TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

DIPLOMARBEIT

Positioning of Single Atoms in a Dipole Trap

ausgeführt am Institut für

Angewandte Physik

der Technischen Universität Wien

unter der Anleitung von
Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Martin Gröschl

durch

Mag. Robert Jindra
Rainergasse 16/1/8, 1040 Wien

09.02.2014 Robert Jindra

Contents

Kurzfassung **iii**

Abstract **iv**

1 Introduction **1**

2 Experimental setup **3**

2.1 Magneto optical trap 3

2.2 Dipole trap 4

2.3 Optical Setup and Hardware 6

2.3.1 Camera 6

2.3.2 FPGA Module and Real-time Controller 8

2.3.3 Field-Programmable Gate Array 9

3 Algorithm **10**

3.1 Mathematical background 10

3.1.1 The deconvolution problem 10

3.1.2 Determining the line spread function 11

3.1.3 Number of atoms 13

3.1.4 Trigonometric moment estimation 14

3.1.5 Levenberg-Marquardt algorithm 16

3.2 Implementation 19

3.2.1 Structure 19

3.2.2 FPGA part 20

3.2.3 RT part 24

4 Performance **27**

4.1 FPGA Performance 27

4.2 DMA Data Transfer 30

4.3 Accuracy 31

4.4 RT Performance 38

4.5 Resolution power 45

5 Measurement results **48**

5.1 Signal Characteristics 48
5.2 Performance 49
5.3 Accuracy 50

6 Conclusion **51**

References **52**

Kurzfassung

Das Ziel dieses Diplomprojekts ist die Positionsbestimmung einzelner Atome in einer Dipolfalle durch die Implementierung eines Echtzeit-Algorithmus. Das Fluoreszenz-Signal der Atome, das von einer Electron Multiplying Charge-Coupled Device (EMCCD) Kamera aufgenommen wird, wird dabei von hoch-performanter Hardware bestehend aus einem Field-Programmable Gate Array (FGPA) und einem Real-Time (RT) System verarbeitet. Der Algorithmus ist für schnelle Datenverarbeitung optimiert, um eine Rückkopplung auf die Positionen der Atome zu ermöglichen, und besteht aus zwei Teilen. In einem ersten Schritt wird eine schnelle aber nur ungenaue Positionsbestimmung mittels einer Methode, die sich der Abschätzung der trigonometrischen Momente bedient, vorgenommen. Diese ungefähre Annäherung wird in einem zweiten Schritt als Startwert für einen präziseren Levenberg-Marquardt-Fit-Algorithmus verwendet. Erste Messungen zeigen sowohl zufriedenstellende Performanz als auch ausreichende Genauigkeit.

Abstract

The aim of the project described in this thesis is the positioning of single atoms in a dipole trap by the implementation of a real-time algorithm. The fluorescence signal of the atoms which is captured by an Electron Multiplying Charge-Coupled Device (EMCCD) camera, is processed by a combination of high performance hardware, including a field-programmable gate array (FPGA) and a real-time (RT) system. The algorithm is optimized for fast data processing to enable feedback on the positions of the atoms. The implemented positioning algorithm consists of two parts. First, a rough and fast position calculation is performed by a trigonometric moment estimation method. These approximate positions are then used in a second step as initial parameters for a more precise Levenberg-Marquardt fitting algorithm. First measurements show that both performance and accuracy of the positioning program are satisfying.

Chapter 1

Introduction

The positioning program developed within the course of this thesis is designed for a quantum optics experiment using single atoms [3][8][17]. Until now, the positions of the atoms have been determined by post-processing software. The aim of this project is to enable a real-time positioning in order to use the data in a feedback loop. High performance hardware such as a field programmable gate array (FPGA) and a real time (RT) controller are used.

In the experiment which is based at the quantum technologies research group at the University of Bonn, Rubidium atoms are cooled and trapped in a magneto optical trap (MOT)[11] and overlapped with a dipole trap to address single atoms inside the MOT [2]. The fluorescence caused by the laser beams of the MOT is used for the positioning process. The signal from the atoms is convoluted with the point spread function (PSF) of the optics that transfers the light to an Electron Multiplying Charge-Coupled Device (EMCCD) camera. The main mathematical task is to reconstruct the original signal by a deconvolution procedure as described by Karski [4].

The signal from the camera is read out by the FPGA. The FPGA allows fast computation but performance is restricted due to its limited resources. Therefore only basic steps of the positioning algorithm are executed on the FPGA. First, the data is separated in regions of interest (ROIs) which contain atom fluorescence and background regions. For the ROIs, the number of atoms are calculated.

This information is then transferred to the RT system which performs the actual deconvolution for each ROI. The unknown source distribution is modelled as a constant background with delta peaks representing the atom fluorescence. This signal is convoluted with a 1D version of the PSF, the so called line spread function (LSF). An analytical model of the LSF is obtained by an iterative self-consistency loop [4].

Due to the form of the source distribution, a spike-convolution model fitting as described by Li et al. [6] is used. The method of trigonometric moment estimation gives a fast but rough approximation of the positions. The accuracy of the position of the atoms is then subsequently enhanced by an iterative maximum likelihood estimation method using the Levenberg-Marquardt fitting algorithm [10].

First measurements show satisfying results concerning both performance and accuracy.

This work has been conducted in strong collaboration with National Instruments (NI) in Munich and the quantum technologies research group of the University of Bonn. It is part of a dissertation project which was realised within the network Circuit and Cavity Quantum Electro-Dynamics (CCQED) which is funded by the European Union through a Marie Curie Action within the Seventh Framework Program Initial Training Network.

Chapter 2

Experimental setup

The positioning program acts on atoms within a magneto-optical trap (MOT) overlapped with a strongly focused standing wave dipole trap. In this experiment single rubidium atoms are used.

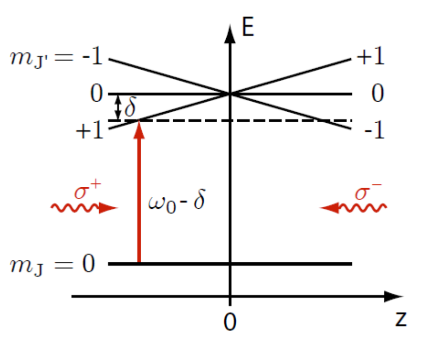
2.1 Magneto optical trap

The MOT represents the first stage in cooling and trapping of atoms. This technique was first implemented in 1987 [11] and is today the basis for most experiments with single trapped atoms.

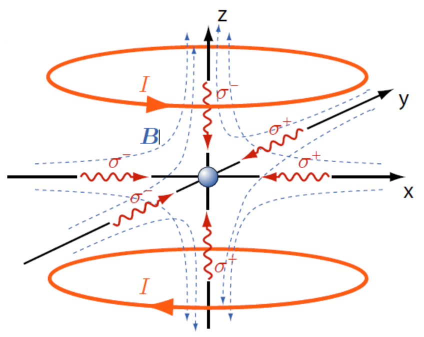
The basic principle is the implementation of a velocity-dependent cooling force and position-dependent restoring force to provide spatial confinement of the atoms.

The cooling force can be realised by three mutually orthogonal pairs of counterpropagating laser beams which are slightly red-detuned to the transition frequency of the trapped atoms. When an atom moves in the direction of a laser source, the frequency of the beam is Doppler-shifted to the atomic resonance. This leads to the absorption and subsequent spontaneous emission of photons. Because this process is anisotropic, the net force resulting from the momentum transfer slows down the atoms and thus cools it. With this techniques temperatures down to the *Doppler limit* $T_D = \frac{\hbar\Gamma}{2k_b}$ can be achieved [18]. Here Γ denotes the line width of the optical transition, k_b the Boltzmann constant.

The spatial confinement is implemented by a quadrupole magnetic field which is generated by a pair of anti-Helmholtz coils and circular polarized laser beams. At the centre of the trap, the magnetic field is zero while it increases linearly in all directions. If an atom moves out of the center, it can only absorb a photon from the counter-propagating beams as the transition gets in resonance due to the Zeeman effect (Fig. 2.1), leading to a confining force.



(a)



(b)

Figure 2.1: (a) One-dimensional scheme of the origin of the spatial force in a MOT. The magnetic field increases linearly with the z direction, lifting the degeneracy of the sublevels of the excited state $J' = 1$. If displaced far enough, the $m_J = 0 \rightarrow m_{J'} = 1$ transition comes into resonance. Due to selection rules, only σ^+ photons are absorbed, resulting into a confining force. (b) Three-dimensional scheme of a MOT, where three pairs of counterpropagating laser beams perform cooling and trapping.
Source: [4]

In the experiment described in this thesis, three laser beams which intercept each other perpendicularly are used. The polarization of the original beam σ^+ is shifted for one beam to σ^- by passing a $\lambda/4$ plate two times. The light is red-detuned by 10 MHz from the $F = 2 \rightarrow F' = 3$ transition of 87Rb at a wavelength of 780 nm. The beams have a power of 20 µW and a waist of 350 µm. An atom can be off-resonantly excited to the $F' = 2$ state from which it can decay to the $F = 2$ or $F = 1$ state, obeying the selection rules. A repumper laser is used to excite the atom from the $F = 1$ to the $F' = 2$ state until it decays to the desired $F = 2$ ground state.

2.2 Dipole trap

In the MOT used in the experiment the atoms are confined in a volume of several cubic micrometers. In order to work with individual atoms, the MOT is overlapped with a standing wave from a dipole trap. This one-dimensional optical lattice consists of conservative trapping potentials [2].

In the classical Lorentz model, an electron is elastically bound to the atom by the harmonic potential of the core. In an external monochromatic electric field $\boldsymbol{E}(\boldsymbol{r},t) = (\boldsymbol{E_0}(\boldsymbol{r})exp(-i\omega t) + c.c.)/2$, the induced dipole moment of the atom $\boldsymbol{d}(\boldsymbol{r},t)$ obeys the equation of motion

$$\ddot{\boldsymbol{d}}(\boldsymbol{r},t) + \Gamma_\omega \dot{\boldsymbol{d}}(\boldsymbol{r},t) + \omega_0^2 \boldsymbol{d}(\boldsymbol{r},t) = \frac{e^2}{m_e}\boldsymbol{E}(\boldsymbol{r},t), \tag{2.1}$$

where ω_0 is the resonance frequency of the oscillator and Γ_ω the energy damping rate due to dipole radiation

$$\Gamma_\omega = \frac{e^2\omega^2}{6\pi\epsilon_0 m_e c^3}. \tag{2.2}$$

Here m_e is the rest mass and e is the electric charge of the electron, and ϵ_0 is the vacuum permittivity. The stationary solution of Eq. 2.1 leads to the complex-valued atomic polarizability

$$\alpha(\omega) = \frac{e^2}{m_e}\frac{1}{\omega_0^2 - \omega^2 - i\omega\Gamma_\omega}, \tag{2.3}$$

which connects the dipole moment with the electric field:

$$\boldsymbol{d}(\boldsymbol{r},t) = \alpha(\omega)\boldsymbol{E}(\boldsymbol{r},t). \tag{2.4}$$

The dipole potential is derived from the time-averaged interaction energy between the induced dipole and its driving field

$$U_{dip}(\boldsymbol{r}) = -\frac{1}{2}\langle \boldsymbol{d}(\boldsymbol{r},t)\boldsymbol{E}(\boldsymbol{r},t)\rangle = -\frac{1}{2\epsilon_0 c}Re\{\alpha(\omega)\}I(\boldsymbol{r}), \tag{2.5}$$

with the field intensity $I(\boldsymbol{r}) = c\epsilon_0|\boldsymbol{E_0}(\boldsymbol{r})|^2/2$. With the rotating wave approximation for far detuned light fields, terms rotating at high frequencies with a time evolution determined by $\Delta = \omega \pm \omega_0$ are neglected. These terms average to zero over the evolution of the system due to $\Delta > \Gamma$. Eq. 2.5 reads then

$$U_{dip}(\boldsymbol{r}) = \frac{3\pi c^2}{2\omega_0^3}\frac{\Gamma}{\Delta}I(\boldsymbol{r}). \tag{2.6}$$

From Eq. 2.6, the basic properties of dipole trapping can be inferred. The dipole potential scales linearly with the intensity of the light field and with the reciprocal of detuning Δ. The direction of the dipole force $\boldsymbol{F}_{dip}(\boldsymbol{r}) = -\nabla U_{dip}(\boldsymbol{r})$ exerted on the atom depends on the sign of the detuning. Blue detuning ($\Delta > 0$) leads to a repulsive, red detuning ($\Delta < 0$) to an attractive force.

In order to realize a one-dimensional lattice, two counterpropagating, linearly polarized Gaussian laser beams are used. For cylindrical coordinates ($\boldsymbol{r} = (\rho, z, \phi)$), the standing wave intensity distribution can be approximated by

$$I(\boldsymbol{r}) \approx I_{max}\frac{w_0^2}{w^2(z)}e^{-\frac{2\rho^2}{w^2(z)}}cos^2(kz), \tag{2.7}$$

with the peak intensity $I_{max} = 4P/\pi\omega_0^2$ (where P is the power of a single beam of the counterpropagating light), the beam radius $\omega(z) = \omega_0\sqrt{1 + z^2/z_0^2}$ (where ω_0 is the waist radius and z_0 the Rayleigh length) and wavenumber $k = \frac{2\pi}{\omega}$.

Together with Eq. 2.6, the dipole potential of a one-dimensional optical lattice reads

$$U_{dip}(\boldsymbol{r}) = -U_0 \frac{w_0^2}{w^2(z)} e^{-\frac{2\rho^2}{w^2(z)}} cos^2(kz), \tag{2.8}$$

with U_0 denoting the maximum trap depth. The periodicity of the lattice is half the laser wavelength. The dipole potential has a Gaussian profile providing a confinement determined by the waist radius w_0.

A Titanium Sapphire laser at 860 nm is used to generate the dipole trap in the experiment. The light is red-detuned from the Rubidium D_2 line by 80 nm. The beam is split into two arms and passed through tapered amplifiers to increase the power. The counterpropagating arms are then focused on one spot with a waist size of 5 µm by using high numerical aperture lenses. To control the frequencies of the two laser beams, Acousto Optic Modulators (AOMs) are used.

2.3 Optical Setup and Hardware

The position of the atoms is acquired by the fluorescence signal of the atoms inside the MOT. The direct and antipodal light which is retro-reflected is collected by a high numerical aperture lens and guided through the optical setup to the camera. From there, the camera signal is sent via a splitter box to the positioning hardware as indicated in Fig. 2.2.

2.3.1 Camera

The camera used in the experiment is an iXon$_3$897 manufactured by Andor Technology[1]. The image sensor has 512 x 512 active pixels with the size of 16 x 16 µm each resulting in a 8.2 x 8.2 mm image area. In order to gain the necessary sensitivity, an Electron Multiplying Charged Coupled Device (EMCCD) is used. This chip is similar to an conventional CCD with the extension of multiplication or gain registers to the readout register shown in Fig. 2.3 (b). During read out the charges are transferred vertically line by line to the readout register and then horizontally to the multiplication register. Within these registers, the charges are amplified by impact ionization, resulting in a higher signal.

The Andor iXon$_3$897 EMCCD camera offers readout rates of 1, 3, 5 and 10 MHz. In the experiment 10 MHz are used. With this rate, each pixel is digitalized to 14 bits.

