CARL VON OSSIETZKY universität OLDENBURG

Fakultät II – Informatik, Wirtschafts- und Rechtswissenschaften
Department für Informatik

SEM-based motion control for automated robotic nanohandling

Dissertation zur Erlangung des Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)

von

Dipl.-Inform. Daniel Jasper

Gutachter:

Prof. Dr. Sergej Fatikow
Prof. Dr. Nicolas Chaillet

Tag der Disputation: 6. Juni 2011

Abstract

Nanoobjects such as carbon nanotubes and other nanowires have become of significant interest for both research and industrial applications due to their unique electrical, chemical, optical and mechanical properties. As humans lack both the motor skills and the sensing capabilities on this scale, nanohandling robots are commonly employed in scanning electron microscopes to precisely characterize individual nanoobjects and assemble nanocomponent-based devices. A key challenge for the automation of such processes is controlling a robot's motion with the required accuracy. In this work, a new method for automatically tracking the position of nanoobjects and robots by conducting line scans with the electron microscope is devised that substantially exceeds the capabilities of existing approaches in terms of resolution, update rate, working range and robustness. Based on this method, visual servoing is implemented for a new robotic system making fast movements along well-defined motion trajectories possible. Position measurement as well as visual servoing is implemented on dedicated hardware leading to high performance and accurate timing. The developed system enables visual servoing along jerk-limited trajectories with deviations of less than 20 nm in a few tens of milliseconds. Thereby, this work represents an important step for taking nanohandling from laboratory-sized experiments to high-throughput automation viable for industrial application.

Zusammenfassung

Nanoobjekte wie Kohlenstoffnanoröhren und andere Nanodrähte gewinnen aufgrund ihrer einzigartigen elektrischen, chemischen, optischen und mechanischen Eigenschaften sowohl in der Forschung als auch in industriellen Anwendungen zunehmend an Bedeutung. Da die Nanoskala aufgrund der motorischen und sensorischen Anforderungen für Menschen unzugänglich ist, werden Aufgaben wie die Charakterisierung einzelner Nanoobjekte oder der Aufbau von auf Nanokomponenten basierenden Systemen von speziellen Robotern im Rasterelektronenmikroskop durchgeführt. Eine Kernaufgabe bei der Automatisierung solcher Prozesse ist die geregelte Roboterbewegung mit der benötigten Genauigkeit. In dieser Arbeit wird eine neue Methode für die automatische Verfolgung der Position von Nanoobjekten und Robotern mit Hilfe von Linienscans des Rasterelektronenmikroskops vorgestellt, die die Leistung von bestehenden Methoden in Bezug auf Auflösung, Aktualisierungsrate, Arbeitsraum und Robustheit deutlich übersteigt. Darauf aufbauend wird eine Regelung für ein neuartiges Robotersystem entwickelt, die eine schnelle Bewegung entlang vorgegebener Trajektorien erlaubt. Die Positionsverfolgung und die Regelung werden durch spezielle Hardwaresysteme implementiert, um eine hohe Leistung sowie ein genaues Zeitverhalten zu gewährleisten. Das entwickelte System ermöglicht eine Bewegung entlang einer Trajektorie innerhalb weniger Millisekunden und mit unter 20 nm Abweichung. Somit ist diese Arbeit ein wichtiger Schritt, um bislang nur im Labormaßstab umsetzbare Nanohandhabung industriell nutzbar zu machen.

Acknowledgements

The presented work has been carried out at the Division Microrobotics and Control Engineering (AMiR) of the University of Oldenburg, Germany, headed by Prof. Dr.-Ing. Sergej Fatikow. I would like to express my gratitude to Prof. Fatikow for supervising my Ph.D. thesis, the division's excellent laboratory equipment as well as the confidence he had in my work. I highly appreciate the freedom I had during the time of my doctorate. Furthermore, I would like to thank Prof. Nicolas Chaillet for refereeing this dissertation.

All of my colleagues, I would like to thank for the excellent team work. I cherish the positive and productive atmosphere as well as the fruitful discussions that developed during the daily coffee breaks. I especially thank Claas Diederichs for leading the development of an excellent software infrastructure and developing several hardware components used by the scan generator, Christoph Edeler for the teamwork regarding the design and optimization of the mobile nanorobots, and Volkmar Eichhorn for constantly providing challenging applications and required references.

I would like to thank my parents Annette and Heinrich Jasper for their constant support throughout my life, especially my father for the time and effort he put into proofreading this thesis. Without you, I would not be where I am today. Finally and most importantly, I would like to thank my girlfriend Elisabeth May for her moral support throughout the years, proofreading many of my papers as well as her knowledge on scientific writing, significantly contributing to the success of my work.

Contents

List of Figures

1 Introduction

Nanoobjects such as carbon nanotubes and other nanowires have become of significant interest for both research and industrial applications due to their unique electrical, chemical, optical and mechanical properties. The precise characterization of individual nanoobjects and the assembly of nanocomponent-based devices is a challenging task that cannot be done by batch processing in many cases. Instead, a nanorobotic approach is used (Dong and Nelson, 2007), i.e. nanohandling robots are used to individually characterize nanoobjects, to assemble nanocomponent-based devices and to integrate nanoelectromechanical systems (NEMS) (Fukuda et al., 2009). These robots can move, manipulate and assemble objects with a resolution of single nanometers (Jasper and Edeler, 2008). Similar to the field of microrobotics, the involved physical effects, e.g. the ratio of gravitational and surface forces, are entirely different from the macroscale (Abbott et al., 2007).

For most nanorobotic applications, microscope systems are used to deliver sensor information. Viable systems are scanning electron microscopes (SEMs), optical microscopes, transmission electron microscopes (TEMs) and atomic force microscopes (AFMs). SEMs are commonly favored for robotic manipulations because of their large zoom range, high resolution and acceptable update rate. Optical microscopes can only be used for microobjects and very large nanoobjects because of their limited resolution. TEMs are limited to the imaging of thin objects and provide little space to integrate robotic systems. AFMs have a very long image acquisition time making the supervision of nanorobotic operations impractical. Thus, for this thesis, SEMs are the only system considered for visual feedback.

The precise alignment of several micro- and nanoobjects is a basic task for all nanohandling and nanoassembly operations, e.g. objects and tools need to be brought into a precise relative position. To implement automation, this positioning needs to be done through closed-loop motion control. On the macroscale, the location of the manipulated objects in a world coordinate system is either known from the process design or can be measured by a variety of available sensors. To perform closed-loop positioning, a robot can move to this location with sufficient accuracy using its internal sensors. On the nanoscale, the position of objects has to be derived by processing images of the employed microscope system. Especially for electron microscopes, the result is often time-variant and non-linear, so that the object's location in a world coordinate system cannot be derived precisely.

Furthermore, the positioning itself is a challenging problem due to the size ratio between the robotic system and the handled objects. Commonly, a robot with a size of a few centimeters is used to manipulate an object with a precision of a few nanometers, i.e. the robot is six orders of magnitude larger than the required accuracy. Effects such as thermally-induced drift and limited mechanical stiffness lead to a non-linear and time-variant relation between the position measured by a nanohandling robot's internal sensors and the real position of the robot's tool. Thus, internal sensors are insufficient for reliable nanopositioning.

To circumvent this problem, visual servoing can be used for closed-loop positioning employing image feedback as an element in the low-level control loop (Hutchinson et al., 1996). For nanohandling operations, the SEM is combined with special image processing algorithms to become a nanopositioning sensor (Sievers and Fatikow, 2006). Moving towards higher positioning speeds, however, current implementations of SEM-based visual servoing become a bottleneck due to several reasons. First, SEM image acquisition is slow with update rates below 50 Hz even for well-selected regions of interest and acceptance of high noise levels (Kratochvil et al., 2009). Second, most image processing algorithms are computationally intensive and thus introduce a delay into the control loop. Third, quickly moving objects appear distorted in SEM images and make tracking hard or impossible. These aspects prohibit high-speed positioning and high-throughput automation using visual servoing.

1.1 Goals

The goal of this work is to create a new closed-loop motion control methodology that combines the speed of positioning based on internal sensors with the accuracy of visual servoing. To this end, a new tracking approach needs to be developed that features:

- A high update rate, preferably higher than 100 Hz
- Low computational overhead leading to a low latency
- The ability to track moving and thus distorted objects

As these features cannot be achieved by SEM image acquisition and processing, the new tracking approach needs to control the electron beam on a lower level in order to capture information relevant for tracking more effectively. Additionally, it needs to be robust so that changing imaging conditions such as brightness and contrast do not impede the tracking. Combined with focus analysis, the approach should enable three-dimensional tracking.

Based on the new tracking approach, vision-based motion control usable for automated nanohandling operations is a second goal of this work. With this motion control, the accurate positioning of robots should be possible with a speed of several operations per second. This is an important step on the way to achieve high-throughput nanohandling suitable for industrial applications.

1.2 Outline / Author's contribution

The author contributes to the advancement of the closed-loop motion control for automated nanohandling and nanoassembly in six chapters following this introduction.

Chapter 2 describes the current state of the art in the area of automated motion control on the nanoscale. For such a motion control, both an actuation and a sensor system are required. Available actuation principles as well as robot types are analyzed and their suitability to high-speed motion control on the nanoscale, i.e. positioning in a few tens of milliseconds with a precision of about 10 nm, is evaluated. Positioning based on the internal sensors of actuators is described and compared to existing visual servoing approaches. Based on the available technologies and their shortcomings, precise requirements for the new positioning system developed in this work are derived.

A key technology for the new positioning system, the high-speed position tracking with the scanning electron microscope, is then analyzed in depth in **Chapter 3**. After formally deriving an inherent bottleneck of image-based tracking, a new approach relying solely on SEM line scans is described. Different methods for evaluating the data gathered during the line scan are introduced and analyzed based on theoretic considerations as well as simulations. The implementation of three-dimensional tracking by analyzing the focus of the electron beam is described.

Chapter 4 introduces a novel nanorobotic system that is designed based on conclusions from Chapter 2. This system combines multiple degrees of freedom in a compact design with other beneficial properties such as a long working range, high resolution and effortless integration into microscopes. An open-loop movement model of the robot is derived, inverted and validated to facilitate open-loop positioning. Based on the effective open-loop control in conjunction with the new line scan-based position tracking, the implementation of high-speed visual servoing is described. With this control, the robots are able to complete a highly accurate positioning in a few tens of milliseconds and can precisely follow predefined trajectories.

Validation results and other measurements are given in **Chapter 5** in order to prove the validity of the presented approaches. The robustness of the tracking against both changing SEM parameters and motions of the robot is verified. Experimental closed-loop motions show the speed and precision with which the robot can be positioned within a large working range.

Chapter 6 shows how the two key technologies developed in this thesis, fast position tracking and motion control of a novel nanorobotic system, can be integrated to implement a fully automated handling scenario.

The final **Chapter 7** contains conclusions and an evaluation of this work's impact on the speed, throughput and industrial applicability of automated nanohandling and nanoassembly. An outlook describes future steps extending this work.

2 Motion control in SEMs

The scanning electron microscope (SEM) is commonly used to support robotic nanohandling and nanoassembly. This section presents the state of the art regarding motion control of robots operating in SEMs. An analysis of the capabilities of SEMs shows, why it is a versatile tool for nanohandling as well as its limitations when acting as a visual position sensor. Different mechanical nanorobot designs are given, highlighting the advantages and disadvantages of certain motion principles. The limited applicability of internal sensors to nanoscale motion control is shown as well as the limited performance of current visual servoing implementations based on tracking in SEM images. The required precision as well as the employed actuators and sensors pose challenges for motion control that are not present in macroscale systems. Several concepts such as latency compensation and trajectory control, however, can be applied to the nanoscale. Implemented automation scenarios show the need for more effective motion control.

2.1 Definitions

Throughout this thesis, the term **nanorobot** will be used for robots that can operate with a resolution of 10 nm or better. The physical size of the described robots is in the cm-range. Nanorobots can be used to manipulate **nanoobjects**, i.e. objects with at least one dimension in the nm-range. This dimension is called critical dimension.

There are several definitions of the term **real-time** in different fields of research. For example, an image processing algorithm works in real time, if it is capable of operating with the same update rate as the image sensor. For this work, the term is used in a strict sense defined for embedded systems, i.e. only for processes with reliable and predictable timing. Thus, a real-time capable sensor delivers its data with a known update rate, a limited latency and a well-defined jitter.

A **pose** describes the combined position and orientation of an object. A **trajectory** defines a movement of a robot or tool along a predefined path with a predefined timing. Thus, with trajectory control, the exact pose of a robot at any given time is controlled.

2.2 Scanning electron microscopes

The scanning electron microscope generates a highly magnified image of a sample by scanning with an electron beam. Here, the key principles of scanning electron microscopy and the resulting properties are introduced with a level of detail necessary for the following considerations. A more detailed description of scanning electron microscopes can be found in Reimer (1998).

The most important consideration is that physically an SEM does not generate an image of the observed specimen, although the SEM's output can be interpreted similar to camera images. In contrast to cameras, that measure the light intensity simultaneously for many pixels, the SEM uses a focused electron beam and can only measure the electron beam interaction at a single point on the specimen at any time. Images are generated by scanning the electron beam line by line over a specific area. Thus, the SEM uses a single sensor to capture all pixels sequentially whereas a camera uses many sensors and captures pixels simultaneously.

A benefit over cameras is that the pixel distance can be chosen arbitrarily enabling the SEM to deliver images in a large zoom range as shown in Fig. 2.1. The physical size of each pixel is defined by the so-called interaction volume, i.e. a certain volume of the specimen that reacts measurably to the electron beam at a specific location. The interaction volume remains constant independent of the chosen zoom. Therefore, for low zoom settings, the electron beam is moved quickly over a specimen and pixels are recorded only at certain locations. Thus, features on the specimen might be skipped and invisible at low magnifications (see Fig. 2.1a). At medium magnifications (Fig. 2.1b), all but the smallest features of the specimen are visible and the image still appears sharp because the pixel size is smaller than the interaction volume. At very high magnifications, the electron beam moves

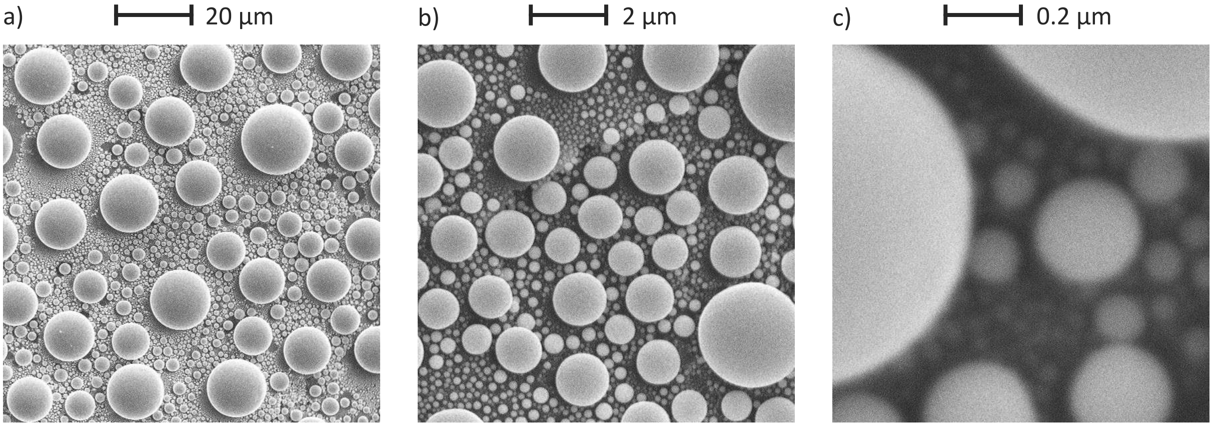


Figure 2.1: Images of an SEM calibration specimen acquired at three different magnifications: a) 1000×, b) 10000× and c) 100000×.

only slightly between pixels, so that an overlap between the interaction volumes occurs. As a result, the image appears smooth although it is perfectly in focus (see Fig. 2.1c). The resolution limit of the SEM is exceeded.

An SEM consist of an electron column and different electron detectors. The basic design of an electron column is shown in Fig. 2.2. The electron gun is responsible for the generation of an electron beam. There are two basic types of electron guns: Thermionic guns and field-emission guns. The exact physical details of beam generation are outside the scope of this thesis. The important difference between the two gun types is that the achievable resolution is about 10 nm for thermionic guns and 1 nm for field-emission guns.

The generated electron beam passes through a series of condenser lenses and apertures with the aim of creating a focus point as small as possible. These are simplified and represented by a single condenser lens and aperture in Fig. 2.2. The condenser lenses are used to concentrate all of the electrons in the center of the electron beam. The apertures intercept scattered electrons that should not reach

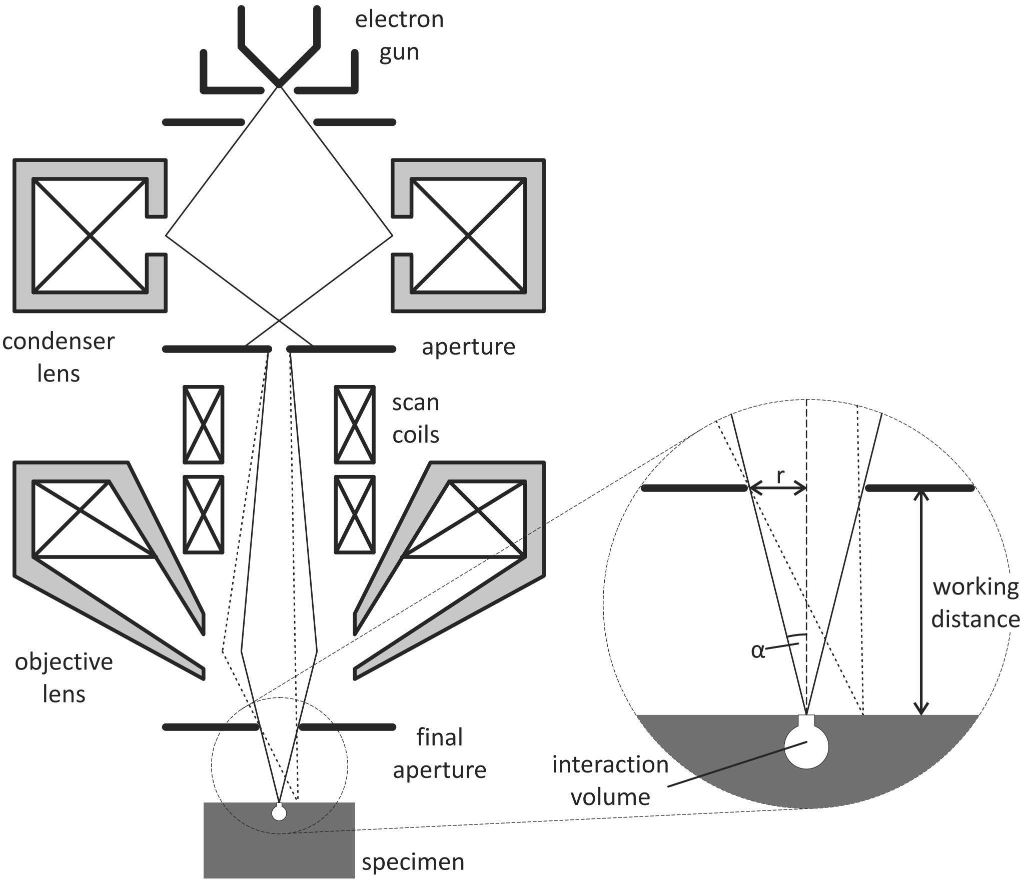


Figure 2.2: Basic principle of a scanning electron microscope.

the specimen as they would widen the focus point. Which each pair of condenser lens and aperture, the achievable focus point becomes smaller.

A pair of scan coils is used to position the electron beam on the specimen. These scan coils tilt the electron beam before it passes the final lens, called objective lens. The objective lens is actually a combination of an electromagnetic and an electrostatic lens and is the final focus element in the electron column. The shown final aperture has the task of improving the beam shape and edge sharpness. It is also known as beam-defining aperture or diaphragm. If it is located at the bottom of the column, this aperture is called "real aperture". In modern SEMs, this aperture is actually located somewhere in the middle of the column and called "virtual aperture". A virtual aperture has the same effect as a real aperture, but has the added benefit that it is located further away from the dirty specimen area. Thus, a virtual aperture requires less maintenance. The final aperture has an influence on the size of the focus point and on the depth of focus (see Section 2.2.2).

The two parameters specimen current and scan speed have a significant effect on the image quality and are thus important for SEM-based motion control. The specimen current, also known as spot size, is a measure for the intensity of the electron beam. A higher specimen current means that more electrons hit the specimen at the same time leading to an increased emission of secondary and backscattered electrons (see Section 2.2.3). As the noise of the corresponding detector remains constant, the signal-to-noise ratio increases. However, an increased specimen current leads to a larger interaction volume and thus a reduction of the achievable resolution. The scan speed describes the speed with which the electron beam is scanned. With a lower scan speed, the detector signal is averaged over a longer period of time for each pixel. Similar to a higher specimen current, this reduces pixel noise.

2.2.1 Distortions in SEMs

The image generated by a scanning electron microscope is distorted by several influences and cannot be assumed to be an ideal projection of the specimen area. These distortions include perspective distortions similar to other imaging systems, environmental distortions caused by the monitored environment itself and distortions created by the dynamic behavior of the electron beam.

Perspective distortions occur because of the mapping of a three-dimensional scene to the image plane. It is commonly assumed that for higher magnification, i.e. 1000× and above, these distortions become marginal and a parallel projection can be assumed. At lower magnifications, the perspective projection should be taken into account, i.e. the size of objects decreases with the distance from the electron column.

Another form of distortions are **environmental** distortions. In a robotic setup, it is infeasible to properly shield all electric and magnetic fields. Furthermore, for handled objects, especially for nanoobjects, a proper grounding cannot be guaranteed throughout the handling process. Thus, these objects get charged by the electron beam introducing an electric field. Environmental distortions can cause a linear image shift, non-linear distortions as well as a change in brightness and contrast.

Dynamic beam distortions are caused by the dynamic properties of the electron beam. The electron beam is deflected by magnetic coils with stored energy in form of a magnetic flux. If the electron beam position is changed, this energy needs to be increased or decreased requiring a certain amount of time. Furthermore, the coils are supplied by amplifiers that are built with the goal of minimal noise and not necessarily fast response time. Thus, the electron beam deflection system acts as a low-pass filter on the set point value of the beam position. For imaging, this is usually counteracted by giving the electron beam a certain amount of time to settle left to the start of each line. Then, the beam is scanned over the observed scene, continuing slightly past its end. The pixels of the line are thus captured during a virtually linear movement of the electron beam. Remaining nonlinearities create low-frequency distortions in the SEM image.

In addition to low-frequency distortions, which are likely to be caused by the effects described above, Cornille et al. (2003) have also found **high-frequency** distortions. They attribute these distortions to the scan control system, i.e. the scan generator. A spatial distortion of up to 0.5 pixels was found which can represent a significant distance dependent on the magnification. Validation measurements will have to show whether similar non-linearities are present in the employed SEM and affect closed-loop positioning.

Another problem that acts as a distortion of the SEM's coordinate system is **drift**. Drift is a slow and continuous unintended movement of objects in a coordinate system. There are a variety of reasons for such a drift. The most important are **magnetization** of the magnetic coils in the electron column, **thermally-induced** drift and other environmental influences. Sutton et al. (2006) measured this drift to be as high as 20 pixels per hour even in a well-controlled environment.

In short, an SEM is a valuable tool to obtain highly magnified images, but is not designed as a nanoscale measuring device. Images are influenced by a variety of distortions and the accuracy of positions and distances measured in SEM images is limited.

2.2.2 Depth of focus

The final aperture (see Section 2.2) has a significant impact on the depth of focus of an SEM. Based on its radius r, which is 15-100 µm dependent on the SEM, and the working distance f, which is usually 5-20 mm, the aperture angle α (see Fig. 2.2) can be calculated:

$$\alpha = \tan^{-1}\frac{r}{f} \approx \frac{r}{f}. \tag{2.1}$$

The approximation is valid, as α is <100 mrad for virtually all SEMs. Dependent on α and the resolution δ, the depth of focus d_f of an SEM can be calculated by (Reimer, 1998):

$$d_f = \frac{\delta}{\alpha}. \tag{2.2}$$

At high magnifications, the resolution δ is limited by the size of the interaction volume. Thus, for an SEM with $\alpha \approx 10\,\text{mrad}$, the depth of focus is $d_f \approx 100 \cdot \delta$. If such an SEM features a 20 nm resolution, the depth of focus is at least 2 µm. This is beneficial for a lot of handling and tracking applications, as multiple objects and tools can be tracked even if they are not exactly in the focus plane. However, it is a disadvantage if the focus needs to be used for depth estimation as commonly done in optical microscopes.

At lower magnifications, the pixel size becomes the lower limit for the resolution. If a 50×50 μm^2 area is imaged with 500×500 pixels, the resolution is limited to:

$$\delta = \frac{50\,\mu\text{m}}{500} = 100\,\text{nm}. \tag{2.3}$$

With $\alpha = 10\,\text{mrad}$, this leads to $d_f = 10\,\mu\text{m}$. An even larger d_f can be achieved by increasing the working distance or using a smaller aperture resulting in a smaller α. This is desirable for overview and initialization scans (see Section 3.6), which are most useful if as many objects as possible are in focus.

2.2.3 Detector types

The interaction between the electron beam and a specimen can be measured using different detectors. The most commonly used detectors are the secondary electron (SE) detector and the backscattered electron (BSE) detector. Both of these detectors can be used for the position tracking described in this thesis and have different characteristics.

The secondary electron detector detects low-energy (<50 eV) electrons that are generated by inelastic scattering when the electron beam hits a specimen. Secondary electrons are created within a certain interaction volume around the point hit by the electron beam. However, due to the limited energy, only secondary electrons generated within a certain distance to the surface can escape the material. Edges and other topographic features lead to a high secondary electron count, as more of the interaction volume is located sufficiently close to a surface. Thus, using the secondary electron detector, features exhibiting significant topographic differences create a high contrast.

The backscattered electron detector detects high-energy electrons that are reflected out of the specimen be elastic scattering. The amount of reflected, or backscattered, electrons mostly depends on the chemical composition of the specimen. Thus, features consisting of different materials lead to a high contrast.

Details on the advantages of each detector in terms of tracking as well as sample images will be given in Section 3.3.1.

2.3 Nanorobots

Nanorobots are designed to perform repeated, controlled movements with an accuracy of nanometers. Due to inherent problems such as backlash, conventional actuation principles cannot simply be scaled down in order to fulfill this requirement. Furthermore, nanorobots commonly need to operate in special environments such as the vacuum chamber of an electron microscope. There, the actuator must not interfere with the microscope's image generation, i.e. all electric and magnetic fields need to be shielded, and the materials and lubricants that can be used are limited due to the necessary vacuum. To fulfill these requirements, novel actuation principles and technologies have been developed.

2.3.1 Actuators for nanorobots

Although there are efforts to scale conventional actuators such as electric motors to enable nano-precision, e.g. backlash-free gears (Degen and Slatter, 2002), novel actuation principles are better suited for this task. An actuator exploits a **physical effect** in order to create motion. Actuators suitable to create nanoscale movements are piezoelectric, electrostatic, thermal and magnetostrictive actuators as well as shape memory alloys.

Piezoelectric actuators are based on the inverse piezoelectric effect, i.e. the controlled deformation of a piezoelectric element using an electric field. Piezoelectric actuators are popular owing to their large forces, high precision and fast response time. Such actuators can create forces in the kN-range and respond to an input signal within microseconds. Major downsides are the limited deformation of about 0.1% of the actuator's length as well as a strongly non-linear behavior exhibiting hysteresis and creep (Croft et al., 2001).

Electrostatic actuators (Tang et al., 1990) use the electrostatic force generated between surfaces of different potentials. Such actuators are often implemented as comb actuators and fabricated using surface micromachining (Bustillo et al., 1998). They are well-suited for miniaturization for two reasons. First, electrostatic forces are a surface effect and do not decrease as fast as forces generated by volume effects with decreased actuator size. Second, electrostatic forces increase quadratically with the decrease of the gap size between the electrodes. A downside for applications in the SEM is that electrostatic actuators combine soft micromachined springs, comparatively large masses with virtually no damping in vacuum environments (Lee et al., 2000). Thus, care must be taken to avoid vibrations.

Thermal actuators (Comtois and Bright, 1997) use the thermal expansion of materials in order to create motion. Common actuator designs are bimorph and ripcage actuators Carlson et al. (2007). Thermal actuators are especially suited for miniaturization, as the thermal mass is reduced with the volume of the material and thus high actuation speeds and frequencies can be used. A common challenge for thermal actuators is drift. If the heat cannot be sufficiently restricted to the actuated element, it spreads to the entire robotic system and leads to thermal expansion.

Shape memory alloys are special alloys that exhibit different behavior dependent on their temperature. In the low temperature state, the so-called martensitic phase, a shape memory alloy can be easily deformed. When it is subsequently heated, in enters the so-called austenitic phase, in which the initial shape is recovered. Using a preload to deform the alloy when entering the martensitic phase, a two-way actuator can be designed. Due to their high power to weight ratio, the large achievable deformations and the low driving voltages, shape memory alloys are well-suited for micro- and nanoactuation (Ikuta, 1990).

Magnetostrictive actuators are similar to piezoelectric actuators but a magnetic field is used instead of an electric field in order to generate motion. Commonly, the material Terfenol-D is used due to its strong magnetostrictive properties (Moffett et al., 1991). The major downside compared to piezoelectric actuators is that a magnetic field is much harder to shield. Thus, in a scanning electron microscope, image distortions are hard to avoid.

Most of the actuators currently employed for nanohandling as well as the nanorobot developed in this thesis are based on the piezoelectric effect. Thus, only piezoelectric actuators will be considered in the following section.

2.3.2 Motion principles

Piezoelectric actuators can be categorized into scanning and step-wise actuators. **Scanning actuators** use a backlash-free kinematic structure based on flexure hinges (Xu and King, 1996) to amplify the displacements of piezo actuators. Although commercial actuators can reach working ranges of up to 500 µm, the working range is a severe limitation and a second coarse positioning actuator is required. Furthermore, the electric field used for actuation cannot be generated without noise. Scanning actuators translate the signal-to-noise ratio into a ratio between working range and position noise. This noise limits the actuators' resolution and there is a trade-off between working range and resolution. This downside is overcome by **step-wise actuators**. Such actuators do not amplify the piezo element's motion using a mechanical structure. Instead, they use the piezo element's motion to perform small steps. Step-wise actuators have a virtually unlimited working range. Most step-wise actuators are hybrid actuators, so that the piezo element can also be used in a scanning mode to create movements smaller than single steps. Thus, they combine coarse and fine positioning in a single actuator.

