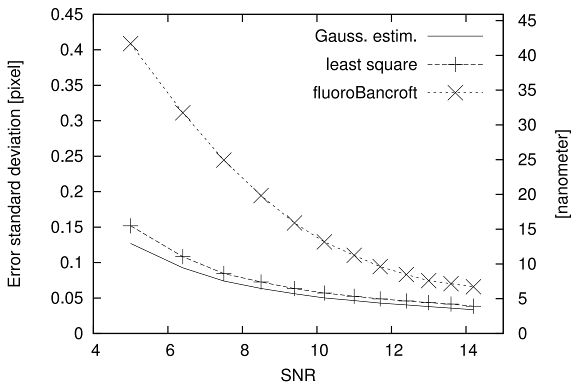
(a) with background (b) mean background subtracted

Fig. 2. A generated noisy signal with total charge $Q = 1000$, background $N_B = 100$ and width $\sigma_{x,y} = 1.4\,\text{px}$ at $\vec{x} = (4.1, 4.2)$. By setting pixels with $q_i < 2\sigma_{N_B}$ to zero the feature extraction becomes resistant to ROIs that were not centered at the true position due to noise. Here, the position was determined with an accuracy of $0.16\,\text{px}$.

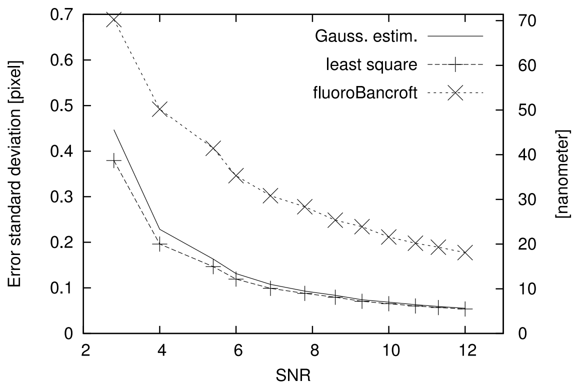


(a) a second signal from top left overlaps the ROI (b) after separation

Fig. 4. The access pattern of the signal separator is from center to left and right and from top to bottom. Values that are past a local minimum in are set to zero. The process is repeated in vertical direction and in horizontal direction again.



(a) $N_B = 10, \sigma_{x,y} = 1.4$, 1 px is 102 nm



(b) $N_B = 100, \sigma_{x,y} = 1.4$, 1 px is 102 nm

Fig. 5. Monte Carlo simulation of the localization accuracy. The algorithm we propose (Gaussian estimator) is about as accurate as the numerical fits within a 5% margin. For low background ($N_B = 10$) it even outperforms. The ROI was chosen 7×7 px.

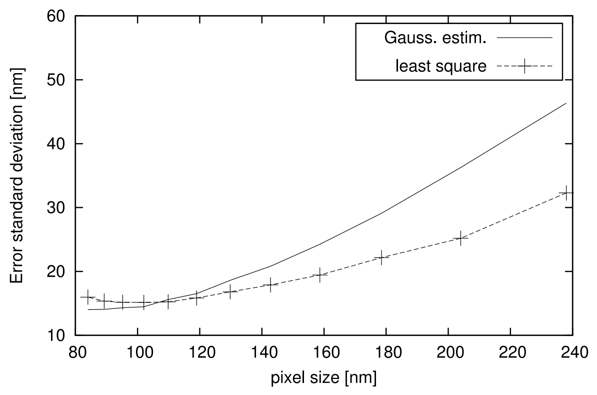


Fig. 6. The localization accuracy was simulated for CCD cameras with different pixel sizes. The accuracy decreases if the pixels are too big and most of the signal is confined to a single pixel. For smaller pixel sizes both methods work well until the ROI gets too small to occupy the spot. The simulated spots had a width of $\sigma = 143$ nm, the SNR was 6 in a 7×7 px ROI. Our camera has a pixel size of 102 nm.

The localization error $\Delta\mu_{x,y}$ matches the expected errors if the signal separator did not remove too much of the total intensity. We found empirically, that close spots can still be successfully separated if the separator does not remove more than 30% of the signal. Otherwise the ROI must be discarded.

The localization accuracy depends on the size of the pixels in the CCD camera sensor (Fig. 6). If these pixels are bigger than the width of a spot, the pixels next to the brightest spot in the center of the ROI will only illuminated by a tiny fraction of the total intensity. Both method loose accuracy in this situation, but the least-square method degrades not as hard as our enhanced Gaussian estimator. However, cameras for visible light microscopy have a smaller pixel size where both methods have their optimum and the Gaussian estimator is as accurate as the least-square fit.