The output signal of the camera is composed of three main parts. First, the information of the data bus which consists of 16 bits, whereby only 14 bits are used in this project, determines the value of each pixel. Second, the

[1]http://www.andor.com

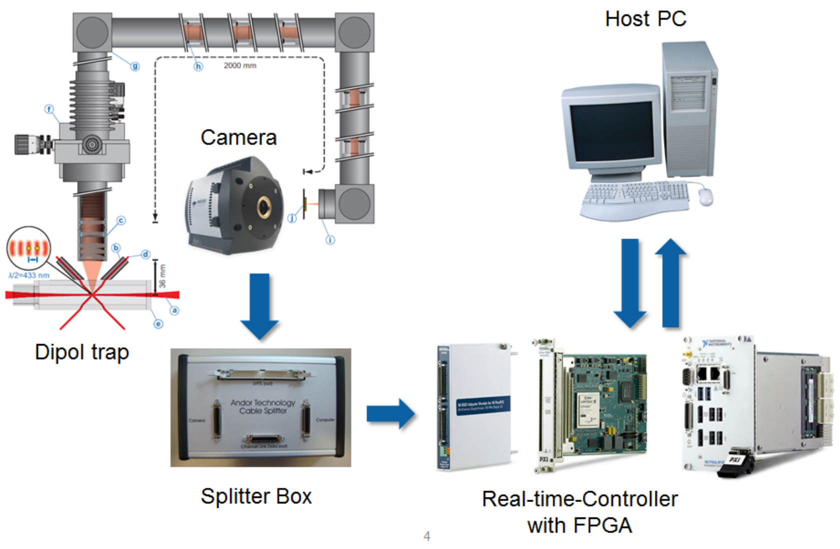
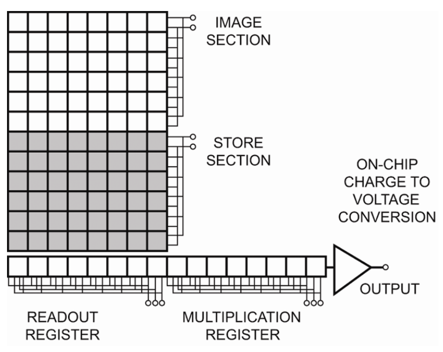

Host PC
Camera
2000 mm
36 mm
λ/2=433 nm
Dipol trap
Andor Technology Cable Splitter
Splitter Box
Real-time-Controller with FPGA
4

Figure 2.2: Schematic view of the experimental setup. The fluorescence light of the atoms in the dipole trap is guided through the optical setup to the camera. The camera signal is sent via the splitter box to the FPGA and the RT controller. The parameters for the processing of the camera data are set by the PC.
Parts taken from [4][16][15][1]

pixel clock whose rising edge indicates at which point in time valid data is available on the data bus. Third, the frame signal which is set to high at

(a)



(b)

Figure 2.3: (a) The Andor iXon$_3$897 EMCCD camera. (b) Structure of the EMCCD chip. Figures are taken from [16]

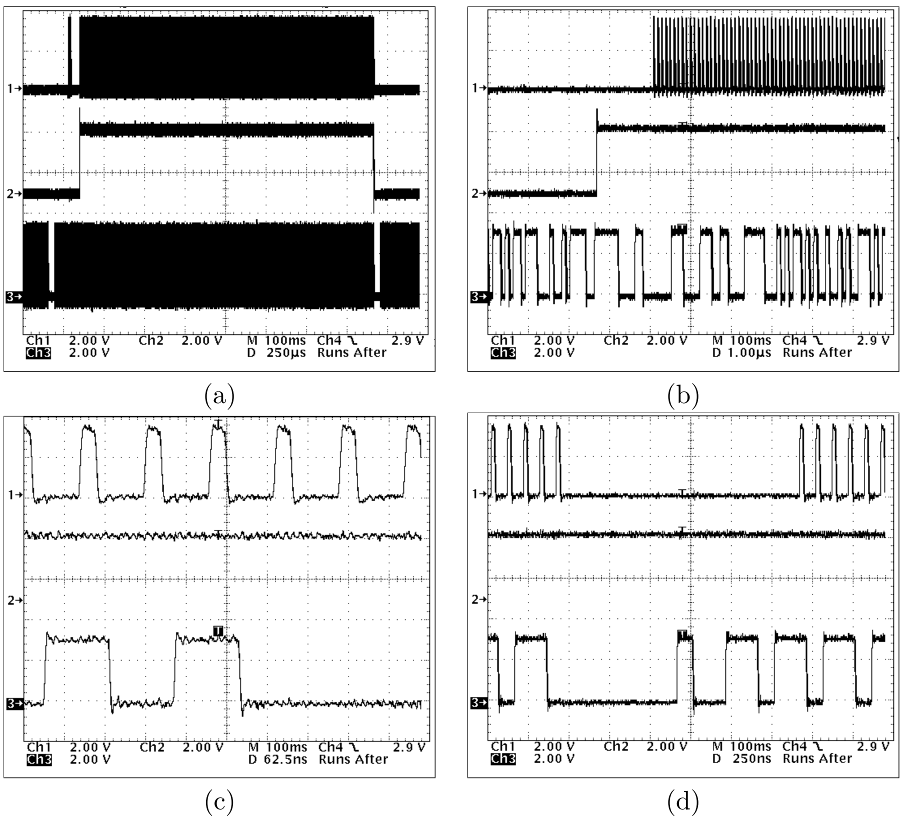


Figure 2.4: Oscilloscope images of the data being sent from the camera. Channel 1 shows the pixel clock, channel 2 the frame signal and channel 3 one bit stream from the data bus. (a) A full sequence of sending an image. The so-called overscanned pixels are sent even before the frame signal is set to logic high, which represents invalid data. (b) Beginning of a sequence. First the frame signal is set to high, then the pixel clock starts. (c) The rising edge of the pixel clock indicates valid data on the data bus. (d) Between two CCD lines a small pause occurs. The new line starts again with overscanned pixels. Figures are taken from [15]

the start of an image and to low at the end. Figure 2.4 shows oscilloscope images taken from the camera output.

2.3.2 FPGA Module and Real-time Controller

The camera signal is sent via the iXon Cable Splitter Box to a PXI-based control system manufactured by National Instruments (NI)[2]. This control system consists of a NI PXIe-7966R FPGA together with a NI 6581 digital adapter module to provide high speed digital input/output channels and a

[2]http://www.ni.com/pxi/

PXIe-8135 real-time (RT) controller, all assembled within a NI PXIe-1062Q chassis. The RT controller communicates with the FPGA via the PXI bus and with the PC holding the user interface via ethernet.

The NI PXIe-7966R FPGA module holds a Xilinx Virtex-5 FPGA chip. For programming the FPGA, the LabVIEW FPGA add-on module for the software development tool LabVIEW is used. The user-defined LabVIEW source code is then auto-generated to the hardware description language (HDL) which is a common programming language used to compile FPGA designs.

2.3.3 Field-Programmable Gate Array

The FPGA card is a central part of the project as it provides the necessary performance for the real-time positioning process. The basic principles of the FPGA technology is briefly explained in the following. Detailed resources on the specifications of the Virtex-5 can be found on the homepage of the manufacturer[3], the FPGA basics are described by Sauer [12], for example.

Field-Programmable Gate Arrays (FPGAs) are reconfigurable semiconductor devices. Configurable Logic Blocks (CLBs) are used to define the FPGA's circuit behavior. Although different manufacturers use different designs, in principle they consist of two basic components, lookup tables (LUTs) and flip-flops. The logic of the FPGA program is implemented as truth tables within the LUTs.

Data can be stored within LUT logic or the so-called Block RAM (BRAM).The Virtex-5 FPGA chip offers individual BRAM slices of 36 kbits which can also be used as a independent 18 kbits memory.

The FPGA's routing matrix connects individual resources such as CLBs and BRAMs. After compiling the user-defined HDL code, a bitfile is generated and downloaded which contains the corresponding parameters for the switches of the routing matrix and for the user-definable resources.

In addition to these FPGA basic parts, higher level functionality has been included in FPGA chips in recent years. For example, multiplying is very resource intensive in combinatorial logic. Therefore, many FPGAs possess prebuilt multiplier circuitry. Also, high speed I/O logic and even embedded processors are combined within one chip.

[3]http://www.xilinx.com

Chapter 3

Algorithm

In the following, the algorithm which is used for the atom positioning as described by Karski [4] is explained. In the first section, the mathematical details of the two steps of the positioning process, namely the trigonometric moment estimation and the Levenberg-Marquardt algorithm, is discussed. In the second section, the implementation of the algorithm is explained.

3.1 Mathematical background

In this section, the mathematical background of the various steps of the positioning algorithm are explained in detail. First, the positioning process is characterized as a deconvolution problem. Then the determination of the line spread function is explained. Finally the mathematics of the trigonometric moment estimation and the Levenberg-Marquardt algorithm are shown.

3.1.1 The deconvolution problem

The positioning process is a deconvolution problem. The fluorescence signal of the illuminated atoms is directed by the optical setup to the EMCCD sensor of the camera. In mathematical terms, the signal of the atoms is convoluted with the *Point Spread Function* (PSF). The 2D imaged intensity distributions reads

$$\begin{aligned} I_{2D}(x,y) &= (P_{2D} * O_{2D})(x,y) + \epsilon_{2D}(x,y) \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} P_{2D}(x-u, y-v) O_{2D}(u,v) \mathrm{d}u\mathrm{d}v + \epsilon_{2D}(x,y), \end{aligned} \tag{3.1}$$

where $O_{2D}(x,y)$ denotes the intensity distribution as originally emitted by the atoms, $P_{2D}(x,y)$ the area normalized PSF and $\epsilon_{2D}(x,y)$ additive measurement errors. $P_{2D}(x,y)$ contains the whole information about the optical setup. The typical form of a PSF for a point source and a aberrations free optical system with a circular entrance pupil is the well-known Airy pattern, which is a central disc surrounded by concentric rings of successively decreasing intensity.

With the camera used in the experiment, 512 x 512 equally spaced sampling points can be taken from the imaged intensity distributions,

$$I_{2D}[x_i, y_j] = (P_{2D} * O_{2D})[x_i, y_j] + \epsilon_{2D}[x_i, y_j]. \tag{3.2}$$

The indices i and j denote the horizontal and vertical position of the pixel, and squared brackets are used to indicate discrete functions in contrast to continuous ones.

Only little information is obtained from the original signal $O_{2D}(x, y)$ reading out the camera data. Furthermore, noise is added during the digitalization process. Therefore, deconvolution is inherently ill-conditioned leading to a non-unique solution for a given measurement.

The problem of reconstructing an original signal from a discrete number of sample points appears in various areas of signal and image processing. Different approaches can be found in literature, depending on used idealization assumptions and available information [14].

For the positioning algorithm, the PSF is known in beforehand. It is extracted from multiple measurements by an iterative process as described in section 3.1.2. Furthermore, the original signal is modelled by a sum of intensity spikes of a homogeneous background. This preconditions lead to the method of parametric deconvolution which is explained in section 3.1.4.

A further simplification for the algorithm is applied as only the axial positions of the atoms in the dipole trap are of interest for the experiment. Therefore, Eq. 3.2 can be reduced to a 1D version by binning the acquired intensity distribution vertically $I[x_i] = \sum_j I_{2D}[x_i, y_j]$, resulting in the convolution equation

$$I[x_i] = (L * S)[x_i] + \epsilon[x_i]. \tag{3.3}$$

$L(x) = (\delta * P_{2D})(x)$ is the area normalized *Line Spread Function* (LSF), which is the 1D equivalent to the 2D PSF. S is the simplified model of the unknown source distribution

$$S(x) = a_0 + \sum_{j=1}^{N} a_j \delta(x - \xi_j), \tag{3.4}$$

where a_0 denotes the constant background signal, $\delta(x)$ the Dirac delta function representing the intensity spikes, a_j the fluorescence contributions, and ξ_j the positions of N atoms.

3.1.2 Determining the line spread function

The knowledge of the LSF is central for the whole deconvolution process. Therefore a method is applied to reconstruct the LSF with sub-pixel accuracy by using the intensity distributions of up to hundreds of single isolated atoms. The determination of the LSF is done once before the position measurements and does not have to be repeated until the experimental setup changes.

Let $I_k[x_{k,i}]$ be K binned intensity distributions with $k = 1, ..., K$ of images of one isolated atom each. They all contain one ROI of the same width $M\Delta$ of M pixels defined by the interval $J_k = \{x_{k,1}, ..., x_{k,M}\}$ and Δ being the pixel distance.

First, an area normalized model of the LSF $L(x)$ roughly approximates the shape of the intensity distribution of a single atom with using a Gaussian function, for example. The position of the isolated atom ξ_k is then determined by fitting $L(x)$ to each of the $I_k[x_{k,i}]$ by minimizing

$$\min_{\{a_k,\xi_k,b_k\}} \sum_{i=1}^{M} \{a_k L(x_{k,i} - \xi_k) + b_k - I_k[x_{k,i}]\}^2. \tag{3.5}$$

Here, a_k denotes the fluorescence contribution of the atom and b_k the background. The position ξ_k can be anywhere in the interval $J_k^c = [x_{k,1}, x_{k,M}]$. An approximation of the continuously sampled representation $I_k(x)$ can be written as

$$I_k(x) = \sum_{i=1}^{M} I_k[x_{k,i}] R_\Delta(x - x_{k,i}) \tag{3.6}$$

with the rectangular function

$$R_\Delta(x) = \begin{cases} 1 & for - \Delta/2 < x \leq \Delta/2 \\ 0 & else. \end{cases} \tag{3.7}$$

.

By superimposing $I_k(x)$, the mean single atom distribution in the interval $\tilde{J}^c = [-d, d]$ with $d = min_k\{\xi_k - x_{k,1}, x_{k,M} - \xi_k\}$ is obtained and leads to

$$\tilde{I}_x = \frac{1}{K} \sum_{k=1}^{K} I_k(x - \xi_k) \tag{3.8}$$

$\tilde{I}(x)$ can be sampled on a $s \,\epsilon\, \mathbb{N}_{>1}$ denser grid $\tilde{J} = \{\tilde{x}_1, ..., \tilde{x}_{sM}\}$ by calculating $\tilde{I}[\tilde{x}_j] = \tilde{I}(\tilde{x}_j)$. If the number of considered intensity distributions K is bigger than the refining factor s, this procedure leads to an intensity distribution with sub-pixel accuracy and noise is reduced by a factor of $\sqrt{K}$. This intensity distribution can then be used as a new initial model function, restarting the procedure again. After some iterations of this self-consistency loop, a precise analytical model of the LSF is obtained.

Applying this method shows that the signals of the single atoms take the form of a central Gaussian peak overlapped with another peak of different amplitude and deviation values. In addition, a smaller Gaussian peak with some offset representing the first concentric ring of the Airy pattern is observed. This leads to a LSF defined as

$$L(x) = \begin{cases} \frac{1}{C_{L0}}[C_{L1}e^{\frac{-x^2}{2C_{L2}^2}} + e^{\frac{-x^2}{2C_{L3}^2}} + C_{L4}e^{\frac{-(x^2-C_{L6})^2}{2C_{L5}^2}}] & x < 0 \\ \frac{1}{C_{R0}}[C_{R1}e^{\frac{-x^2}{2C_{R2}^2}} + e^{\frac{-x^2}{2C_{R3}^2}} + C_{R4}e^{\frac{-(x^2-C_{R6})^2}{2C_{R5}^2}}] & x > 0 \end{cases} \tag{3.9}$$

where the parameters C_{L0} and C_{R0} normalize the LSF over the full integral and the other C-parameters depend on the experimental setup.

The measured intensity distribution can be written as

$$I[x_i] = a_0 + \sum_{j=1}^{N} a_j L(x_i - \xi_j) + \epsilon[x_i]. \tag{3.10}$$

When $L(x)$ is known, the positioning problem breaks down to a parameter estimation of a_j and ξ_j.

3.1.3 Number of atoms

For the parameter estimation, the number of atoms N within the trap is needed to be determined. Before N is calculated, it is more efficient to divide the 1D data $I[x_i]$ into ROIs which contain atoms and redundant regions which only contain background noise first. The positioning process is then executed for each single ROI. A ROI is defined by exceeding a certain intensity threshold over a fixed range. These ROIs are then extended by a fixed width wing on both sites in order to incorporate the whole fluorescence contributions of marginal atoms. If any ROIs overlap after that procedure, they are merged as indicated in Fig. 3.1 (c).

The calculation of the number of atoms N_k in the ROI k is straightforward as it can be assumed that each atom has more or less the same fluorescence contribution. Therefore only the integrated fluorescence contribution I_k^{tot} of a ROI subtracted by the background a_0 needs to be divided by the single atom fluorescence contribution I_a

$$N_k = I_k^{tot}/I_a, \;\; with \;\; I_k^{tot} = \sum_{i=1}^{M_k}(I_k[x_{k,i}] - a_0), \tag{3.11}$$

where M_k denotes the number of pixels and $I_k[x_{k,i}]$ the measured intensity values of the ROI. The background a_0 is calculated as the mean value of the fluorescence contributions of redundant regions without atoms. Detailed calculations as done by Karski [4] show that as long as the signal-to-noise ratio and the spatial inhomogeneity of atom illumination and detector sensitivity is sufficiently accurate, the reliability of the counting procedure is extremely high.