Scanning actuators have the advantage of generating smooth motion over the entire working range. Furthermore, due to the high force generated by a piezoelectric actuator, high accelerations and speeds can be obtained. Another advantage is that a rough knowledge of the actuator's position can be directly derived from the applied voltage. Homing is not required for scanning actuators and even open-loop movements can be done with a certain reliability. A downside is, that a scanning actuator always returns to its default position on power loss. Thus, microrobotic systems have to be designed in a way that this movement cannot result in a severe damage to the employed robots or tools. Furthermore, the non-linear characteristics including hysteresis and creep are proportional to the full working range. Thus, these effects have a significant influence and make sensor feedback necessary.

In contrast, step-wise actuators have opposed properties. Their working range is only restricted by the mechanical guides used to generate motion in a specific direction, usually in the cm-range. They require a position sensing because the executed steps are not sufficiently repeatable. Furthermore, dependent on the sensor, homing is required to find the initial location of an actuator. In addition to the large working ranges, major advantages of step-wise actuators are that they do not perform a significant movement if the power is lost and that hysteresis and creep are proportional to the step length. The disadvantage of step-wise actuators is that each step introduces vibrations into the system.

The most commonly applied step-wise motion principle based on piezoelectric actuators is the stick-slip actuation principle (Breguet et al., 1996). This principle makes use of the high movement speed and acceleration of a piezoelectric actuator.

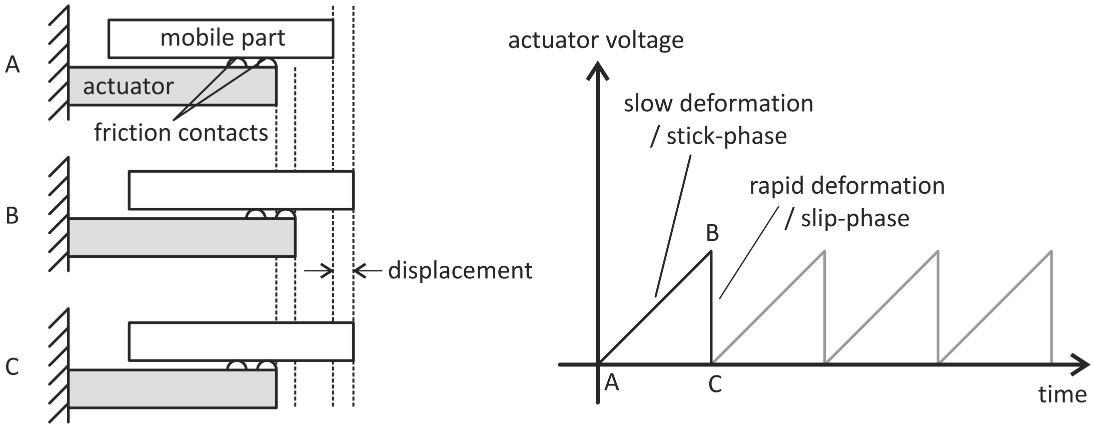


Figure 2.3: Stick-slip actuation principle (Bergander and Breguet, 2003).

A mobile part is connected to a fixed piezo actuator mechanically using a contact with a specific friction (see Fig. 2.3). If the piezoelectric actuator is deformed slowly the friction is sufficient and the mobile part follows this motion. As the mobile part sticks to the actuator, the slow deformation is called stick-phase. However, if the actuator is deformed rapidly, the inertia of the mobile part hinders the motion and the friction force is overcome. Thus, the mobile part slips and this phase is called slip-phase. A net displacement between actuator and mobile part is generated. Alternating slow and rapid deformations into opposite directions, a step-wise motion over long travel ranges is generated.

2.3.3 Types of robots

Robots are used to position objects such as tools in multiple degrees of freedom. There are two basic approaches to build robots with multiple degrees of freedom: Serial and parallel robots. Serial robots combine multiple actuators with one degree of freedom each in a serial kinematic chain. Thus, only the first actuator is fixed whereas all the other actuators move with the first. Parallel robots use multiple fixed actuators that are all directly or indirectly attached to a single moving part. Thus, all the actuators in parallel influence the moving part's position.

A common serial robot type is the Cartesian robot. A Cartesian robot uses multiple linear actuators mounted orthogonally in order to create two- or three-dimensional motion. Depending on the employed actuators, such robots can be used for micromanipulation. Fig. 2.4 shows a robot comprised of three orthogonal linear axes. The actuators and the robot were developed by SmarAct GmbH, Germany, and can be considered among the most advanced Cartesian systems available today. The actuators use stacked piezo elements to create motion and preloaded roller slides to create a virtually one-dimensional, backlash-free movement. For each degree of

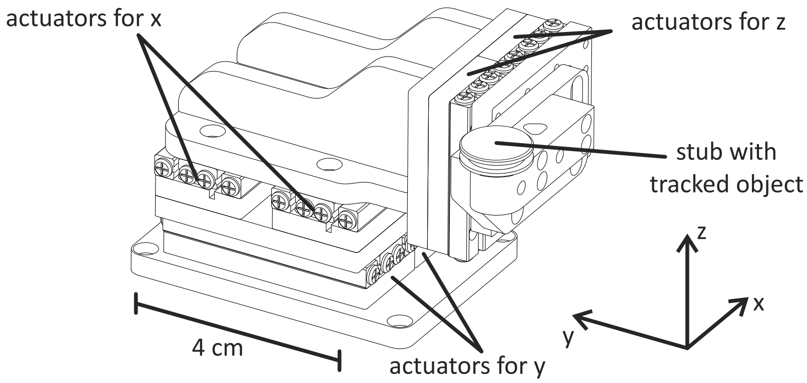


Figure 2.4: Cartesian nanorobot using three pairs of linear axes.

freedom, two actuators are used in parallel to increase the effective span. Internal optical encoders can determine the position with a resolution of 5 nm.

Another serial robot architecture is the so-called Selective Compliant Assembly Robot Arm (SCARA). A SCARA robot uses two articulated arms in order to create motion along the x- and y-axes whereas the z-axis is fixed. Using backlash-free drive principles and miniaturization, such a robot can be utilized to perform micromanipulation (Burisch et al., 2007). However, the accuracy on the nanoscale is limited because of the required rotational joints. Compared to Cartesian robots, SCARA robots offer faster motion and less space requirements.

Steward platforms (Dasgupta and Mruthyunjaya, 2000) are a class of parallel robots that can generate movement in six degrees of freedom. Delta robots (Hesselbach et al., 2004) are a simplification of Steward platforms with only the three translational degrees of freedom. Using flexure hinges as joints, such robots can avoid the typical backlash and friction effects and create motion with nanometer resolution. Disadvantages are the tight coupling of the individual degrees of freedom and the difficult integration of sensors that measure the tool center point with nanometric resolution.

A common downside of parallel robots is the requirement for joints capable of bending in multiple degrees of freedom. Mobile robots form another approach to combine multiple degrees of freedom in a parallel fashion. Such robots either use legs (Martel and Hunter, 2002; Driesen et al., 2005; Murthy et al., 2008) or actuated spheres (Kortschack and Fatikow, 2004) to move on a flat working surface. Thereby, they can move in three degrees of freedom – x, y and rotation. No hinges other than the friction contacts with the surface are required and the robots can be designed to be highly rigid. Furthermore, mobile robots integrate the three degrees of freedom in a comparatively small robot of a few cm^3. Another major advantage of mobile robots is the virtually effortless integration into different microscopes or robotic systems. They can be placed on a flat surface such as a microscope slide without the need for mechanical fixation. The major downside of mobile robots is that, up to now, no sensor is available that can be integrated into the robot and measure its pose with a resolution suitable for nanohandling. However, this downside is alleviated by the fact that positioning based on internal sensors has limited applicability to the nanoscale, as presented in the following section.

2.3.4 Internal position sensors

On the macro scale, the motion control of robots is commonly implemented using internal position sensors. Typically, each linear or rotary axis has a dedicated encoder that can determine the position of the axis with sufficient accuracy. For nanoscale applications, the accuracy achievable by this approach is insufficient for several reasons.

A major challenge for actuators is the limited precision of mechanical guides. Fig. 2.5 shows a measurement of the repeat accuracy of positioning an object with the Cartesian robot depicted in Fig. 2.4 using the stick-slip actuators for the x- and y-axes. A fixed position was approached ten times from ten different positions on a circle with a 1.5 mm radius. An object on the actuator was tracked and the final position in the SEM image is shown in Fig. 2.5a. Although the internal sensor reported an accuracy higher than 5 nm for each positioning, the position of the tracked object deviated by more than 500 nm. For the x-position, the periodical behavior suggests that the movement's direction has an impact on the final position.

A second challenge is the size ratio of several orders of magnitude between robot and required precision that also leads to a mismatch between internal sensor and tool position. Commonly, the employed robots have dimensions of at least a few cm^3 but need nanometer accuracy. Thus, the limited mechanical stiffness can change the

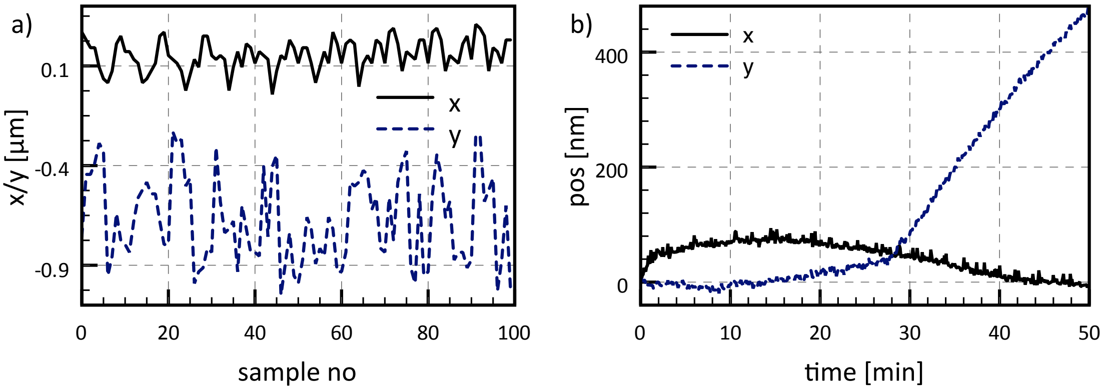


Figure 2.5: a) Repeat accuracy of current linear stick-slip nanopositioners. b) Drift measurement over a 50 min period.

tool's position if the mechanical load changes. Furthermore, thermal expansion of the actuators and mounts leads to drift. A robot of several centimeters can expand by several nanometers even for minute temperature changes. Fig. 2.5b shows the results of a drift measurement. The position tracked in the SEM's image is shown over a 50 min time period. The drift leads to a displacement of almost 500 nm. Furthermore, the drift rate is time-variant with a significantly accelerated x-drift after 28 min. A likely cause is a change in the room temperature, e.g. because a door was opened.

Even if an ideal actuator with accurate internal sensors could be constructed, i.e. the internal sensor's position would exactly correspond to the mounted tool's position in a world coordinate system, motion control using only the internal sensors would still be insufficient due to the distortions of the SEM image itself (see Section 2.2.1). The location of a nanoobject found in the SEM image cannot be reliably converted to a precise location in the world coordinate system or the coordinate system of a robot's internal sensors. To manipulate the object, a tool has to be aligned to the object, i.e. brought to a specified relative location in the SEM image. Depending on the required accuracy, this constraint can only be met by visual servoing.

2.4 Visual servoing in SEMs

To overcome the limitations of positioning based on internal sensors described in the previous section, visual information can be used to increase the control accuracy. Using vision feedback as a sensor for robotic manipulation is promising as it is a non-contact measurement and mimics human behavior.

The initial technique, first described by Shirai and Inoue (1973), extracts a position deviation from an image and then moves the robot either open-loop or based on internal sensors to decrease this deviation. When the robot stops, another image is taken, again extracting and compensating for the position deviation. This is called the look-then-move approach. Two advantages of the approach are that low throughput rates are sufficient for the image processing and that the entire setup can be handled in a quasi-static fashion neglecting dynamic aspects. The effectiveness, however, entirely depends on the speed of the visual sensor and the accuracy of the robot's open-loop control. A slow sensor leads to a long time between movements and an inaccurate open-loop control can fail to reduce the position deviation.

An alternative is to use visual feedback as a low-level element in the motion control loop of a robot. This is called visual servoing (Hutchinson et al., 1996). Two challenges need to be solved to achieve effective visual servoing. First, the image processing needs to be at least as fast as the image acquisition. As the available processing power has significantly increased over the recent years, several image processing algorithms can be used for visual servoing. More complex algorithms, however, still require computation times on the order of seconds and do not allow for effective servoing. Second, the dynamic aspects of visual servoing need to be considered (Corke and Good, 1996). The robot's movement as well as resulting vibrations lead to movement during image acquisition. For camera-based systems, this movement can result in blurred as well as distorted images dependent on the shutter type. For SEM-based systems, images of moving objects are sharp but distorted. An in depth analysis is done in Section 3.1.2.

Nanorobotic systems with feedback from SEMs use a variety of tracking algorithms adopted to SEM images. As non of these tracking algorithms is capable of providing the required speed or of handling image distortions, nanorobotic systems have so far been treated in a quasi-static fashion. Vision feedback was either used in a look-then-move approach or slow visual servoing control loops.

2.4.1 Tracking with the SEM

Over the recent years, a variety of algorithms for the recognition and tracking of objects in SEM images have been developed. These algorithms have to take the special characteristics of SEM images into account. The high level of additive noise is the most prominent difference to camera-based visual feedback (Sievers and Fatikow, 2006). Furthermore, SEM images are affected by drift and non-linearities (Sutton et al., 2006) and, due to the line-wise image acquisition, moving objects appear distorted. For these reasons, only a subset of the available tracking algorithms can be reasonably applied to SEM images. As described by Kratochvil et al. (2009), existing image-based tracking algorithms can be divided into feature-based and model-based approaches. Hybrid approaches try to incorporate aspects of both techniques in order to combine their benefits and to overcome limitations.

Feature-based tracking approaches

Feature-based tracking approaches track specific features visible in images. Shi and Tomasi (1994) state that feature-based tracking is mostly a solved problem and refer to normalized cross correlation-based tracking and tracking based on the sum of squared differences. SEM imaging however shows special characteristics so that the tracking approaches and considerations cannot be applied directly. Reasons are the high level of noise in fast SEM image acquisition, suddenly occurring brightness and contrast changes (Sievers and Fatikow, 2006) and the distorted appearance of moving objects. Nevertheless, two feature-based tracking algorithms have been successfully applied to SEM tracking: Cross correlation and active contours.

Cross correlation itself is not a tracking technique but a pattern recognition algorithm. However, if the same pattern is detected within a specific region of consecutive images, the approach can be used for tracking. Owing to its robustness against additive noise, cross correlation can be applied even to SEM images acquired at a high scan speed (Sievers and Fatikow, 2006). Furthermore, cross correlation requires little configuration and can track a vast variety of different objects, as long as the objects contain sufficient detail and can be unambiguously matched to a specific location. The downsides of the cross correlation approach are a high computational complexity as well as no invariance to object shape, rotation and magnification. This is especially problematic as moving objects in an SEM appear distorted (see Section 3.1.2). Furthermore, common techniques for decreasing the computational effort such as sub-sampling or multi-resolution matching (Rosenfeld, 1977; Gharavi-Alkhansari, 2002) cannot be used due to the high noise level in SEM images.

Active contours, also known as snakes, have been introduced by Kass et al. (1988) and adapted to SEM imaging by Sievers and Fatikow (2006). Using the active contour approach, a contour is fitted around the tracked object. This contour usually consists of several splines. With each new image, the contour is adapted to fit around the changed object. The adaptation can be performed using different methods, either based on detected edges or the encircled region (Ronfard, 1994; Blake and Isard, 1998). An important advantage of active contours is the ability to not only track the position of an object but also its rotation, magnification and deformation. However, for edge-based contour modification, a robust detection of an object's edge is required which is difficult in noisy SEM images. Thus, region-based active contours show better performance (Sievers, 2006; Fatikow et al., 2008). The computational complexity of current region-based implementations, however, is as high as that of cross correlation.

In general, feature-based tracking approaches that work in spite of the substantial noise in SEM images have a high computational complexity. This complexity prevents high update rates and introduces additional latency.

Model-based tracking approaches

Model-based tracking approaches use knowledge of a tracked object's geometry and possible movements to determine the position in a given image. A basic matching between geometrical features and their appearance in an image needs to be feasible.

Tracking microobjects based on a three-dimensional computer-aided design (CAD) model was presented by Yesin and Nelson (2005). With such a CAD model, an object's pose in all six degrees of freedom can be extracted from a optical microscope image. Similar to a camera, a microscope exhibits perspective distortions, i.e. objects appear smaller if they are located further from the microscope. Thus, as the physical dimensions of an object are known from the CAD model, the distance from the microscope can be estimated from the size of the object in the image. Furthermore, as object tracking relies solely on fitting the model into the image and the tracking is independent of distinct features, the approach can handle occlusions, which commonly occur during handling operations.

Kratochvil et al. (2009) adopted the idea of CAD model-based tracking to SEM images. The measurement of the z-position is limited however, as SEMs barely show perspective distortions, especially at magnifications above 1000× (see Section 2.2.1). Due to the flexibility of SEM image acquisition, the speed of the tracking can be increased by only scanning small regions of the tracked object that contain significant information. The regions to scan are selected based on an analysis of the model.

Speed optimization

A lot of work has also been done on making SEM-based position tracking approaches fast and real-time capable (Sievers and Fatikow, 2006). Real-time capable in this context does not comply to the definition given in Section 2.1 but means that the position tracking can be done with the same rate as the image acquisition. Thus, a live stream of SEM images can be analyzed and the derived position information can be used for closed-loop control.

Three basic approaches can be distinguished that increase the speed of SEM-based tracking. First, the SEM acquisition speed, i.e. the scan speed, can be increased. This however also increases noise on the pixel value and can make tracking algorithms more difficult and less precise. Second, the scanned region of interest can be reduced so that less pixels are acquired leading to higher update rates. This however also limits the tracking robustness, as the tracking is lost as soon as an object no longer resides in the scanned region. This aspect is carefully analyzed in Section 3.1.2. Third, the computation itself can be optimized, as it is a bottleneck for several tracking approaches.

Although several optimizations have been implemented in order to make visual servoing based on SEM images possible, both image acquisition and image processing remain time consuming operations. No approach has yet been developed that takes the dynamic aspect of visual servoing into account and positioning based on SEM images remains a quasi-static technique.

2.4.2 Three-dimensional positioning

The SEM is a valuable tool when supporting nanohandling operations. However, the high depth of focus (see Section 2.2.2), an initially beneficial property of the SEM, is also a drawback, as accurate height information cannot be derived from the focus. Thus, the precise z-measurement and z-alignment of objects in SEMs is a recurring problem.

Depth detection using focus

Focus-based depth detection methods are based on focus measures, i.e. the conversion of the focus in an image to numeric values. Although there is a variety of sharpness measures to evaluate the focus in images, the normalized variance of the pixels' gray values shows the best overall performance for computer microscopy applications (Sun et al., 2005). This is especially true for SEM images because of the substantial level of noise. Fig. 2.6 shows a typical plot of the normalized variance when an object moves through the focus plane. As the SEM with the used aperture and working distance has a depth of focus d_f of approx. 5 µm, there is a peak with a width of approx. 5 µm making the determination of the precise height of an object difficult.

There are two basic challenges when using the focus to determine the z-position of an object in an SEM. First, an approach must be able to obtain height information with a resolution better than the depth of focus in order to be useful as z-position sensor. Second, there is no direct relation between any available focus measure and an actual relative or absolute z-position. For robot positioning however, such

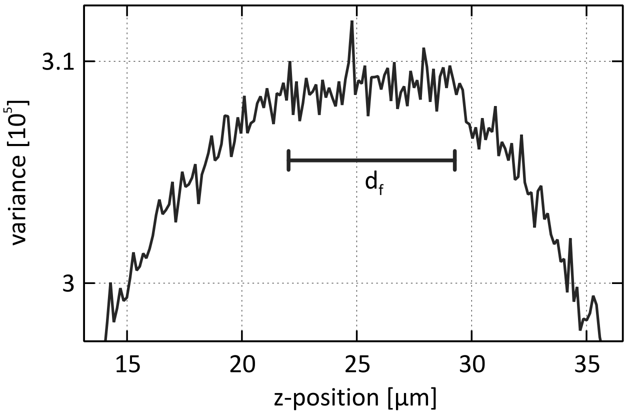


Figure 2.6: Normalized variance when moving an object through the point of highest focus with approximated depth focus d_f.

a relation is required. The minimum requirement is the possibility to align two objects at the same height.

Two approaches have been developed to determine the height of an object in an SEM image. The **depth from focus** approach (Dahmen, 2008) determines the height of an object by changing the SEM's working distance setting and thereby sweeping the focus plane through the object of interest. A variance curve similar to the one in Fig. 2.6 is recorded and the peak of the curve is matched to a working distance setting which delivers the z-position of an object. This approach is similar to auto focusing methods implemented in many current SEMs. For auto focusing however, a setting anywhere within the depth of focus d_f (see Fig. 2.6) is acceptable. For the depth of focus approach, the peak's center needs to be determined with as high a resolution as possible. Although this approach can determine the absolute z-position of an object, it has certain limitations. First, sweeping through a focus range as well as acquiring and processing multiple images is time consuming. Second, the focusing lenses in an SEM are often affected by hysteresis and other non-linear behaviors. This makes an accurate absolute positioning with this approach impossible. Nevertheless, measuring the relative z-position between two objects is possible, if sharpness measures for both objects can be obtained during a single focus sweep.

To enhance the measurement speed of focus-based depth measurement, the **depth from defocus** approach (Dahmen, 2009) was developed. It is divided into two phases. During the initialization phase, an entire focus sweep is performed by either moving the object or changing the working distance. The variance values for the different z-positions are stored. During the measurement phase, the normalized variance of a single frame is determined. Using the data acquired during initialization, this normalized variance is then attributed to a specific z-position. The information of whether an object is located above or below the focus plane can be obtained from a single image, if there is astigmatism in the image. To measure the astigmatism, the horizontal and vertical variances are calculated independently. One will be higher for locations above the focus plane, whereas the other will be higher for locations below the focus plane. A major drawback of this approach is that most focus measures are affected by a change in contrast or brightness as well as a changing scene. Thus, the relation between the focus measure and a corresponding z-position is time-variant. Furthermore, due to the noise in SEM images, the focus measure shows significant noise, limiting the approach's resolution. Dahmen (2009) reported a resolution on the order of 10 µm. However, the absolute accuracy has not been measured and the possibility to align multiple objects on the same height has not been demonstrated.

Depth detection without using focus

Eichhorn et al. (2008) describe a so-called **touchdown sensor** which detects the contact between two objects on a nanoscale level using a piezoelectric bimorph bending actuator. The piezoceramic element on one side of this actuator is driven with a low-amplitude alternating current (AC) signal. The other side is used as a sensor. A frequency close to the bending actuator's resonance is chosen inducing an oscillation with an amplitude of approx. 50 nm. A contact between this sensor and an object results in a different resonance behavior, which can be detected with high sensitivity by a lock-in amplifier. Although the sensor was successfully employed to detect the contact between a sharp tungsten tip and a nanotube, the reliability is limited, as the sensor is affected by environmental disturbances such as vibrations.

Another approach for depth detection is **stereoscopic imaging**. Two images of a scene have to be acquired from different angles. Cornille et al. (2003) demonstrated, that it is possible to obtain highly accurate three-dimensional measurements of a structure using a single SEM, if the specimen can be rotated. Rotating the specimen is similar to observing the specimen by multiple electron columns. Thus, a three-dimensional reconstruction is possible and the obtained resolution is in the range of tens of nanometers. However, the distortions present in SEM images (see Section 2.2.1) limit the performance and need to be compensated for. Furthermore, as the images showing the specimen from different angles are recorded with a significant time difference, drift needs to be factored into the calculation (Sutton et al., 2006). This method of three-dimensional measurement, however, is not applicable to a positioning task, because it is too slow and most robotic systems do not allow for being tilted.

An alternative to rotating the system is tilting the electron beam by a magnetic beam deflection system (Jähnisch and Fatikow, 2007). The stereoscopic analysis in SEM images acquired using a beam deflection system faces three additional challenges in comparison to camera-based stereoscopy. First, the images exhibit a high level of pixel noise. This makes finding corresponding points difficult. Second, the two images are not acquired simultaneously. Although this does not affect the three-dimensional imaging of stationary objects, it makes three-dimensional tracking of moving objects infeasible. A moving object creates a position offset in the stereoscopic image pair and the offset cannot be clearly attributed to either movement or z-position. Third, the angle between the two images, also known as vergence angle, is comparatively small, usually less than 1°. Thus, the obtainable z-resolution is inherently limited. A solution currently under investigation is the employment of cross-beam SEM/FIB (focused ion beam) systems. However, this entails additional costs and makes the correspondence analysis significantly more complex due to completely different imaging modalities.

An approach similar to laser triangulation can also be used in SEMs. Schmoeckel et al. (2001) attached a cathodoluminescent material to a microgripper. When the SEM beam hits a point on this material, the point is illuminated. The location of this illumination can be detected using a camera with microscope optics mounted in a certain angle with respect to the electron beam. A downside of this approach is that only objects coated with cathodoluminescent material can be measured. Furthermore, the camera has substantial space requirements and a limited accuracy. Schmoeckel et al. (2001) achieved an accuracy of 30 µm. A similar system has been used by Buerkle and Fatikow (2000) under an optical microscope. A laser was mounted in a certain angle with respect to the optical axis and moved horizontally to create a line. Due to the high magnification of the optical microscope, the achieved resolution was about 1 µm. Both systems however, are unable to measure the position of nanoobjects, because nanoobjects cannot be coated with cathodoluminescent material and are not visible in optical microscopes.

To conclude, precise z-positioning still is a challenging problem. No approach has been developed so far, that can be used to quickly align two objects with a relative accuracy higher than 1 µm.

2.4.3 Line scan-based tracking

The technique of line-scan based tracking with the SEM, which is a major part of this thesis, is a new approach with dedicated challenges that has not been used before. However, line-scan based tracking has been successfully applied in several other environments. Mokaberi and Requicha (2006) used line scans conducted by an atomic force microscope (AFM) to iteratively find the center of a spherical particle. The highest point on a line scan along the x-direction is determined and used as x-coordinate for a subsequent scan along the y-direction. The highest point on the latter is then used as y-coordinate of a second scan along the x-axis. This process continues until a predetermined accuracy is achieved. The authors do not describe how the highest point is determined.

A similar approach is employed by Onal et al. (2009). Line scan-based tracking is used for the automated handling of nanoparticles with an AFM. In their experiments, the AFM is used for both sensing and manipulation. After an initial imaging scan, the goal is to arrange several nanoparticles into a predefined target position. As the pushing of nanoparticles with the AFM is an unreliable operation, the precise position of a nanoparticle after each operation needs to be measured. As repeated image acquisitions would be a major bottleneck, this measurement is conducted based on two line scans. The so-called watershed algorithm is used to determine the position of the particle on the line scan.

For both approaches, two important differences exist with respect to the tracking algorithm presented in this thesis. First, due to the comparatively low scanning rate of an AFM, the tracking algorithm does not need to be fast or performed during the line scan. Second, the tracked particle can be assumed not to move during the position measurement. This condition, however, would only be fulfilled using a look-then-move approach.

2.5 Motion control

Motion control describes the task to move a robot to a specific coordinate in a given coordinate system. Open-loop control describes the control of a robot not using sensors, i.e. calculating the actuation parameters for a desired motion based on a model of the robot. Utilizing the open-loop motion controller, a closed-loop motion controller incorporates a sensor's feedback to compensate for errors in the open-loop control as well as external disturbances.

2.5.1 Open-loop motion control

The challenges of open-loop control strongly depend on the employed actuation principle and robot type (see Section 2.3). For piezoelectric actuators used in a scanning mode, the open-loop control mainly has to counteract the non-linear behavior of the piezoelements, i.e. hysteresis and drift (Croft et al., 2001). For step-wise actuators, time variance is an additional challenge, as actuation principles such as stick-slip actuation are influenced by environmental conditions and wear. For nanorobots with multiple coupled degrees of freedom such as mobile nanorobots, the degrees of freedom influence each other and the inverse model, i.e. the mapping from desired movement to actuation parameters, is ambiguous. Thus, in general, it is hard to derive a dynamical mathematical model that describes a robot's behavior with sufficient accuracy.

A common method that can still be used to derive an adequate model is system identification (Ljung, 1986). With system identification, statistical methods are used on the results of multiple test movements in order to derive the required movement model. The model can either be a black box model, assuming no prior knowledge of the system, or a gray box model, using certain knowledge of the system in conjunction with free parameters. System identification, however, does not solve the control problem posed by a strongly time-variant system.

A method that is especially suitable to work with time-variant systems is an adaptive or self-learning open-loop controller. Hülsen (2007) developed a self-learning controller based on self-organizing maps. This controller can learn a wide range of mapping functions between an input space and an output space. It was successfully applied to learn an inverted model of the behavior of two different mobile microrobots. With supervised online-learning, the controller can adapt the learned mapping functions and handle time-variant control problems.

2.5.2 Closed-loop motion control

Closed-loop control for nanorobots has been studied mostly for scanning actuators. Similar to open-loop control approaches, a major task for the closed-loop control is the compensation for hysteresis, drift and other non-linearities of the employed actuators (Croft et al., 2001; Mokaberi and Requicha, 2008). Additionally, the speed, i.e. bandwidth and settling behavior, of the closed-loop controlled system is a subject of interest. For example, closed-loop positioning with a high bandwidth is required for the scanning stage of AFMs if line frequencies of more than 5 Hz are desired.

There are several approaches for the high-speed closed-loop control of the scanning stages used in AFMs. Li and Bechhoefer (2007) propose a feedforward control approach using the deviation measured during the current movement to predict the best possible control signal for the next movement. This is feasible, because an AFM scanner performs periodic movements with no significant external disturbances. Another approach to control an AFM scanner is proposed by Salapaka et al. (2002). A robust control approach is chosen using an H_∞ controller. With this approach, the objectives of precisely and quickly following the control input are an integral part of the problem formulation. The actual control law is then derived by solving the optimization problem containing these and other objectives, e.g. robustness and stability. The paper shows that significantly better results can be obtained compared to a classic PI-control scheme, even with highly optimized parameters. With both approaches, line frequencies of up to 50 Hz become usable.