We measured the width of a thin molecular line to analyze the accuracy of the enhanced Gaussian estimation with real world data. The profile of the line is shown in Fig. 7 alongside with the profile that results from the least-square fit. The line profile produced by our method is 12% percent thinner than the profile of the least-square method. The steps for background removal and signal finding were the same for both methods.

B. Computing Time

Before our method was implemented on an FPGA in hardware, we wrote a Matlab version of both algorithms. Here, our algorithm was about 100 times faster than the numerical least-square fit and 5 times faster than fluoroBancroft on an Intel i5 450 with Matlab 7.10.0. It analyses a 256×256 px frame with 10 spots on the same machine within 73.9 ms. Hence, the bandwidth was 0.89 Mpx/s. The feature extraction step consumed about 68% of the computing time and scaled with the number of points found. We also wrote a C++ implementation that was up to 2 times slower than our heavily optimized Matlab implementation.

The hardware implementation on the Xilinx Virtex 5 LX-330T achieved a clock frequency of 200 MHz. Every clock cycle a pixel is consumed. The bandwidth therefore is 200 Mpx/s, independent of the number of spots found in the recording. We achieve a hardware acceleration factor compared to the Matlab implementation of 225.

For a typical image stack with 2000 frames with 280×340px each and a total of 300 000 signals, the second kernel is only occupied 7.7% of the total time the calculation is running.

V. Resource Usage

Table I lists the FPGA resources of the hardware design on a Xilinx Virtex-5 LX330. The FPGA is occupied by about 1/3. The background measurement consumes most of the BRAMs to store the background map. The signal separation does mainly consist of FIFOs to implement the access pattern mentioned above and some comparators. The feature extraction step implements the formulas of the Gaussian estimator and consumes the majority of flip-flops and DSPs for the accumulators. All other resources are used for synchronization and the communication with the host CPU.

VI. Discussion

As it is often the case with hardware acceleration, most of the effort is spent on rewriting the computer program and

TABLE I

RESOURCE USAGE ON A XILINX VIRTEX-5 LX330. THE CLOCK FREQUENCY IS 200 MHZ. THE RESOURCES FOR SYNCHRONIZATION AND I/O ARE INCLUDED IN THE TOTAL NUMBER.

	LUTs	FFs	BRAMs	DSPs
background	1 179	1 308	51	0
signal finder	211	198	0	2
signal separation	1 401	3 303	0	0
feature extraction	23 146	33 817	0	42
total	42 044	63 881	108	44
% of the FPGA	20%	31%	33%	23%

proving that the results are still correct. Done right, the following implementation on hardware is more straightforward. We chose the MaxCompiler solution because it simplified the implementation of the pipeline and let us focus on the algorithm.

The results show that the Gaussian estimation with zero suppression is as accurate as the least square fit Matlab provides. This is surprising given its relative simplicity. No iterations are needed and all features like the location of each spot and the confidence can be calculated within a single scan over the ROI. This made a pipelined implementation suitable. The resource usage indicates that the algorithm would also fit on smaller and cheaper FPGAs like the Xilinx Artix-7 A105. An implementation of the iterative least-square fit on FPGAs would require the calculation of the fit function (2) and its derivatives to approach the optimum numerically, resulting in an increase of resources occupation at slower speed.

The change in the algorithm accounts for an acceleration of more than 100, and the translation to hardware multiplied a second acceleration of 225 on top of this. For the end-user this means an acceleration of 22 500 compared to the previously used least-square method in software. Compared to the MaLiang method the speedup is 2.8 [6].

The low degree of capacity utilization of the second kernel suggests that the design can be optimized further. A (slower) FSM could probably substitute the second kernel and save resources without loosing performance for typical use cases.