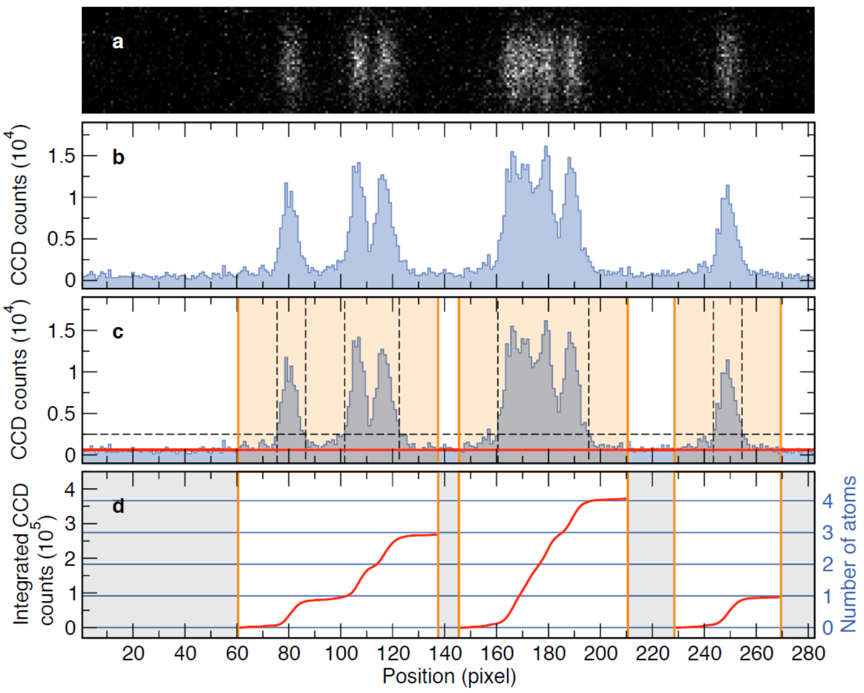


Figure 3.1: Process of image segmentation and atom counting. The 2D data in (a) is binned to 1D data in (b) and divided into regions of interest in (c) with overlapping ROIs being merged. (d) The step-function of the integrated fluorescence contribution. Figures are taken from [4]

3.1.4 Trigonometric moment estimation

With the background a_0 and the number of atoms N known, all necessary parameters for the first step of the positioning algorithm are available. The aim is to get an initial approximation for the parameters in Eq. 3.10. This guess is then used as starting values for the iterative Levenberg-Marquardt algorithm, where the positions are refined. The deconvolution method itself is a slightly modified version of the spike-convolution model fitting described by Li [6]. Although this method is originally designed for the problem of base-calling in DNA sequencing, the signal structure itself is the same as in the positioning task, namely Dirac deltas convolved with a known PSF and some additive noise. It can be shown that the parameter estimation in Eq. 3.10 is the Fourier dual of the so-called hidden frequencies problem. In signal processing, this problem denotes the search for ω_j in

$$s = \sum_{j=1}^{p} A_j \cos(t\omega_j + \theta_j) + \epsilon, \tag{3.12}$$

with positive amplitude value A_j and ϵ as white noise. In literature one solution strategy suggests the use of Toeplitz matrices constructed from autocovariance functions [9]. A similar idea is used for the positioning problem by estimating the peak locations when connecting deconvolution and the spectral structure of Toeplitz matrices constructed from the Fourier coefficients of the measured intensity distributions.

At first, a binned intensity distribution $I[x_l]$ of a single ROI defined by the interval $J = \{x_1, ..., x_M\}$ with N atoms and a distance Δ between neighbouring pixels is assumed. The LSF is expected to be twice continuously differentiable and to have a finite support in $[-\beta, \beta]$, with $\beta = M\Delta/2$ being the half width of the ROI. Under this conditions, the Fourier coefficients of the LSF

$$\hat{v}_j = \frac{1}{2\pi} \int_{-\beta}^{\beta} L(x) e^{ij\pi x/\beta} \mathrm{d}x \tag{3.13}$$

do not vanish. In the next step the Fourier coefficients of the binned intensity distribution

$$\hat{f}_j = \frac{1}{M} \sum_{l=1}^{M} I[x_l] e^{ij\tau(x_l)} \tag{3.14}$$

with the transformation $\tau : [x_l, x_M] \to [-\pi, \pi], x \mapsto \tau(x) = -\pi + \pi(x - x_1)/\beta$ are computed.

This leads to the calculation of the Fourier coefficients $\hat{g}_j$ of our signal

$$\begin{aligned} \hat{g}_0 &= \hat{f}_0 \\ \hat{g}_j &= \frac{\hat{f}_j \hat{v}_0}{\hat{v}_j} \\ \hat{g}_{-j} &= \hat{g}_j^* \end{aligned} \tag{3.15}$$

to construct the Toeplitz matrix

$$\hat{G}_N = (\hat{g}_{m-j})_{j,m=0,...,N}. \tag{3.16}$$

The smallest eigenvalue of $\hat{G}$ equals the background a_0 of the modelled source distribution of Eq. 3.4. The corresponding eigenvector $\alpha = (\alpha_0, ..., \alpha_n)$ determines the polynomial equation

$$\prod_{j=1}^{N} (z - e^{i\tau_j}) = \sum_{j=0}^{N} \alpha_j Z^j = 0, \tag{3.17}$$

of which the roots lie on the unit circle $\{e^{i\tau_j} | j = 1, ..., N\}$. They lead to the trigonometric moment estimates of the atomic positions $\xi_j = \tau^{-1}(\tau_j)$. For that the inverse transformation $\tau^{-1} : [-\pi, \pi] \to [x_1, x_M], x \mapsto \tau^{-1}(t) = x_1 + \beta(t + \pi)/\pi$ is used.

The fluorescence contributions a_j can subsequently be estimated by minimizing

$$\min_{\{a_j | j=1,\dots N\}} \sum_{l=1}^{M} \left\{ I[x_l] - a_0 - \sum_{j=1}^{N} a_j L(x_l - \xi_j) \right\}^2 . \tag{3.18}$$

Karski [4] proposes a linear least squares method based on Givens decomposition for this task.

3.1.5 Levenberg-Marquardt algorithm

The use of the trigonometric moment estimation method provides fast computation as shown in Fig. 4.10. However, the position estimations are quite inaccurate. In order to refine this position estimations, in the second step of the positioning algorithm they are used as an initial guess for an iterative non-linear maximum likelihood minimization algorithm applying the Levenberg-Marquardt algorithm. In the following the underlying principles of this algorithm will be discussed as presented in [10].

The approach of the Levenberg-Marquardt algorithm for nonlinear fitting is to define a merit function χ^2 which depends on the set of M unknown parameters a_k. The best-fit parameters are determined by minimizing χ^2. In our case this minimization problem is written as

$$\min_{\{a_0, a_j, \xi_j | j=1,\dots N\}} \sum_{l=1}^{M} \frac{1}{\sigma_l^2} \left\{ I[x_l] - a_0 - \sum_{j=1}^{N} a_j L(x_l - \xi_j) \right\}^2 , \tag{3.19}$$

where the noise for each pixel $\epsilon[x_l]$ can be considered optionally by weighting the fitted data with $\sigma_l^2 = Var(\epsilon[x_l])$.

In a first step, the merit function is approximated near the minimum by a quadratic form

$$\chi^2(\mathbf{a}) \approx \gamma - \mathbf{d} \cdot \mathbf{a} + \frac{1}{2} \mathbf{a} \cdot \mathbf{D} \cdot \mathbf{a}, \tag{3.20}$$

where $\mathbf{d}$ is an M dimensional vector and $\mathbf{D}$ is a $M \times M$ matrix, namely the Hessian matrix.

If the approximation is sufficiently accurate, one can calculate the minimizing parameters a_{min} directly from the current ones a_{cur} in a single step with the *inverse-Hessian method*:

$$\mathbf{a}_{min} = \mathbf{a}_{cur} + \mathbf{D}^{-1} \cdot [-\nabla \chi^2(\mathbf{a}_{cur})]. \tag{3.21}$$

For a weak local approximation, one can only make one step in the direction of the gradient, expressed by the steepest descent method

$$\mathbf{a}_{next} = \mathbf{a}_{cur} - constant \times \nabla \chi^2(\mathbf{a}_{cur}). \tag{3.22}$$

The *constant* has to be chosen sufficiently small in order to avoid overstepping the minimum. Because the Hessian matrix $\mathbf{D}$ is known, either Eq. 3.21 is used as long as it improves the fit or Eq. 3.22 in case it worsens the fit.

Be

$$y = y(x|\mathbf{a}) \tag{3.23}$$

the model to be fitted, the merit function can be written as

$$\chi^2(\mathbf{a}) = \sum_{i=0}^{N-1} \left[\frac{y_i - y(x_i|\mathbf{a})}{\sigma_i}\right]^2 . \tag{3.24}$$

The components of the gradient of χ^2 with respect to the parameters $\mathbf{a}$ are therefore

$$\frac{\partial \chi^2}{\partial a_k} = -2 \sum_{i=0}^{N-1} \frac{[y_i - y(x_i|\mathbf{a})]}{\sigma_i^2} \frac{\partial(x_i|\mathbf{a})}{\partial a_k} \quad k = 0, 1, ..., M-1. \tag{3.25}$$

The second derivation yields

$$\frac{\partial^2 \chi^2}{\partial a_k \partial a_l} = 2 \sum_{i=0}^{N-1} \frac{1}{\sigma_i^2} \left[\frac{\partial y(x_i|\mathbf{a})}{\partial a_k} \frac{\partial y(x_i|\mathbf{a})}{\partial a_l} - [y_i - y(x_i|\mathbf{a})] \frac{\partial^2 y(x_i|\mathbf{a})}{\partial a_l \partial a_k}\right]. \tag{3.26}$$

By convention, the factor 2 is removed when defining

$$\beta_k \equiv -\frac{1}{2}\frac{\partial \chi^2}{\partial a_k} \quad \alpha_{kl} \equiv \frac{1}{2}\frac{\partial^2 \chi^2}{\partial a_k \partial a_l} \tag{3.27}$$

with α being called the *curvature matrix*. Eq. 3.21 can be rewritten as a set of linear equations

$$\sum_{l=0}^{M-1} \alpha_{kl} \delta a_l = \beta_k \tag{3.28}$$

which is solved for δa_l. Adding the current approximation $\mathbf{a}_{cur}$, this yields the next approximation $\mathbf{a}_{next}$. Similar, Eq. 3.22 translates to

$$\delta a_l = constant \times \beta_l \tag{3.29}$$

Marquardt found, based on an earlier suggestion of Levenberg [5], a method to effectively switch between the inverse-Hessian method in Eq. 3.28 and the steepest descent method given by Eq. 3.29 [7]. The former is used close to the minimum whereas the latter when being far away from the minimum.

Marquardt's first insight was that there is some information about a reasonable scale of the *constant* in the steepest decent method given from the Hessian matrix. As the quantity χ^2 is nondimensional and β_k has the dimensions of $1/a_k$, the constant must have the dimensions of a_k^2 given by the relation in Eq. 3.29. Only the reciprocal of the diagonal element of matrix α have these dimensions which defines the scale of the *constant*. Another fudge factor δ is introduced in case this scale is too big. With this, Eq. 3.29 can be replaced with

$$\delta a_l = \frac{1}{\lambda\, \alpha_{ll}} \beta_l \; or \; \lambda\, \alpha_{ll}\, \delta \alpha_l = \beta_l. \tag{3.30}$$

Marquardt's second idea was to combine Eq. 3.30 with Eq. 3.28 by defining a new matrix α'

$$\begin{aligned} \alpha'_{jj} &\equiv \alpha_{jj}(1+\lambda) \\ \alpha'_{jk} &\equiv \alpha_{jk} \quad (j \neq k), \end{aligned} \tag{3.31}$$

which leads to one single equation

$$\sum_{l=0}^{M-1} \alpha'_{kl} \delta a_l = \beta_k \tag{3.32}$$

For very large λ, the matrix α' is forced into being diagonally dominant, so Eq. 3.32 gets identical with Eq. 3.30. For $\lambda << 1$ on the other hand, it merges to Eq. 3.28.

With these definitions, the Levenberg-Marquardt algorithm can be implemented as follows:

1. Compute $\chi^2(\mathbf{a})$.
2. Pick a modest value for λ (e.g. 0.001).
3. Solve Eq. 3.32 for $\delta\mathbf{a}$.
4. Evaluate $\chi^2(\mathbf{a} + \delta\mathbf{a})$.
5. If $\chi^2(\mathbf{a} + \delta\mathbf{a}) \geq \chi^2(\mathbf{a})$, increase λ by a substantial factor (e.g. 10) and go back to step 3.
6. If $\chi^2(\mathbf{a} + \delta\mathbf{a}) < \chi^2(\mathbf{a})$, decrease λ by a factor of 10, update the trial solution $\mathbf{a}_{next} = \mathbf{a}_{cur} + \delta\mathbf{a}$, and go back to step 3.

The Levenberg-Marquardt algorithm is already implemented within the Nonlinear Curve Fit Virtual Instrument (VI) of LabVIEW. There are two parameters, the maximum number of iterations and a tolerance value. The tolerance value specifies the relative change in the weighted distance between the measurement points and the current fit, given by $(\chi^2(\mathbf{a} + \delta\mathbf{a}) - \chi^2(\mathbf{a}))/\chi^2(\mathbf{a})$. The Levenberg-Marquardt algorithm tends to oscillate near the minimum in a flat valley of complicated topography. This behaviour stems from a problem with regard to near-degeneracy of the minimum. A small pivot leads to a large correction which according to step 5 is then rejected. At the same time, the value of λ is increased. For large λ, the matrix α' becomes positive-definite. Therefore it has no small pivots, avoiding failure by a zero pivot. This leads to the use of the steepest descent method in very unsteep degenerate valleys, which causes the oscillation.

The stopping condition of the algorithm is defined by when the relative change falls below a certain threshold. For example, a tolerance of 0.001 is mentioned by Press et al. [10] which is close to the value of 0.01 that emerged from measurements in Chap. 4.

3.2 Implementation

In this section the implementation of the positioning software, which has been developed within the course of this thesis, is described.

3.2.1 Structure

The positioning program consists of three main parts, each executed on different hardware as described in Chap. 2. Due to the amenities of NI LabVIEW[1], this does not mean that each part has been written in a different programming language. Instead, all pieces of code are written in LabVIEW 2012 extended with the Real-Time[2] and the FPGA[3] add-on modules. Fig. 3.2 shows the basic structure of the software project.

The FPGA part mainly performs the read out of the camera data transferred via the splitter box. Some basic operations needed for the position algorithm are already implemented at the FPGA such as binning the 2D data into 1D data, the segmentation into ROIs and the calculation of the number of atoms per ROI and the background.

The advanced mathematical methods are processed within the Real Time part. This includes the trigonometric moment estimation and the Levenberg-Marquardt algorithm. The logic to save the raw camera data and the position information to disk is also implemented in this part.

The Host PC serves as a user interface to set the configuration parameters of the algorithm. Once configured, the real time system works as a standalone system.

[1]http://www.ni.com/labview/
[2]http://www.ni.com/labview/realtime/
[3]http://www.ni.com/fpga/

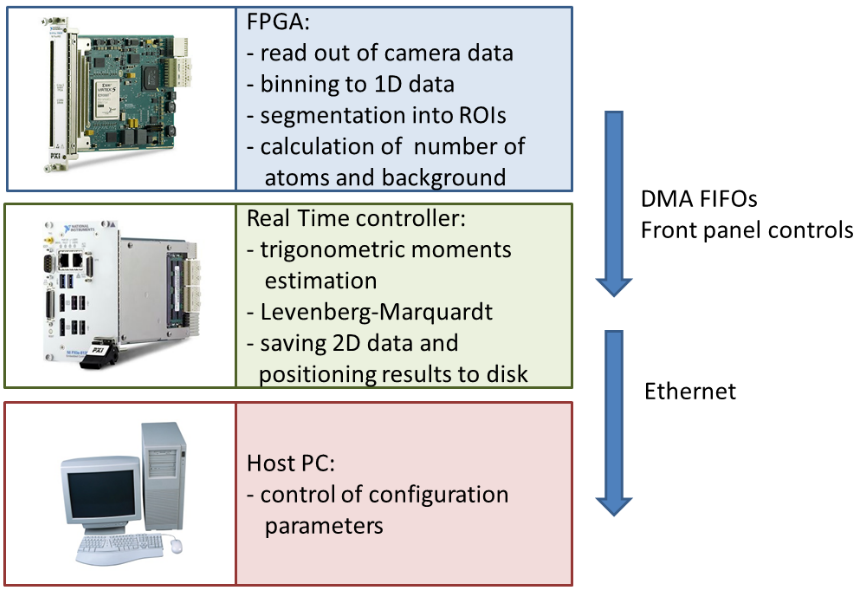

FPGA:
- read out of camera data
- binning to 1D data
- segmentation into ROIs
- calculation of number of atoms and background
Real Time controller:
- trigonometric moments estimation
- Levenberg-Marquardt
- saving 2D data and positioning results to disk
Host PC:
- control of configuration parameters
DMA FIFOs
Front panel controls
Ethernet

Figure 3.2: The program consists of three parts implemented on different hardware. For the communication between FPGA part and Real Time part, Direct Memory Access (DMA) channels and Front Panel elements of the FPGA VI are used. Ethernet communication is used between Real Time part and Host PC.