Motion control as described above can be applied for the scanning mode of stick-slip actuators, i.e. the slow deformation of the actuator without performing steps. To perform longer, closed-loop control motions, a motion control has to incorporate the step-wise motion. Thus, a hybrid controller is necessary which performs a step-wise motion until the target position is within scanning range. A design for such a controller was outlined by Breguet and Clavel (1998). However, a careful analysis and optimization in terms of accuracy and speed has not been performed.

2.5.3 Trajectory control

While there have been many applications of trajectory control in the macro world, barely any research has been conducted on the trajectory control of micro- and nanorobots. This is contradictory to the fact that trajectory control is essential for many nanohandling applications due to the huge differences in scale. A robot which is several cm^3 in size moves tools and objects on the nanometer scale. Thus, it is necessary to control the robot as precisely as possible to its target location without intermediate states being unknown. However, the lack of adequate sensors that measure the position of moving nanoobjects with the required precision makes nanorobot trajectory control challenging.

A sort of trajectory control for a mobile microrobot has been described by Fatikow et al. (2007). However, the use of the term trajectory differs from the definition in Section 2.1 as no precise timing is used for the movement. Instead, the robot is set to follow a given path as quickly as possible. At any given state, the measured robot pose differs from the predetermined path at least slightly. Using the measured pose, the controller calculates the local deviation from the path and the orthogonal desired local pose, i.e. the pose on the path with the shortest distance to the measured pose. The motion controller then balances the two goals of moving towards the path and moving along the path.

The planning of trajectories is a common optimization problem in a variety of robotic fields. Dependent on the application, a trajectory can be optimized with respect to different goals, including minimum time (Piazzi and Visioli, 1998), minimum acceleration and minimum jerk (Piazzi and Visioli, 2000). A common type of trajectory is the so-called point-to-point trajectory (Kyriakopoulos and Saridis, 1988). To this end, a trajectory is defined as a straight movement between two points implementing a specific speed, acceleration and jerk profile. Common applications for such a trajectory are machining applications such as milling or turning. More general trajectories that require curved movements are often calculated using cubic splines (Lin et al., 1983).

For point-to-point trajectories, a third-order motion profile is commonly used in order to limit the jerk during a movement (Lambrechts et al., 2005). Excessive jerks need to be prevented for most applications. In machining applications, jerk leads to a vibration of the tool and thus to chatter. Other robotic operations, e.g. assembly operations, need settling time in order for vibrations to subside. Furthermore, jerk is translated into an impulse on moved objects. In general, limiting jerk improves path tracking accuracy and reduces wear on the employed robots.

For many robotic applications, it is necessary to calculate trajectories in real-time during the process. Macfarlane and Croft (2003) proposed a concept of calculating fifth-order trajectories in real-time with a guaranteed limit on computation time. This concept is extended by Haschke et al. (2008) in order to allow for arbitrary initial states.

To conclude, no significant research has been done on the trajectory control of nanorobots. However, the calculation of a trajectory for a given point-to-point movement is well-understood and is applicable to the nanoscale. Based on such a calculation, a trajectory control is implemented in Section 4.3.

2.5.4 Latency compensation

Latency, also known as time delay, is a major challenge for a variety of process control systems. It increases the complexity of analytical control system design and makes fast and stable control difficult to achieve. Latency can be divided into system latency and sensor latency. System latency describes the time it takes for the control signals issued by the controller to be applied to the system and impacts setpoint control. Sensor latency is the time between a change in system state and the corresponding change of a sensor value in the controller and impacts the attenuation of load disturbances.

To compensate for latency, the system behavior during the time delay needs to be predicted. The simplest form of prediction is the derivative part of a PID controller. However, the derivation of the measurement signal can only be used to predict short latencies. For processes with longer latencies, the derivative part in fact becomes counterproductive and is often turned off leading to a PI process control.

Smith (1958) proposed a method to compensate for the latency of a system by adding a second feedback loop to a conventional controller. The additional feedback loop predicts the system's behavior during the latency duration based on a mathematical open-loop model and adds the predicted behavior to the conventional feedback. Thereby, the system can be converted into and handled as a system without latency. This is known as the **Smith predictor** method.

The original Smith predictor, however, has limitations. As it is based on a precise model of the system, a constant load disturbance leads to a steady-state error. Furthermore, it does not adequately handle initial conditions both in the system and in the controller (Watanabe and Ito, 1981). Thus, Watanabe and Ito (1981) proposed a modification, coupling a Smith predictor with a PI or PID controller. The integral part in this controller removes any steady-state error. The performance of the controller was later improved by Åström et al. (2002) by decoupling the setpoint from the load response.

All predictors have in common that they need a model of the controlled process. In a simple form, a process can be modeled by a gain, a delay and a time constant. Together with the two parameters for a PI controller, five parameters need to be chosen adequately for an effective control. Hägglund (1996) proposed a concept of reducing these five parameters to three. The proposed concept is also applicable to processes with varying latency.

Although latency is a challenge for control systems, it can be effectively compensated for if the amount of latency is known and a precise model of the system behavior is available. A PI controller should be used to avoid steady-state errors in spite of a constant load disturbance.

2.6 Automation on the nanoscale

Similar to macroscale automation, the automation of nanohandling processes requires intensive research and development on multiple aspects (Fahlbusch et al., 2005; Jasper et al., 2009). First, a robotic system needs to be designed that is capable of performing the required task. Second, a software system needs to be built, that controls the individual, heterogenous components of the robotic system as well as related processes such as image processing. Such a system is commonly divided into low-level controllers and a high-level controller (Arbuckle et al., 2006). The low-level controllers encapsulate the different hardware components and provide primitive functions, e.g. move to point B, to the high-level controller. The high-level controller, also called automation server, can be programmed to execute a specific handling sequence using the primitive functions provided by the low-level controllers.

In other aspects however, automation on the nanoscale is significantly different from macroscale automation. For example, on the nanoscale, surface forces are a much stronger influence than gravitational forces (Fearing, 1995). Thus, processes such as gripping and releasing objects become significantly harder and the result of specific operation is afflicted with a higher degree of uncertainty. Nevertheless, there are examples of successfully automated nanohandling.

The goal of the project ZuNaMi (Wich et al., 2009) was the automated mounting of CNTs onto specific targets. Two Cartesian robots with three degrees of freedom each were used to carry the CNT substrate and an array of sharp tungsten tips, respectively. Two key technologies used for the process were electron beam-induced deposition (EBiD) as well as electron beam-induced etching (EBiE). After bringing a tungsten tip into contact with the tip of a CNT, a chemical bond was formed using EBiD. Then, the bond between the CNT and the substrate was removed using EBiE. Thus, the CNT was held in place by at least one strong chemical bond at any time during the experiment to achieve a highly reliable automated operation.

A common approach to tackle the problem of precise positioning are the so-called zoom-and-center steps. Starting from a low magnification, the specimen and tools are positioned to the center of the SEM image. Then, the magnification is increased and the process repeated in order to get all objects into the center of the working area at a very high magnification. Wich et al. (2009) derived a formula of the amount of zoom-and-center steps required to get to a specific target magnification.

As several imaging parameters need to be adjusted after each step, this has a significant impact on the process time.

Within the scope of the European project NanoHand (Eichhorn et al., 2009), a robotic architecture for the automated assembly of carbon nanotubes onto the tips of AFM probes was developed. The enhanced AFM probes are used for measuring high-aspect-ratio structures such as deep trenches. Using electrothermal microgrippers (Andersen et al., 2009), CNTs grown vertically using chemical vapor deposition were successfully detached from a substrate and placed on standard AFM probes. The final setup was fully automated and able to repeatedly create an enhanced probe in less than three minutes. Several aspects described in this thesis such as the robot control architecture were developed within the scope of the NanoHand project.

Several research groups have investigated the automation of AFM-based manipulation (Onal et al., 2009). However, an AFM only allows for specific nanomanipulations such as pushing and cutting. Furthermore, visual feedback is missing during the actual handling, so that the possibilities for process control are limited.

2.7 Conclusions

The full automation of specific nanohandling processes has been achieved, but the process times are on the order of minutes. For successful industrial application, the speed and throughput of the processes need to be increased by several orders of magnitude. Major bottlenecks are the speed of highly accurate positioning as well as the tuning of SEM parameters for image-based process control.

For automated closed-loop positioning, an adequate position sensor is required. Current robots do not allow for positioning with the accuracy required for nanohandling solely based on internal position sensors. All current implementations of vision-based closed-loop control however treat the positioning in a quasi-static fashion, commonly using the look-then-move approach. This approach is inadequate to high-speed positioning and it cannot be used to perform a precise trajectory control as the location of the robot during a movement cannot be controlled. High-speed visual servoing is hindered by the bottleneck of SEM image acquisition and image processing. Thus, to facilitate high-speed motion control, a new position sensing approach is required. Even accepting low positioning speeds, there is no practicable solution for the nanoscale alignment of two objects at the same height due to the missing z-information in SEM images. Several approaches reach accuracies of about 1 µm but a real nanopositioning has not yet been achieved.

Different robot architectures have been implemented and can be categorized into serial and parallel architectures. A serial system comprised of linear actuators has the advantages of internal sensors and a straightforward control. However, due to the long force flow, the mechanical stiffness is significantly reduced with each additional degree of freedom. Mobile nanorobots are a promising alternative that offers multiple degrees of freedom in a parallel fashion. With a new, external position sensing approach, the major downside of mobile nanorobots, the lack of internal sensors, becomes insignificant.

The closed-loop control of nanorobots still is at an early stage. Different control approaches have been implemented and tested for scanning piezoactuators used in AFM stages. However, hardly any research has been done on the control of step-wise actuators, trajectory control of nanorobots or the compensation of sensor latency required for high-speed closed-loop control in SEMs.

3 Line scan-based tracking in SEMs

As described in the previous chapter, the performance of vision-based servoing using SEM images is limited. Although many approaches have been developed for tracking objects in SEMs, all of these approaches have in common that they operate on images. However, the SEM physically is not an imaging device and scanning the electron beam line by line to acquire an image is a bottleneck for the tracking performance. Furthermore, objects that move during the image acquisition can become impossible to track as they appear distorted. Thus, a new tracking methodology is required. To understand the inherent bottleneck of SEM image-based tracking and to derive a better approach, a formal analysis of the tracking process and its performance is necessary.

3.1 Tracking with the SEM

There are three major characteristics that describe the performance of a tracking algorithm:

- Resolution
- Robustness
- Speed

The **resolution** describes the smallest movement that can be reliably detected by a tracking algorithm. Algorithms tracking objects in images can have a resolution of multiple pixels, single pixels or even sub-pixels. Furthermore, the tracking's noise, i.e. the variation of the result when tracking stationary objects, is highly important. Changes of the tracking result smaller than the noise cannot be assumed to be movements of the tracked object. Due to the SEM's high level of pixel noise, this tracking's noise is high and limits the possible resolution. If the update rate is less important than the reduction of noise, averaging over multiple tracking calculations can be used. With averaging, noise is reduced whereas the resolution remains unchanged.

The **robustness** describes the capability of a tracking approach to continually track an object under changing conditions. If the tracking cannot continue, a so-called loss of tracking, the tracked object's position becomes unknown, which can result in damage of the involved components. For high-speed positioning, the maximum velocity of a tracked object is an important aspect of robustness. Additionally, changes in brightness, contrast and focus as well as occlusions should not prevent a successful tracking.

The **speed** is measured by the update rate and the latency. The update rate is the number of position updates generated per second. For online algorithms, the update rate is identical to the frame rate of the SEM. The frame rate can be increased by acquiring fewer pixels or accepting a higher noise level. Higher update rates enable faster control during a closed-loop movement, e.g. a more precise following of predetermined trajectories and a shorter settling time. Latency is the delay between pixel acquisition and availability of the tracking result. With a significant latency, each sensor update is outdated limiting the performance of closed-loop control (see Section 2.5.4).

The tracking efficiency, i.e. the amount of information retrieved from each pixel, can be used as a common measure that influences all of these characteristics as detailed below.

3.1.1 Tracking efficiency

At any given time, the SEM beam only delivers information of one specific location on the observed specimen. The electron detector signal of an SEM is digitized using an analog to digital converter, which usually operates at a fixed frequency f. Each new digitization delivers an update of the electron interaction at the current position of the electron beam and represents the smallest piece of information. For the following considerations, this smallest piece of information is called **pixel**.

The effectiveness of a tracking algorithm can be measured by evaluating the amount of information obtained from each pixel. In order to make the position tracking as effective as possible, the amount of information gathered from each pixel needs to be maximized. Four classes of pixels can be distinguished:

- Unusable pixels
- Unused pixels
- Potentially used pixels
- Pixels with information

Unusable pixels are generated during rapid beam movements, e.g. when jumping from one line to the next as a part of image acquisition. For each rapid beam movement, the beam needs a certain amount of time to settle to its new position. During this time, the correspondence between beam location and pixel cannot be inferred and the pixels need to be discarded. As the electron beam is positioned using electromagnetic coils, the response behavior can be approximated by a low-pass filter. Such a filter has a constant settling time t_s. Thus, for each rapid beam movement, a constant number of pixels needs to be discarded. A detailed analysis of the dynamic beam behavior is conducted in Section 3.5.3.

Unused pixels are acquired during data acquisition but it is known a priori that the information will not contribute to the tracking calculation. For tracking based on images these pixels are often caused by the scanned region having a rectangular shape while the shape of the object itself is more complex. The number of unused pixels should be minimized for any tracking approach.

Potentially used pixels are acquired and evaluated but do not contribute to the tracking calculation for this specific frame. A common reason for acquiring this type of pixel is the handling of object movement. Commonly, the tracked object has a certain size. However, as it is expected to move, an area larger than the object itself needs to be scanned in order to locate the object within this area. All pixels not belonging to the tracked object do not deliver information about its location. They might however contain this information for another frame.

All other pixels are **pixels with information**. The information value is dependent on the tracking algorithm and can vary significantly. Ideally, each pixel is unique and the pixel's gray value can be unambiguously used to determine the pixel's position. However, this ideal case cannot be achieved due to the manifold uncertainties in an SEM system. These uncertainties include changing focus of the electron beam, as well as changing amplitude, offset and noise of the detector signal.

Fig. 3.1 shows the four pixel classes for the image-based tracking of a particle on an untextured background (Fig. 3.1a). The green circle (see Fig. 3.1b and c) contains the pixels with information as these pixels show different parts of the particle. The blue area contains potentially used pixels. Depending on the motion of the particle, it could have been located anywhere within the blue area. Thus, for tracking based on a particular image, the green circle might be located anywhere within the blue

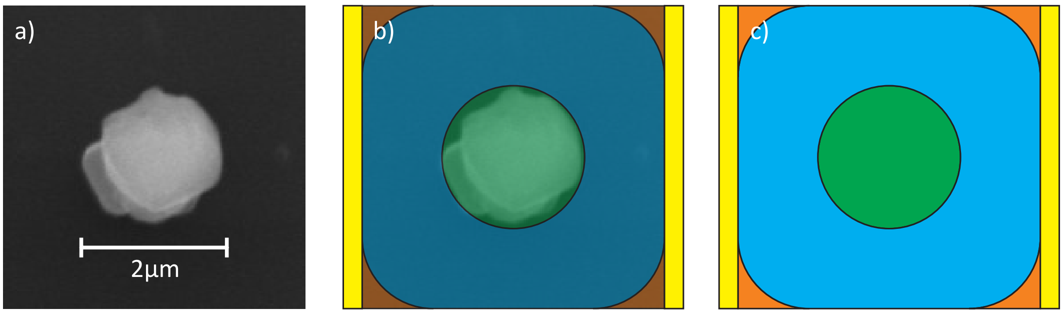


Figure 3.1: Pixels with different information content during tracking: a) SEM image of tracked object, b) image overlaid with pixel types and c) pixel types.

area. However, the amount of pixels in each area is constant. The red corners are unused pixels as they can never show parts of the tracked object. This is assuming that the tracking algorithm does not work if a part of the object is cut off as e.g. the case for cross correlation (see Section 2.4.1). The unusable pixels are symbolized by the yellow areas on the sides of the images. These pixels are not actually part of the image but are generated as the electron beam needs time to settle to the beginning of the next row and scanning usually starts left of the image and exceeds past the right end of the image (see Section 3.5.3).

Each pixel contains a certain amount of information. For unusable pixels or unused pixels, this amount is 0. They nevertheless are acquired due to limitations of the electron beam movements. Potentially used pixels do not contain information other than the absence of the tracked object. These pixels need to be acquired as the location of an object is not known before then scan. Thus, these pixels are acquired to take the tracked object's movement into account. All other pixels deliver direct information on the tracked object's location. This information can be used to determine the object's position, counteract noise as well as amplitude, offset or focus changes of the object.

3.1.2 Moving objects

Tracking objects, especially with the goal of robotic motion control, is only useful for objects that are moving. The tracking of moving objects based on SEM images faces two challenges:

- The object must not leave the view field or scanned region of interest.
- The object appears distorted due to the linear scanning line by line.

Both challenges arise during tracking, but are not adequately described in the literature. Furthermore, due to the special imaging conditions in an SEM, both aspects need to be handled considerably different from tracking with optical cameras.

The first challenge appears primitive, but a careful consideration reveals a more complex problem. The basic idea is that a tracked object most not leave the scanned region. Otherwise, the tracking algorithm has no information on the object's location and the tracking is lost. For the following considerations, a rectangular shaped object with a width of w pixels and a height of h pixels is assumed, which has to be fully contained within the scanned region in order to be tracked. The highest possible update rate $r_{\max}$ is achieved when the scanned region is $w \times h$ pixels large. In this situation, all pixels are used for tracking aside from a few unusable pixels (yellow regions in Fig. 3.1). Nevertheless, the tracking is useless, as even a small movement of the object leads to a loss of tracking.

Thus, the area has to be extended past the object, introducing potentially used pixels (blue region in Fig. 3.1). If n additional pixels are scanned around the object in all directions, it can move up to n pixels into an arbitrary direction between two consecutive updates. However, also the update rate is reduced from $r_{\max}$ to $r(n)$:

$$r(n) = \frac{1}{t_p \cdot (w + 2n) \cdot (h + 2n) + t_u \cdot (h + 2n)}, \tag{3.1}$$

with t_p being the time to acquire a pixel and t_u the time required for jumping from one line to the next. This function falls almost quadratically. To be trackable, an object must not leave the scanned region between one tracking update and the next. Thus, the maximum movement speed $v_{\max}(n)$ of an object is limited and defined as:

$$v_{\max}(n) = n \cdot r(n). \tag{3.2}$$

This equation starts at 0 for $n = 0$, increases to a maximum which is dependent on the values of w and h and then drops for higher values of n because of the rapid decrease of the update rate r. Fig. 3.2 shows both $r(n)$ and $v_{\max}(n)$ for a typical object size ($w = h = 50$), a short pixel acquisition time $t_p = 100\,\text{ns}$ and a short beam settling time $t_u = 10\,\mu\text{s}$. As expected, there is a well-defined maximum. Several experiments have shown that the maximum is located at n in the same range as w and h. In this case, the maximum is located at $n \approx 40$. For the example of the particle shown in Fig. 3.1, the 50 pixels width and height correspond to $2\,\mu\text{m}$, i.e. $1\,\text{px} \approx 40\,\text{nm}$. Thus, the maximum theoretically trackable movement speed is a little over $520\,\mu\text{m/s}$ ($v_{\max} > 13\,\text{px/ms}$).

For several reasons, this theoretical limit is hard to achieve in a real implementation. First, due to the low pixel acquisition time t_u, the image will have a low signal-to-noise ratio. Second, no time is reserved for the actual calculation of the tracking

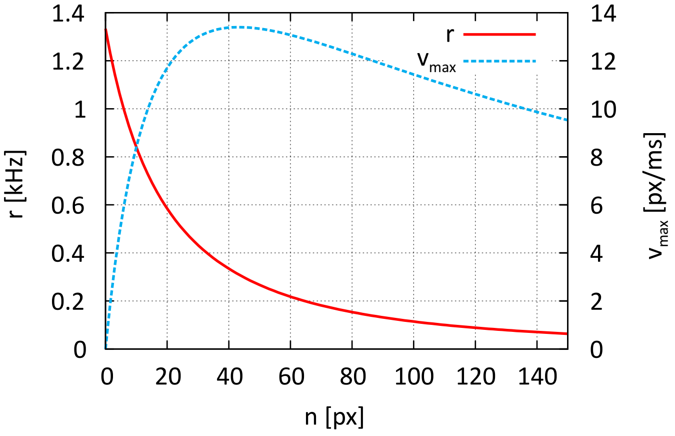


Figure 3.2: Dependency of scanned region, update rate and movement speed limit.

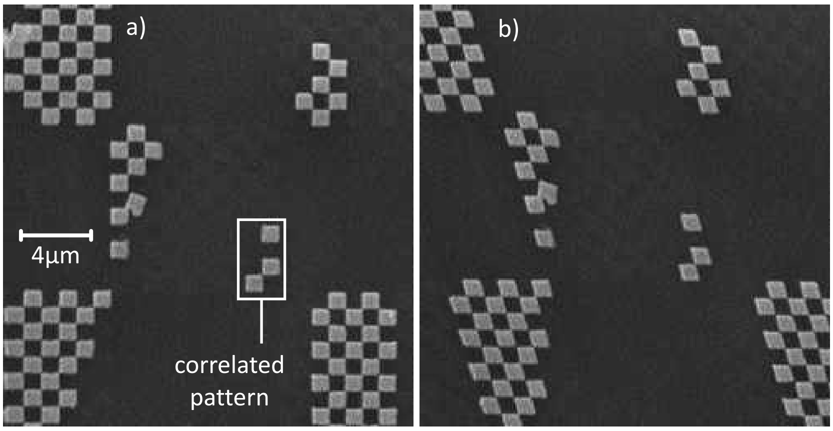


Figure 3.3: Moving Chessy SEM-calibration pattern: a) stationary and b) moving right at a constant 100 µm/s.

or the setup of the new region of interest. Immediately after the completion of a scan, the new position of the object needs to be known, so that a new scan can be conducted around the new position. Third, the tracking algorithm is assumed to find an object anywhere within the scanned region, independent of the region's size. This is not the case for some algorithms, e.g. the active contour approach. Fourth, with such high movement speeds, the object will appear distorted, leading to the second challenge for the tracking of moving objects.

Most optical cameras use a shutter to expose all sensor pixels of the camera chip simultaneously to the incoming light. The movement of an object can thus not lead to a distorted appearance. Due to the comparatively long exposure time however, an object might get blurred as the same part of the object contributes to the value of multiple pixels. In an SEM, these properties are reversed. At any given time, only a single sample location is observed. Thus, an object appears distorted if it moves during image acquisition. However, the movement does not create blurring as the acquisition time of each pixel is only about 50 ns. Thus, an image in the SEM is distorted but not blurred by the motion.

The following experiment shows the effect of image distortions on the performance of a tracking approach. Cross correlation-based tracking was used as an example. Model-based tracking would show similar results, whereas tracking based on active contours could not be used as the tracked object moves too much between two consecutive frames. The imaging conditions were chosen to ideally suit cross correlation-based tracking. A high specimen current of 1 nA leads to a high signal-to-noise ratio despite the 25 Hz frame rate on a 512×256 pixel sized region of interest (RoI). The pattern used for correlation is shown in Fig. 3.3a. It features a high contrast and is significantly different from other objects.

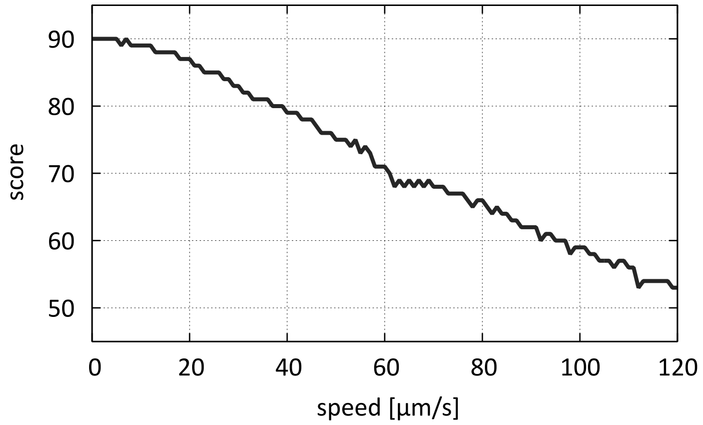


Figure 3.4: Correlation scores when tracking a moving object.

Because of the linear scanning, the motion distorts the object as can be seen in Fig. 3.3b. This distortion makes recognizing an object hard and lowers the correlation score for tracking based on cross correlation. Fig. 3.4 shows the correlation score for different movement velocities. At movements velocities <10 µm/s, the correlation score remains at an almost constant value of 0.9. It does not reach 1.0 because of the noise in the SEM images which is unavoidable at update rates as high as the used 25 Hz in spite of the high specimen current. The score then drops linearly and falls below 0.7 at 60 µm/s. At this point, other objects in the distorted image have similar scores and the tracking becomes unreliable. When moving with 100 µm/s as seen in Fig. 3.3b, tracking with cross correlation becomes impossible. Also, it has to be kept in mind, that image acquisition parameters are chosen to ideally suit the tracking and the pattern is easily recognizable in this experiment. In a real scenario, the usable speeds are likely to be lower.

Additionally, the experiment was conducted using an actuator that creates a very smooth motion, i.e. a movement with virtually constant speed and no vibrations. Many actuators generate vibrations, either by using steep acceleration or deceleration ramps or by a step-wise motion principle such as stick-slip actuation. Such vibrations can lead to severe distortions even for low movement speeds and further affect the tracking performance.

3.1.3 Efficiency of image-based tracking approaches

Feature-based tracking approaches such as cross-correlation or active contours need to acquire a significant amount of potentially used pixels (see Section 3.1.1) in order to allow for viable movement speeds. For the example of a round object with a 50 pixel diameter, roughly 2000 pixels are used for the tracking. A region of interest with at least 40 additional pixels in each direction around the object should be scanned (see Section 3.1.2) to allow for the highest possible movement speed. Thus, an area of 130×130 = 16900 pixels is scanned. Adding the unusable pixels that are lost when jumping from one line to the next and allowing the beam to settle, roughly 20000 pixels are acquired. Thus, only during about 10% of the time, the SEM generates pixels that are actually used for tracking.

Selecting multiple small scanned regions for a model-based tracker can reduce the number of potentially used pixels for large objects that contain a limited number of features. However, the potentially used pixels required for allowing object movement still need to be acquired. Additionally, the number of unusable pixels increases as the electron beam has to perform more rapid movements between shorter line scans. Thus, this approach cannot increase the efficiency substantially.

The major problem for all image-based approaches is the uncertainty about the object's position. As the object can move into an arbitrary direction, a region larger than the size of the object needs to be scanned. In order to allow for movement of up to n pixels, the number of necessary pixels increases quadratically with n. Additionally, scanning line by line requires rapid beam movements creating unusable pixels. Thus, the new tracking approach must not exhibit this quadratic increase of potentially used pixels and must use as few rapid beam movements as possible.

3.2 A new tracking approach

As described above, tracking performance cannot be significantly improved based on images. The linear scanning line by line inherently leads to a low efficiency. Thus, a new method of moving the electron beam to obtain information needs to be created. For the initial considerations, it is assumed that the shape of the tracked object can be chosen arbitrarily. Thus, the tracked object is well-known and exhibits required features. In a real tracking application, such well-known features could be created on arbitrary specimens using different methods such as electron beam-induced deposition (EBiD). In this thesis such position markers are called **patterns**.

Ideally, the electron beam would be left at a fixed point and the tracked object's position would be derived from the detector value. This could be implemented by the electron beam being located on a point of a soft edge, i.e. the detector signal increases if the pattern moves left and decreases if the object moves right (see Fig. 3.5a-c). A closed-loop controller could move the electron beam to keep the detector signal constant. According to the pixel definitions described above, all pixels are used and the tracking has a 100% efficiency. However, such an approach is limited to one-dimensional tracking and strongly affected by changing signal conditions, e.g. brightness and contrast. Furthermore, a changing focus would

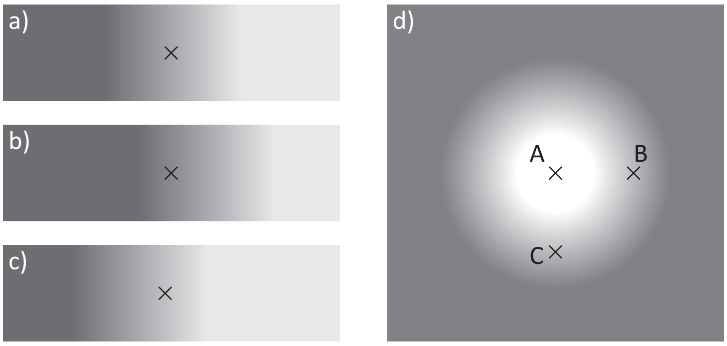


Figure 3.5: Tracking an object with point measurements. a) One-dimensional measurement, b) moving right and c) moving left. d) Two-dimensional measurement with detector value reference.

change the slope of the gray values. Thus, the proportional gain between detector signal and position changes, affecting the speed and the stability of the beam position control.

The approach can be extended to measure two degrees of freedom and to be robust against changing signal parameters by measuring multiple points. Three points could be sufficient for such a measurement as shown in Fig. 3.5d. Point A is used as a motion-independent reference for the detector value. Thus, the values at points B and C are referenced to point A to eliminate the influence of changing signal parameters. Point B then gives a signal corresponding to the x-coordinate whereas point C is used for the y-coordinate. If movement is detected, all three points are adapted. The downside of this extension is that the electron beam cannot jump between the measured points with arbitrary speed. A significant amount of settling time is required after each change creating unusable pixels and limiting the tracking's efficiency. Furthermore, the approach is still strongly affected by changing focus and maintaining the electron beam at a constant point creates charging and beam-induced deposition so that the detector value at point A cannot be assumed to be a fixed reference.

For these reasons, more electron beam movement is necessary in order to obtain sufficient information for a robust tracking. A basic approach is to use the SEM beam to perform **line scans** and extract the position of a pattern based on the result of this line scan. As a first assumption, the pattern looks like a square in an image acquired in a stationary state.