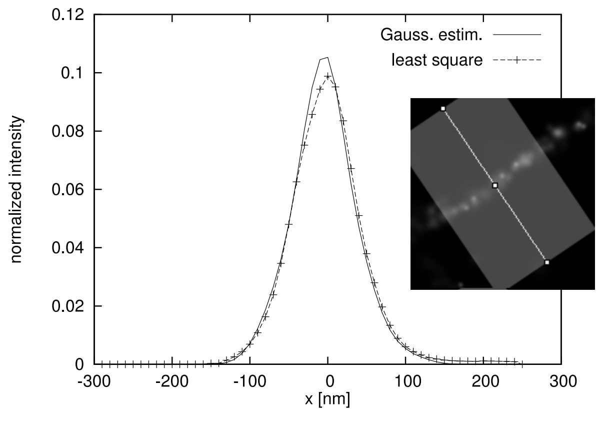


Fig. 7. Line profile of real world data. The line is formed by a molecular structure of a cell and is considerably thinner than the diffraction limit of light microscopy. The results from the Gaussian estimator have a slightly sharper width $\sigma_{\mathrm{estim}} = 47.5\,\mathrm{nm}$ than the least-square method ($\sigma_{\mathrm{lsq}} = 41.7\,\mathrm{nm}$).

VII. CONCLUSION

The here proposed enhanced Gaussian estimation combines the accuracy of previously used least-square fits with the speed of an non-iterative solution. The translation to hardware accelerates the program additionally. This leads to real-time processing of recordings from cameras with big CCD/CMOS sensors and high frame rates on FPGAs.

The Matlab version of our algorithm is in productive use already in the research group of Prof. Christoph Cremer and Prof. Michael Hausmann at the University of Heidelberg and has replaced the slower least-square method.

In the future the demand for fast image analysis will continue to rise as faster detectors and more efficient fluorescent molecules are being developed. Our hardware implementation can satisfy this demand and increase the usability of localization microscopy in practice. We argue that FPGAs can transform localization microscopy from a time-consuming process to a real-time tool for biology and medical science where the microscope and its computer are embedded in a single device.

ACKNOWLEDGMENTS

We like to thank Torsten Alt and Heiko Engel for hints and discussions about the Gaussian estimation, and the research group of Prof. Christoph Cremer for providing us with test imagery. This work was funded by the FRONTIER innovation fund, Heidelberg, and supported by the Maxeler Technologies University Program MAX-UP.

REFERENCES

[1] E. Betzig, G. H. Patterson, R. Sougart, O. W. Lindwasser, S. Olenych, J. S. Bonifacino, M. W. Davidson, J. Lippincott-Schwartz, and H. F. Hess, "Imaging intracellular fluorescent proteins at nanometer resolution," *Science Express*, vol. 313, pp. 1642–1645, 2006.

[2] S. T. Hess, T. P. K. Girirajan, and M. D. Mason, "Ultra-high resolution imaging by fluorescence photoactivation localization microscopy," *Biophysical Journal*, vol. 91, pp. 4258–4272, 2006.

[3] M. J. Rust, M. Bates, and X. Zhuang, "Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (STORM)," *Nature Methods*, vol. 3, pp. 793–796, 2006.

[4] P. Lemmer, M. Gunkel, D. Baddeley, R. Kaufmann, A. Urich, Y. Weiland, J. Reymann, P. Müller, M. Hausmann, and C. Cremer, "SPDM: light microscopy with single-molecule resolution at the nanoscale," *Applied Physics B: Lasers and Optics*, vol. 93, pp. 1–12, 2008.

[5] S. Andersson, "Precise localization of fluorescent probes without numerical fitting," in *Biomedical Imaging: From Nano to Macro, 2007. ISBI 2007. 4th IEEE International Symposium on*, april 2007, pp. 252–255.

[6] T. Quan, P. Li, F. Long, S. Zeng, Q. Luo, P. N. Hedde, G. U. Nienhaus, and Z.-L. Huang, "Ultra-fast, high-precision image analysis for localization-based super resolution microscopy," *Optics Express*, vol. 18, no. 11, pp. 11 867–11 876, 2010.

[7] S. M. Anthony and S. Granick, "Image analysis with rapid and accurate two-dimensional Gaussian fitting," *Langmuir*, vol. 25, no. 14, pp. 8152–8160, 2009, pMID: 19419178.

[8] S. Brandt, *Data Analysis.* Springer, 1998, ch. 7. The Method of Maximum Likelyhood.

[9] *ALICE Technical Design Report.* Cern, 2004, ch. 13.2.3 FPGA coprocessor / detector interface.

[10] U. Kubitscheck, O. Kückmann, T. Kues, and R. Peters, "Imaging and tracking of single GFP molecules in solution," *Biophysical Journal*, vol. 78, no. 4, pp. 2170–2179, 2000.

[11] Maxeler Technologies, "MaxCompiler white paper," February 2011, http://maxeler.com/content/software/.