3.2.2 FPGA part

The advantage of the FPGA is the deterministic execution at low clock cycles. The camera data arriving with 10 MHz (Fig. 2.4) are sampled by the default timebase of the FPGA module of 40 MHz. According to the sampling theorem ([13]), a sampling rate of at least twice the maximum signal frequency is necessary to reconstruct the original signal which is fulfilled when using the FPGA. When using the Single Cycled Timed Loop[4] within the LabVIEW FPGA module, all contained code is executed within one tick of the FPGA's timebase.

The source code of the FPGA design is shown in Fig. 3.3. It consists of two loops, one containing the sampling and processing code, the other one is responsible for the handshake mechanism to communicate with the RT part.

In order to understand this communication, a short introduction to the possible communication mechanisms between the FPGA and the RT module is given[5]. In principle, there are two possibilities to transfer data between a FPGA and a CPU.

[4]http://digital.ni.com/public.nsf/allkb/722A9451AE4E23A586257212007DC5FD
[5]http://zone.ni.com/reference/en-XX/help/371599F-01/lvfpgaconcepts/pfi_data_transfer/

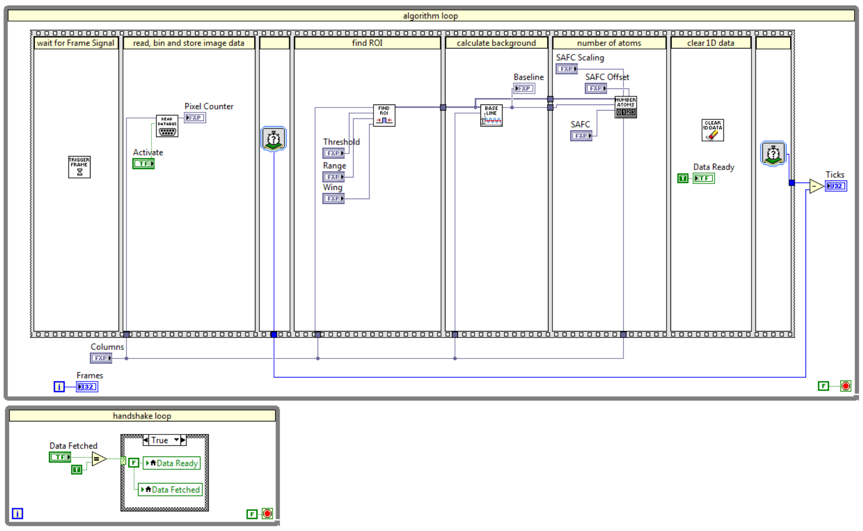


Figure 3.3: The main structure of the FPGA part ([FPGA] Main.vi).

The first one is the programmatic front panel communication. With a low throughput rate and a small call overhead, it is recommended to exchange single values, such as configuration parameters and flags, which need to be transferred immediately. It is very easily implemented as the RT part simply calls the Front Panel elements of the FPGA Virtual Instrument window when using the Read/Write Control functionality of the FPGA interface palette of LabVIEW.

For larger amounts of data, Direct Memory Access (DMA) FIFOs are the best choice. A DMA FIFO allocates memory on both the RT system and the FPGA target. The big advantage of DMA transfers is the direct access of the RT system's memory without involving its processor. Transfer latencies are introduced due to the overhead of the communication architecture and therefore it is recommended for big amounts of data.

A handshake mechanism for the safe data communication between FPGA and RT is used. When a picture has been processed on the FPGA and all relevant data has been written to the respective DMA channels, a flag *Data Ready* is set to true which tells the RT program to read out the DMA FIFOs and to start processing. The RT program controls another flag, *Data Fetched*, which is set to true after emptying the FIFOs at the RT part to tell the FPGA program to reset *Data Ready* and *Data Fetched* back to false, getting the system ready for a new data transfer.

Four DMA FIFOs are necessary to transfer the relevant data from the FPGA to the RT system:

- *Data 2D DMA*: the actual image data as obtained from the camera is written pixel by pixel. With a size of 8000 elements, it can buffer image data up to a size of 512 x 15 pixels. The loop which reads the FIFO at the RT side has a period of 500 µs. With respect to the 10 MHz camera clock, only 5000 pixels are sent to the RT in order to avoid overflow.

- *Data 1D DMA*: the binned camera data is transferred. The size of this FIFO is 512 elements according to the maximum image width.
- *ROIs DMA*: the result of the image segmentation is written. It contains the pixel number of the start and end of all ROIs. A size of 256 elements allows the transfer of up to 128 ROIs, which in practice is unlikely to happen.
- *Number Atoms DMA*: the number of atoms for each ROI is transferred.

In addition to the DMA FIFOs Front Panel communication is implemented in order to set the relevant algorithm parameters:

- *image width*: the horizontal width of the image in pixels which is sent from the camera.
- *Threshold*: the fluorescence value above which the segmentation mechanism starts to classify the pixels as part of a ROI.
- *Range*: the minimum number of pixels above the threshold value which determines a ROI.
- *Wing*: the number of pixels by which a ROI is extended to both sites in order to include the complete fluorescence contribution of an atom peak.
- *SAFC*: the Single Atom Fluorescence Contribution is the mean integrated fluorescence of a single atom.
- *SAFC Scaling* and *SAFC Offset*: these parameters are used to compensate spatial differences in the sensitivity of the EMCCD chip. The SAFC is modified by the formula $SAFC = SAFC + x^2 * Scaling$, where x is the difference between the centre of the current ROI and the centre of the total image width shifted by the offset value. This allows to calculate a correct number of atoms even if the chip is more sensitive in peripheral areas.

The [FPGA] Main.vi consists of two loops, the algorithm and the handshaking loop as indicated in Fig. 3.3.

In idle mode, when there is no signal from the camera, the inner Flat Sequence Structure of the algorithm loop will stop at its first frame. This is determined by the *[FPGA] Trigger Frame Signal.vi* whose source code is shown in Fig. 3.4. The loop will run as long as the value of the IO Module\Frame Signal Node is set to *low* equivalent to $false$. When the *Frame Signal* switches to *high* respectively *true*, the stopping condition is met and the *Flat Sequence Structure* proceeds. This basic principle for positive edge detection is also used in other parts of the FPGA program.

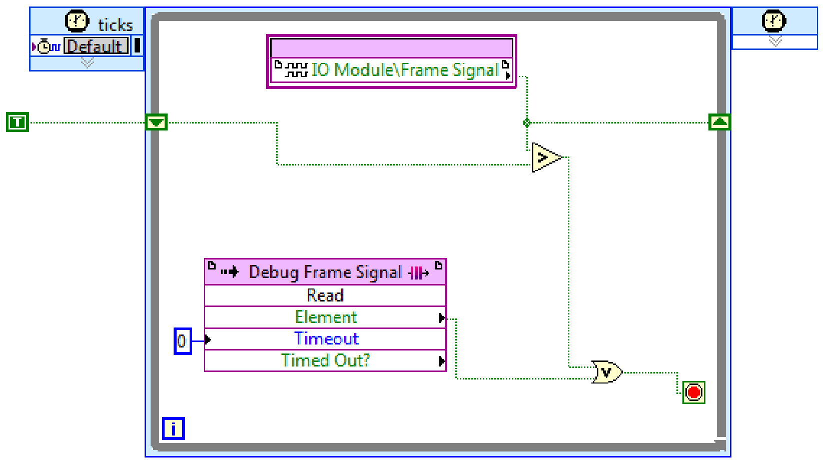


Figure 3.4: Basic principle for positive edge detection in the FPGA program. The loop will run as long as the *Frame Signal FPGA I/O Node* is set to *false*. When set to true, the stopping condition is met. Also a Debug Frame Signal is included for algorithm testing with simulated data.

As shown in Fig. 2.4, a rising Frame Signal indicates the begin of an image. With this positive edge, the program continuous with the second frame of the Flat Sequence Structure and stays there until the Frame Signal drops again, which indicates the end of an image sequence.

In the *[FPGA] Read Databus.vi*, for each rising edge of the pixel clock the 14 bits of the data bus are read from the I/O nodes. In the same step the 2D pixel values are accumulated and stored into LUT logic, resulting in the binned 1D data. In contrast to BRAM slices LUT can be read and written within one tick which is necessary for the binning mechanism. To save resources on the FPGA, mainly the fixed-point data type is used.

The falling edge of the frame signal determines the end of an image sequence and stops the read out loop. In the next step, the 1D data from the LUT is written to the Data 1D DMA FIFO and at this point in time already available in the RT system to store the 1D data to disk.

After receiving all data from the camera, the actual data processing can start. In the *[FPGA] Find ROI.vi*, the positions of the ROIs are identified and the merging of overlapping ROIs takes place. The pixel number of start and end of each ROI are written to the ROIs DMA FIFO.

The next step is determined by the source code within the [FPGA] Calculate Baseline.vi. The fluorescence values of all pixels which are not within a ROI are summed up, which denotes the mean value of the background.

This background value is then used in *[FPGA] Number Atoms.vi* to calculate the number of atoms per ROI according to Eq. 3.11. For each

ROI, an integer value is written to the Number Atoms DMA FIFO.

At the end of the sequence structure, the Data Ready flag is set to true as explained above, signalling the RT part to start data processing.

3.2.3 RT part

In addition to the control parameters for the FPGA explained above, more configuration parameters are needed for the data processing at the RT part:

- *LSF parameters*: these parameters define the form of the line spread function as indicated in Eq. 3.9. If the second and the third Gaussian peak are not used, the respective parameters can be set to zero.
- *SAFC Threshold*: the trigonometric moment estimation algorithm becomes very imprecise for atoms in close proximity. A good indicator for the validity of estimated positions is the fluorescence contribution a_j (Eq. 3.4). The SAFC Threshold defines a maximum for the absolute difference between the SAFC and a_j. If the difference exceeds this maximum for a ROI with a pair of atoms, two positions near the centre of the ROI are used as initial values for the Levenberg-Marquardt algorithm instead of the estimated positions [4].
- *LM Configuration*: the maximum iterations and the tolerance parameter for the stopping condition of the Nonlinear Curve Fit VI.
- *Filepath* and *Filename*: file path and name where the data should be saved on the RT controller.
- *# per File*: the number of images written into one single file.

The source code of the RT part ([RT] Main.vi) is structured into three loops as indicated in Fig. 3.5. One corresponds to a deterministic loop with a period of 500 µs. The other two are also timed with 500 µs holding non-time critical code.

In the deterministic loop, the communication and data transfer with the FPGA takes place. The Data 2D DMA FIFO is read continuously. The pixel values are written to the single process shared variable *2D Data* which is configured as a RT FIFO with a array size of 8000 elements. Single process shared variable with RT FIFO enabled secure lossless data communication.

When the number of elements in the Data 1D DMA FIFO equals the specified image width, the elements are read from the DMA FIFO and written to the shared variable *1D Data*. At the same time, the time stamp for the measurement is generated. This time stamp is written to two shared variables, *Timestamp Data* and *Timestamp Positions*. The former determines the time stamp for the 2D data file. The latter is for the positions data file and only written if its current value is equal to the text string *ready*. This handshaking is needed to deal with the case when the positioning process takes longer than the exposure time of the images. In this case, new image data arrives although the positioning algorithm is still occupied with

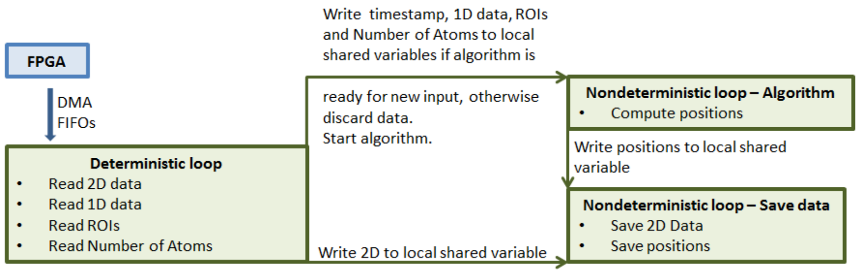

FPGA
DMA FIFOs
Deterministic loop
Read 2D data
Read 1D data
Read ROIs
Read Number of Atoms
Write timestamp, 1D data, ROIs and Number of Atoms to local shared variables if algorithm is ready for new input, otherwise discard data. Start algorithm.
Write 2D to local shared variable
Nondeterministic loop – Algorithm
Compute positions
Write positions to local shared variable
Nondeterministic loop – Save data
Save 2D Data
Save positions

Figure 3.5: The basic structure of the RT part of the program consists of three loops. The deterministic loop reads out all data from the DMA FIFOs and starts the algorithm which is located in the second loop. If the algorithm is still busy with the previous image, the data is dumped and an overflow indicator is set. In the third loop the data is written to disc. The 2D data is always written, the positions only if no overflow has happened.

the old one. To handle this possible error source, another shared variable named *Algorithm Task* has been added. It acts as a flag for the status of the positioning VI. When positioning starts, its text string value is set to *busy*, when positioning finishes, it is set to *ready*. When the FPGA changes the Data Ready flag to true but the Algorithm Task variable is set to the text string value *busy*, the data from ROIs DMA FIFO and Number Atoms DMA FIFO is dumped and the *Overflow* indicator is set to true. The 1D data is saved to disk but no positioning takes place for this image.

If *Algorithm Task* is not set to *busy* and the FPGA indicates that new data is available, *Algorithm Task* is changed to the text string value *go* and the ROI and number of atoms information is written to shared variables *ROIs* and *Number Atoms*.

With *Algorithm Task* set to *go*, the *[RT] Positions.vi*, which is located in non-deterministic loops, starts the positioning process. For the software architecture of *[RT] Positions.vi*, the state machine[6] design pattern with three states is used:

1. *State 1*: depending on the value of *Algorithm Task* the positioning process is either started or stopped.
2. *State 2*: the actual positioning takes place. The logic iterates through the ROIs and calculates the atom positions within *[RT] Positions ROI.vi*, accumulating them in a *shift register*[7].
3. *State 3*: the position results together with the total number of atoms, the time stamp, the processing time and the 1D data are merged into an array and written to the RT FIFO *Positions*. There they are buffered and then written to a file. The shared variables Timestamp

[6]http://www.ni.com/white-paper/7595/en/
[7]http://www.ni.com/getting-started/labview-basics/shift-registers

Positions and Algorithm Task are set to *ready*, preparing the system for a new set of image data.

The positioning process for a ROI is a straightforward implementation of the algorithm as described by Karski [4]. It is divided into three SubVIs. In the first one, *[RT] Trigonometric Moments.vi*, the Fourier coefficients v_j and f_j of the LSF and the measured intensity distribution are calculated. The second one, *[RT] Trigonometric Positions.vi*, estimates the positions. In the third one, *[RT] LM Positions.vi*, the fluorescence contributions a_j are calculated, and the *Nonlinear Curve Fit VI*[8] is fed with all initial parameters.

[8]http://zone.ni.com/reference/en-XX/help/371361H-01/gmath/nonlinear_curve_fit/

Chapter 4

Performance

The overall aim of the project is to read out the camera data and to calculate the atom positions in realtime. A typical exposure time for the experiments is 10 ms which specifies the time frame for the realtime positioning. In the following, the performance of the implemented algorithm is presented.

First, the execution time of the positioning procedure and parts of it with varying parameters e.g. number of atoms, number of pixels and noise intensity will be examined. These benchmark results show the basic capabilities of the positioning program.