Two line scans are conducted exactly over the last known position of the square's center (see Fig. 3.6a). If the pattern remains stationary, the detector shows a square wave signal on each of the line scans. If the pattern moves between to tracking updates, the line scans are no longer conducted over the square's center and a deviation of the detector signal during the line scan is visible (Fig. 3.6b). Each

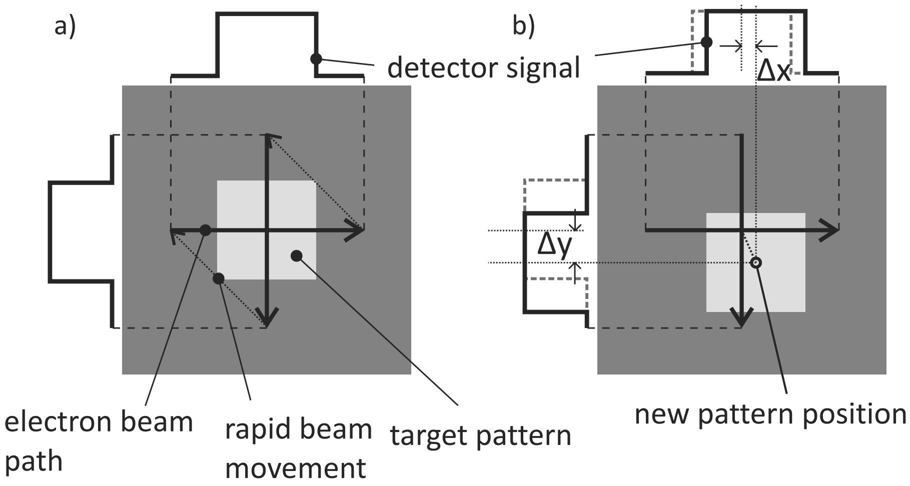


Figure 3.6: Determining the position of a pattern with line scans. a) For a stationary pattern, the line scans are conducted over its center. b) If the pattern moves, each line scan reveals an offset in one degree of freedom.

line scan is used to calculate the pattern's position in one degree of freedom. A movement orthogonal to the scan direction does not change the detector signal due to the so called aperture problem. The position of the next line scan can then be adapted to the new position of the pattern's center and continuous tracking over long distances is possible.

To facilitate the line scan-based tracking approach as outlined above, several key steps have to be considered:

- How can the pattern's position on a line scan be determined?
- Can the approach track moving and thus distorted objects?
- How fast can objects move and how should the scanning be timed?
- What are the parameters and properties of the tracking?

These questions are answered in the following sections.

3.2.1 Position calculation on a line scan

A key step for the line scan-based position tracking is the deduction of an object's position from the pixel data. For the following considerations, two basic cases are considered. The line scan can either be used to detect the center of a pattern similar to Fig. 3.6 or the line scan can be conducted over an edge of a pattern. The resulting pixel values form a trapezoidal or a sloped function, respectively (see Table 3.1). Accordingly, the calculation must either determine the center of the feature or the center of the edge. A detailed description on when to use center or edge detection is given in Section 3.3.

A precise calculation of the position on each line scan is the key to a robust and precise tracking with this approach. There are multiple performance requirements that this calculation has to fulfill. First, it needs to determine the position on the line scan correctly, even with a high level of noise (see Table 3.1). Noise on pixel values of the line scan should not translate into significant noise on the position readout. Second, the calculation should be robust against changing imaging conditions such as gain, offset and focus. Third, as it is hard to create patterns that lead to a specific detector signal independent of environmental conditions and focus, the calculation of the position should be independent of the exact signal shape. Fourth, independent of the signal's position offset and changing width caused by a moving object, the calculation needs to determine the position correctly.

The following sections describe three approaches for this calculation. The **matched filter** is a so-called optimal filter. It can be used as an upper limit of the conceivable performance for some parameters. An approach called **quadrature interpolation**, adapts the working principle of linear encoders to SEM line scans. An alternative calculates the **center of gravity** in order to derive the pattern's position. All approaches are compared with respect to the performance requirements mentioned above. In the following, each line scan returns N pixels with the pixel values $p[i]$ with $0 \leq i < N$. The goal is the estimation of the position R of the pattern on the line scan. R is normalized, i.e. $0 \leq R \leq 1$, where 0 means that the pattern is located

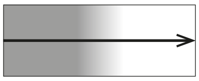
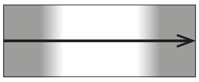
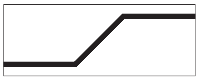
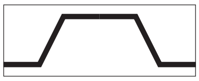
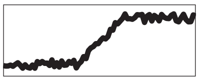
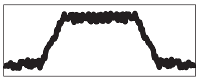
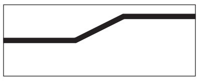
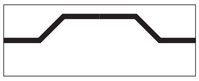
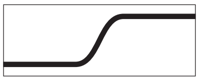
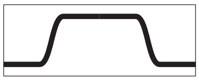
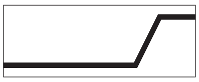
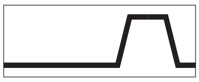
	Edge	**Center**
Scanpath		
Default		
Noise		
Gain and offset		
Shape		
Position and width		

Table 3.1: Signals obtained during line scans.

at the first pixel of the line scan and 1 means that the pattern is located at the last pixel.

Matched filter

A matched filter extracts the position of a well-known signal called template within an unknown signal using correlation. It is an optimal filter in the sense that there is no other linear filter with a better signal-to-noise ratio for a given level of additive noise (Smith, 1997). To implement the matched filter, a template signal h is chosen, representing an ideal signal without noise, e.g. the default signal in Table 3.1. Furthermore, h is extended on both sides so that it is twice as long as p. The filter result y is calculated using correlation:

$$y[n] = \sum_{m=0}^{N} p[m] \cdot h[n+m]. \tag{3.3}$$

The index of y's maximum is divided by $N-1$ and used as calculated position R of the signal. The matched filter exhibits optimal robustness against pixel noise. However, as the precise signal shape needs to be known to create the template, applying the matched filter has severe drawbacks for a changing signal shape, e.g. changing focus, as well as a changing signal width, e.g. through movement (see Section 3.2.2). Thus, the matched filter is mainly used as reference to evaluate the noise performance of the other approaches.

Quadrature interpolation

Quadrature Interpolation (QI) is a well-known method to calculate the position in linear encoders. Two periodic signals can be used for quadrature interpolation if they are of the same period and separated in phase by 90°. Many laser interferometers and optical encoders have two sensor channels generating a periodic sine and a cosine signal, the phase of which depends on the displacement. Based on the two channels' current values, the precise phase, i.e. displacement within the current period, can be calculated using the arctangent function.

The signal generated during a line scan over an edge can be looked at as half a period of a signal. Thus, it creates $N/2$ pairs of points $s[i]$ and $c[i]$, which have a 90° phase offset (see Fig. 3.7a). Thus for $0 \leq i < N/2$:

$$s[i] = p[i] \text{ and } c[i] = p[i + N/2]. \tag{3.4}$$

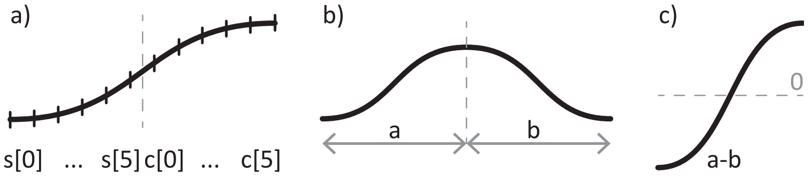


Figure 3.7: Extracting quadrature signals from a line scan: a) Edge signal leads to half a signal period with $N/2$ measurement points. b) Center scan leads to full period with parts a and b. c) Subtracting a and b leads to offset-invariant half period.

A scan over the center of a pattern generates a full period of a signal. To unify the two cases, the second half of the signal is subtracted from the first half (see Fig. 3.7b and c). This increases the signal-to-noise ratio and makes the approach immune to an offset in pixel values, i.e. the resulting signal is always centered around the value 0. Thus, $N/4$ values $s[i]$ and $c[i]$ are calculated $(0 \leq i < N/4)$:

$$s[i] = p[i] - p[i + N/2] \text{ and } c[i] = p[i + N/4] - p[i + 3N/4]. \tag{3.5}$$

Assuming $s[i]$ and $c[i]$ form clean sine waves with a 90° phase offset, the Lissajous curve, i.e. a plot of $s[i]$ against $c[i]$, shows a quarter circle (see Fig. 3.8). For each pair of $s[i]$ and $c[i]$, the angle φ_i can be calculated using the inverse tangent function (Fig. 3.8a). This angle can then be used to obtain the position of the pattern's center:

$$x[i] = i + \frac{N}{2\pi} \cdot \varphi_i = i + \frac{N}{2\pi} \arctan \frac{c[i]}{s[i]}. \tag{3.6}$$

By adding i, each $x[i]$ corresponds to the position of the pattern's center. With equation (3.6), the approach becomes immune to changes in contrast because a change in contrast is a constant factor to both $c[i]$ and $s[i]$ and thus cancelled out

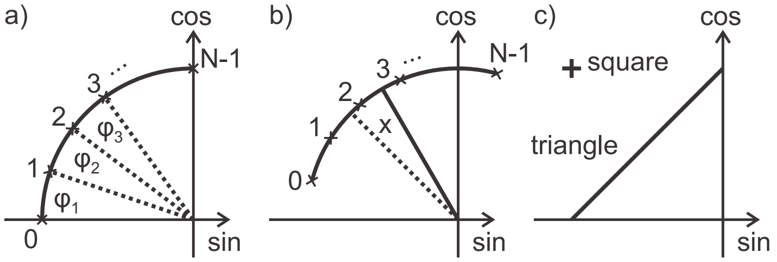


Figure 3.8: Quadrature interpolation: a) For pixels are used to derive a sine and a cosine value with a corresponding angle φ_i, b) the average angle represents the pattern's center and c) the influence of different signal shapes.

in the fraction. In other words, the scale of the quarter circle in the Lissajous curve changes, but the angle towards each point remains identical. Theoretically, each value $x[i]$ is the correct position of the pattern's center. However, if only one such $x[i]$ is used, the calculation is highly sensitive to noise. Instead, all $N/4$ quadruples are used and averaged in order to obtain a robust estimate x of the position:

$$x = \frac{4}{N} \sum_{i=0}^{\frac{N}{4}-1} x[i] = \frac{4}{N} \sum_{i=0}^{\frac{N}{4}-1} \left(i + \frac{N}{2\pi} \text{arctan} \frac{c[i]}{s[i]} \right) = \frac{N}{4} - \frac{1}{2} - \frac{2}{\pi} \sum_{i=0}^{\frac{N}{4}-1} \text{arctan} \frac{c[i]}{s[i]}. \tag{3.7}$$

Simply put, this equation calculates x from the average angle determined by the different $s[i]$ and $c[i]$. The offset between this angle and the -135° line shows the movement of the pattern (see Fig. 3.8b). For signals that are not exactly sine-shaped, the Lissajous curve is not circular. Examples of a triangular and square wave are plotted in Fig. 3.8c. For the square wave, the line collapses to a single point. For both the triangular and the square wave, the different $x[i]$ obtained from each $s[i]$ and $c[i]$ do not correctly calculate the pattern's position. However, the average angle and thus x still is correct. The normalized position R is calculated by:

$$R = \frac{x}{N-1} \tag{3.8}$$

Center of gravity calculation

Another approach to find the current position of the target pattern on a scanned line is the calculation of the center of gravity (CoG). To find the center of a pattern, each pixel is weighted with its gray value and the CoG of these values is likely to represent the real center of the pattern independent of whether it is located in the center of the line scan (see Fig. 3.9a) or at another position (see Fig. 3.9b). On a line of length N, the center of gravity c is calculated by:

$$c = \frac{\sum_{i=0}^{N-1} i \cdot p[i]}{\sum_{i=0}^{N-1} p[i]} = \frac{s_w}{s_v}. \tag{3.9}$$

Thus, s_w is the weighted sum and s_v the sum of the values, which will later be calculated in hardware during the line scan (see Section 3.5.2). In this form, the calculation is highly sensitive to an offset in the pixel value, which can be caused by a change in brightness of the SEM image. Such an offset leads to a changing influence of the pixels not belonging to the feature, and the CoG does not correctly represent the pattern's center (see Fig. 3.9c). This downside can be circumvented by applying a threshold function before calculating the CoG. This function removes

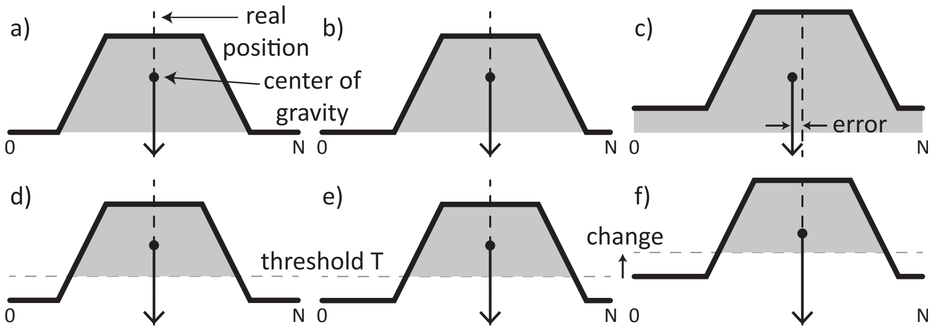


Figure 3.9: Center of gravity calculation.

the pixels not belonging to the pattern from the center of gravity calculation. Thus, for a given threshold T, the equation can be changed to:

$$c = \frac{1}{N-1} \cdot \frac{\sum_{i=0}^{N-1} i \cdot t(p[i])}{\sum_{i=0}^{N-1} t(p[i])} = \frac{s_{wT}}{s_{vT}}, \text{ with } t(p[i]) = \begin{cases} 0, & \text{if } p[i] < T \\ p[i] - T & \text{else.} \end{cases} \tag{3.10}$$

In this form, the position of the pattern can be still determined precisely independent of its position (see Fig. 3.9d and 3.9e). However, the threshold T has to be changed according to the scan in order to make the approach robust against a changing offset of the pixels' gray values (see Fig. 3.9f). For an efficient hardware implementation (see Section 3.5), the new threshold is not available during the current scan and thus cannot be used to calculate the two sums s_w and s_v. However, as brightness does not change rapidly, T can be adapted based on the previous scan. After each scan, the minimum and maximum pixel values are determined and the threshold is changed.

To calculate the position of an edge, the threshold T is chosen as the medium gray value as shown in Fig. 3.10. Then, two calculations are performed to determine the CoG of both the values above and the values below the threshold (see Fig. 3.10). The distances between each CoG and the center of the scanned line are computed and named a and b. The difference of a and b corresponds to the distance of the edge from the center.

For both the tracking of an edge and the tracking of a center, the center of gravity c needs to be divided by $N-1$ in order to obtain the normalized result R:

$$R = \frac{c}{N-1} \tag{3.11}$$

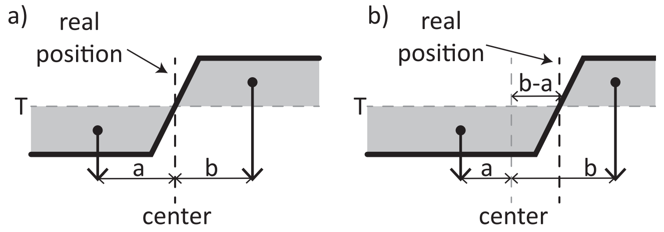


Figure 3.10: Center of gravity calculation for step functions. a) Edge located in the center of the line scan. b) Difference between a and b equals the displacement of the edge.

Comparison

To select the best option, the position calculation approaches described above need to be compared with respect to several characteristics such as noise, linearity and robustness to changing signal shape and focus (see Table 3.1). A change in brightness or contrast does not influence any approach as can be deduced from the mathematical derivation in the previous sections. The comparison is done by applying the approaches to several simulated test signals with well-defined properties.

The involved test signals used for simulating the tracking of a pattern's center are shown in Fig. 3.11a. The signals are chosen to represent several special cases. The square wave signal represents an ideal pattern with maximum sharpness. In contrast, the triangle signal demonstrates the worst case of out-of-focus images with no edge and only a slow and continuous ramp. The trapezoidal signal lies between these two and represents a realistic, not entirely sharp pattern. The cosine

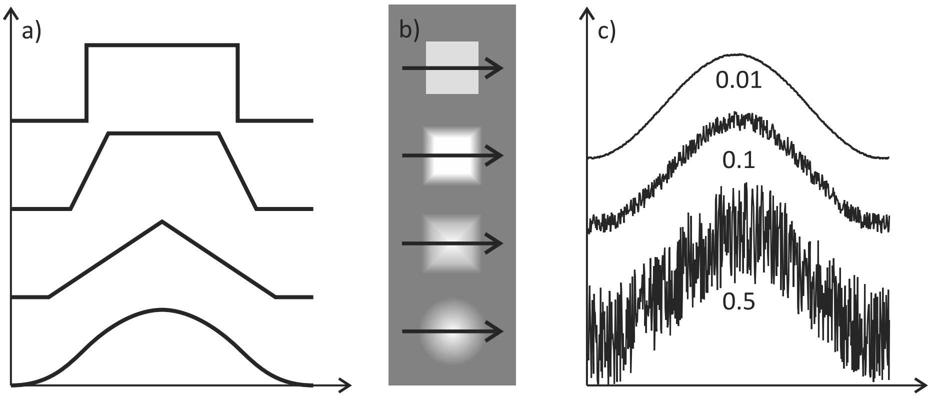


Figure 3.11: Test signals for line calculations: a) Square, trapezoidal, triangle and cosine signals, b) images of patterns leading to these waveforms and c) different levels of noise added to the cosine signal.

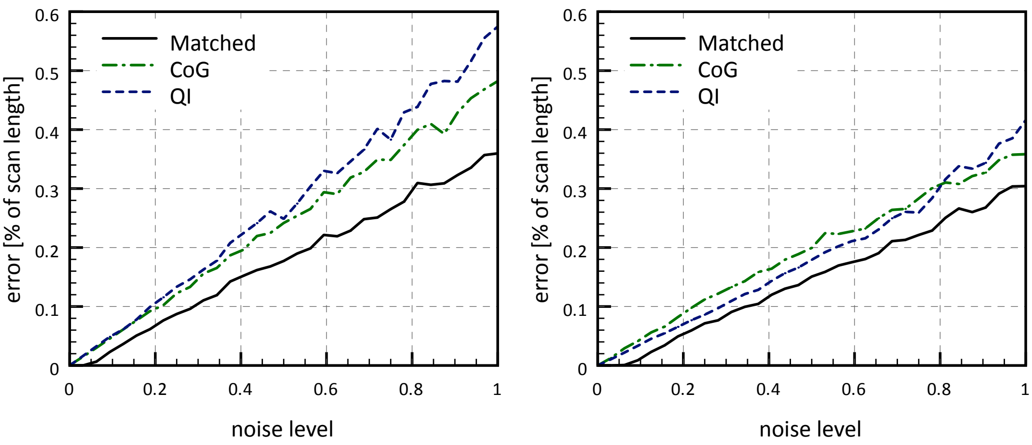


Figure 3.12: Noise performance of the approaches: Left) calculating the position of an edge, right) calculating a signal's center.

signal is used to check if all approaches can deal with a different signal shape. To simulate the detection of an edge, only the first half of a signal is used. Fig. 3.11b shows what an image of a pattern creating the corresponding signals would look like. All of the following simulations are conducted on generated test signals with 512 sample points and calculate an error as a ratio of the scan's length.

To evaluate each approach's robustness against noise, signals with a varying level of signal noise are evaluated. The noise level can be interpreted as the signal-to-noise ratio in a real application. A noise level of 0.5 means that the amplitude of the randomly added noise is half as high as the signal amplitude (see Fig. 3.11c). This is a realistic value for fast SEM pixel acquisition. Fig. 3.12 shows the average error of 100 computations on signals with randomly added noise of the described level. The results show that the matched filter approach provides the best results. This is expected as it is an optimal filter. The center of gravity calculation and the quadrature interpolation exhibit similar results, both exhibiting about 25% increased error compared to the matched filter. In general, all approaches exhibit a good robustness against noise as even the noise level of 1.0 leads to an error of less than 0.6%. Furthermore, all approaches perform about 20% better for the case of determining the center.

In addition to the insensitivity to noise, an approach must be robust against changing focus, as a tracked object might leave the focus plane. Fig. 3.13 shows the simulation results for changing focus. Multiple test signals have been chosen between a square shaped signal, a trapezoidal signal and a triangle signal (cf. Fig. 3.11). The average errors over 100 simulations with a noise level of 0.5 added to the signals are shown. The error of the center of gravity calculation and quadrature interpolation rises slowly with decreased focus. However, even in the worst case, i.e. for a triangle

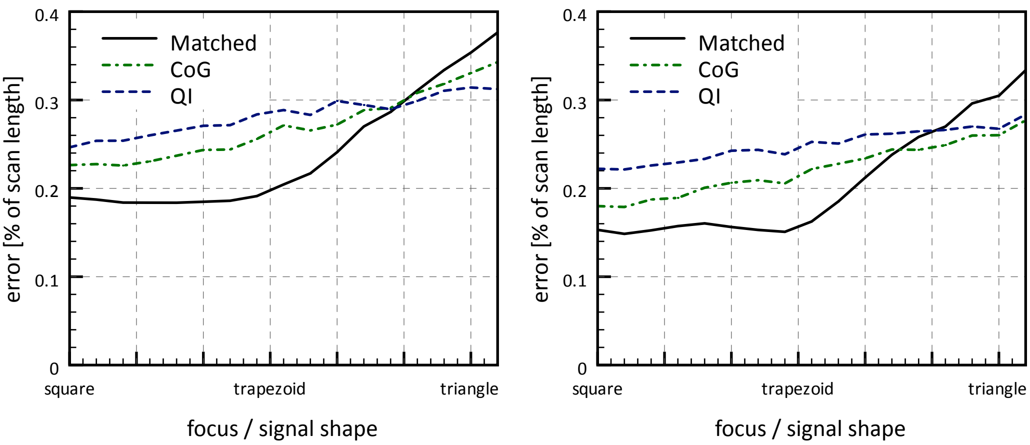


Figure 3.13: Performance with changing focus: Left) calculating the position of an edge, right) calculating a signal's center.

signal, the approaches can be applied successfully. The matched filter was used with a trapezoidal reference signal. For signals similar to the reference or sharper, it constantly outperforms the other two approaches. For signals less sharp than the reference, e.g. the triangle signal, the error of the matched filter increases significantly.

Lastly, to be useful for the tracking of moving objects, an approach must be capable of tracking patterns and objects that vary in size, i.e. calculate correctly on signals that vary in length. This is necessary, due to distortions created by movement and is explained in detail in Section 3.2.2. Fig. 3.14 shows the simulation result for signals with varying length. As expected, the matched filter outperforms the other approaches for the default signal length but shows an increased error for both wider

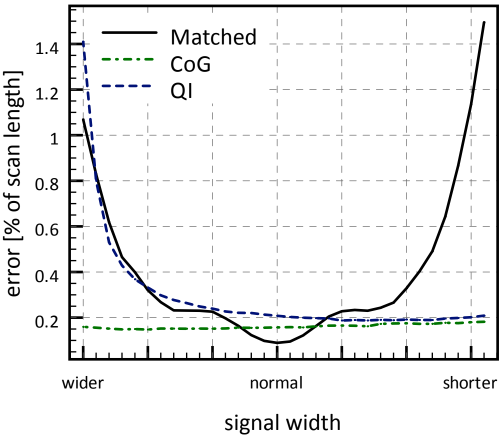


Figure 3.14: Accuracy with changing signal width.

and shorter signals. Quadrature interpolation is robust against shorter signals but shows increased error with wider signals. Only the center of gravity calculation is not influenced by the signal's length. The computation is not repeated for the tracking of edges, as this would be identical to a changing focus.

To conclude, the center of gravity calculation exhibits a good performance and is robust with respect to all required changes of the signals. Thus, it should be used for line scan-based tracking.

3.2.2 Movement distortions

In order to be a major improvement over existing SEM image-based tracking, which treats tracking in a quasi static fashion, line scan-based tracking needs to be able to track moving objects. If an object moves significantly during a line scan, the signal on the line scan is distorted because the relative speed of object and electron beam changes. Only movement components that are parallel to the scanned line lead to such distortions, either expanding or contracting the signal as shown in Fig. 3.15.

The position detection approaches described in Section 3.2.1 can reliably extract the position of the object despite the signal distortions. Furthermore, the measured position on a line scan approximately represents the position at the time at which half the scan is completed independent of the movement's direction (see Fig. 3.15). This is true for movements with constant speed and a good approximation for other movement types. Thus, if the scan starts at time t_1 and ends at time t_2, then the measured position p can be assumed to correspond to the position at time t with:

$$t = \frac{t_1 + t_2}{2} \tag{3.12}$$

Thus, it can be concluded, that in spite of distortions created by movement, the tracking result correctly measures the position at time t. However, there is a speed limit. If an object moves too far between two consecutive line scans, a line scan

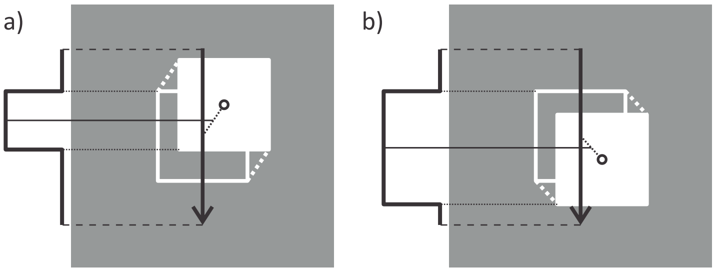


Figure 3.15: Distorted signal during line scan on moving object: Electron beam and object moving into (a) the opposite or (b) the same direction.

might entirely miss the object. In this case, the position cannot be determined and the tracking is lost. To evaluate the latter limitation, a careful analysis of the tracking's timing is required, which is conducted in the next section.

3.2.3 Timing analysis and trackable velocities

In the form described above, the tracking algorithm conducts a horizontal and a vertical line scan. The horizontal line scan determines the x-coordinate of the tracked object whereas the vertical line scan determines the y-coordinate. Although these two line scans can be conducted in rapid succession, they cannot be conducted concurrently. At a given number of pixels, pixel acquisition time and beam settling time, each line scan consumes a time T_L. Furthermore, there is a slight computational overhead T_C from the end of the line scan to the availability of a newly measured position data, which can then be used to calculate the start of the next line scan. Two timing-related design decisions need to be taken.

First, the **reporting** of the tracking result to the robot controller has to be done at a specific point during the tracking. The decision here is whether to report the x- and y-coordinates individually or together. If they are reported together, one of the coordinates has been measured earlier and is thus older than the other. If the controller can capitalize on an individual but quicker reporting of the two coordinates, then this should be preferred.

Second, **pipelining** can be used to remove T_C from the update rate as shown in Fig 3.16. Without pipelining, the next orthogonal line scan is delayed so that it can start at the newly measured position (Fig. 3.16a). With pipelining, each line scan is conducted over a slightly outdated position. This increases the update rate but limits the maximum trackable movement speed, as there is a longer delay from a detected movement to an adjustment of the scanned line (Fig. 3.16b).

Based on whether or not pipeling is used, the tracking system has a different update rate. The update rate f_N without pipelining can be calculated based on the sum of the times T_L and T_C. With pipelining, either T_C or T_L can be omitted, as the two operations can be performed concurrently. Thus, the update rate with pipelining f_P is determined by the longer of these two operations:

$$f_N = \frac{1}{2(T_L + T_C)} \text{ and } f_P = \frac{1}{2 \cdot \max(T_L, T_C)}. \tag{3.13}$$

The trackable speed is limited, as a line scan must not miss the pattern. If a pattern moves too quickly, the line scan can partly or entirely miss the pattern leading to an incorrect result of the position calculation and a loss of tracking. The automatic detection of such a situation is described in Section 3.2.5. The movement can either

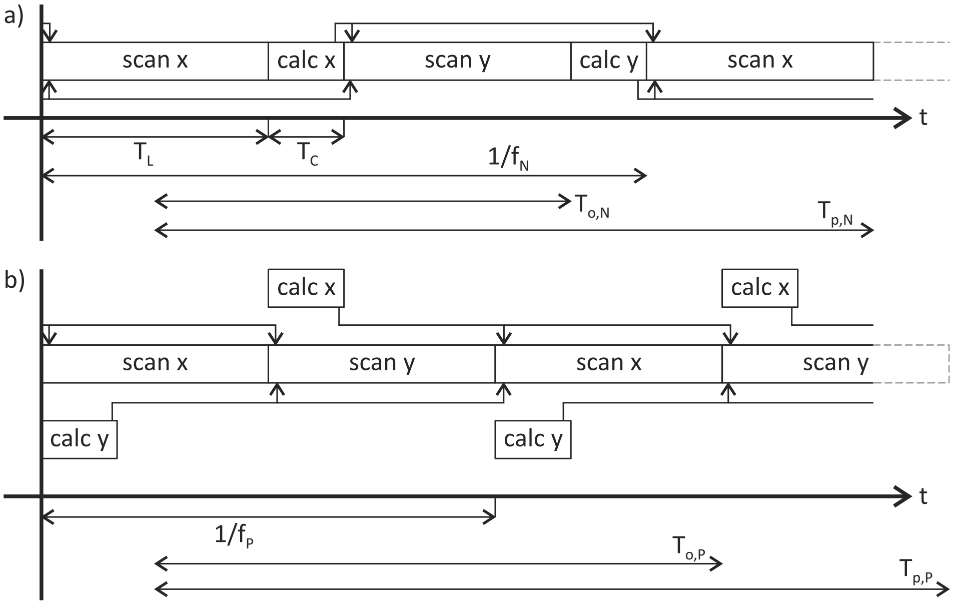


Figure 3.16: Timing of line scanning (a) without and (b) with pipelining.

be too fast in a direction parallel or orthogonal to the scanning path as shown in Fig. 3.17. The pattern has a size s and the length of the line scan l can be chosen. Based on these values, the maximum movement distance d_o and d_p for the orthogonal and parallel movement direction, respectively, can be calculated:

$$d_o = s/2 \tag{3.14}$$

$$d_p = (l - s)/2 \tag{3.15}$$

With the timing diagram given in Fig. 3.16, the exact intervals during which the pattern must not move further than d_o and d_p can be calculated. These intervals are called $T_{o,N}$, $T_{o,P}$, $T_{p,N}$ and $T_{p,P}$ for the orthogonal and parallel cases as well as tracking with and without pipelining (see Fig. 3.16). To determine the interval, the

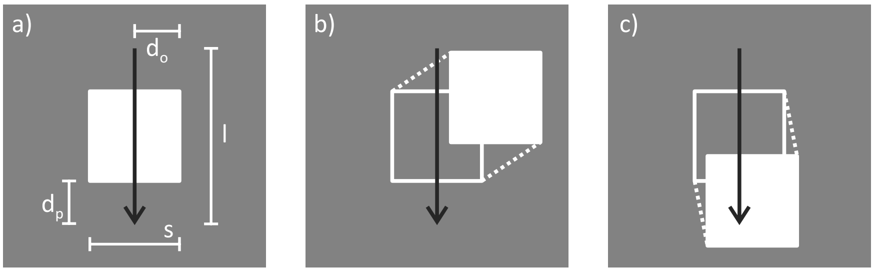


Figure 3.17: a) Maximum movement between scans. Pattern is missed by the line scan, because it is moving to fast in the b) orthogonal or c) parallel direction.

result of a line scan is assumed to represent the pattern's position when half of the line scan is completed (see Section 3.2.2):

$$T_{o,N} = 1.5 \cdot T_L + T_C \tag{3.16}$$
$$T_{o,P} = 3.5 \cdot T_L \tag{3.17}$$
$$T_{p,N} = 2.5 \cdot T_L + 2 \cdot T_C \tag{3.18}$$
$$T_{p,P} = 2.5 \cdot T_L \tag{3.19}$$

In order for the line scan's result to be correct, the pattern must not move further than d_o and d_p until the corresponding line scan's completion. The movement in the parallel direction is not a limitation, if the scan length is chosen long enough. The orthogonal movement provides a stronger limitation for the movement speed, if

$$\frac{d_o}{T_o} \leq \frac{d_p}{T_p} \tag{3.20}$$
$$\Rightarrow l \geq s\left(\frac{T_p}{T_o} + 1\right) = l_{\min}. \tag{3.21}$$

Line scans of length $l_{\min}$ should be used because, according to the definition in Section 3.1.1, a longer line scan contains less used pixels and more potentially used pixels. Thus, the information content of the shorter line scan is higher and the resulting signal will have lower noise.