The second main part of this chapter provides the accuracy of the algorithm, i.e. the deviation of the estimated positions from the actual ones. The real physical positions of the atoms are not known, therefore camera data is simulated and fed into the algorithm for this purpose.

4.1 FPGA Performance

The camera is read out via the iXon Cable Splitter Box and the NI PXIe-7966R FPGA module with the NI 6581 Digital Adapter module. As explained above, this step also includes transferring the 2D data to the RT controller and binning it into 1D data. All these steps are performed within a Single-Cycle Timed Loop. A benchmark of this part with a varying quantity of pixels has been conducted (Fig. 4.1). The read out time starts with 2747 ticks at 4 pixels which is the minimum picture size the camera software which is provided by Andor allows to configure. At a FPGA clock rate of 40 MHz, this corresponds to about 67 µs. If a whole picture with 512x512 pixels is read out, it takes 1143317 ticks equivalent to 28583 µs. The typical image size taken during the experiments normally does not exceed 4000 pixels. A simple linear interpolation leads to a maximum read out time of 459 µs. With respect to the 10 ms threshold, this part of the algorithm does not seem to limit the realtime constraints. Anyhow, as the speed of the read out phase is determined by the 10 MHz camera clock, there are no

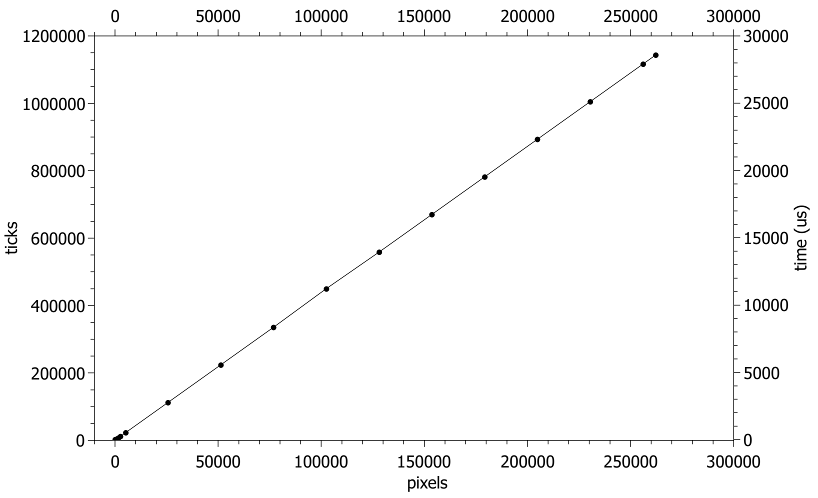


Figure 4.1: Ticks and execution time of the camera read out part of the algorithm ranging from 4 pixels (minimum camera can send) to 262144 pixels (full 512 x 512 resolution image) at 10 MHz camera clock rate.

further improvements of the read out part of the FPGA design achievable. The remaining part of the algorithm performed on the FPGA contains the fragmentation of the 1D data into region of interests (ROIs), the calculation of the background and the determination of the number of atoms per ROI.

In order to obtain comparable and repeatable timing benchmarks of this part of the code, it is not suitable nor possible to work with real physical image data from the camera as this changes from measurement to measurement. Also the exact positions of the physical atoms are not known, which are especially necessary to analyse the accuracy of the algorithm. Therefore the algorithm is fed with simulated data which is calculated on the RT controller and passed on to the FPGA by a DMA channel. For the simulation of the camera data, the form of the measured intensity distribution in Eq. 3.10 combined with the line spread function (LSF) in Eq. 3.9 is used. The parameters of the LSF are chosen in such a way that a prototypical form is achieved (Fig. 4.2 (a)).

The parameter a_0 which denotes the background is assumed to be constant over all pixels. Also the fluorescence contributions a_j are equal for each atom. The noise $\epsilon[x_i]$ is modelled by a Gaussian-distributed, pseudo-random pattern provided by the LabVIEW Gaussian White Noise PtByPt VI, where the standard deviation can be defined as a parameter. The value of this parameter is denoted in the following as the noise level. An example for simulated camera data is shown in Fig. 4.2 (b). This method of simulating camera data is used for all benchmark and measurement results provided in this chapter.

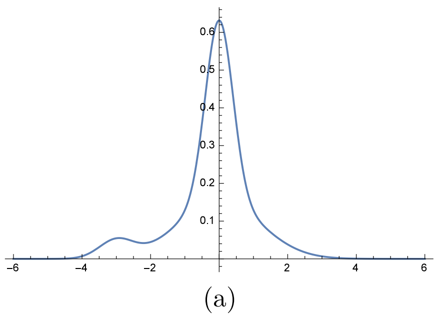


Figure 4.2: (a) Prototypical line spread function with a central Gaussian peak and another smaller Gaussian peak with some offset representing an asymmetric airy disc. (b) Simulated 1D data with 512 pixels containing 10 equidistant atoms. Background is set to 600, the fluorescence contributions to 3000, and the standard deviation of the Gaussian noise to 50 counts.

A benchmark of the FPGA program without the read out part with a simulated data of well separated, equidistant atoms is shown in Fig. 4.3 (a). The processing time reaches from 40 µs for one single atom up to 65 µs for 29 atoms. In Fig. 4.3 (b) the atoms are so close that the ROIs are finally merged into one single ROI. This leads to a slightly better performance. While one atom still needs about 40 µs, 29 atoms are processed within about 45 µs.

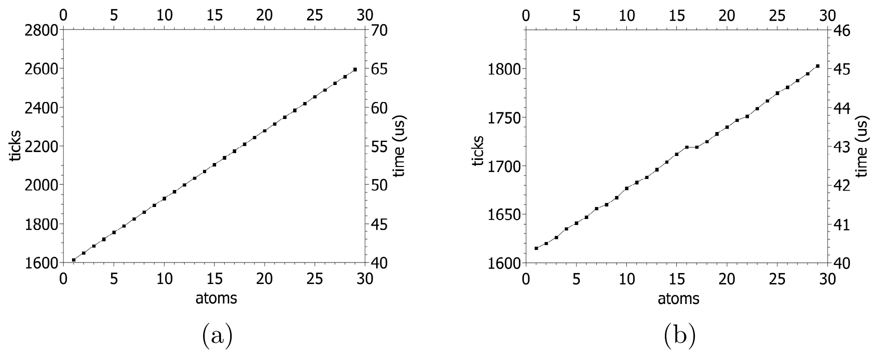


Figure 4.3: Benchmark of the processing part of the FPGA code. The graphs show a linear relationship between the processing time and the number of atoms. (a) Well separated atoms resulting in one ROI for each atom. (b) Close atoms which are finally merged into one single ROI result in a slightly better performance.

Altogether, the FPGA part of the algorithm takes about half a millisecond to execute and does only slightly depend on the number of atoms. As long as the image size does not exceed some several thousand pixels, this part represents only a small contribution to the overall duration of the image processing.

4.2 DMA Data Transfer

There are four types of data which are transferred from the FPGA to the RT system for further processing. First, the 2D pixel data which is read from the camera, represented as a stream of 16 bit unsigned integers[1]. This DMA FIFO is read continuously from the RT system. Because the 2D data is not used for further image processing but only saved to disk, the speed of this transfer has no influence on the overall algorithm performance and is therefore not benchmarked. The RT system also continuously reads the 1D data, which is written to the respective DMA FIFO right after the readout from the camera has finished. Before the RT algorithm can start, the segmentation into ROIs and the calculation of the number of atoms has to be performed on the FPGA. This gives enough time (Fig. 4.3) to transfer the 1D data without delaying the rest of the procedure and is therefore also not benchmarked. When the FPGA part of the algorithm has finished, a data ready flag is set to tell the RT system to start data processing. At this point, two other DMA FIFOs are read, which contain the ROIs and the number of atoms data. A benchmark of the whole process which includes the readout of these FIFOs, transferring the data to local shared variables and the handshake mechanism is shown in Fig. 4.4. No significant relationship between the number of ROIs and the processing time can be observed, which means the overhead is high compared to the actual data transfer. On average, the RT algorithm will start approximately 115 µs after the FPGA has indicated that all data is available.

[1]At 1 MHz readout rate, the camera is in principal able to digitalize 16 bit. Although only 14 bit are used for the current experiment, future applications may use this feature.

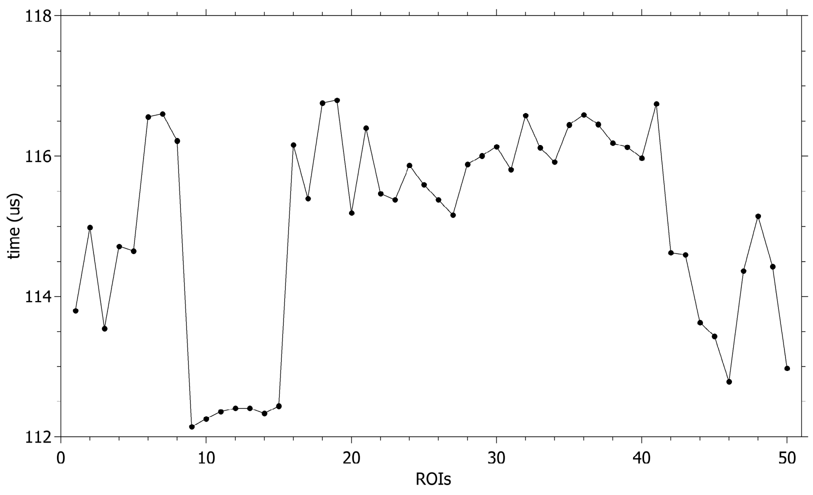


Figure 4.4: Duration of reading out the ROI and the number of atoms DMA FIFOs, transferring the data to local shared variables and the respective handshake mechanism. No significant dependence on the number of ROIs can be observed. (average over 1000 measurements for each data-point.)

4.3 Accuracy

The aim of the algorithm is to achieve sufficient accuracy within the shortest possible execution time. Accuracy in this respect refers to the deviation of the estimated positions from the real positions of the atoms.

For the trigonometric moments estimation part, there are no parameters which can be set in order to influence neither execution time nor accuracy. The only possibility for further increasing the performance lies in analysing and streamlining the code itself.

For the Levenberg-Marquardt part, some configuration of the algorithm during runtime is possible. As explained in Chap. 3, a stopping condition has to be defined for the iterative fitting process. The Nonlinear Curve Fit VI offers two parameters for this purpose. The *maximum iterations* value defines the maximum number of loops of the Levenberg-Marquardt algorithm. The *tolerance* parameter specifies the minimum relative change in the weighted distance between the input data points and the current fit. The execution time increases with the value of the maximum iteration parameter and decreases with the value of the tolerance parameter.

In Figure 4.5, the absolute deviation of positions for a setup of two atoms within a ROI of 15 pixels for four different noise levels against the tolerance parameter is shown (black). In addition, also the residue is shown (red). This value is provided by the Nonlinear Curve Fit VI and represents the weighted mean square error between the best fit and the camera data. Although strongly correlated for measurements with moderate noise, for high noise the position deviation starts to fluctuate while the residue stays quite smooth.

A significant drop of the deviation between a tolerance 1 down to 0.01 is observed. Although expected for low noise, even for higher noise level of 500 CCD counts and more this drop can still be observed, especially when looking at the residue. For smaller tolerances, deviation does not decrease any further, which is consistent with the findings in Chap. 3. Using other atom configurations, e.g. with more or differently distributed atoms provides the same result. Therefore, a tolerance of 0.01 seems to be a good choice for all measurements.

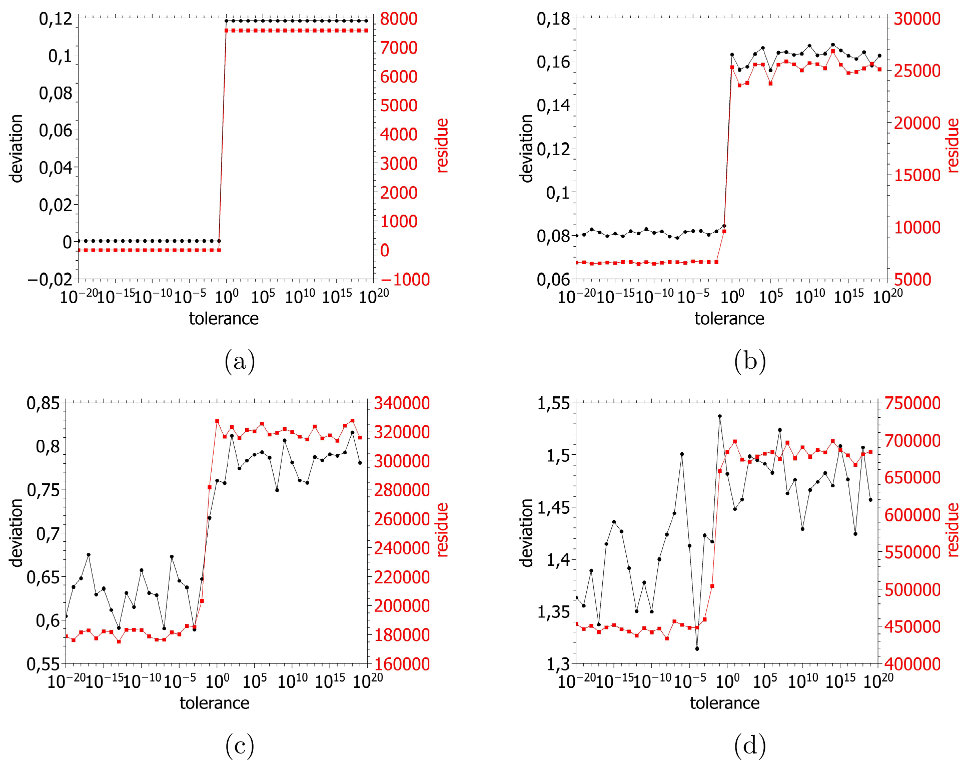
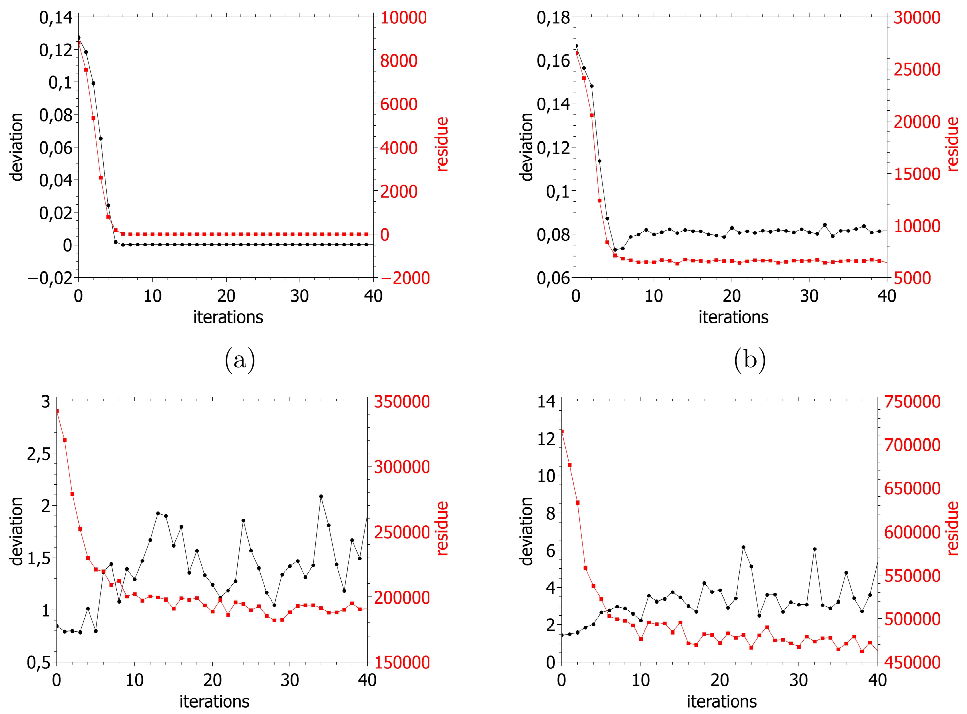


Figure 4.5: Deviation of positions (black) and residue (red) over tolerance for a noise level of 0 (a), 100 (b), 500 (c) and 1000 (d) CCD counts. (average over 1000 measurements, 2 atoms within 15 pixels, maximum iterations 2000)

In Figure 4.6, the same configuration as above is shown for a varying number of maximum iterations. Because tolerance has been set to zero for these measurements, the maximum iterations parameter provides the only stopping condition. Therefore, this parameter determines the actually performed iterations. Again for low noise measurements, no more improvement of the deviation above a certain threshold is achieved with increasing iterations. For the current configuration with two atoms within 15 pixels, a deviation minimum lies at 5 iterations. A surprising behaviour is observed for higher noise (500 CCD counts in (c) and 1000 CCD counts in (d)). Although the overall residue (red) decreases with the number of iterations, the position deviation even seems to increase a little, at least starts to fluctuate. This effect is analysed in more detail further below.

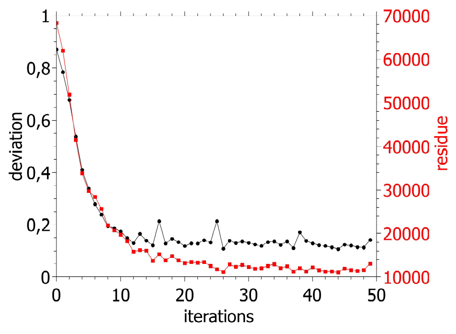


(a) (b)

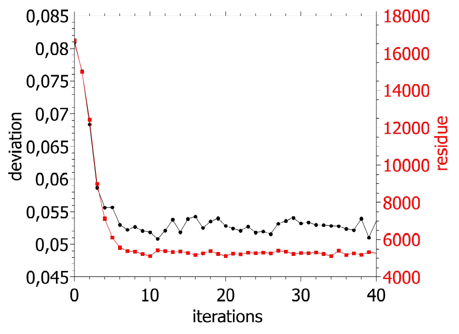
(c) (d)

Figure 4.6: Deviation of positions (black) and residue (red) against iterations for a noise level of 0 (a), 100 (b), 500 (c) and 1000 (d) CCD counts. (average over 1000 measurements, 2 atoms within 15 pixels, zero tolerance)

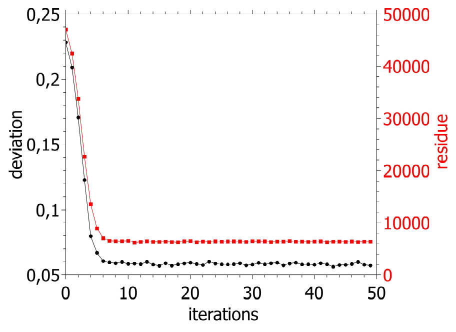
In contrast to the tolerance parameter, for the maximum iterations value there does not seem to exist a single optimal choice for all atom configurations. In Figure 4.7, example measurements for 1, 3, 4 and 5 atoms are shown. Especially for one atom, more iterations are necessary to reach a deviation minimum. For further improvement of the algorithm, these behaviour could be investigated in more detail. Maybe a mechanism which sets the maximum iterations value according to the number of atoms (and maybe also the number of pixels) within a ROI may be reasonable. At the moment, ten iterations seem to be a good compromise for most atom configurations.



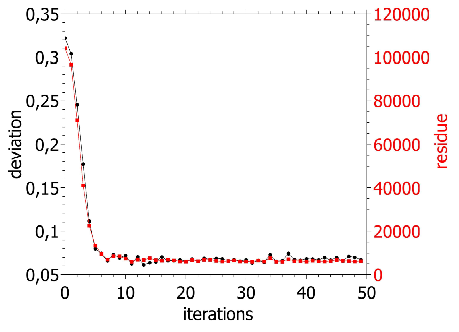
(a)



(b)



(c)



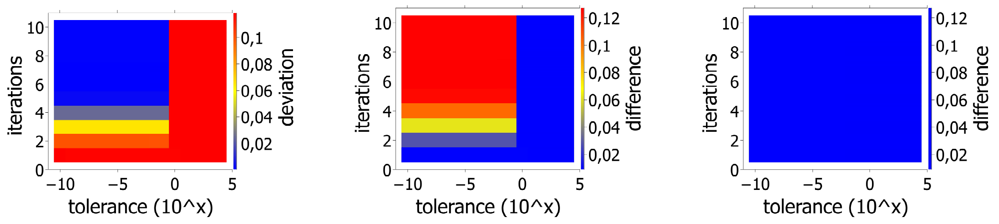
(d)

Figure 4.7: Deviation of positions (black) and residue (red) over iterations for a noise level of 100 CCD counts and 1 (a), 3 (b), 4 (c) and 5 (d) atom(s). The decrease of the position deviation depends on the specific atom configuration. (average over 1000 measurements, zero tolerance)

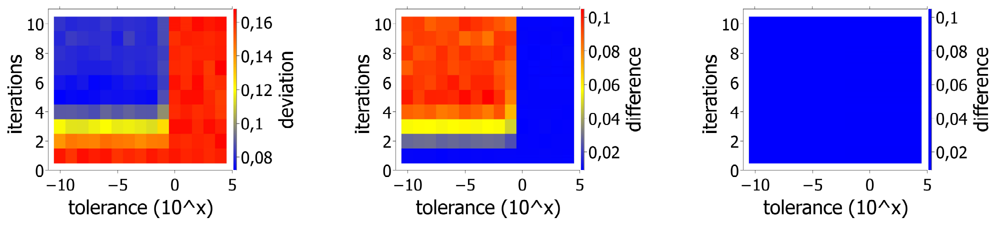
As mentioned above, for a high noise level more iterations lead to a higher position deviation. In the following, this behaviour is analysed in more detail.

In Fig. 4.8, the deviation of the estimated positions from the real positions over tolerance and maximum iterations is shown for various noise levels. In the first column, the absolute value of the deviation is shown. In the second column, the difference between the trigonometric moments estimation deviation and the Levenberg-Marquardt algorithm is shown. In the third column, the same difference is displayed, this time with only two colours. Blue areas show where Levenberg-Marquardt delivered more accurate positioning then the trigonometric moment estimation, red ones indicate a deterioration of the position estimates. In the first row where zero noise is applied to the simulated camera signal, the result is as expected. The

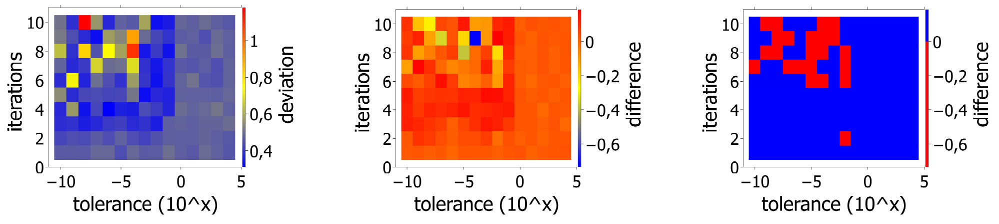
Levenberg-Marquardt VI delivers more accurate positions over the whole configuration space. Especially for areas with maximum iterations greater than 4 and tolerance smaller than 1, the difference becomes significant. With a moderate noise level of 100 CCD counts applied (compared to an atom amplitude of 3000 CCD counts) as shown in (b), the behaviour remains the same. In row (c), with a noise level of 333 CCD counts, a different result can be observed. Although still small improvements for areas with high tolerance and only a few iterations are achieved, above certain critical thresholds the effect becomes inverted. For a noise level of 1000 CCD counts in row (e), all Levenberg-Marquardt configurations show worse accuracy than the trigonometric moments estimation results.



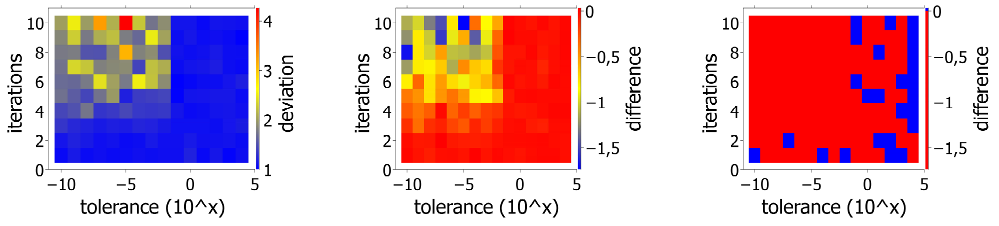
(a)



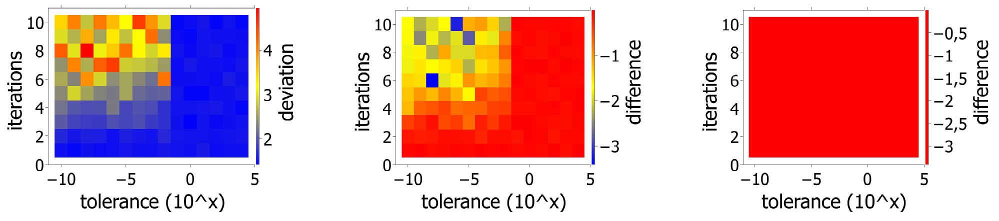
(b)



(c)



(d)



(e)

Figure 4.8: First column: Absolute mean deviation from real atom positions over maximum iterations and tolerance for a noise level of 0 (a), 100 (b), 333 (c), 666 (d) and 1000 (e) CCD counts. Second column: Mean difference between the deviation of the trigonometric moments estimation and the Levenberg-Marquardt algorithm positions. Third column: The same difference, displayed in two colours. Blue areas indicate improvement of positioning by Levenberg-Marquardt, red ones deterioration. (average over 1000 measurements, 2 atoms within 15 pixels)

This raises the question what causes this behaviour and if any improvement could be applied to the algorithm to solve this issue. One possible explanation is that due to the high noise level, the initial positioning from the trigonometric moments estimation introduces already a significant deviation. Given by the nature of the Levenberg-Marquardt algorithm, this could lead to slow converging or even divergent behaviour, resulting in amplifying rather than improving the position error. In order to check this assumption, a test bench is implemented which allows to feed the Nonlinear Curve Fit VI directly with initial position parameters instead of using the trigonometric moments estimation. These position parameters are then subsequently shifted from zero derivation to an error of 2 pixels as shown in Fig. 4.9.

In the first two graphs with a noise level of zero (a) and 100 (b) CCD counts, the final deviation of the Levenberg-Marquardt results decreases with the number of iterations, independent from the initial deviation. For higher noise levels as e.g. 500 (c) or 1000 (d) CCD counts, the opposite effect is observed.

This proves that the behaviour of divergent position errors does not originate from the initial position guess but solely from the noise from the camera signal. It seems that for highly distorted signals it is better to abstain from the use of the second step of the algorithm and directly use the trigonometric moment estimation results. In general it is doubtful if measurements with such poor signal-to-noise ratios can even be used for any reasonable analysis.

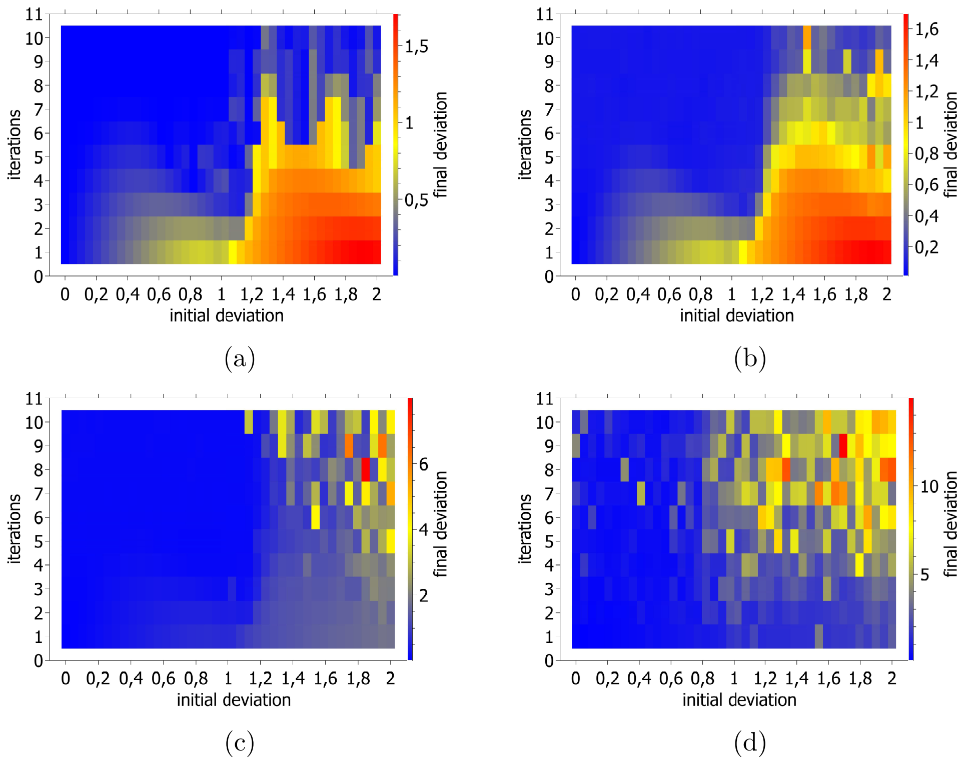


Figure 4.9: Final position deviation of the Levenberg-Marquardt algorithm over maximum iterations and initial deviation for a noise level of 0 (a), 100 (b), 200 (c), 1000 (d) CCD counts. Only for moderate noise levels an improvement of the positioning is achieved. (average over 1000 measurements, 1 atom within 10 pixels, zero tolerance

4.4 RT Performance

In the following section, various measurements concerning the execution time of the positioning algorithm are presented. In particular, the influence of the two configuration parameters of the Nonlinear Curve Fit VI, namely the maximum iterations and the tolerance value, is analysed. Also the influence of the noise level is examined.

Fig. 4.10 (a) shows the execution time for an increasing number of atoms within one single ROI. For the black graph, each atom is placed in the middle of a space of 10 pixels. This means e.g. for three atoms, that the ROI has a width of 30 pixels with atoms at position 5, 15 and 25. For the red graph, each atom has a space of 4 pixels. The noise level in this measurement is set to zero to clearly point out the relationship between the execution time and the number of atoms respectively the width of the ROI.

For the 10 pixel spaced atoms, the limit of 10 ms is already approached with three atoms (9376 µs), while four atoms even exceed it by the double (20421 µs). For the 4 pixel spaced atoms, 4 atoms still lay within the limit (8428 µs).

Fig. 4.10 (b) shows the same measurement for the trigonometric moments estimation positioning only. Although more imprecise as explained in section 4.3, the performance advantage is enormous. Even the positioning process for 15 atoms is with a duration of approximately 5 ms still within the 10 ms limit. Also, there does not seem to be a big dependency on the specific atom distribution, at least for equally spaced peaks.

These measurements show that for the Levenberg-Marquardt algorithm, the execution time strongly depends on the specific atom distribution. However, it seems that in any case precise positioning for more than 3 atoms within one ROI below 10 ms is difficult to achieve. Even if the trigonometric moment estimation may still be optimised, the Nonlinear Curve Fit VI hardly offers any possibilities to gain shorter execution times.

In practise, the atoms are predominantly not concentrated in one single ROI but distributed over a wider range. If there are multiple ROIs within one image, the execution times simply accumulate. The additional logic for the iteration through multiple ROIs is very straightforward and does not cause much overhead time. According to Fig. 4.10, the processing of a single atom in a ROI takes about 1180 µs for 10 pixels width respectively 985 µs for 4 pixels width. Adding half a millisecond from the FPGA part, this means that for single atoms located in their own ROI 8 or 9 atoms can be positioned within 10 ms. If there is one ROI with a pair of close atoms and all others are well separated, still 6 or 7 atoms can be positioned in time.