With the scan length $l_{\min}$, only the movement distance d_o during the time $T_{o,N}$ and $T_{o,P}$ is crucial and the maximum movement speed $v_{\max}$ can be calculated for tracking with and without pipelining:

$$v_{\max,N} = \frac{d_o}{T_{o,N}} = \frac{s}{2} \cdot \frac{1}{1.5 \cdot T_L + T_C} \tag{3.22}$$
$$v_{\max,P} = \frac{d_o}{T_{o,P}} = \frac{s}{2} \cdot \frac{1}{3.5 \cdot T_L} \tag{3.23}$$

Thus, the maximum velocity for tracking without pipelining is 1.4 times (for $T_C = T_L$) to 2.3 times (for $T_C \approx 0$) higher than for tracking with pipelining. In the implemented hardware-based system (see Section 3.5) with its default parameters, the times are $T_L = 400\,\mu\text{s}$ and $T_C = 100\,\mu\text{s}$. Thus, the maximum velocities for a pattern with $s = 1\,\mu\text{m}$ are $v_{\max,N} \approx 0.71\,\text{mm/s}$ and $v_{\max,P} \approx 0.36\,\text{mm/s}$, respectively. The update rate of 1 kHz for tracking without pipelining is only slightly increased to 1.25 kHz by using pipelining. For tracking without pipelining, $l_{\min}$ is approx. 2.7 µm.

To conclude, if a significant computation time T_C is required for each line scan, tracking with pipelining can show superior performance in terms of update rate at the cost of decreased maximum movement speed. If T_C is marginal, the advantages of pipelining become insignificant and pipelining should not be used. This is the case for the developed hardware-based implementation.

Motion prediction

The limitation of the maximum movement speed can be a significant limitation for high-speed motion control, especially if movements as long as several tens of micrometers need to be performed. For example, a 100 µm-long movement is bound to require at least 200 ms at a velocity of 500 µm/s. Furthermore, if nanoscopic patterns or objects are to be tracked, the maximum movement speed is very low.

This limitation can be overcome by incorporating a movement model of the robot into the position tracking. Up to now, the electron beam moves with a constant velocity vector $\mathbf{v}_b$ in the x-y plane during each line scan. If the robot can be assumed to move with a known velocity $\mathbf{v}_r$, this velocity can be added to the beam's movement vector to create the adapted movement vector $\mathbf{v}_b'$:

$$\mathbf{v}_b' = \mathbf{v}_b + \mathbf{v}_r \tag{3.24}$$

If such an adapted beam motion is used for a line scan, $\mathbf{v}_r$ must also be added to the position calculated after a tracking update. Each line scan is assumed to be conducted within a certain time T_L, a certain angle α relative to the tracked pattern and a length l. If the approach uses horizontal and vertical line scans as described above, α equals 0 and $\pi/2$, respectively. The angle α needs to be determined with respect to the tracked pattern and not with respect to the SEM's coordinate system. Thus, after each line scan, the new position $\mathbf{p}(t + T_L)$ of the tracked object can be calculated based on the line scan result R and the estimated motion $\mathbf{v_r}(t)$:

$$\mathbf{p}(t + T_L) = \mathbf{p}(t) + T_L\mathbf{v}_r(t) + R \cdot l \cdot \begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix}. \tag{3.25}$$

An example of such a scanning method is shown in Fig. 3.18. The example shows that lines scanned horizontal or vertically with respect to the tracked pattern become diagonal in the SEM's coordinate system.

A major challenge is, that the robot's actual movement speed $\mathbf{v}_r$ is unknown. However, it can be approximated using two different approaches:

1. Coupling with open-loop controller
2. Assuming a limited acceleration

If the robotic system is adequately modeled and a precise open-loop controller was derived, then this controller can provide a good approximation of $\mathbf{v}_r$ based on the

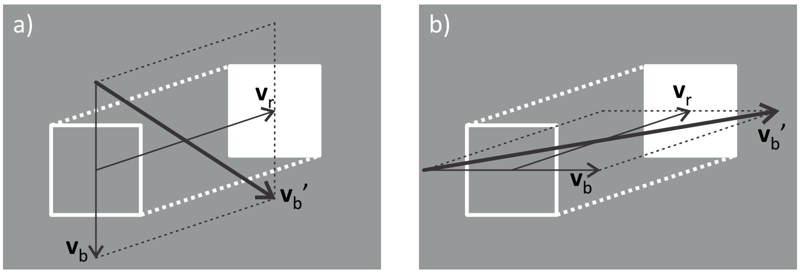


Figure 3.18: Scanning paths adapted to robot movement for both vertical (a) and horizontal (b) scans.

actuation parameters applied to the robot. In this case, the tracking only needs to be able to compensate for the controller's deviations. With a good open-loop controller, there is virtually no limit on the trackable velocities with this approach. However, there are two major downsides. First, there is a tight coupling between the controller and the tracking algorithm. An additional channel must be provided for communication, so that the controller can transfer information to the tracker. This makes the integration of new actuators or controllers tedious. Second, under special circumstances, the open-loop controller can be significantly wrong, e.g. if the robot's movement is restricted by the environment. In such a situation, the tracking would be lost almost instantly and exactly at the time when it is needed the most, i.e. when the robot does not execute the desired motion.

An alternative approach assumes a limited acceleration and deceleration of the robot. If the acceleration of the tracked object is limited, then the tracking can use the position offset detected on a line scan as an indicator for $\mathbf{v}_r$. Thus, if the robot moves with a constant velocity, the tracking will be conducted precisely over the pattern's center, even for high velocities as shown in Fig. 3.18. If the robot's velocity changes during the scan, this will be detected as an offset in the pattern position and the assumed $\mathbf{v}_r$ can be changed accordingly.

A limited acceleration is not a limitation for micro- and nanohandling implementations. Many handling operations require a limited acceleration or even a limited jerk (see Section 2.5.3) to create smooth movements with reduced vibrations. In such applications, the motion controller needs to limit accelerations in any case and the tracking's acceleration limit does not reduce the performance.

The estimated motion of a robot is derived from the result of each line scan. For the following considerations, a non-pipelining approach is used, i.e. each line scan is fully evaluated before the next line scan starts. Initially, the robot is assumed to be stationary, i.e. $\mathbf{v}_r = (0, 0)$. Based on the line scan's result R, the new speed of the robot can be estimated by:

$$\mathbf{v}_r(t + T_L) = \mathbf{v}_r(t) + R \cdot l \cdot \begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix} \tag{3.26}$$

Using this approach, the maximum trackable velocity becomes unlimited. Thus, all speeds of current nanohandling robots, usually up to 20 mm/s, can be used. The electron beam itself is not a limit as it is able to move with a speed of several m/s (see Section 3.5.3). The maximum acceleration a_{max} is calculated similar to v_{max} (compare Eq. 3.22). However, at the maximum acceleration, the distance limit d_o between a line scan and the next orthogonal scan is reached by an accelerated movement with $d_o = 0.5 \cdot a_{\text{max}} \cdot T_o^2$:

$$a_{\text{max}} = \frac{2 \cdot d_o}{T_o^2} = \frac{s}{(1.5 \cdot T_L + T_C)^2}. \tag{3.27}$$

Fig. 3.19 shows an example of the line scans conducted when tracking a diagonal movement with movement prediction. A tracked pattern moves along a 43 µm-long diagonal line with a sine shaped velocity profile. Although line scans are conducted horizontally or vertically with respect to the moving object, they are diagonal in the SEM's coordinate system. The experimental setup and tracking results can be found in Section 5.1.7.

3.2.4 Properties

The line scan-based tracking, even in the basic form described above, has a variety of parameters that can be changed and have a significant influence on its properties. These properties are the update rate f, the maximum movement speed v_{max} and the resolution r. The following considerations apply for tracking a pattern of size $s \times s$ without pipelining. The length of each line scan is assumed to be $l = 2.7 \cdot s$ (see Section 3.2.3).

The update rate f is dependent on the number of pixels n acquired on each line scan, the pixel acquisition time T_P and the beam settling time T_S during which unusable pixels are acquired. In addition, as pipelining is not used (see Section 3.2.3), the calculation time T_C further reduces the update rate (compare f_N in Eq. 3.13):

$$f = \frac{1}{2nT_P + 2T_S + 2T_C}. \tag{3.28}$$

The maximum movement speed is limited if the tracking does not include a movement model of the robot (see Section 3.2.2). As shown in Section 3.2.3, the maximum

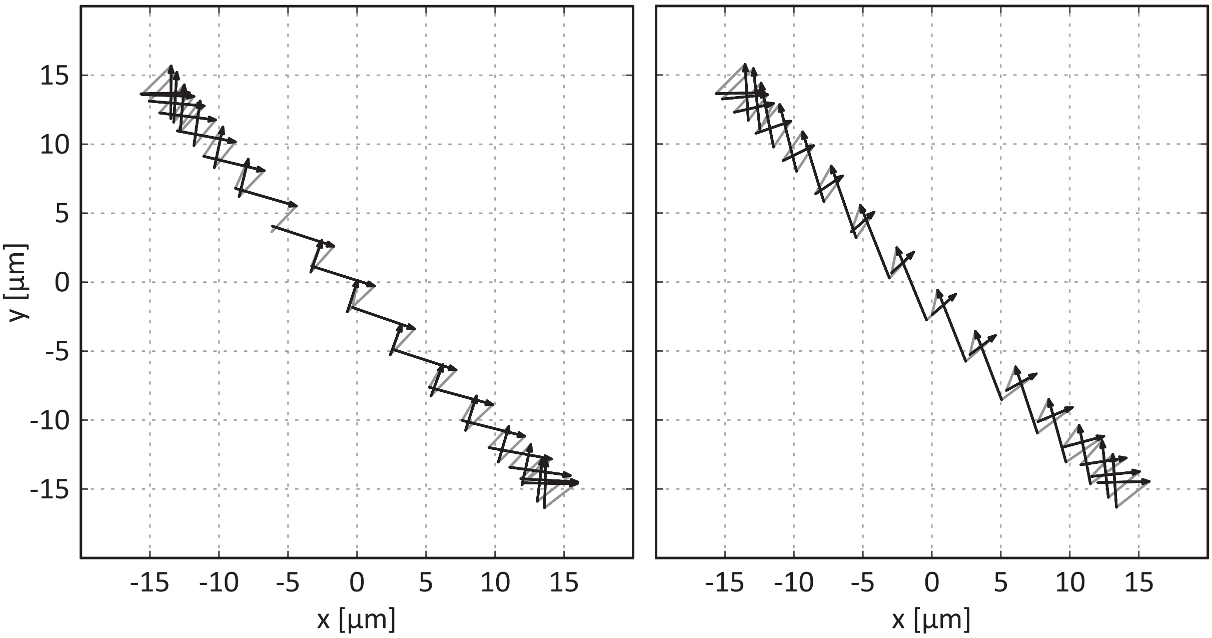


Figure 3.19: Line scans conducted while tracking a 43 µm-long diagonal motion with a sine shaped velocity profile. Line scans are conducted horizontally and vertically with respect to the moving object. Left) Moving top-left to bottom-right, right) reversed motion.

movement without pipelining is (compare Eq. 3.22):

$$v_{\max} = \frac{s}{2} \cdot \frac{1}{1.5 \cdot (nT_P + T_S) + T_C}. \tag{3.29}$$

If the tracking can use the described movement model of the robot, the maximum acceleration $a_{\max}$ is limited. According to Eq. 3.27, $a_{\max}$ can be estimated:

$$a_{\max} = \frac{s}{(1.5 \cdot (nT_P + T_S) + T_C)^2}. \tag{3.30}$$

The resolution r of the tracking is limited by its noise. The noise of the tracking in turn is influenced by the pixel noise and the number of pixels N that contribute to the position calculation. Additionally, as the noise is scaled with the scan length and thus the pattern size s, a smaller pattern leads to a higher resolution, if the number of pixels is kept constant. Thus, for an unknown constant coefficient K:

$$r = K \cdot \frac{I_S \cdot N \cdot T_p}{s}. \tag{3.31}$$

A higher specimen current I_S, reduces the noise of each pixel and thereby the noise of the measured position. However, it also leads to a wider focus diameter and thus a reduction of the achievable imaging resolution. This in turn increases the depth of focus d_f. To reduce the depth of focus, e.g. for focus-based depth estimation (see Section 3.4), the aperture angle α can be increased, either using a wider aperture or a smaller working distance.

All parameters and their effect are summarized in Table 3.2.

Parameter		**Effect on**			
name	symbol	f	$v_{\max}/a_{\max}$	r / SNR	d_f
pattern size	s		↑	↓	
pixels on line scan	n	↓	↓	↑	
pixel acquisition time	T_P	↓	↓	↑	
specimen current	I_S			↑	↑
aperture angle	α				↓

Table 3.2: Important tracking parameters.

3.2.5 Detecting a loss of tracking

Reliability is a key feature of a nanohandling implementation. Therefore, the tracking's limitations, e.g. the limited acceleration, should be taken into account so that situations where the tracking fails do not occur. However, if such a loss of tracking occurs, this failure should be detected quickly, so that all movements can be stopped, the tracking can be reinitialized, and the handling can resume from a well-defined state. Otherwise, the tracking would report invalid positions of the robot leading to an unpredictable behavior.

If a line scan does not scan the pattern as expected, this results in a sudden change of the normalized variance of the line scan's signal. Thus, for each line scan, the normalized variance v is calculated. For N pixels p_i, v is defined as:

$$v = \frac{1}{\mu} \sum_{i=0}^{N} (p_i - \mu)^2, \tag{3.32}$$

with μ being the average pixel value:

$$\mu = \frac{1}{N}\sum_{i=0}^{N} p_i. \tag{3.33}$$

In order to detect a loss of tracking quickly, v should be calculated in hardware during the acquisition of each line scan similar to the position detection (see Section 3.5.2). To calculate the normalized variance efficiently in a hardware-based system, Eq. 3.32 needs to be reformulated so that μ is not needed during the line scan:

$$\begin{aligned}
v &= \frac{1}{\mu}\sum_{i=0}^{N}(p_i-\mu)^2 = \frac{1}{\mu}\sum_{i=0}^{N}(p_i^2 - 2p_i\mu + \mu^2) && (3.34)\\
&= \frac{1}{\mu}\sum_{i=0}^{N} p_i^2 - 2\sum_{i=0}^{N} p_i + N\mu && (3.35)\\
&= \frac{1}{\mu}\sum_{i=0}^{N} p_i^2 - 2\sum_{i=0}^{N} p_i + \sum_{i=0}^{N} p_i && (3.36)\\
&= \frac{1}{\mu}\sum_{i=0}^{N} p_i^2 - \sum_{i=0}^{N} p_i = \frac{N\cdot S_q}{S_p} - S_p && (3.37)
\end{aligned}$$

Thus, as the number of pixels N is known, only the sum of the pixels S_p and the sum of the pixels' squares S_q have to be calculated. Both values can easily be computed in hardware during a line scan by adding each new pixel value p_i and its square p_i^2 to two different accumulators.

With this calculation, the loss of tracking can be detected by comparing the new variance value of each line scan with a corresponding reference. The reference is calculated by averaging the variance values of the previous tracking iterations on the same line scan. Several tracking experiments have shown that, despite changing focus and movement distortions affecting the normalized variance, the variance does not rapidly change by more than 50% as long as the tracking is not lost. Thus, a tracking loss is reported, if the normalized variance varies more than 50% from the calculated reference.

3.3 Scanning sequences

The approach as described above is capable tracking a square-shaped pattern by scanning the electron beam in two straight lines. In this section, the approach is analyzed in terms of the efficiency criteria described in Section 3.1.1. Other pattern shapes are proposed that facilitate increased efficiency, better properties or tracking in more degrees of freedom. Furthermore, the approach is extended to track multiple patterns and to track objects directly without using patterns.

Tracking the edge of a pattern has almost identical characteristics to tracking the center of a pattern (see Fig. 3.20a and b). The tracking can operate successfully, if the pattern is fully captured by the scanned line. The maximum movement distance d_o orthogonal to the line scan is defined by the pattern's shape. Over the entire distance d_o, the pattern must remain identical. Due to the so-called aperture problem, a shape deviation would also be detected as a movement along the scanned line. Thus, for shape deviations or other objects in the vicinity of a pattern, d_o is reduced (see Fig. 3.20c). The distance d_p parallel to the scanned line is given by the length of the scanned line itself. Scanning the center of a pattern enables scanning arbitrary symmetrical patterns (see Fig. 3.20d) and provides a better signal-to-noise ratio than the tracking of an edge (see Fig. 3.12).

In terms of efficiency, 30% to 50% of the acquired pixels can be actively used during a tracking iteration (see Fig. 3.20a and b). About 5% of the pixels is unusable when rapidly moving from one line scan to the next. The remaining pixels are potentially used pixels which have to be acquired to allow for pattern movement. This already is a significant improvement to image-based tracking with a 10% upper bound on efficiency (see Section 3.1.3). However, the pattern shape and scan path can be changed in order to improve the efficiency and other tracking properties.

3.3.1 Tracking a pattern

Tracking based on SEM line scans works most efficiently if the tracked object is well-known and has certain key properties. Section 3.2 introduced a first method of tracking a square-shaped pattern in two degrees of freedom. This approach is straightforward, but it also has downsides and can be improved with respect to several considerations. Both the electron beam path and the pattern shape can be altered in order to improve the tracking's performance.

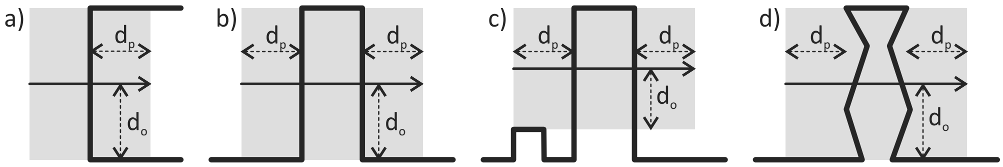


Figure 3.20: Analysis of edge and center-based tracking: a) Tracking an edge, b) tracking a center, c) limits of movement and d) center tracking of symmetrical patterns.

First, with the basic approach, two rapid beam movements are required for a position detection. Each of these movements requires a certain amount of settling time. During this time, unusable pixels are acquired decreasing the efficiency by about 5%. Thus, a continuous beam path is preferred. A pattern shape that allows for such a continuous path is shown in Fig. 3.21. The x- and y-coordinates are measured alternately while the beam follows a square shape. Due to manufacturing differences in the pattern, there is always a difference between positions detected on lines A and C as well as B and D. However, these differences can be measured while the pattern is not moving and programmed as a fixed offset for a given pattern. Furthermore, the tracking calculation has to take the scan direction into account, as the opposing line scans are conducted into opposite directions. This can be incorporated into the tracking as a different angle α in Eq. 3.25. Such a cross-shaped pattern increases the tracking efficiency as in makes rapid beam movements unnecessary.

Second, certain applications might require the measurement of in-plane rotation. This can also be solved with the scanning sequence shown in Fig. 3.21. If the robot performs purely translational movement, the offset on the two opposite line scans are identical (Fig. 3.21b), taking into account the difference in measurement time (see Section 3.2.3). Rotational movement leads to opposite position readings from the two horizontal or vertical line scan pairs (see Fig. 3.21c).

Third, the size of a pattern influences the relation between resolution and maximum acceleration (compare Section 3.2.4). To further improve the control performance, a pattern could be tracked with long line scans for coarse positioning to obtain high trackable accelerations while accepting a limited resolution. When the robot has decelerated and is close to the final position, shorter line scans could be used to allow for high resolution. The pattern and scanning sequence described above can already be used in a scale-invariant fashion by making the pattern bigger and

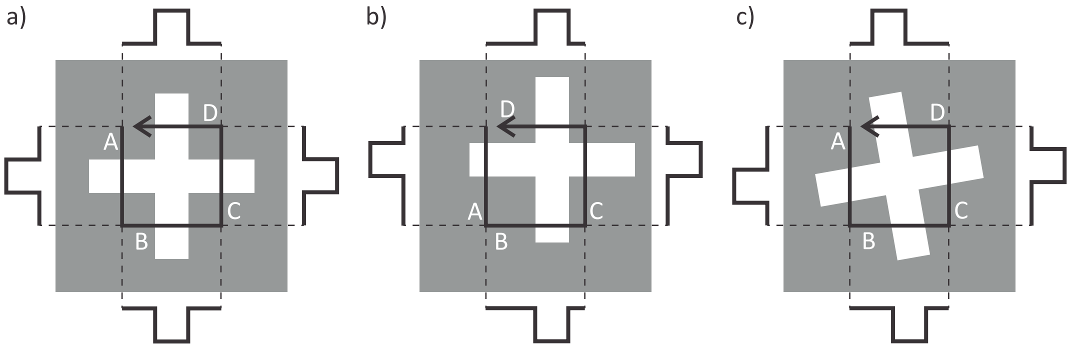


Figure 3.21: Continuous scanning of a square path: a) Scanning path, b) translational movement leads to identical offsets on the two opposite line scans and c) rotational movement leads to opposite values on opposite line scans.

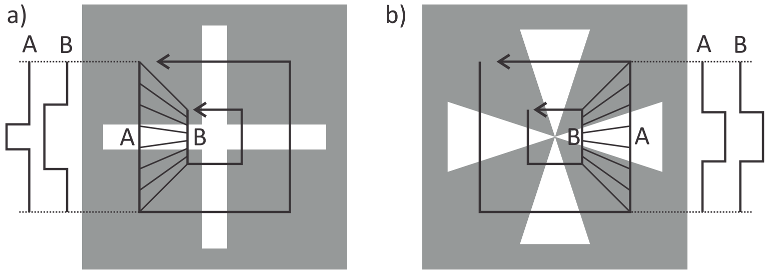


Figure 3.22: Scale-invariant pattern tracking.

changing the size of the square-shaped beam path (see Fig. 3.22a). Thus, as the same number of pixels are acquired on the line scans A and B, line scan A focuses on accepting high accelerations, whereas scan B provides a higher resolution. The pattern in Fig. 3.22a, however, is not ideal for scale-invariant tracking because it has a limited minimum size and the actual amount of pixels belonging to the pattern is very low for long scans. This means that the amount of potentially used pixels to used pixels is low. Fig. 3.22b shows a pattern better suited for scale-invariant tracking. The minimum size is only limited by the possibilities of pattern fabrication and the detector signal remains identical independent of the size of the scanned square.

To conclude, different patterns can improve the tracking efficiency, enable the tracking of more degrees of freedom or facilitate scale-invariant tracking. A pattern should be chosen according to the requirements of the handling operation.

Pattern fabrication

The fabrication of patterns for line scan-based tracking can be done with a variety of technologies. For tracking, a pattern should exhibit a high contrast, i.e. large signal in order to have a high signal-to-noise ratio. Thus, the difference in the detector signal dependent on whether the electron beam hits the pattern or not needs to be high. The method of how to create a pattern that generates a large signal depends on the detector type. Two detector types are readily available in most modern SEMs: The secondary electron (SE) detector and the backscattered electron (BSE) detector.

For the SE detector, a high contrast is achieved by using high aspect ratio structures. A very promising and also simple to manufacture solution is the milling of deep cavities using a focused ion beam. While secondary electrons are easily emitted by the surroundings of this cavity, the cavity itself does barely allow for any secondary electron to escape. Thus, pixels inside the cavity lead to a very low detector value, whereas pixels outside the cavity lead to higher values. Fig. 3.23a shows an example

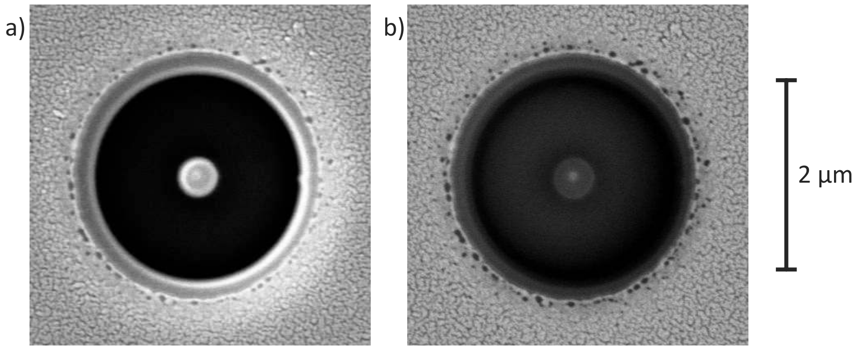


Figure 3.23: Pattern created by focused ion beam-milling: a) Secondary electron and b) backscattered electron image.

of such a pattern. A 2 µm deep circular groove is milled into a silicon surface leaving a central column. For the SE detector, the central column produces a bright signal whereas the surrounding groove barely emits any secondary electrons. In contrast such a structure only leads to a small signal on the BSE detector (see Fig. 3.23b).

Structures with a high aspect ratio can be milled with an FIB or also produced by etching during the batch fabrication process of tools. Complex shapes needed for tracking more degrees of freedom can be created. Fig. 3.24 shows an electrostatic gripper decorated with a variety of FIB-milled patterns. A simple pattern is a square-shaped hole but also the shapes shown in Fig. 3.22 have been realized. Note that the patterns are created in a way to appear undistorted in spite of the gripper be tilted by about 30 degrees.

The BSE detector exhibits a high contrast when scanned over different materials (see Section 2.2.3). Thus, a pattern can be created by depositing a material or by removing a thin film of a material to expose the material underneath. Although the achievable contrast is not as high as the contrast for high aspect ratio structures imaged with the SE detector, there are two advantages. First, as only a very thin film of material needs to be added or removed, the shape and function of the tracked tool or object are barely influenced. Thus, patterns can even be created on very thin objects or delicate membranes. Second, the backscattering of electrons is constant over flat surfaces of the same material. Thus, even large patterns can be created with a well-defined shape. This not possible for the SE detector as edges lead to a larger signal. The difference is shown in Fig. 3.25. The BSE image shows almost constant pixel values on the gold pattern, whereas the SE image shows a strong highlight on the pattern's edges. The corresponding line scan results in an almost ideal trapezoidal signal observed by the BSE detector (see Fig. 3.25b). Note that the same effect is visible in Fig. 3.23, but the highlighting of the edges is insignificant compared to the contrast created by the high aspect ratio.

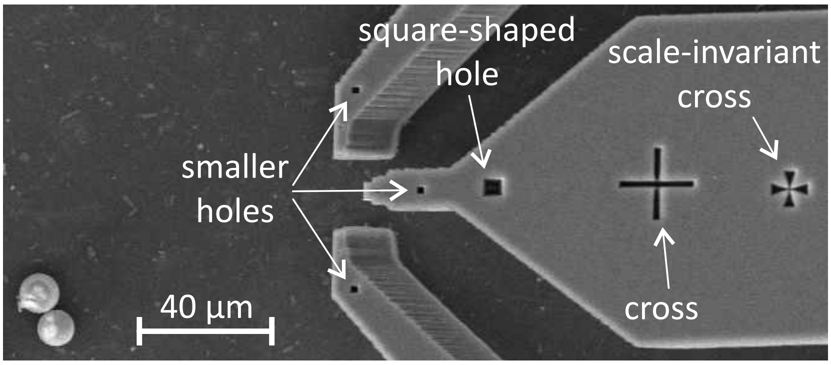


Figure 3.24: SE detector image of electrostratic gripper (Chen et al., 2009) decorated with a variety patterns using a focused ion beam.

3.3.2 Tracking multiple patterns

With line scan-based tracking, it is possible to track multiple patterns and objects at the same time. The SEM beam can only perform one line scan on one object at any time, but due to the high speed of the electron beam, multiple patterns can be tracked alternately in rapid succession. Tracking multiple patterns is useful in a variety of scenarios. First, tracking two patterns on the same object with a significant distance leads to a highly accurate rotational measurement. Second, tracking multiple patterns on the same object can provide redundancy in case a pattern is rendered useless, e.g. by occlusions. Third, if one pattern is located on the end of a flexible structure and the other on a fixed reference, the structure can be used for tracking-based force measurements. Lastly, multiple robots can be concurrently used to combine their movement capabilities, e.g. for increased speed or more degrees of freedom.

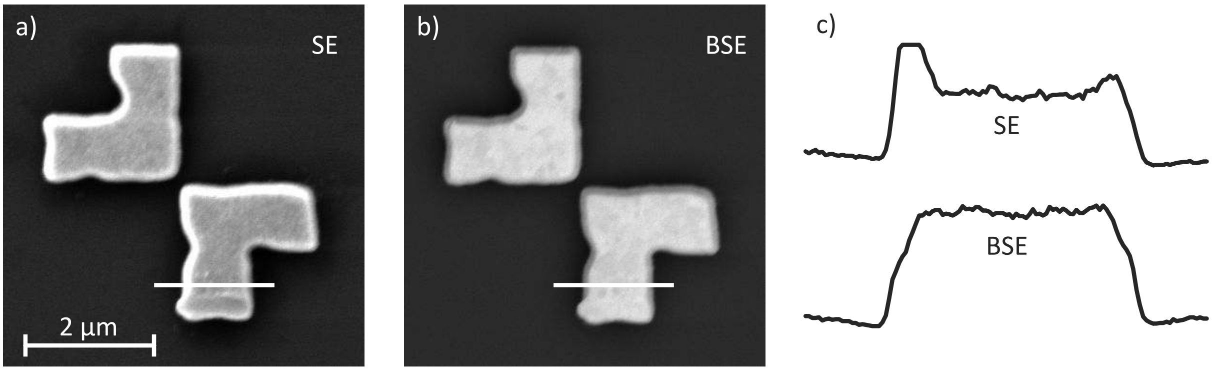


Figure 3.25: Comparison of SE and BSE detectors when imaging a gold pattern deposited on a silicon surface: a) SE image, b) BSE image and c) pixel values of the shown line scan.

If multiple patterns are tracked, the tracking performance of one pattern is reduced. First, rapid beam movements are unavoidable when changing between the patterns. Second, the update rate calculated by Eq. 3.28 is divided by the number of patterns. Third, the time between a line scan and its next orthogonal line scan changes significantly. Thus, the maximum trackable velocity or acceleration is reduced. The time to the next parallel line scan also changes, but this can be compensated for by performing longer line scans. The amount of reduction depends on the scheduling of the line scans. There are two basic strategies for this scheduling (see Fig. 3.26):

- Perform all line scans of one pattern then switch to the next.
- Perform one line scan on each pattern then the next.

Performing all line scans of one pattern before switching to the next pattern has the advantage that, for patterns that allow for a continues beam motion (see Section 3.3.1), as few rapid beam movements as possible are used. Thus, this approach maximizes the efficiency of the tracking. However, a major downside is that the time T_o between a line scan and the next orthogonal line scan on the same pattern is long. In Fig. 3.26a, the time between completing scan 2 and starting scan 1 is long, as the entire second pattern is scanned in between and this time limits the movement speed. For the following considerations, two simplifications are used (compare Section 3.2.3). First, all patterns are assumed to have the same size, i.e. d_o is identical for each pattern. Second, each line scan is assumed to return the pattern's position at the time of the scan's completion. With these simplifications, the maximum movement speed $v_{\max}(n)$ for an arbitrary number n of patterns is:

$$T_o(n) = (2n-1) \cdot T_L + (n-1) \cdot T_C \quad (3.38)$$

$$v_{\max}(n) = d_o \cdot \frac{1}{T_o(n)} = \frac{s}{2} \cdot \frac{1}{(2n-1) \cdot T_L + (n-1) \cdot T_C} \quad (3.39)$$

Note that the computation time T_C is added only once per pattern as only the calculation for its first scan needs to be done in sequence. The computations for the second scan can be done in parallel as they do not influence the following scan,

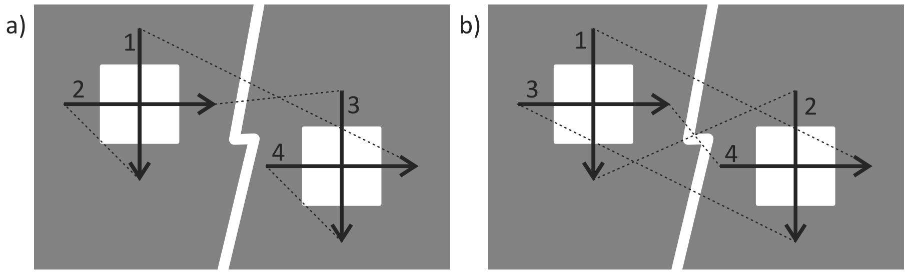


Figure 3.26: Beam path when tracking two patterns: a) Consecutive patterns. b) Consecutive line scans.

i.e. pipelining can be used without negative effects. However, when tracking two patterns with this scheduling, the maximum movement speed is almost divided by 3 in comparison to tracking a single pattern. When tracking three patterns or four patterns, it is divided by approx. 5 and 7, respectively. Thus:

$$v_{\max}(n) \approx \frac{v_{\max}(1)}{2n} \tag{3.40}$$

This is solved by scheduling one line scan on each pattern before scheduling the next. For the case shown in Fig. 3.26b, the time between scan 1 and 3 is identical to the time between 3 and 1. The maximum speed is:

$$\begin{aligned} T_o(n) &= n \cdot T_L && (3.41) \\ v_{\max}(n) &= d_o \cdot \frac{1}{T_o(n)} = \frac{s}{2} \cdot \frac{1}{n \cdot T_L} && (3.42) \end{aligned}$$

The computation time T_C can be entirely omitted, as any two consecutive scans do not measure the same pattern. Thus, the computation can always be done in parallel. For this scheduling, the maximum movement speed of tracking a single pattern is simply divided by the number of patterns, i.e.:

$$v_{\max}(n) \approx \frac{v_{\max}(1)}{n} \tag{3.43}$$

If a model of the robot's motion is used, higher velocities can be used and only the maximum acceleration is limited. The calculation of the maximum acceleration is done similar to Eq. 3.27.

3.3.3 Tracking objects

Most nanohandling scenarios can be solved by only tracking patterns. Even if the involved nanoobjects are arranged randomly on a substrate, the substrate itself could be manufactured with a regular grid of patterns. If a nanoobject needs to be moved to a specific location, it can first be located in an image, measuring the distance to the closest pattern. Then, the pattern could be positioned with the corresponding offset. If the patterns are arranged in a close grid, e.g. a pattern is available every 50 µm, thermal drift and other uncertainties have a negligible influence due to the small distance between object and pattern. However, this approach is not always applicable. Some manufacturing processes might not allow for the structuring of patterns on the substrate. More importantly, if the nanoobject's movement needs to be monitored during the operation, e.g. during an assembly operation, the tracking of patterns in the vicinity cannot provide the necessary information. In these cases, objects need to be tracked directly.

There are several image-based tracking approaches that rely on the detection of edges. Examples are edge-based active contours and rigid-body tracking (compare Section 2.4.1). These approaches can be directly used as line scan-based tracking. Instead of first acquiring an image and then detecting the object's edges on multiple virtual lines in the image, lines are directly scanned at characteristic locations. Fig. 3.27 shows examples of tracking a rectangular shape with an active contour. With each update, the contour (dashed line) is update by detecting the edges on several orthogonal measurement lines (solid lines). For image-based tracking, the measurement lines are extracted from the image's pixel values. However, a similar result can be obtained by performing line scans along these measurement lines. In this case the benefit in terms of tracking efficiency becomes obvious. For the image-based tracking, only the pixels on the measurement lines are considered for tracking, although all pixels of the image are acquired. For line scan-based tracking, all pixels apart from a few unusable pixels during rapid beam movements are used.

A key challenge is the selection of edges that are best suited for tracking. The goal of the edge selection is to maximize the possible movement distances d_o and d_p (see Fig. 3.20), i.e. to select a straight edge in large clear area. Such edges lead to a robust tracking with a high trackable acceleration. Furthermore, to successfully track an object in two dimensions, it is necessary to find two edges that are as close to being orthogonal as possible. Otherwise, two degrees of freedom become coupled which amplifies the tracking's noise. In contrast, two line scans along parallel edges with a significant offset are best suited to measure rotational movement. Other considerations include that edges should be selected that are not prone to being occluded or changed during the actual nanohandling sequence.

Fig. 3.28 shows examples of edges that could be used to track the electrostatic gripper shown in Fig. 3.24 without using patterns. Using filters such as the Canny filter (Canny, 1986) to highlight an object's edges (see Fig. 3.28b) helps selecting edges. Instead of tracking the created patterns with the line scans A, the gripper can also be tracked by other combinations of line scans. The combination of the

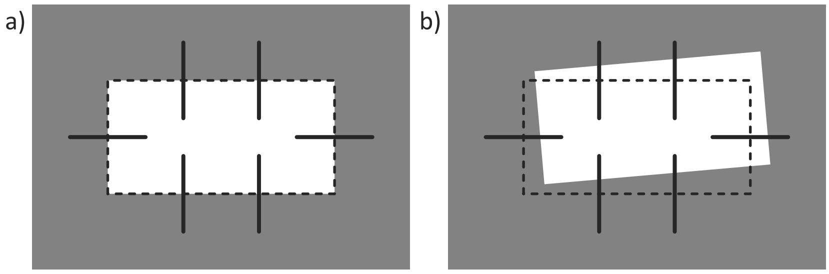


Figure 3.27: Tracking the contour of an object: a) Measurement lines (solid) are scanned orthogonally to contour (dashed line). b) Movement is detected on measurement lines to update contour.

scans B and C is well-suited for a two-dimensional tracking allowing high movement speeds as the gripper can move considerably between updates without risking a tracking loss (see Fig. 3.20). Other Combinations for two-dimensional tracking are the pairs (G, F) and (H, I). The latter, however, can be problematic for handling operations during which the gripper is used as edges F or H can change significantly. Using B' in addition to (B, C) can be used to measure the gripper's rotation. The gripper's opening can be measured by the distance between D and B. In summary, there is a variety of edges that can be used for tracking. Choosing appropriate edges should be based on the requirements on a specific nanohandling operation.

There are several special cases, in which tracking is possible, even though the selection of edges is not as straightforward as outline above. The first case are symmetrical objects without straight edges, such as the microspheres shown in Fig. 3.29a and b. As the center of a symmetrical object is tracked, both line scans A and B are invariant to orthogonal movement. In fact, even if the objects are not perfectly symmetrical, tracking is still possible as will be shown in Section 5. The second special case is the tracking of very thin, almost one-dimensional structures such as CNTs (see Fig. 3.29c and d). Here, no two long straight edges with a significant angular difference can be found. Line scan B can determine the vertical movement of the CNT easily, but there is no edge to determine the horizontal movement. A solution is to use line scan A, although the tracked edge is only 100 nm to 200 nm long. Thus, the maximum acceleration for tracking such nearly one-dimensional objects is highly limited. A second vertical line scan B' can be used to track rotations.

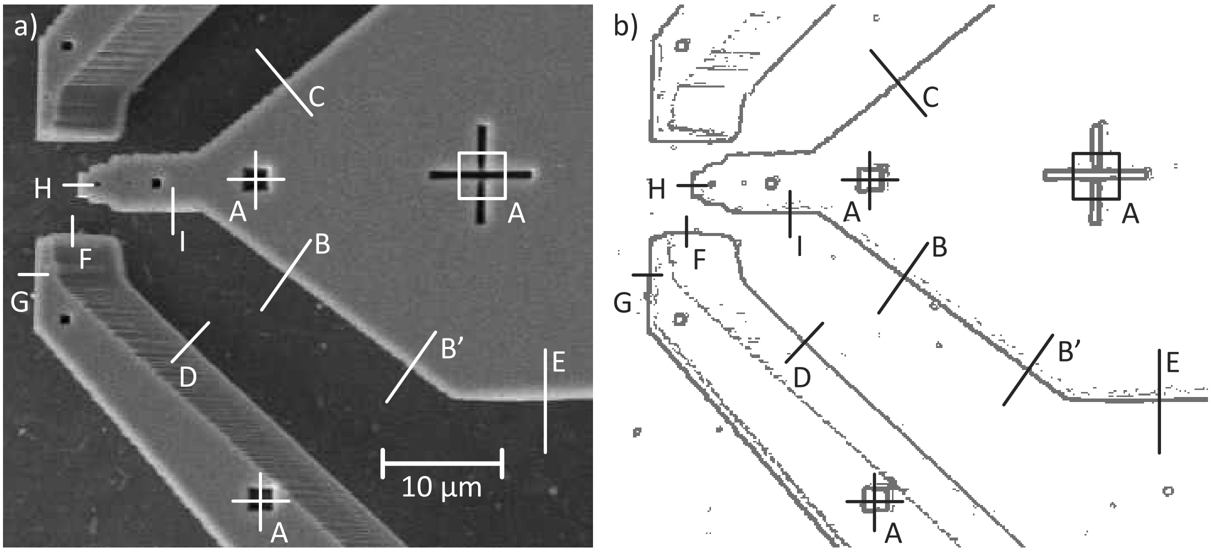


Figure 3.28: Directly tracking a microgripper: a) SEM image with examples of line scans and b) edges detected with Canny filter.

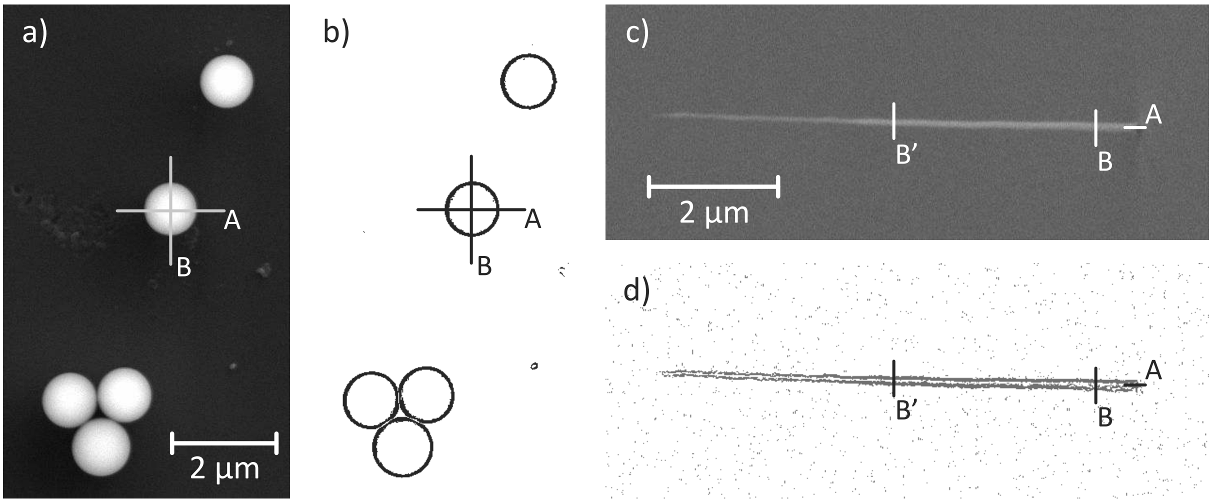


Figure 3.29: Special cases for direct tracking: a) and b) tracking microspheres, c) and d) tracking a CNT.

To conclude, the tracking of a variety of objects is possible although careful consideration is required in special cases. So far, the tracking can only be used to determine the position of an object in a plane, i.e. movements along the x- and y-axes as well as rotations. Although a major part of handling scenarios can be solved using such two-dimensional tracking, some scenarios require information about the third dimension.

3.4 Three-dimensional tracking

As described in Section 2.4.2, multiple ways have been considered to obtain three-dimensional information with the SEM. This work focuses on combining line scan-based tracking with focus-based depth measurements for two reasons. First, focus-based depth measurements are universally applicable, i.e. can be done in any SEM with any robotic setup. Second, the improved efficiency of line scan-based tracking significantly improves the performance of focus-based measurements as shown below.

This section first shows the benefits of line scan-based focus evaluation in comparison to focus evaluation in images. Then, a method for aligning two objects with high accuracy is described. The method uses the depth from defocus approach for coarse alignment and the depth from focus approach for fine alignment (see Section 2.4.2). Lastly, SEM parameters with influence on the alignment speed and precision are described and typical values are given.

3.4.1 Focus evaluation based on line scans

Section 3.2.5 describes how to calculate the normalized variance for each line scan. As the normalized variance is a common focus measure, the obtained value can be directly used for focus evaluation. Using line scans tackles the four challenges of focus evaluation based on SEM images:

- Large depth of focus
- High level of noise
- Lack of features
- Non-monotonic focus measure

The SEM's large **depth of focus** leads to a broad peak of the used focus measure, e.g. the normalized variance (see Section 2.2.2). Thus, SEM parameters should be chosen to make the depth of focus as small as possible (see Section 3.4.4). This however restricts other robot movements, as objects have to be closer to the focus plane to be clearly visible. In contrast to most image-based tracking algorithms, however, line scan-based tracking alleviates this downside because it is robust against changing focus. Objects can still be tracked even if they are defocused, i.e. not located in the focus plane.

There is significant **noise** in SEM images compared to optical imaging modalities. This noise prevents using gradient-based focus measures and affects static-based focus measures (Sun et al., 2005). With the increased efficiency of line scan-based tracking, a higher ratio of the acquired pixels is used for tracking and focus evaluation. Thus, a high percentage of pixels contains used information and the signal-to-noise ratio is increased.

All sharpness measures rely on evaluating **features** visible in an image, e.g. edges or texture. If there are no features, the focus cannot be determined. For micro- and nanoobjects, this is a significant problem as they lack distinct features. For example, the 2 µm-sized particle shown in Fig. 3.30a has an almost uniform surface so that only the edges can be used to determine sharpness (see Fig. 3.30b). The same holds true for the spheres and CNTs shown in Fig. 3.29. Thus, only about 1% of the pixels in an SEM image can actually be used for sharpness evaluation (dark pixels in Fig. 3.30b). For line scan-based tracking, this ratio can be significantly increased as the line scans are conducted exactly over known features and there is a predetermined amount of features on each line scan.

Furthermore, features in images can be located at different heights, even if they belong to the same object. An example is the CNT shown in Fig. 3.33. For height differences smaller that the depth of focus, this makes the focus peak wider and limits the resolution of the depth measurement (compare Section 2.4.2). For even larger differences, it leads to multiple peaks on the focus curve and thus a **non-**

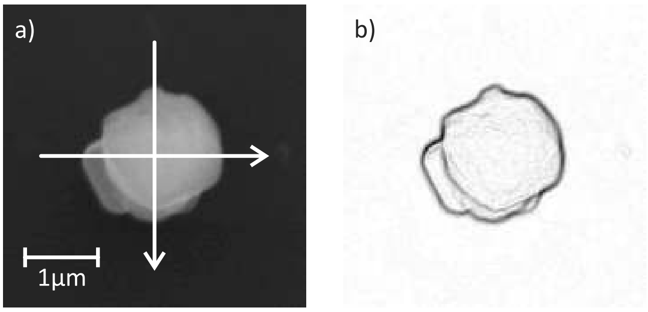


Figure 3.30: Tracked particle. a) SEM image with electron beam path used for direct tracking. b) Information content of each pixel with respect to image focus; darker pixels representing more information.

monotonic relation between the focus measure and the distance to the focus plane. Thus, the depth from defocus approach becomes infeasible. However, the features of patterns tracked by line scans are flat. Even if objects are directly tracked (see Section 3.3.3), each edge is located at one specific height. Thus, the focus measure on each line scan exhibits a narrow peak and a monotonic decrease with degree of defocus.

A detailed analysis of the focus resolution and signal-to-noise ratio that can be achieved with line scan-based depth measurements will follow in Section 5.1.9. It will be shown that steps as small as 250 nm lead to a detectable step in the measured variance value. However, a high resolution focus measure alone does not provide the means for an accurate height measurement or alignment, as there is no fixed relation between the focus measure and an absolute or relative height.

The ultimate goal of three-dimensional tracking is the alignment of multiple objects in all degrees of freedom. As described in Section 2.4.2, there are two basic approaches to align objects in three dimensions: Depth from focus and depth from defocus. With the increased effectivity of calculating focus measures on line scans, these methods can be refined and allow for accurate alignment. This alignment is subdivided into coarse and fine alignment. During coarse alignment, a height measurement is returned for each line scan using the depth from defocus approach, so that fast visual servoing towards a target height can be implemented. The depth from focus approach can then be used as it is slower but has the accuracy required for fine positioning.

3.4.2 Depth from defocus

The depth from defocus approach exploits the monotonic decrease of the focus measure with increasing defocus. A full focus sweep is shown in Fig. 3.31. The working distance is changed in 1 µm steps and the resulting value for the variance is shown. The object is located at the peak of this curve, i.e. at a working distance of 8.033 mm. Close to this working distance, a focus reading cannot be used to determine the height of an object, because a similar result is obtained anywhere within the depth of focus d_f. However, in the defocus area, a linear behavior can be observed and the 1 µm steps of the working distance are clearly resolved.

To implement the depth from defocus approach, an initial focus sweep is performed resulting in a curve similar to Fig. 3.31 and a line is fitted into the linear defocus interval. This line leads to a linear equation for the z-position h of the tracked object in relation to its variance value v:

$$h(v) = 8.024\,\mathrm{mm} + (v - 825000) \cdot \frac{10\,\mu\mathrm{m}}{300000} \tag{3.44}$$

The initial working distance is set to the center of the linear defocus area, e.g. working distance 8.024 mm in Fig. 3.31, and the tracking is started. Each line scan results in a new value of the normalized variance. By inverting Eq. 3.44, this variance value is converted into a height. To allow for continuous tracking over large height ranges, the SEM's working distance setting can be changed to keep the relative distance between object height and focus plane constant. Thereby, the focus measure is ensured to remain in the linear defocus region.

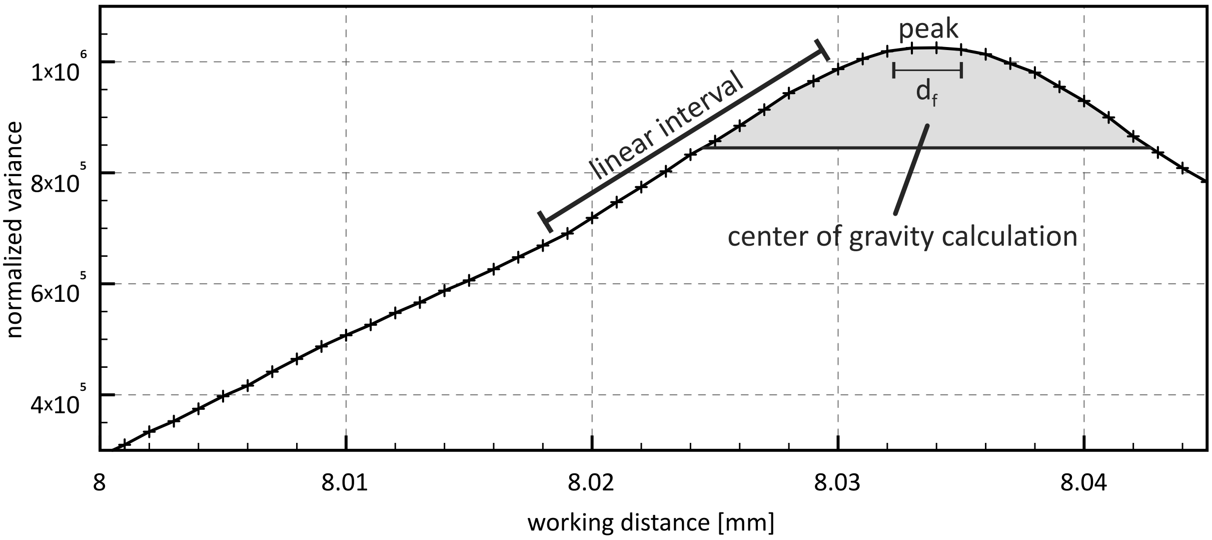


Figure 3.31: Normalized variance during a focus sweep. The linear interval is used for depth from defocus whereas the center of gravity calculation is used for depth from focus.

A challenge for this approach is that changes to brightness and contrast, i.e. amplitude and offset of the detector signal, lead to a change in the calculated normalized variance. Line scan-based tracking, however, provides a way to keep these conditions constant by adjusting the SEM's settings to keep the minimum and maximum pixel values constant. A further description of this process is given in Section 3.5.4.

3.4.3 Depth from focus

For the depth from focus approach, the working distance setting of the SEM is swept through the objects of interest and a curve of the normalized variance is recorded for each tracked object. Then, the center of gravity calculation approach presented above is used to determine the location of the center of the normalized variance peak for each object (see Fig. 3.31). The threshold value is chosen as 80% of the maximum value so that a sufficient amount of values is used but the curve is still symmetrical. The symmetry is essential for the center of gravity calculation to correctly identify the center of the peak.

For alignment, the detected difference between two objects can be corrected. Robots with internal sensors can use closed-loop control. Other robots can use depth from defocus based on a line fitted into the newly acquired focus sweep. However, similar to positioning along the x- and y-axes, neither the measured distance nor the executed robot movement can be assumed to have an arbitrarily high precision. Thus, the combination of focus sweep and robot movement needs to be repeated until the objects are at the same height. This implements a form of look-then-move control.

The depth from focus approach is more time-consuming than depth from defocus for two reasons. First, it needs to acquire a significant amount of samples along the focus sweep in order for the center of gravity calculation to correctly identify the peak. Second, the working distance needs to be changed repeatedly during this sweep. On most SEMs, it is possible to change the working distance using a remote interface (see Section 3.5), but this process is time consuming and the precise timing of the focus change is unknown. Thus, to synchronize the working distance change with the tracking updates, a certain amount of settling time is required until the working distance change is complete. However, as will be shown by validation measurements, it can be used to accurately align objects with an accuracy of about 100 nm. It is the only focus-based method suitable for alignment on the nanoscale.

3.4.4 SEM parameters for depth estimation

In order to maximize the resolution of a depth estimation based on focus, the SEM needs to be used in a way so that the depth of focus d_f is minimized. As described in Section 2.2, d_f depends on the aperture angle α and the resolution δ (see Eq. 2.2). Thus, to obtain a small d_f, the resolution δ should be as high as possible, i.e. δ should have a small value, and the aperture angle α should be as large as possible.

Fig. 3.32a sketches the influence of the two parameters α and δ on the curve of the normalized variance. A lower resolution, i.e. higher δ, leads to a widening of the peak, whereas a lower α leads to a gentler slope. Both changes limit the accuracy of focused-based depth measurements, because the peak of the noisy variance signal can be determined less precisely. Both depth from focus and depth from defocus use the linear defocus areas to obtain a highly accurate depth measurement. Thus, α is the main parameter that influences the accuracy of the depth measurement.

The aperture angle α in turn is influenced by the working distance and the diameter of the final aperture (compare Section 2.2.2). To obtain a high aperture angle, the working distance should be chosen as low as possible. However, this also limits the space for robotic movements. Furthermore, working distances that are too small prohibit a large proportion of the generated secondary electrons to be collected by the detector reducing the image intensity and signal-to-noise ratio. In a variety of experiments and with different robotic setups, a working distance of 8 to 10 mm has proven to be an acceptable trade-off. For SEMs with multiple or exchangeable apertures, an aperture as large as possible should be chosen. Large apertures, however, severely reduce the resolution and need to be avoided as no clear images of nanoobjects can be obtained. The used tungsten cathode-based SEM is equipped

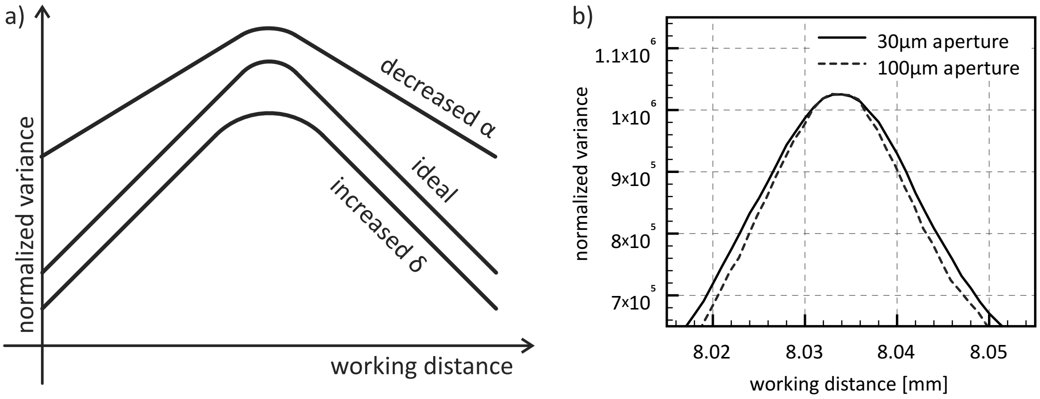


Figure 3.32: Influence of SEM parameters on curve of normalized variance: a) Influence of aperture angle α and resolution δ, b) measured curves with 30 µm and 100 µm apertures.

with three different apertures: 5 µm, 30 µm and 100 µm. The 5 µm aperture has shown to lead to a poor signal-to-noise ratio which can barely be used for imaging or tracking. Fig. 3.32b shows the focus curves for the other two apertures. As expected, the 100 µm aperture leads to a steeper curve of normalized variance. The difference, however, is marginal an either aperture can be used for accurate depth estimation.

Fig. 3.33 shows the image acquisition of a CNT at a working distance of 9 mm when using the 100 µm aperture. The 14 µm-long CNT was grown vertically on a substrate which is mounted in a 45°-degree angle with respect to the electron beam. Thus, the tip of the CNT is approximately 10 µm higher than the base and there is a linear relation between the x- and the z-coordinate. Three images are acquired while changing the working distance in 5 µm steps. It can be seen that Fig. 3.33a is focused on the CNT's base, Fig. 3.33b on the CNT's center and Fig. 3.33c on the CNT's tip. The depth of focus is in the range of 3 µm.

Choosing the right SEM parameters for depth estimation leads to a significantly lower depth of focus and thus a higher resolution.

3.5 SEM scan generator

To implement the tracking as described above, a precise and fast control of the electron beam is necessary. Additionally, the result of each line scan needs to be available quickly after the completion of the line scan, i.e. the computation time T_C should be as small as possible (see Section 3.2.3). These prerequisites cannot be achieved with any commercially available scan generator. Thus, a new scan generator was implemented. This scan generator makes use of a standard interface, available on most SEMs. The following sections briefly introduce the key ideas and

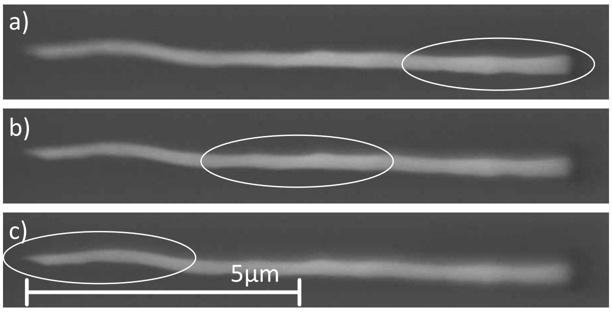


Figure 3.33: Imaging a 14 µm-long CNT mounted in a 45° angle with low depth of focus. Sharp regions are highlighted: a) Base in focus, b) center in focus and c) tip in focus.

technologies implemented in the newly developed scan generator. This includes the choice of the appropriate hardware components, the counteracting of SEM distortions, considerations during image acquisition as well as the implementation of the center of gravity calculation.

3.5.1 Hardware design

The developed scan generator uses a standard external scanning interface in order to be compatible with a variety of different SEMs. This interface consists of two analog output voltages that are applied to the electron microscope to control the x- and y-position of the electron beam, respectively. An analog input voltage returns the current value of the selected electron detector. This interface gives a low-level access to an SEM, but provides only a basic level of control. Other SEM parameters such as the working distance, magnification, brightness and contrast are not available through a standardized interface and have to be set via the SEM's control program. Most control programs however provide means of remote control, either through a software interface by a program running on the same machine or by a hardware interface, e.g. network or serial port.