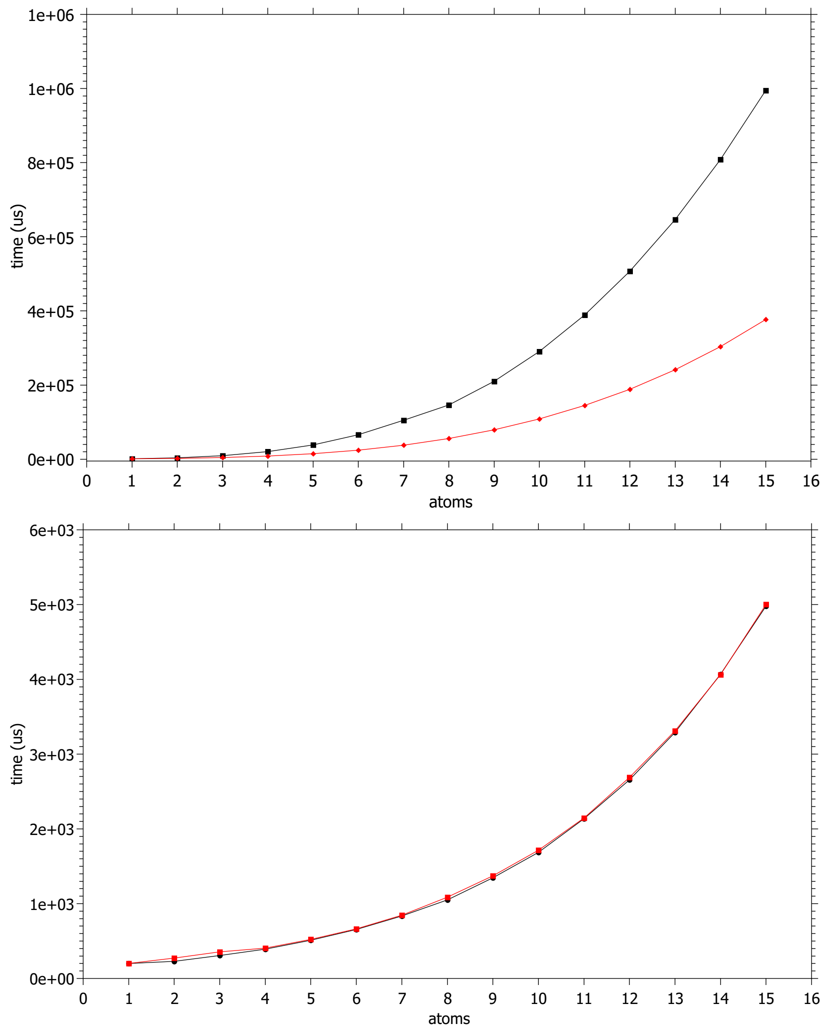


Figure 4.10: (a) Execution time of the RT positioning algorithm for one single ROI. The atoms are separated by 10 (black) and 4 (red) pixels each, which means the width of the ROI is growing with the number of atoms. (b) Same measurement for the trigonometric moments estimation part only. (average over 100 measurements for each data-point, 10 max. iterations, 0.01 tolerance, zero noise)

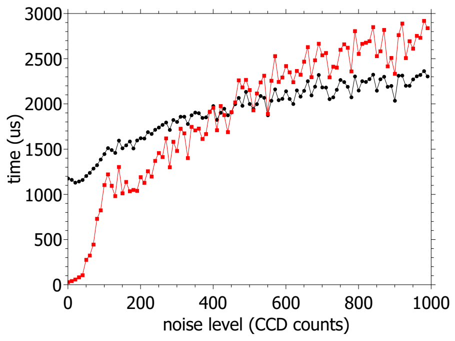
As stated above, the execution time does not only depend on the number of atoms but also on the noise level. Fig. 4.11 shows the processing time of one (a,b), two (c,d) and three (e,f) atom(s) within one ROI of 10, 15 and 20 pixels over the noise level. The amplitude of the atom peaks is set to 3000, the background to 600 CCD counts. As explained above, the noise is modelled by Gaussian-distributed, pseudo-random pattern provided by a LabVIEW VI. The level of noise denotes the standard deviation of the

Gaussian distribution. Noise is generated and added for each pixel, according to the model in Eq. 3.4 of the measured intensity distribution. Each data point in the graphs represents the mean value of 1000 measurements, each single one performed with a unique noise pattern.

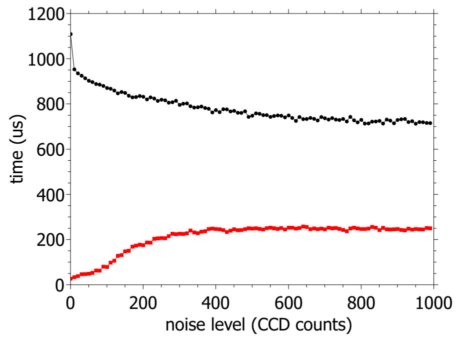
For the first column (a,c,e), the maximum iterations parameter is set to 20000 and the tolerance parameter is set to 10^{-8}. The higher the noise level the higher the mean execution time of computing the positions. There is a the strong increase of the standard deviation. For one atom, already at a noise level of about 500 CCD counts the standard deviation equals the mean value, for two and three atoms even earlier. This decreases the number of atoms which can be reliably positioned within 10 ms even further down to one or two atoms, depending on the noise level.

As shown in section 4.3, 20000 maximum iterations are an extremely high number, whereas 10^{-8} tolerance is very small for the demands of the positioning algorithm. A tolerance of 0.01 and 10 maximum iterations are for most cases reasonable settings to achieve quite a significant improvement of the position accuracy.

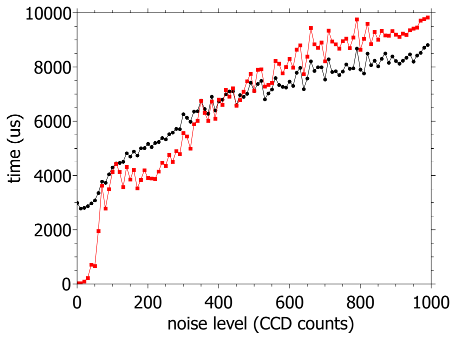
In the second column (b,d,f), the measurements are performed with these settings, leading to a totally different result. Whereas the standard deviation at the beginning still increases with the noise intensity, it finally levels off at a value considerably smaller than the corresponding mean execution time. The mean execution time itself even decreases with the noise level, also becoming nearly constant for high noise. Even for three atoms, the vast majority of the positioning processes will be performed far below 10 ms.



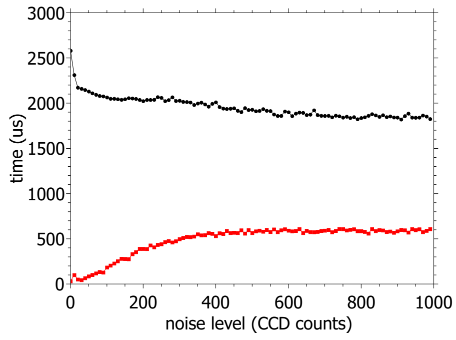
(a)



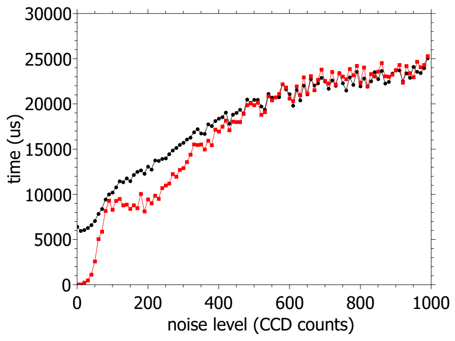
(b)



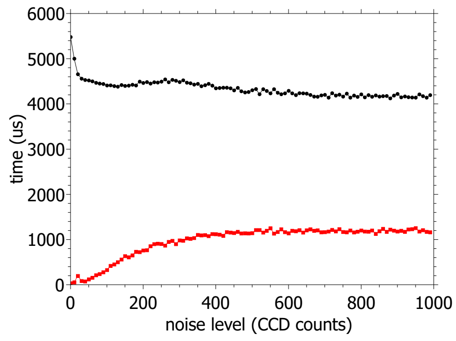
(c)



(d)



(e)



(f)

Figure 4.11: Execution time (black) and its standard deviation (red) for 1 (a,b), 2 (c,d) and 3 (e,f) atom(s) with a ROI width of 10,15 and 20 pixels over the noise level. In the first column, the tolerance of the Nonlinear Curve Fit VI is set to 10^{-8} and the maximum iterations to 20000. For the second column, tolerance is 0.01 and maximum iterations are 10. (average over 1000 measurements for each data-point)

In the following section, the influence of the tolerance respectively maximum iterations parameter on the execution time for various noise levels is discussed.

In Figure 4.12, the mean execution times for the positioning of two atoms within 15 pixels at a noise level of 0 (a), 333 (b), 666 (c) and 1000 (d) CCD counts in relation to the tolerance parameter are shown. For noise levels greater than zero, it can be observed that below a certain tolerance value, execution time begins to fluctuate. Above that threshold, execution time shows an exponential behaviour up to the tolerance of 1 and stays constant after that. The algorithm shows a different behaviour without noise, where constant values for certain tolerance ranges are observed (Fig. 4.12 (a)). In general, the higher the noise, the longer the algorithm needs to achieve a certain tolerance in case the maximum iterations is not the constraining parameter.

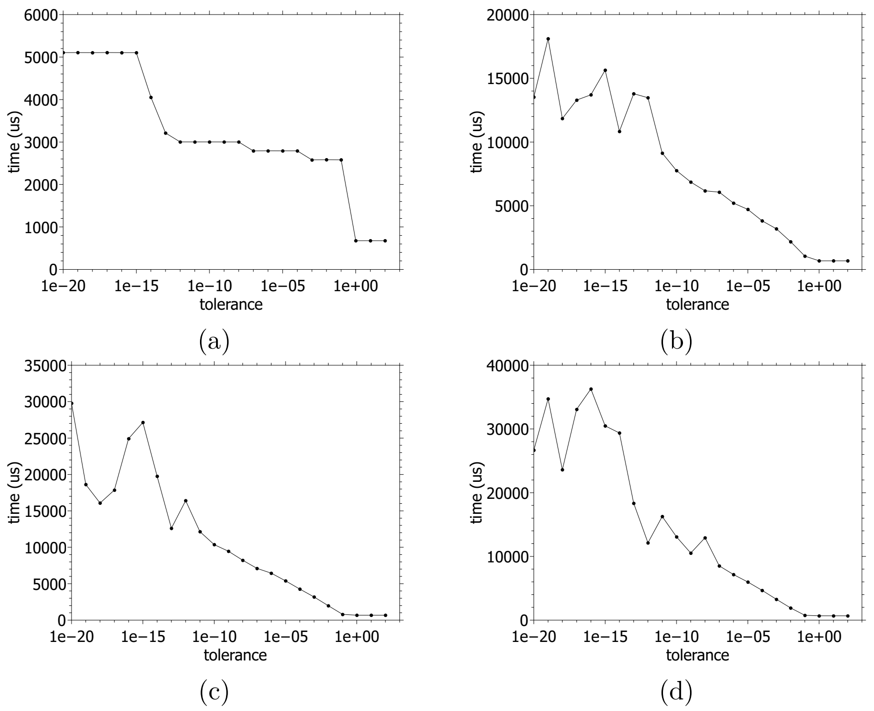


Figure 4.12: Execution time for a noise level of 0 (a), 333 (b), 666 (c) and 1000 (d) CCD counts over the tolerance parameter. (average over 1000 measurements, 2 atoms within 15 pixels)

For a given number of maximum iterations and tolerance set to zero, the execution time does not significantly depend on the noise level. In Fig. 4.13, the execution time for the same setup as above is investigated, now over the maximum iterations parameter. In Fig. 4.13 (a), for zero noise a perfect fit is achieved after 20 iterations. This leads to the determination of the tolerance criterion of the Nonlinear Curve Fit VI and therefore constant execution times. For any other level of noise greater than zero (Fig. 4.13 (b)), a perfect fit can not be achieved. Because the tolerance parameter is set to zero, the maximum number of determined iterations is performed. For few iterations up to about 20, the mean execution time seems to grow linearly with the iteration number. Above, the curves begin to level off. The execution times for various noise levels do not seem to distinguish themselves significantly.

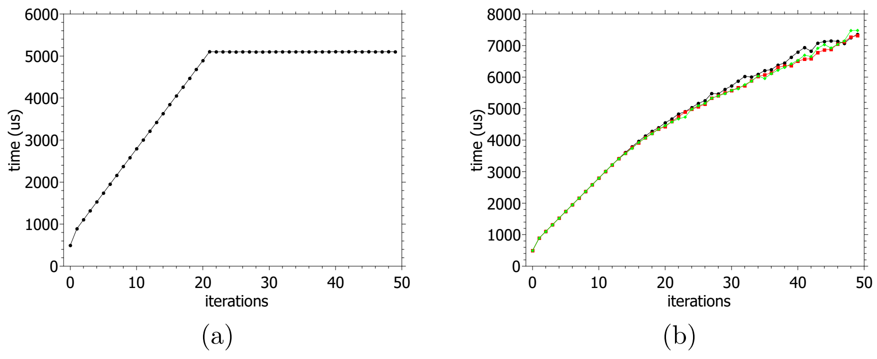


Figure 4.13: Execution time for a noise level of 0 (a), 333 (b,black), 666 (b,red) and 1000 (b,green) plotted against the maximum iteration parameter of the Levenberg-Marquardt VI. For zero noise, a perfect fit is achieved after 20 iterations. (average over 1000 measurements, 2 atoms within 15 pixels, tolerance 0)

In Fig. 4.14, a 3D plot of the mean execution time against the tolerance and the maximum iteration parameter of the Nonlinear Curve Fit VI is shown. As discussed, the execution time increases with the number of iterations influenced by both parameters and shows a fluctuating behaviour when exceeding certain thresholds.

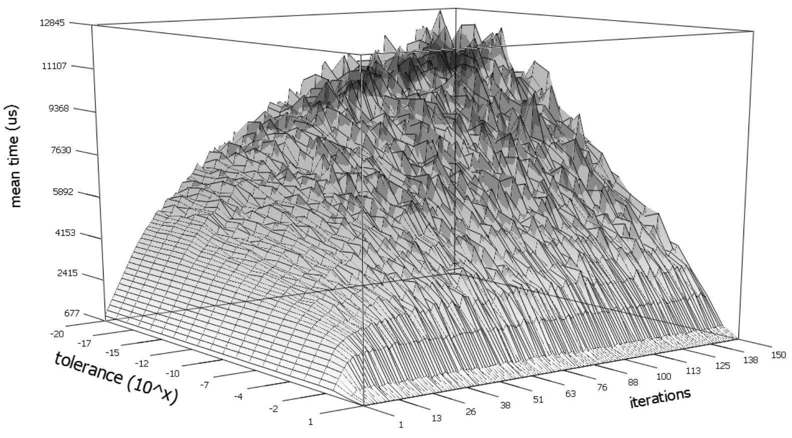


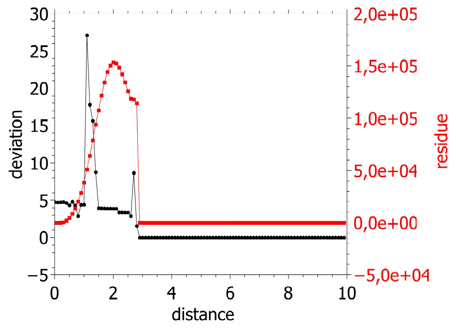
Figure 4.14: Execution time against maximum iterations and tolerance of the Levenberg-Marquardt VI. (average over 200 measurements for each data-point, noise level 300, 2 atoms within 15 pixels)

4.5 Resolution power

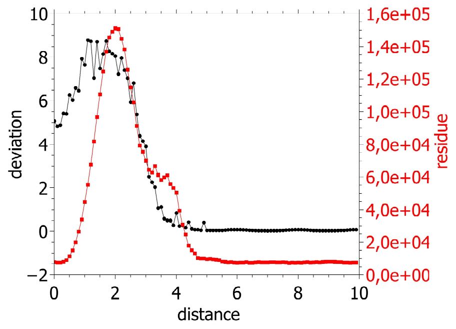
In the following, the ability of the algorithm to distinguish between two neighbouring atom peaks is discussed. Once more the prototypical LSF as shown in Fig. 4.2 (a) is used.

In Fig. 4.15, the position deviation and residue for two atoms with a distance between each other from zero up to ten pixels is shown. For zero noise in Fig. 4.15 (a), there is a clear threshold of about 2.6 pixels when positioning becomes accurate. Below that, very high deviation indicates basic resolution problems. The behaviour of the residue curve can be interpreted in such a way that at zero distance the two atoms can be fitted smoothly as one peak. The greater the distance between the atoms, the higher the residue for a single peak gets. At the maximum deviation the algorithm occasionally switches to two peak fitting, therefore starting to reduce the residue. With zero noise, the algorithm finally switches to sole two peak fitting at a distinct distance, leading to the sharp drop. For a noise level of 100 CCD counts (b), the drop gets more smooth. For even higher noise levels ((c) and (d)), positioning deviation still decreases with growing distance.