Fig. 3.34 shows the architecture of the developed scan generator. As it is a prototype of a system that is partly implemented in hardware, a field-programmable gate array (FPGA) is chosen as central component. The employed development board "Nexys II" featuring a Xilinx Spartan III FPGA is provided by Digilent Inc.. A custom analog extension board contains two high-speed digital-to-analog converters (DACs) and a high-speed analog-to-digital converter (ADC). These converters operate at a sampling frequency of 25 million samples per second (25 MSps), so that the pixel time, i.e. time for the digitization of the current detector value, is 40 ns.

A component called line scanner handles all aspects of scanning a single line, including moving the electron beam, ensuring proper beam settling and pixel averaging. To move the electron beam, two 28-bit accumulators are used so that the beam can be moved from any point within the 14-bit DAC coordinate system to any other point in the same coordinate system while capturing up to 2^{14} pixels. The state machine takes the 14-bit values generated by the ADC reading the value of the electron detector and applies the predetermined averaging. The resulting values can then be either transferred directly to the hardware implementation of the tracking calculator or to an USB device chip on the development board for capturing an image. Using the controller area network (CAN) bus, the tracked positions can be directly transferred to the robot controller described in Section 4.4.

Six additional DACs provide means to scale, rotate and shift the scanning signals generated by the high-speed DACs. These DACs perform an analog four-quadrant multiplication, i.e. an analog input signal is multiplied with a digital signal in

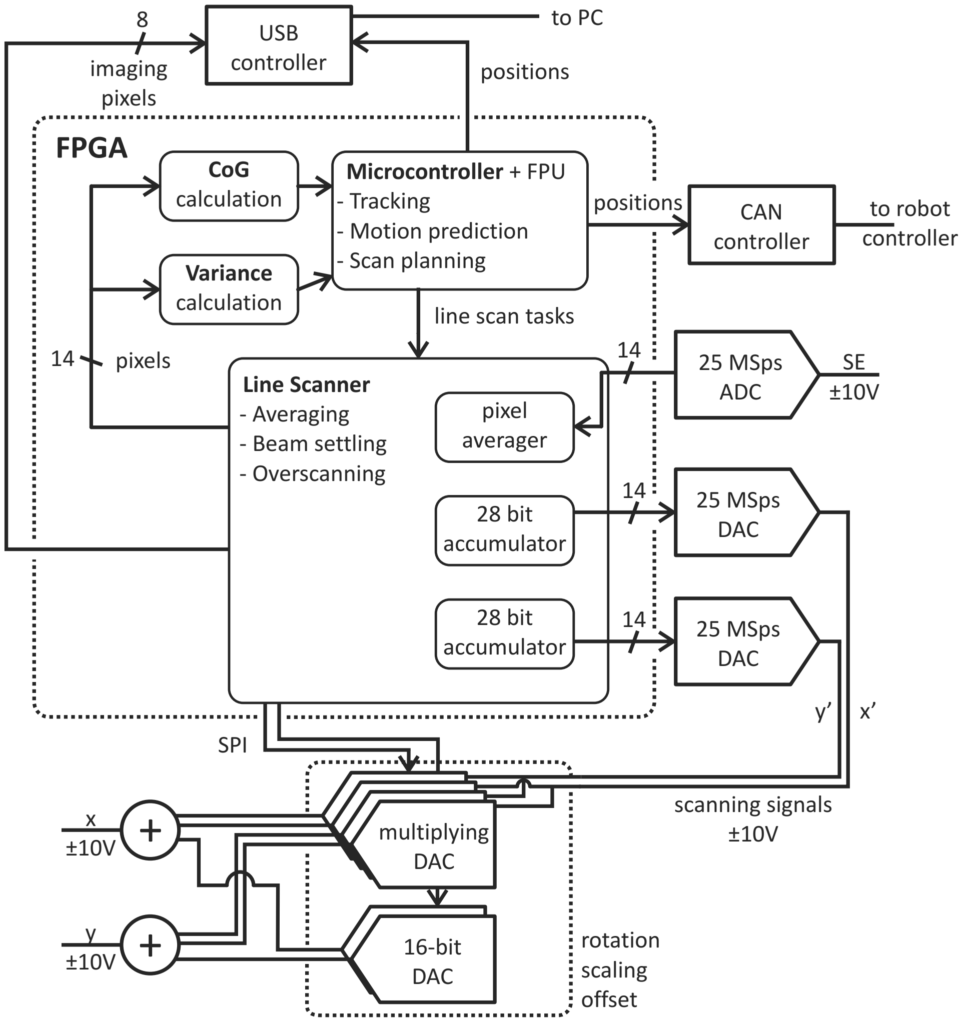


Figure 3.34: Schematics of the developed scan generator.

the range [-1, 1]. Given the outputs x' and y' of the high-speed DACs, these six converters can produce x and y scan signals with:

$$x = ax' + by' + c \tag{3.45}$$
$$y = dx' + ey' + f \tag{3.46}$$

The variables a through f are the values programmed into the corresponding DACs using the serial peripheral interface (SPI) and must be in the interval $[-1, 1]$. Thus, $a = e = 1$ and $b = c = d = f = 0$ lead to an image without scaling, rotation or offset. By choosing $a = e = k\cos\alpha$ and $b = d = k\sin\alpha$, the image can be rotated by the angle α and scaled by a factor k.

Fig 3.35 shows a picture of the implemented scan generator with the development board on the left and the analog extension board on the right.

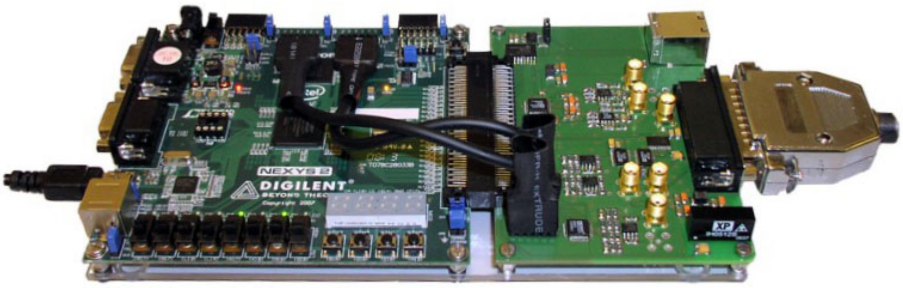


Figure 3.35: Picture of the developed scan generator.

3.5.2 Hardware-based tracking

A key component of the scan generator is the integrated hardware-based tracking implementation. The implementation is based on the center of gravity calculation approach. During each line scan, the hardware receives the value of each pixel and calculates the two sums in Eq. 3.10. The calculation is slightly extended in order to include a transformation into a goal coordinate system. The line scan has a length l, that is defined in the output DACs coordinate system. On this line scan, N pixels are acquired. Instead of calculating c according to Eq. 3.10 that calculates the center as the pixel index on the line scan, a value c' is calculated that returns the center position with respect to the scan length l:

$$c' = \frac{l}{N-1} \cdot c = \frac{\sum_{i=0}^{N-1} l \cdot i \cdot t(p[i])}{\sum_{i=0}^{N-1} (N-1) \cdot t(p[i])} = \frac{s_w}{s_v} \tag{3.47}$$

Upon the completion of a line scan, the quotient of s_w and s_v is calculated in a dedicated hardware divider. The hardware component then raises an interrupt to the embedded microcontroller which updates the tracking information according to Eq. 3.25 and Eq. 3.26 and requests the next line scan.

3.5.3 Distortion-free line scans

In order to compensate for the dynamic beam distortions described in Section 2.2.1, the dynamic behavior of the used SEM's electron beam needs to be carefully characterized. The dynamic characteristics need to be taken into account by both image acquisition and line scan-based tracking in order to generate undistorted images and precise measurement results. For the experiments described in this section, a Zeiss LEO 1450 SEM was used. However, the considerations apply to all SEMs.

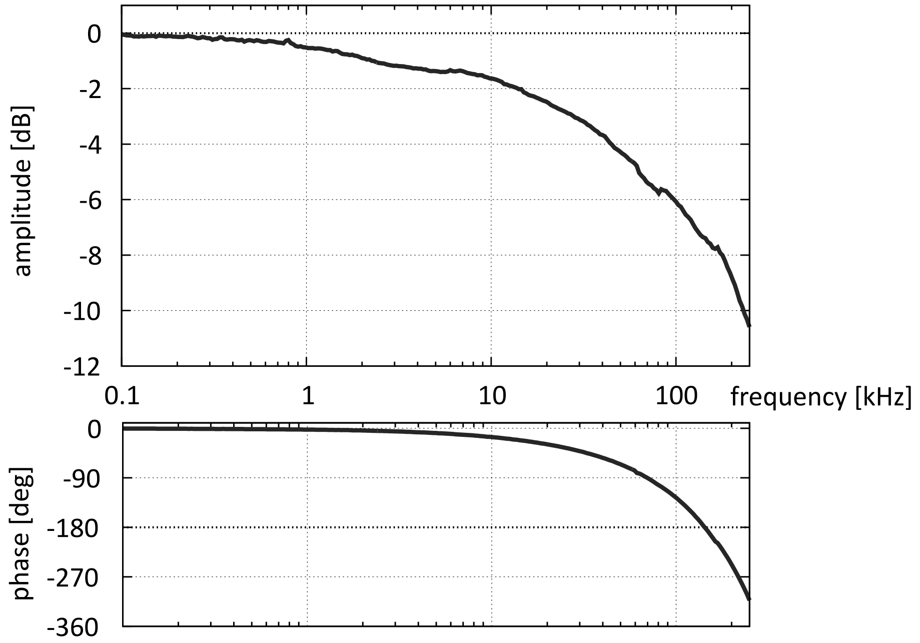


Figure 3.36: Amplitude and phase plots of the SEM beam's movement.

A simple method to evaluate the dynamic motion behavior of an SEM's electron beam is using a specimen that leads to a linear gradient of emitted electrons between two points A and B similar to Fig. 3.5a. Thus, if the electron beam is moved from A to B, the detector signal changes accordingly and can be used to derive the beam's location. Such a linear gradient can be created by using an edge visible on an arbitrary specimen and a slight defocus. With this method, two experiments are conducted.

For the first experiment, the electron beam is set to periodically move between point A and B using a sine-shaped velocity curve. Starting at 1 Hz, the rate of movements between A and B is increased to perform a frequency sweep up to a maximum frequency of 250 kHz. Higher frequencies could not be tested, as they were out of the range of the employed lock-in amplifier. For each frequency, the amplitude as well as the phase offset of the detector signal is measured using a lock-in amplifier. This leads to the amplitude and phase plots shown in Fig. 3.36. The phase plot mainly shows the exponentially falling phase angle caused by signal latency. If this decline was only caused by the latency, the latency could be approximated to roughly 2.4 µs as the phase lags by 180°, i.e. half a period, at a frequency of at about 105 kHz. However, as the low-pass behavior also causes a certain phase lag, the actual latency is shorter. The order of the low-pass filter cannot be derived from the phase plot due to the undetermined latency. The amplitude plot, which is not influenced by the latency, shows an over-dampened low-pass behavior.

To further prove these observations, a second experiment is used to measure the step response. The electron beam is controlled by a square wave at a frequency of 100 Hz. An oscilloscope is used to measure both the input signal as well as the response. The result is shown in Fig. 3.37a. Note that the output of the secondary electron detector is noisy and the figure shows averaged signals of 16,384 step responses. As expected from the frequency behavior, the step response shows a dead time of about 2 µs and a following over-dampened low-pass filter of at least second order. Actually, two different settling behaviors can be observed, the first ending around 6 µs and a much slower settling after that. A total settling time of at least 14 µs is necessary, before the beam can be assumed to be roughly at the requested location.

For virtually any scanning operation, the ramp response of the electron beam is of importance. In order to obtain an undistorted image or line scan, the electron beam needs to move precisely with the required velocity. For very low velocities such as the 2 ms-ramp in Fig. 3.37b, the electron beam can be assumed to be at the location set by the input signal. The remaining distortions are caused by the test setup, as the SE detector signal does not perfectly correspond to the beam's location. For higher velocities such as the 40 µs ramp in Fig. 3.37c, the start and the end of the ramp are non-linear, but there is a linear region. For very high velocities (e.g. the

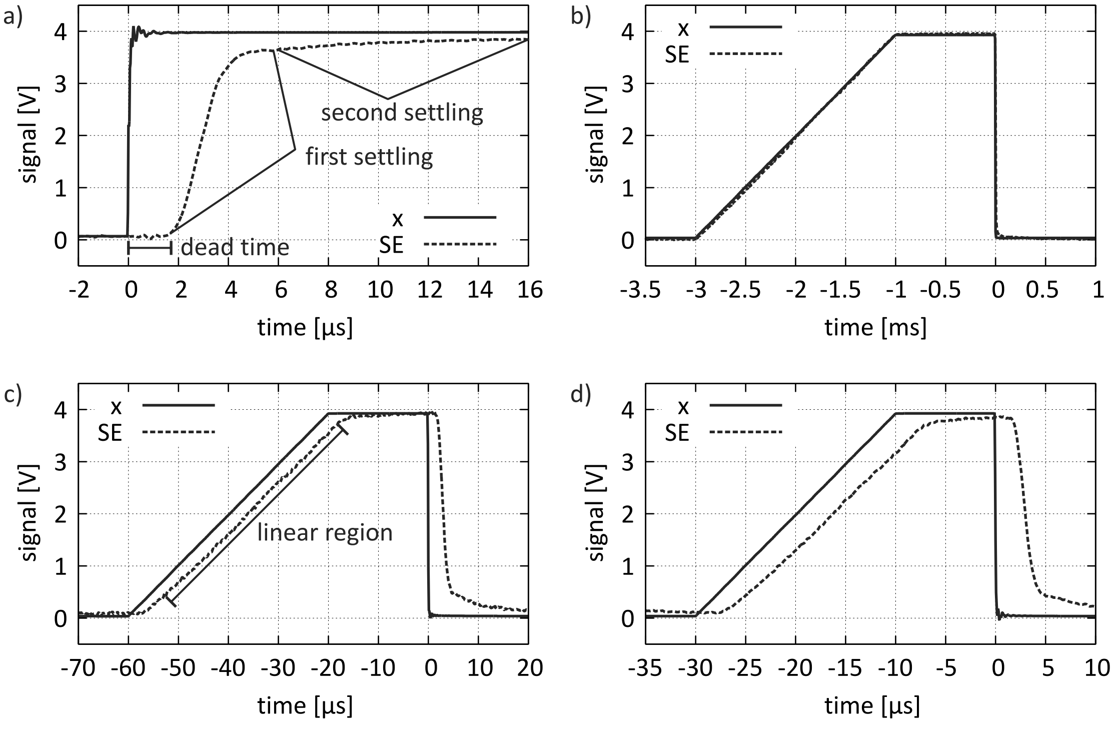


Figure 3.37: Step and ramp responses of the electron beam. The input to the x scan coil is controlled and the value of the SE detector is measured.

20 µs ramp in Fig. 3.37d), there is no linear relation between the input signal and output signal.

Both the step and the ramp response behaviors need to be considered for scanning. The step response is important, as it occurs for every rapid beam movement, e.g. when moving from one line to the next during image acquisition or tracking. Thus, after each such rapid beam movement, at least 14 µs are needed in order to avoid distortions. The non-linearities on the ramp responses (see Fig. 3.37b-d) need to be taken into account in order to obtain undistorted images.

The so-called **overscanning** is a well-known technique for performing undistorted line scans. It is used in most SEMs and all AFMs. For a horizontal line scan, the electron beam is moved to a position left of the first pixel of interest. The beam control signals then remain constant for a few microseconds (14 µs for the used SEM) to allow for the beam to settle to this location. Then, the input signal to the x scan coil linearly changes to a position right of the last pixel of interest. Pixels are only captured during the linear movement phase of the electron beam so that the returned pixels correspond to an undistorted line scan.

3.5.4 Automatic adjustment of brightness and contrast

As described in Section 3.2.1, position tracking with line scans and CoG calculation is virtually immune to changes in brightness and contrast. For other image processing algorithms and focus-based depth estimation however, the brightness and contrast obtained by the SEM is of high significance. Thus, it is desirable to keep brightness and contrast constant during a handling operation. This can be achieved with the data collected during the line scans.

For most SEMs, the two settings brightness b and contrast c can be controlled in a range from 0 to 100%. The brightness is a linear offset to the obtained values, whereas the contrast describes an exponential gain. Thus, a given count of detected electrons s leads to a pixel value p according to the equation:

$$p = s \cdot e^{c \cdot k_1} + (b - k_2) \cdot k_3 \tag{3.48}$$

The constants k_1, k_2 and k_3 can be approximated based on several experiments. k_2 and k_3 are determined by setting c to 0 and changing b. Afterwards, k_1 can be measured by altering c and fitting an exponential curve. For the used Zeiss LEO 1450 SEM, the values $k_1 = 28$, $k_2 = 0.496$ and $k_3 = 26000$ have been determined.

Each line scan returns a brightest pixel value p_b and a darkest pixel value p_d. In order to track the brightness and contrast conditions, the values of p_b and p_d should remain constant. At the start, the desired values d_b and d_d for the brightest and darkest pixel are stored. If the contrast and brightness conditions change, p_b and p_d deviate from d_b and d_d. According to 3.48, this deviation can be compensated for by changing the brightness and contrast settings. To change the contrast, the missing factor f between current and desired contrast is calculated and translated into the new exponential contrast setting c':

$$f = \frac{d_b - d_d}{p_b - p_d} \tag{3.49}$$

$$c' = c + \frac{\ln f}{a} \tag{3.50}$$

To compensate for brightness changes, a new setting b' is calculated using the medium of the brightest and darkest pixels:

$$b' = b + \left(\frac{d_b + d_d}{2} - \frac{p_b + p_d}{2} \right) \cdot \frac{1}{k_3} \tag{3.51}$$

The changes to brightness and contrast are low-pass-filtered because the SEM exhibits significant pixel noise at the employed scan speed. Otherwise, this noise would lead to rapidly and continuously changing brightness and contrast settings.

3.6 Initializiation

Line scan-based tracking can be used to track the position of patterns and objects, but it cannot be used for object detection. Hence, from an unknown state, the initial positions of the tracked objects need to be found by other means in order to fully automate the positioning system. However, such initializations have to be done infrequently during an automated handling operation making performance a secondary concern in the initialization phase. Furthermore, all objects can be assumed to remain stationary. Thus, SEM image acquisition in combination with any established technique used for object recognition in SEM images can be used. A promising approach is cross correlation-based template matching. Template matching can be used directly or indirectly to find the position of the patterns or objects.

The **direct** initialization is especially useful for the tracking of micro- or nanoobjects such as the spheres and CNTs in Fig. 3.29. A template image of such an object is used on images with high magnification. Once the targeted object is found, it can be tracked with line scan-based tracking. This however is impracticable for small patterns (e.g. 2 µm) that are possibly located anywhere within a large working range (e.g. 1×1 mm^2). Imaging such a large area with sufficient resolution for template matching as well as the following image processing is time consuming. For example, if the 2 μm pattern needs to be imaged with a resolution of at least 20×20 pixels for reliable template matching, the 1×1 mm^2 spans 10000×10000 pixels.

For the latter cases, the **indirect** approach should be employed, capitalizing on additional information of the relative position between patterns and larger objects. This approach is especially useful for tools with tracking patterns (cf. Fig. 3.24). Template matching is used to find the tool's position in an image with low magnification. Once the position of the tool is determined, a higher resolution image can be acquired of a small area known to contain the target pattern using the known relative position between tool and pattern. Then, the pattern's exact location is determined in the high-resolution image and the line scan-based tracking is initialized.

A challenge for initialization are the SEM's imaging parameters including, brightness, contrast, focus and astigmatism. If brightness and contrast are set incorrectly, the detector value might lie outside the measurable range. Objects might be unrecognizable or even invisible if they are not located on the focus plane. Astigmatism might need to be corrected especially for high-resolution images in which small micro- or nanoobjects need to be found. These challenges are not specific to line scan-based tracking but persist in any SEM-based tracking scenario. Thus, they are not analyzed in detail in this thesis.

3.7 Conclusions

The inherent bottleneck of tracking based on SEM images is caused by the inefficient use of pixels, i.e. the low ratio between pixels acquired and pixels used for tracking. A new approach based on SEM line scans has been introduced that significantly increases the efficiency and can thus provide a high update rate coupled with a high robustness against noise, focus, brightness and contrast changes. A special SEM scan generator was developed that facilitates the tracking's high efficiency by performing most of the calculations during the line scan itself using dedicated hardware components. The presented approach is designed for tracking special target patterns, but directly tracking arbitrary objects is also possible.

Combining line scan-based tracking with focus-based depth detection also leads to a significant boost in performance. Due to this boost, the depth from focus and depth from defocus approaches can be employed to perform coarse and fine alignment with a high resolution. For the first time, an alignment on the nanoscale becomes feasible only relying on the SEM's focus information.

The presented approach is the first approach taking the dynamic characteristics of a moving object and the corresponding distortions in the SEM signal into account. Thus, vision-based servoing does not have to be treated in a quasi-static fashion. Instead, visual servoing can be implemented using the line scan-based tracking as an element in a robot's low-level control loop. The next section describes a robotic system that can effectively use the new tracking approach's capabilities.

4 Automated motion control

Using SEM-based position tracking as described in the previous chapter provides means for fast and accurate positioning. In this chapter, a robotic system is developed, characterized and controlled that can make full use of the tracking's capabilities.

4.1 Robotic architecture

Over the recent years, a variety of robot designs capable of operating with precisions of single nanometers have been developed (see Section 2.3.3). Mobile robots have proven to have several advantages over other robot architectures. First, they can be effortlessly integrated into different microscopic setups. Mobile robots can simply be placed on a flat surface in SEMs or on microscope slides of optical microscopes. Second, they combine multiple degrees of freedom as well as a large working range in a small robot. The small form factor is beneficial as it decreases vibrations and drift effects. The lack of internal sensors is not a limitation as positioning with the required accuracy needs to be done using visual servoing anyway (see Section 2.3.4). With the line scan-based position tracking described in the previous chapter, fast sensor feedback is available.

All mobile nanorobots use a step-wise motion principle (compare Section 2.3). Thus, they have all the benefits of step-wise actuators such as a large working range combined with a high resolution in scanning mode. The motivation for the new nanorobot described in this work is to alleviate the downside of generated vibrations of such actuators. The approach is to design a robot where the accelerated masses of the actuators as well as the step length of the robot are reduced by one order of magnitude compared to other designs. In addition to reducing vibrations, the small step length leads to a higher possible resolution in scanning mode. The small accelerated masses make high step frequencies possible, so that the robot can reach high velocities in spite of the small steps.

4.1.1 Design

The design of the mobile nanorobot's actuation platform is shown in Fig. 4.1. It is based on laser-structured piezoceramic actuators. The basic building block of the actuator consists of a ruby hemisphere glued to three active segments of the ceramic actuator. Using an appropriate control voltage, each segment contracts or expands. Voltages between -150 V and +150 V are within the actuator's specifications and lead to displacements of up to 100 nm. Each active segment has an approximated size of $0.5 \times 0.25 \times 0.25\,\mathrm{mm}^3$, i.e. a volume of about $0.03\,\mathrm{mm}^3$. With the three active segments, the ruby hemispheres can be rotated around their center points with an arbitrary horizontal axis dependent on which segments contract or expand.

A single actuator unit is comprised of three such basic building blocks, i.e. three ruby hemispheres. The three ruby hemispheres transfer their motion to a steel sphere via friction contacts. A permanent magnet is used to increase the normal force and keep the spheres in place if the robot needs to be transported. If all ruby hemispheres rotate by the same amount into the same direction, the steel sphere will also rotate by this amount, but in the opposite direction. Thus, for the maximum rotation, the surface of the steel sphere also moves by 100 nm. Three actuator units are embedded into a printed circuit board, which forms the actuation platform of the mobile nanorobot. Aside from the electrical connection of all 27 active segments, the circuit board features two infrared LEDs that can be used for camera-based tracking (see Section 4.1.2).

The actuation is based on the stick-slip principle. During the slip phase, the piezoelectric elements slowly rotate the ruby hemispheres. Due to friction, the steel spheres stick to the ruby hemispheres and rotate as well. During the slip phase, the voltage on all active segments is rapidly inverted. Thus, the ruby hemispheres are moved quickly enough, so that the friction is overcome due to the steel sphere's moment of inertia. In contrast to other actuators, however, the scanning range is only ±100 nm leading to a small step size. Furthermore, the combined volume of all 27 active segments is less the $0.85\,\mathrm{mm}^2$. Similarly, the mass of each ruby

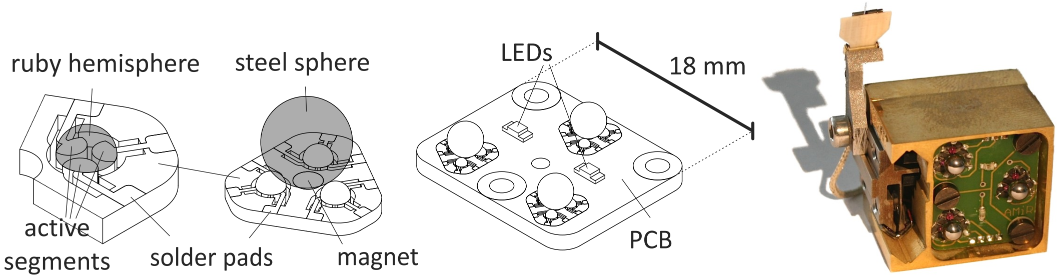


Figure 4.1: Design and picture of a nanorobot.

hemispheres is only about 1 mg leading to a small moment of inertia. These values lead to reduced vibration during the stick-slip motion as well as a reliable operation even for high actuation frequencies as will be shown in Section 4.2.2. Furthermore, as the friction contact used for stick-slip actuation is between the ruby hemispheres and the steel spheres the robot does not impose special requirements on the working surface. Virtually any flat surface can be used without a noticeable change in the robot's behavior, as it practically rolls on the steel spheres. Soft surfaces can be used without wear.

4.1.2 Nanohandling robot cells

A robotic system employing mobile nanorobots in an SEM is shown in Fig. 4.2a. In the setup, two mobile robots work on a glass surface. A camera mounted underneath the surface tracks two infrared LEDs mounted to each robot's bottom. The tool robot (see Fig. 4.2b) carries a tool such as a gripper using an additional vertical axis. The stage robot (see Fig. 4.2c) carries two SEM specimen holders called stubs. The tool and objects on the stubs can be tracked by the SEM monitoring the scene from above. Multiple individual hardware components are used for the control loop. The LED tracker is responsible for coarsely determining the position of the two robots (Diederichs, 2010). The scan generator is used to track tools and objects with the SEM. Two control units implement closed-loop trajectory control for the

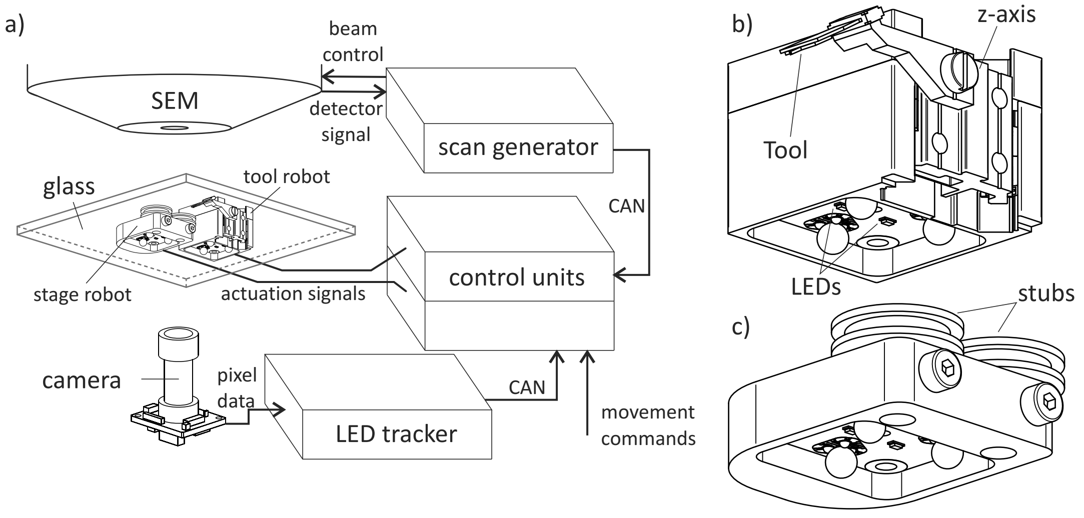


Figure 4.2: a) Microrobotic handling cell with two robots working under an optical microscope. A camera underneath the glass surface facilitates coarse positioning. b) The tool robot is equipped with an additional vertical axis. c) A stage robot carries two specimen stubs.

robots. The communication is implemented using the real-time capable controller area network (CAN) bus.

4.1.3 Control loop and architecture

Fig. 4.3 shows the control loop of the robot cell described above. The controlled nanorobot is tracked by the line scan-based tracking for fine positioning and by the LEDs for coarse positioning. External influences such as the robot's cabling or an uneven working surface influence the robot's motion. The controller itself can be divided into three tiers: physical tier, open-loop control and closed-loop control.

The physical tier is responsible for supplying the necessary signals to a nanorobot's individual actuators. On the higher tiers, mostly digital controllers are employed. Thus, the first step is the generation of analog actuation signals. These signals have a low voltage (<5 V) but can have complex signal shapes such as sawtooth signals for stick-slip actuation. Furthermore, multiple voltage signals have to be generated synchronously for multi-channel actuators such as mobile robots. Moreover, changes to the signals' parameters, e.g. amplitude or frequency of a sawtooth signal, must be applied smoothly, i.e. without creating discontinuities or delays in the signal. Otherwise, vibrations are created and the robot's behavior becomes unreliable. Thus, the signal generation has to be done using dedicated hardware. The low-voltage signal then has to be amplified in order to meet the actuators' demands on voltage and current. The most commonly applied amplifiers are high-voltage amplifiers that linearly amplify the control signal, increasing voltage and current. However, as piezoceramic actuators act as capacitive loads and actuation principles require high slew rates, stable amplifier circuits can be difficult to achieve.