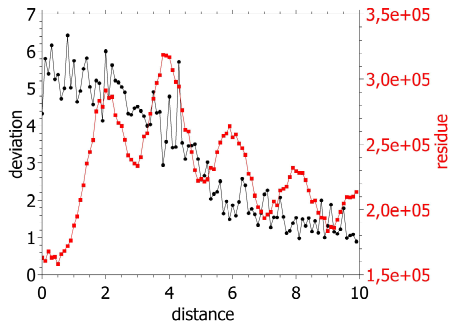
In order to overcome this basic positioning problem for at least two neighbouring atoms, a special mechanism is implemented as proposed by Karski [4]. Instead of using the positions from the trigonometric moment estimation, which become extremely imprecise for close atoms and are the



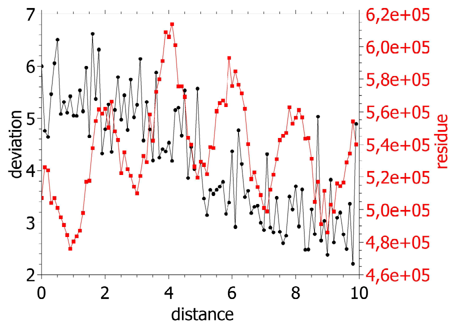
(a)



(b)



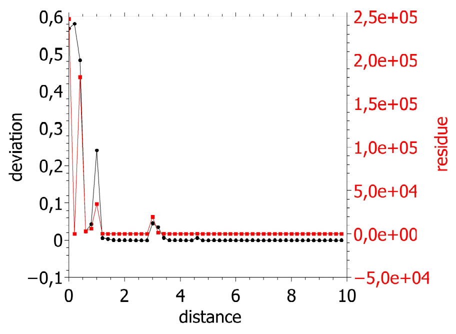
(c)



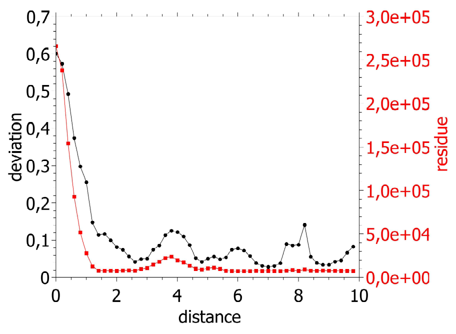
(d)

Figure 4.15: Deviation of positions (black) and residue (red) over distance between two neighbouring atoms with a noise level of 0 (a), 100 (b), 500 (c) and 1000 (d) CCD counts. When the LSF of the single atoms begin to overlap, the deviation increases sharply. (average over 1000 measurements, ROI of 20 pixels)

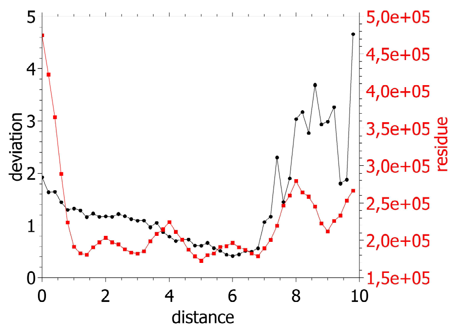
cause of the huge deviation, two positions close to the centre of the ROI are chosen instead. As shown in Fig. 4.16, the effect is enormous. For a moderate noise level of 100 CCD counts, the deviation stays below 0.6 for all distances, compared to a maximum of 9 without the mechanism. Also for higher noise levels, improvements are observed.



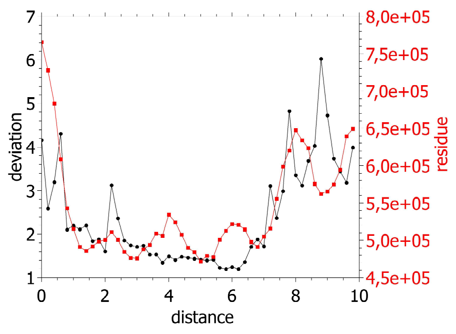
(a)



(b)



(c)



(d)

Figure 4.16: Deviation of positions (black) and residue (red) over distance between two atoms with a noise level of 0 (a), 100 (b), 500 (c) and 1000 (d) CCD counts. By the use of the refined positioning mechanism, the deviation stays moderate even for small distances. (average over 1000 measurements, ROI of 20 pixels)

Chapter 5

Measurement results

In this chapter the first actual measurements performed with the new experimental setup as described in Chap. 2 is presented. Because during the time this thesis has been written the experiment was under construction, only a small data set is available for analysis. Nonetheless a first impression of the performance of the program under real conditions is gained. In the first section, the characteristics of the camera signal is discussed. In the second section, the two important factors of speed and accuracy are examined.

5.1 Signal Characteristics

Fig. 5.1 shows some example measurements taken with an exposure time of 100 ms.

A significant spatial deviation of the fluorescence contributions is observed which indicates either inhomogeneous illumination or a problem with the optics or inhomogeneous sensitivity of the EMCCD chip. The question arises whether this deviation could cause a problem for the algorithm, especially for the calculation of the number of atoms per ROI. In Fig. 5.1 (d), the mean SAFC is 8849 CCD counts with a standard deviation of 1897. The number of atoms per ROI is the accumulated fluorescence contribution divided by the mean SAFC, the result rounded to the next integer. This means that for a single atom peak the SAFC must be smaller than 4425 or bigger than 13274 to deliver a wrong result. Regarding the standard deviation of the SAFC, this seems very unlikely for the present example. However, if the deviation gets to high, the correction mechanism with the *SAFC Scaling* parameter as explained in Chap. 3 can be applied.

The background value in Fig. 5.1 (d) calculated as the mean value of all pixels which are not included within in a ROI is 874 CCD counts with a standard deviation of 66. Compared to the noise levels applied to the simulated data in Chap. 4, this standard deviation indicates rather moderate noise. Therefore, problems with the signal-to-noise ratio at least for this experimental configuration is not expected.

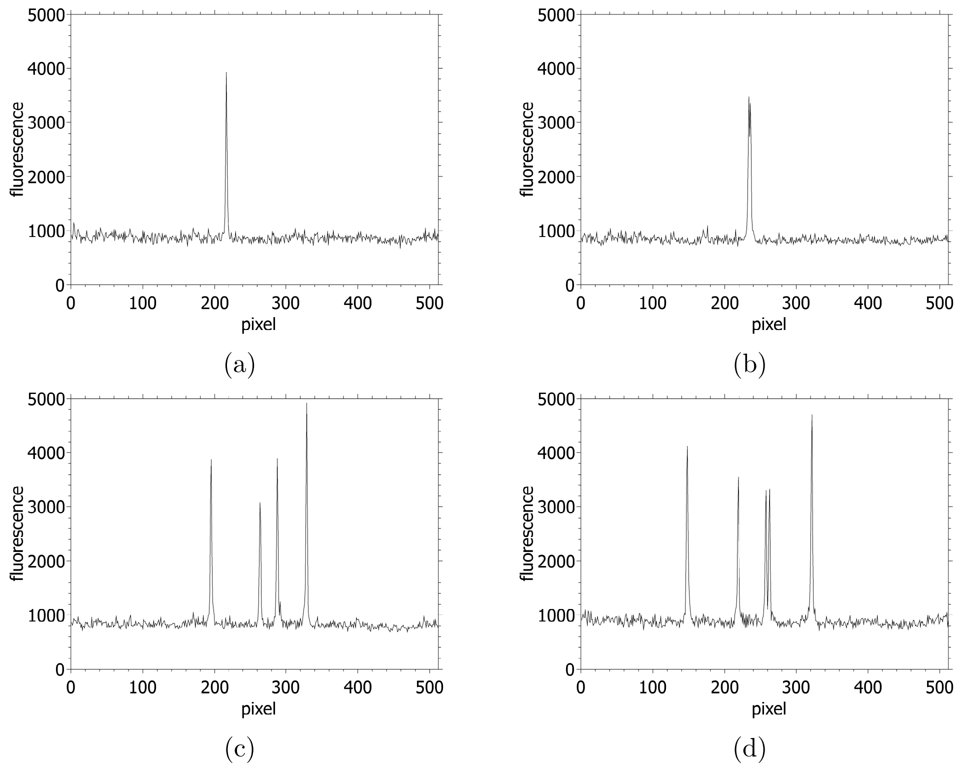


Figure 5.1: Real experiment data with various number of atoms. (a) One single atom, (b) two very close atoms, (c) four well separated atoms, (d) five atoms with two of them within a single ROI.

5.2 Performance

In Fig. 5.2, the RT processing time for various numbers of atoms and the corresponding standard deviation is shown. Each data point is the mean value of 10 measurements, except for 4 atoms where only 2 data sets were available. The form of the curve and the difference in the standard deviation stem from the varying distribution of the atoms. For images with more than one atom, there is always the possibility of multiple atoms in a single ROI. Because the processing of such cases takes longer than the single atom ROIs, a mixture of both types leads to a huge standard deviation. For the data set shown in Fig. 5.2, only one out of ten measurements for the two atoms data point has double atoms ROI. For the three atoms data point, six out of ten cases have double atoms ROIs, which leads to the huge error bar. While the two measurements of the four atoms data point consist of single atom ROIs only, for the five atoms data point always one double atoms ROI is included.

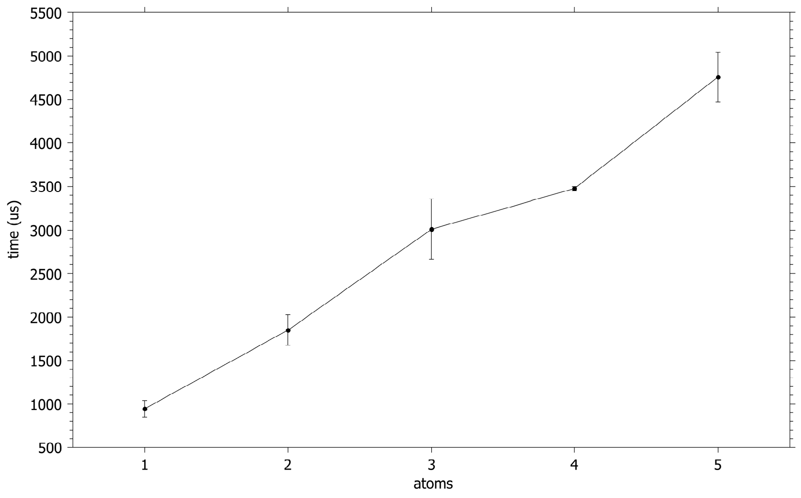


Figure 5.2: Actual execution times for various numbers of atoms. The data points contain a mixture of well separated and close atoms. (average over 10 measurements, for four atoms only two measurements)

5.3 Accuracy

Although it is not possible to compare the positioning results to the 'real' positions of the atoms, at least one can look at the differences between the previously used post-processing and the new realtime program. The mean difference between 38 atom positions in various configurations from single atom ROIs to close neighbouring atoms is 0.0470 pixels with a standard deviation of 0.0259 pixels.

Considering the reproducibility, former measurements have shown that the positions of the atoms in the dipole trap should be quite stable [4]. Only a minor drift due to thermal expansion and some fluctuations are to be expected.

Observing the distance between two atoms over 11 consecutive measurements with a exposure time of 100 ms gives a relative position fluctuation of 0.1075 pixels with a standard deviation of 0.0796 pixels.

Chapter 6

Conclusion

The aim of the project was to implement a positioning algorithm for single atoms in a dipole trap. The focus was to achieve a performance to enable realtime tracking of the atoms so that the position information can be used in a feedback loop. A typical exposure time for an image taken for the positioning is 10 ms which sets the time scale for the execution time. The algorithm was implemented on two different hardware levels.

In a first step, a FPGA handles the read out of the data from the camera and some basic low-level processing. The execution time of this part mainly depends on the image size respectively the number of pixels which are used. As the read out rate of 10 MHz is predetermined by the camera, hardly any performance improvement is possible. For a typical image of 4000 pixels, the read out takes about 450 µs, while the processing is finished after about 50 µs.

The positioning algorithm itself takes place on a RT controller. The execution time of this part mainly depends on the number of atoms and the spatial atom distribution. The determination of the position of a single atom ROI takes about 1 ms. While the execution times for multiple single atom ROIs in one image can simply be accumulated, it increases significantly with multiple atoms in one ROI. Depending on the specific configuration, no more than 3 atoms in one ROI can be determined within 10 ms.

Although only little real measurement data was available, the accuracy compared to the already existing post-processing program seems sufficient. The deviation between the two positioning programs is in the order of some hundredth of a pixel.

References

[1] URL: http://ni.com.

[2] R. Grimm, M. Weidemüller, and Y. B. Ovchinnikov. "Optical dipole traps for neutral atoms". In: *Adv. At. Mol. Opt. Phys.* 42 (2000), pp. 95–170.

[3] Tobias Kampschulte. "Coherently driven three-level atoms in an optical cavity". PhD thesis. Bonn: Rheinische Friedrich-Wilhemls-Universität Bonn, Mathematisch-Naturwissenschaftliche Fakultät, 2011.

[4] Michał Karski. "State-selective transport of single neutral atoms". PhD thesis. Bonn: Rheinische Friedrich-Wilhemls-Universität Bonn, Mathematisch-Naturwissenschaftliche Fakultät, 2010.

[5] K. Levenberg. "A Method for the Solution of Certain Problems in Least Squares". In: *Quart. Appl. Math.* (1944), pp. 164–168.

[6] Lei Li and Terence P. Speed. "Parametric deconvolution of positive spike trains". In: *Ann. Statist.* (2000), pp. 1279–1301.

[7] D. W. Marquardt. "A Method for the Solution of Certain Problems in Least Squares". In: *Journal of the Society for Industrial and Applied Mathematics,* (1963), pp. 431–441.

[8] Lucie Paulet. "Raman and Microwave Manipulation of Small Atomic Ensembles". Master thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, Mathematisch-Naturwissenschaftliche Fakultät, Oct. 2014.

[9] V.F. Pisarenko. "The retrieval of harmonics from a covariance function". In: *Geophys. J. R. Astr. Soc.* (1973), pp. 347–366.

[10] William H. Press et al. *Numerical recipes: the art of scientific computing.* 3rd ed. New York: Cambridge Univ. Pr., 2007.

[11] E. L. Raab et al. "Trapping of neutral sodium atoms with radiation pressure". In: *Phys. Rev. Lett.* 59 (1987), pp. 2631–2634.

[12] Peter Sauer. *Hardware-Design mit FPGA.* Aachen: Elektor Verlag GmbH, 2010.

[13] C.E. Shannon. “Communication in the presence of noise”. In: *Proc. Institute of Radio Engineers* 37 (1 1949). Reprint as classic paper in: Proc. IEEE, vol. 86, no. 2, (Feb. 1998), pp. 10–21.

[14] J. L. Starck, E. Pantin, and F. Murtagh. “Deconvolution in Astronomy: A Review”. In: *Publications of the Astronomical Society of the Pacific* (2002), pp. 1051–1069.

[15] Andor Technology. *iXon Cable Splitter Boxes Design Description (DD 01412/01711).* Version 1.3. 2010.

[16] Andor Technology. *iXon3 Hardware Guide.* Version 1.5. 2011.

[17] Yannik Völzke. “Simultaneous Non-Destructive State Detection of Neutral Atoms”. Master thesis. Rheinische Friedrich-Wilhelms-Universität Bonn, Mathematisch-Naturwissenschaftliche Fakultät, Nov. 2014.

[18] D. J. Wineland and W. M. Itano. “Laser cooling of atoms”. In: *Phys. Rev. A* 20 (1979), pp. 1521–1540.

Acknowledgements

There are a number of people I want to thank for their help with this thesis.

First, Prof. Martin Gröschl of the Sensors and Ultrasonics group of the Institute of Applied Physics at the Vienna University of Technology for offering me the opportunity to conduct the thesis within his team.

Second, Dr. Maria Bernard-Schwarz who was my direct supervisor and gave me her advice and support whenever questions arose and who also proofread my thesis.

Third, the team of National Instruments Munich who supplied the necessary hardware equipment, offered some LabVIEW training courses and always showed a keen interest in my project.

Fourth, the quantum research team at the University of Bonn under the supervision of Dr. Lothar Ratschbacher who granted me online access to their experiment and supplied me with test data.