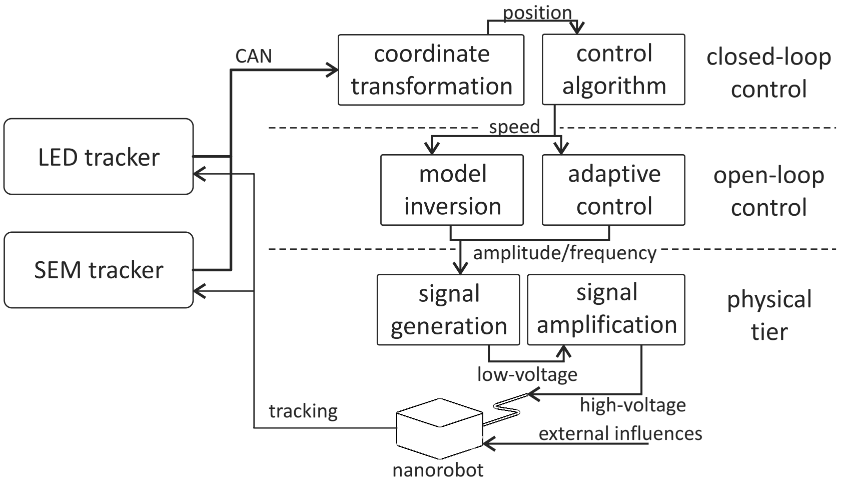


Figure 4.3: Tiers of the nanorobot control architecture.

The task of the open-loop controller is the selection of actuation parameters that lead to a desired motion. This is usually done using an inverse model of the actuators' behavior. For micro- and nanoscale actuators, this model cannot be completely derived from the design, because many effects are not fully understood. This includes static effects of the piezoceramic materials such as hysteresis and creep as well as dynamic effects, e.g. vibration and friction during stick-slip actuation. Therefore, parts of the model are derived using system identification (Jasper and Edeler, 2008). If the behavior is highly unreliable or time-variant, a self-learning model inversion can be necessary (Hülsen, 2007). The latter is commonly referred to as adaptive control.

The first step of the closed-loop control tier is the transformation of the sensor data into a common coordinate system. This transformation has to include aspects such as the robot's rotational center and the relative location of the tracked tool or object. The challenges for the closed-loop control algorithm differ greatly between the nanoscale and the macroscale. On the macroscale, control engineering mostly solves challenges with regard to the dynamic behavior of systems such as inertia and damping. These effects are mostly negligible on the nanoscale, because the required movement distances, velocities and accelerations are comparatively small. Therefore, virtually any movement can be realized instantly. Other challenges such as the attenuation of external influences as well as compensation for the sensor's latency remain similar to the macroworld.

The following sections detail the implementation of the different control tiers.

4.2 Open-loop motion control

The task of the open-loop control is to select actuation parameters so that the robot performs a desired movement, e.g. moves into a specific direction with a specific velocity. The robot is a physical system with a transfer function G_R that maps a set of actuation parameters $\mathbf{p}_a$ to a measured motion vector $\mathbf{g}_m$:

$$G_R : \mathbf{p}_a \rightarrow \mathbf{g}_m. \tag{4.1}$$

The goal of the open-loop motion controller is to invert this behavior, i.e. to select a vector of actuation parameters $\mathbf{p}_a$ in order to perform a desired motion vector $\mathbf{g}_d$. The transfer function of the controller G_C thus performs the mapping:

$$G_C : \mathbf{g}_d \rightarrow \mathbf{p}_a. \tag{4.2}$$

An ideal motion controller leads to a measured motion $\mathbf{g}_m$ that is identical to $\mathbf{g}_d$. The nanorobot described above is controlled using a total of six actuation channels as shown in Fig. 4.4a. Each channel is actuated with a sawtooth-shaped signal of a specific amplitude u. All signals are chosen to be synchronous, i.e. have the same frequency and phase, so that the slip phase occurs simultaneously on all actuation segments. Thus, all channels share the same actuation frequency f. The arrows depicted in Fig. 4.4 can be interpreted as movement vectors. A positive actuation amplitude on a specific channel leads to a movement in the direction of the corresponding movement vector. The robot's transfer function G_R translates from six amplitudes and a frequency to three motion velocities:

$$G_R : \begin{pmatrix} u_{1a} & u_{1b} & u_{2a} & u_{2b} & u_{3a} & u_{3b} & f \end{pmatrix}^T \to \begin{pmatrix} v_x & v_y & \omega_\varphi \end{pmatrix}^T . \tag{4.3}$$

Here, v_x and v_y are the velocities along the x- and y-axes, respectively, whereas ω_φ is the angular velocity of the robot's rotation.

An open-loop controller implementing the inverse function of G_R is derived in three steps:

1. Partially modeling the robot
2. Completing the model by system identification
3. Inverting the model

The following sections describe these steps in detail.

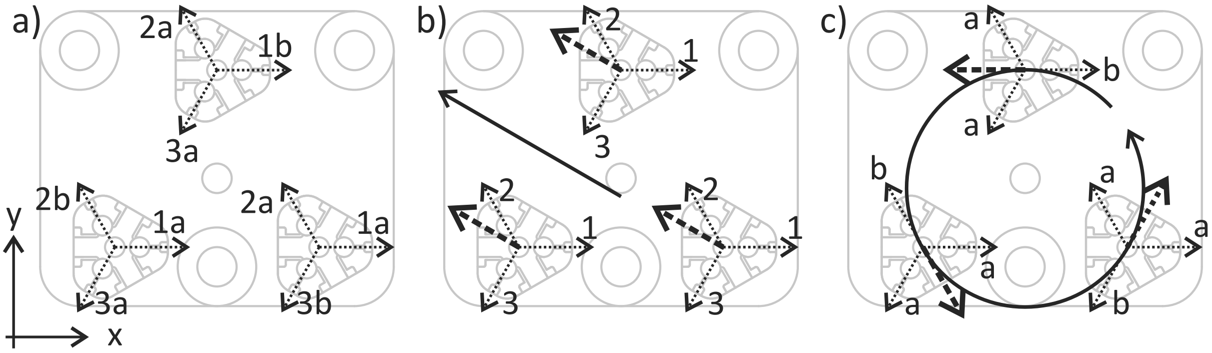


Figure 4.4: The nanorobot's actuation channels: a) All 27 active piezo segments can be grouped into six channels. b) A parallel actuation of the a- and b-channels leads to translational movement. c) A parallel actuation of the 1-, 2- and 3-channels leads to rotational movement.

4.2.1 Partially modeling the robot

The goal of the partial model is to describe the interaction of the control amplitudes on the different channels without knowledge of the maximum velocity which depends on the step frequency f. The maximum velocity could also be modeled either using the elongation coefficients of the piezoceramic material or a finite element simulation. However, due to a variety of uncertainties, a later system identification results in a better model. Therefore, the velocities are modeled with respect to the maximum velocities $\hat{v}(f)$ and $\hat{\omega}(f)$, which will be determined by system identification in Section 4.2.2. Furthermore, a linear behavior between actuation amplitude and step length is assumed. The validity of this assumption will also be checked by system identification.

To model the robot's behavior, translational and rotational movement are separated. The a- and b- channels can be combined in order to create purely translational movement (see Fig. 4.4b). Geometric considerations lead to the following equations for motion velocities:

$$
\begin{aligned}
v_x &= \frac{\hat{v}(f)}{\hat{u}} \cdot \frac{2u_1 - u_2 - u_3}{4} && (4.4)\\
v_y &= \frac{\hat{v}(f)}{\hat{u}} \cdot \frac{u_2 - u_3}{2} \cdot \sin 60^\circ = \frac{\hat{v}}{\hat{u}} \cdot \frac{u_2 - u_3}{2} \cdot \frac{\sqrt{3}}{2}, && (4.5)
\end{aligned}
$$

where $\hat{u}$ is the maximum actuation amplitude. The 1-, 2- and 3-channels can be combined for purely rotational movement (see Fig. 4.4c). Geometric considerations lead to the rational velocity ω_φ:

$$
\omega_\varphi = \frac{\hat{\omega}(f)}{\hat{u}} \cdot \frac{u_a - u_b}{2}. \tag{4.6}
$$

If both movement types are used concurrently, the robot performs a rotation around an arbitrary point P as shown in Fig. 4.5a. Each sphere performs a tangential movement on the corresponding circle around P with a velocity that is proportional to the radius (r_1 through r_3) of the circle. The movement equations can be derived from Eq. 4.4-4.6 by replacing each combined channel value by the average of the individual channels, e.g. replacing u_1 with $(u_{1a} + u_{1b})/2$:

$$
\begin{aligned}
v_x &= \frac{\hat{v}(f)}{\hat{u}} \cdot \frac{2u_{1a} + 2u_{1b} - u_{2a} - u_{2b} - u_{3a} - u_{3b}}{8} && (4.7)\\
v_y &= \frac{\hat{v}(f)}{\hat{u}} \cdot \frac{u_{2a} + u_{2b} - u_{3a} - u_{3b}}{4} \cdot \frac{\sqrt{3}}{2} && (4.8)\\
\omega_\varphi &= \frac{\hat{\omega}(f)}{\hat{u}} \cdot \frac{u_{1a} + u_{2a} + u_{3a} - u_{1b} - u_{2b} - u_{3b}}{6} && (4.9)
\end{aligned}
$$

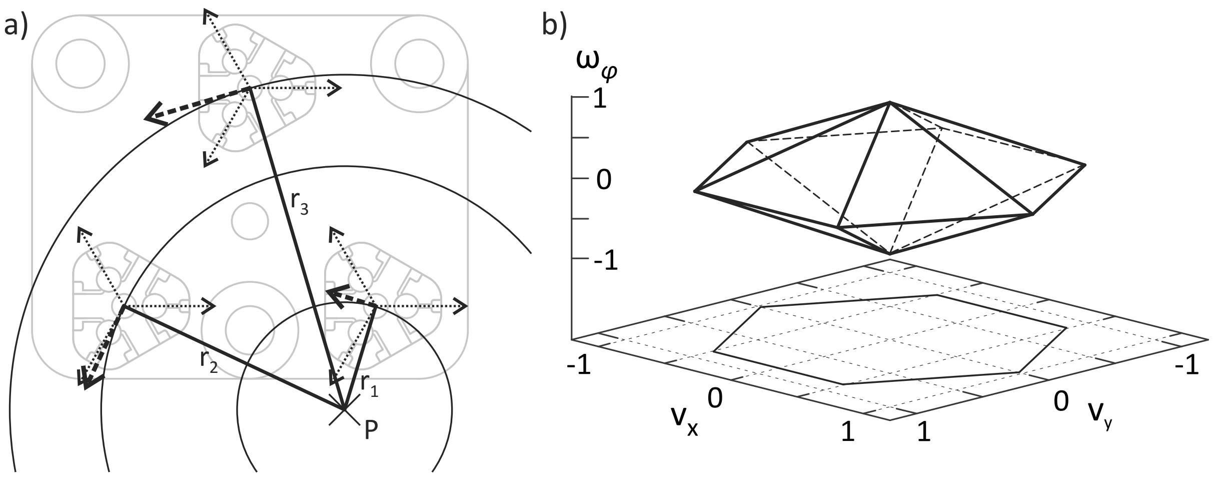


Figure 4.5: a) Using all six channels, the robot can perform a rotation around an arbitrary point P. b) The amplitude limit on each channel leads to a diamond-shaped velocity limit.

This can be rewritten into a matrix relating the actuation amplitude vector to the motion velocity vector:

$$\begin{pmatrix} v_x \\ v_y \\ \omega_\varphi \end{pmatrix} = \begin{pmatrix} \frac{\hat{v}(f)}{\hat{u}} & 0 & 0 \\ 0 & \frac{\hat{v}(f)}{\hat{u}} & 0 \\ 0 & 0 & \frac{\hat{\omega}(f)}{\hat{u}} \end{pmatrix} \begin{pmatrix} \frac{1}{4} & \frac{1}{4} & -\frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} \\ 0 & 0 & \frac{\sqrt{3}}{8} & \frac{\sqrt{3}}{8} & \frac{\sqrt{3}}{-8} & \frac{\sqrt{3}}{-8} \\ \frac{1}{6} & -\frac{1}{6} & \frac{1}{6} & -\frac{1}{6} & \frac{1}{6} & -\frac{1}{6} \end{pmatrix} \begin{pmatrix} u_{1a} \\ u_{1b} \\ u_{2a} \\ u_{2b} \\ u_{3a} \\ u_{3b} \end{pmatrix} \tag{4.10}$$

As the amplitude on each actuation channel is limited, the different motion velocities limit each other. Fig. 4.5b shows that the possible movements form a diamond-like shape with all realizable motion vectors being inside the diamond. The motion vectors are normalized, i.e. $\hat{v}(f) = \hat{\omega}(f) = 1$. A motion vector outside of the diamond would require at least one channel to exceed $\hat{u}$.

Fig. 4.6 shows two-dimensional projections of the movement capabilities. The maximum movement along the x- and y-direction forms a hexagonal shape. This shape is caused by the mechanical structure of the piezoactuators with three segments moving each ruby hemisphere. The corners of the hexagonal shape are the directions with a maximum absolute amplitude applied to all three segments. To move into a different direction, the amplitude of at least one channel has to be decreased leading to a shorter step length. At the smallest maximum step length, i.e. the movement vectors on the middle of the hexagon's edges, only two segments are actuated.

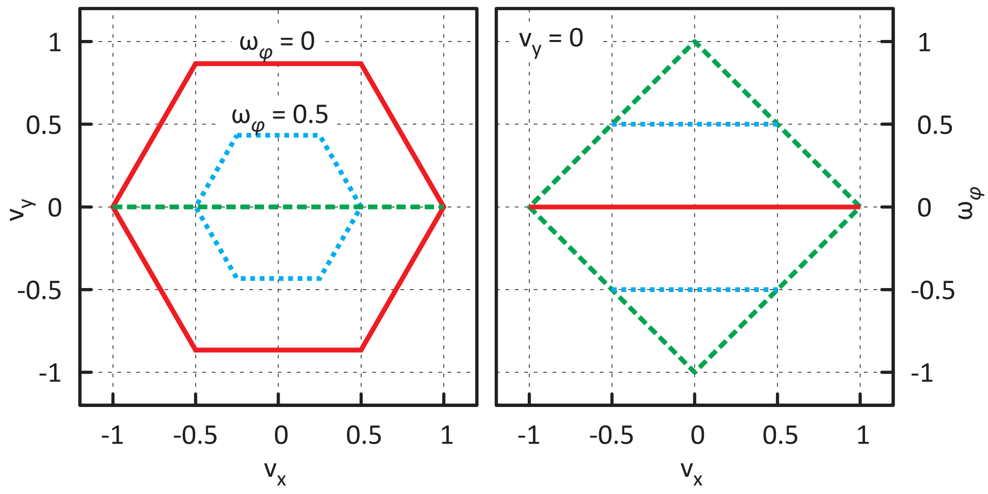


Figure 4.6: Maximum normalized motion velocities.

The relation between rotational and translational behavior is linear. If a normalized rotation velocity ω_φ of 0.25 is desired, only 75% of the maximum translational velocity can be used. The shown case for $\omega_\varphi = 0.5$ shows that the shape of the maximum translation speed remains hexagonal.

4.2.2 System identification

System identification is required to complete the partial model described above by adding the frequency-dependent maximum velocity. For this task, a measurement station with three individual laser interferometers was developed (Jasper and Edeler, 2008). The concept is shown in Fig. 4.7a. Three interferometer beams are used to capture all three degrees of freedom of the mobile robot. The interferometers feature a measurement resolution of about 1 nm at an update rate of 100 kHz. Thus, they can measure the robot's behavior with a high spatial and temporal resolution. A coarse positioning camera can be used to move the mobile robot from any position on the working surface into the working range of the interferometer setup.

To characterize the frequency dependency, the robot is actuated with identical actuation amplitudes and different frequencies. For a better comparison, the net length of a single step is recorded instead of the measured velocity. Ideally, the step length remains constant independent of the actuation frequency. Fig. 4.7b shows the step length derived for actuation frequencies between 2 kHz and 70 kHz. The maximum amplitude $\hat{u} = 150$ Vpp of the used amplifiers was applied to all channels, positively on channel group 1 and negatively on the groups 2 and 3 (see Fig. 4.4b), in order to create the maximum step length along the x-axis. To exclude acceleration effects from the measurements, the difference between a 1200 step movement and a

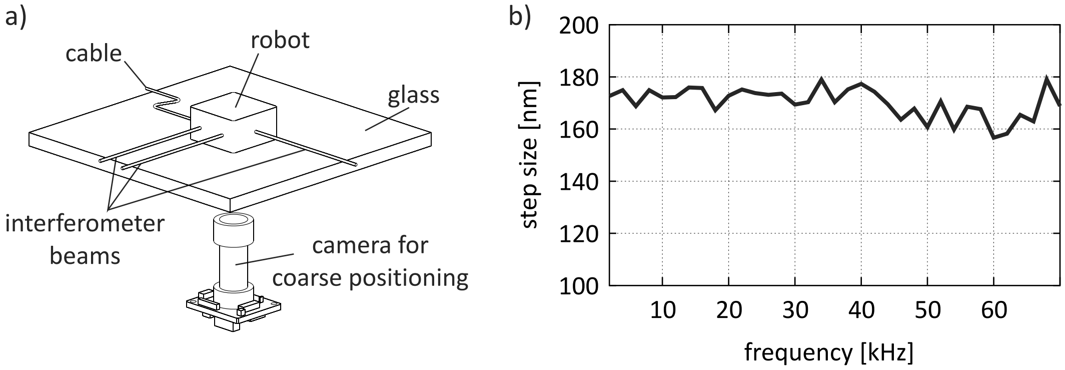


Figure 4.7: a) System identification setup. b) Measurement of step length dependent on frequency.

1000 step movement is calculated. The difference are 200 steps without acceleration effects, as the platform is accelerated identically during both movements. It can be seen that the step length remains virtually constant around 170 nm. The same measurement for rotations reveals frequency-independent maximum rotational steps of 0.0015°. This allows for a modeling of the maximum velocities as linear functions of the maximum step lengths:

$$\hat{v}(f) = f \cdot 170\,\mathrm{nm} \tag{4.11}$$

$$\hat{\omega}(f) = f \cdot 0.0015^\circ. \tag{4.12}$$

In order to measure the relation of amplitude and step length, an amplitude sweep is conducted. Similar to the frequency measurement described above, each amplitude is applied positively to channel group 1 and negatively to channel groups 2 and 3. An average step length over a ten step movement is measured in order to minimize the influence of measurement noise on the obtained result. This leads to the amplitude sweep response presented in Fig. 4.8. There is no movement for amplitudes less than 50 Vpp in any degree of freedom. For such low amplitudes, the acceleration and movement distance of the piezoactuators is insufficient to create a slip. Thus, even though the actuator moves quickly, the robot is able to follow this motion. For amplitudes higher than 50 Vpp, there is a linear increase in step size. The minimum movement amplitude has to be taken into account by the open-loop controller. Nevertheless, the step size and thus the velocity at a given frequency can be assumed to have a linear relation to the actuation amplitude.

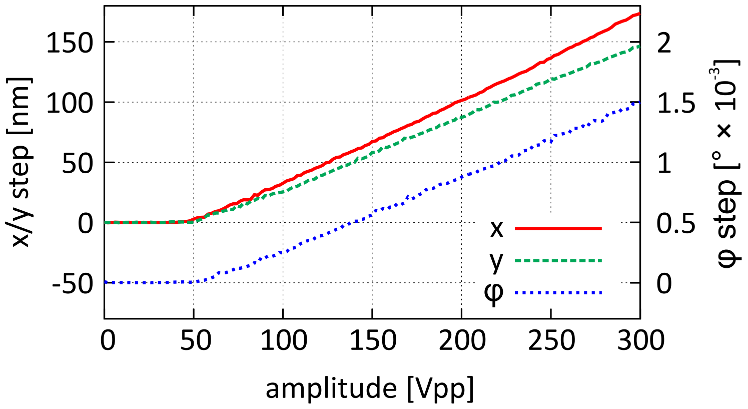


Figure 4.8: Amplitude sweep.

4.2.3 Inverting the model

An open-loop controller for the mobile nanorobot can be developed by inverting the complete model described in the previous sections. Thus, the controller needs to implement the inverse function G_C:

$$G_C : \begin{pmatrix} v_x & v_y & \omega_\varphi \end{pmatrix}^T \to \begin{pmatrix} u_{1a} & u_{1b} & u_{2a} & u_{2b} & u_{3a} & u_{3b} & f \end{pmatrix}^T . \tag{4.13}$$

As a side condition, the velocities should be applied for a time Δt and then stop. During this time, an integer number of steps should be executed, so that the actuation can be stopped or changed after complete steps. This will later be useful for the closed-loop controller. Thus:

$$f \cdot \Delta t = n, \text{ with } n \in \mathbb{N} \tag{4.14}$$

The calculation is done in multiple steps. First, the ratio between v_x and v_y is translated into the movement angle α, total velocity v and the maximum step length l (see Fig. 4.9a):

$$\alpha = \arctan \frac{v_y}{v_x} \tag{4.15}$$

$$v = \sqrt{v_x^2 + v_y^2} \tag{4.16}$$

$$l = \cos(\alpha \bmod 30°). \tag{4.17}$$

The value of l is used to account for the hexagonal shape of the maximum velocities (see Fig. 4.9a).

Next, the minimum actuation frequency f_{min} is calculated, assuming that the translational and rotational steps are executed at the maximum possible amplitude.

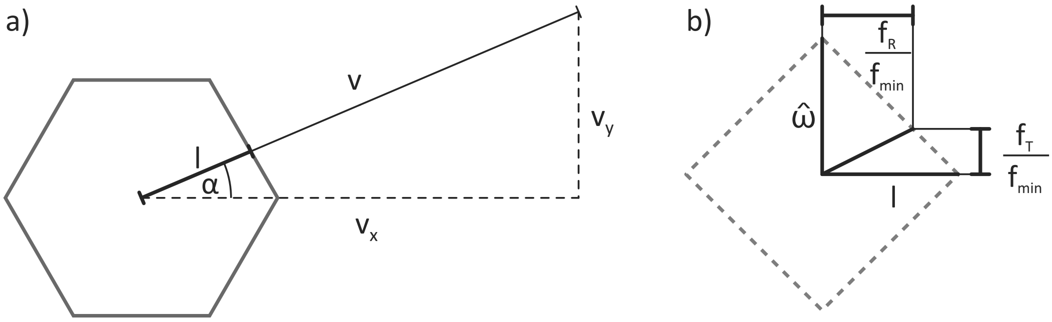


Figure 4.9: Inversion of the robot's movement model: a) Translational movement in v_x and v_y is translated into velocity v, movement direction α and a direction dependent factor l. b) Translational and rotational movement are combined by summing the individual frequencies.

The total frequency is the sum of the frequency required purely for translational steps f_T and the frequency required for purely rotational steps f_R. The two frequencies can simply be summed, as there is an inversely proportional relationship between the translational movement and rotations (see Fig. 4.9b):

$$f_{\min} = f_T + f_R = \frac{v}{l \cdot 170\,\mathrm{nm}} + \frac{\omega_\varphi}{0.0015^\circ}. \tag{4.18}$$

To fulfill Eq. 4.14, the actuation frequency is chosen as:

$$f = \frac{\lceil f_{\min} \cdot \Delta t \rceil}{\Delta t}. \tag{4.19}$$

In summary, the open-loop controller performs amplitude modulation to control the movement's direction in all three degrees of freedom as well as the step length. Frequency modulation is then employed to move with the desired speed only using frequencies that are multiples of Δt.

The last step of the open-loop control is the inversion of Eq. 4.10. A direct inversion of the matrix is impossible as it is not a square matrix. A certain motion vector can be realized by different combinations of the actuation channel amplitudes. Thus, the actuation amplitudes cannot be unambiguously derived for a specific motion vector. To resolve this ambiguity, amplitudes are selected so that the maximum amplitude of all channels is smallest. Thus, if there are multiple combinations that lead to the same movement, the combination is chosen where the maximum amplitude of all channels is smaller than for any other combination. Aside from resolving the ambiguity, this approach is also beneficial in terms of energy consumption and strain in the piezo elements.

For the translational part, the amplitudes can be limited to $v/\hat{v}(f)$. This limitation creates a hexagon of possible translational movement vectors (see Fig. 4.6) with the required vector (v_x, v_y) lying on it. Thus, there is only one combination of amplitudes that implements this vector and does not exceed the amplitude limitation. For the rotational part, the maximum amplitude for the required ω_φ is minimal, if the amplitudes supplied to the a- and b-channels are inverse ($u_a = -u_b$). Calculating the actuation amplitude for a specific motion vector is done using the following steps:

1. Calculate preliminary vector (u_1', u_2', u_3') for the translational movement component by:

$$\begin{pmatrix} u_1' \\ u_2' \\ u_3' \end{pmatrix} = \begin{pmatrix} 2 & 0 \\ -1 & \sqrt{3} \\ -1 & -\sqrt{3} \end{pmatrix} \begin{pmatrix} \frac{\hat{u}}{\hat{v}(f)} & 0 \\ 0 & \frac{\hat{u}}{\hat{v}(f)} \end{pmatrix} \begin{pmatrix} v_x \\ v_y \end{pmatrix} \tag{4.20}$$

2. Limit u_1, u_2 and u_3, so that the step length l is used:

$$\forall i \in \{1, 2, 3\} : u_i = \min\left(\max\left(-\frac{v \cdot \hat{u}}{\hat{v}(f)}, u_i'\right), \frac{v \cdot \hat{u}}{\hat{v}(f)}\right) \tag{4.21}$$

3. Combine with rotation:

$$\begin{pmatrix} u_{1a} \\ u_{1b} \\ u_{2a} \\ u_{2b} \\ u_{3a} \\ u_{3b} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ \frac{\hat{u}}{\hat{\omega}(f)} \cdot \omega_\phi \end{pmatrix} \tag{4.22}$$

With these calculations, the robot moves with the desired $(v_x, v_y, \omega_\varphi)$. Furthermore, the movement can be changed after an interval Δt, after which an integer number of steps have been executed. This is important for the following closed-loop control, which operates at a specific frequency and needs to change the robot's movement after each control iteration.

4.3 Closed-loop motion control

The goal of the closed-loop is precisely controlling a robot using the available sensory feedback. During nanohandling processes, tools and objects need to be brought into close proximity with only a few nanometers remaining between them. Nevertheless, unintended collisions need to be avoided as tools and handled objects are easily damaged. Therefore, closed-loop control needs to be implemented as trajectory control, i.e. the path, velocity and acceleration of each robot needs be controlled. Thus, the precise location of the robot is known at any point in time and collisions can be effectively avoided.

As the sensor data is delivered in digital form with a certain update rate f (update interval $T = 1/f$), the controller is implemented as a digital control. The control algorithm is implemented on a microcontroller embedded into the microrobot control unit (see Section 4.4). The execution of the control loop for each sensor update is called control iteration. Each control iteration consists of three steps:

1. Calculation of the current target pose $\mathbf{p}_t(t)$
2. Compensation for the sensor's latency
3. Computation of the desired movement to remove the control deviation

These steps are explained in the following sections. In addition, an auto-configuration sequence can automatically determine all parameters required to implement the described control approach.

4.3.1 Definition and calculation of trajectories

For the closed-loop controller, a trajectory is defined as a time-variant target pose $\mathbf{p}_t(t)$. Thus, for each iteration of the digital control loop, a new target pose is calculated according to the desired trajectory type. As described in Section 2, point-to-point trajectories should use a third-order motion profile. Fig. 4.10 shows such a trajectory. The covered distance $s(t)$ starts at 0 and finishes at the distance $\hat{s}$ between start and end point. The absolute values of the jerk $j(t)$, the acceleration $a(t)$ and the velocity $v(t)$ are all limited to $\hat{j}$, $\hat{a}$, and $\hat{v}$, respectively. The jerk needs to be limited in order not to create excessive vibrations during the movement. The acceleration and velocity limits stem from the motion capabilities of the robot and the line scan-based tracking properties.

As shown in Fig. 4.10, the third-order motion profile can be subdivided into seven phases. Taking symmetries into account, there are only three different time intervals:

$$t_{\hat{j}} = t_1 - 0 = t_3 - t_2 = t_5 - t_4 = t_7 - t_6 \tag{4.23}$$

$$t_{\hat{a}} = t_2 - t_1 = t_6 - t_5 \tag{4.24}$$

$$t_{\hat{v}} = t_4 - t_3 \tag{4.25}$$

Thus, phases 1, 3, 5 and 7 all last $t_{\hat{j}}$. During these phases, the jerk remains constant, either $\hat{j}$ or $-\hat{j}$. Phases 2 and 6 require $t_{\hat{a}}$ and provide a constant acceleration or deceleration with $\hat{a}$. Phase 4 is the traveling phase, i.e. the robot moves with a

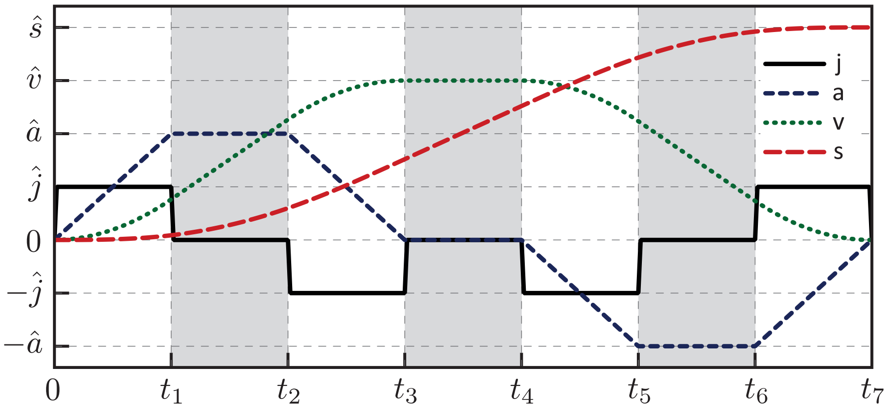


Figure 4.10: Third-order trajectory motion profile

constant velocity $\hat{v}$ for the duration $t_{\hat{v}}$. Dependent on the traveling distance and the limits, the maximum acceleration or velocity might not be reached for a specific movement. In these cases, the duration of the corresponding phases is 0.

The three durations $t_{\hat{j}}$, $t_{\hat{a}}$, $t_{\hat{v}}$ and the maximal jerk $\hat{j}$ fully describe the timing behavior of the trajectory. Together with a movement direction α, they can be used to iteratively calculate the steps of a trajectory in two-dimensional space. During a trajectory movement, the current time t can be used to determine the current phase and dependent on the phase, the current jerk $j(t)$ is either $-\hat{j}$, 0 or $\hat{j}$. Then, the acceleration $a(t)$, the velocity $v(t)$ and the position $\mathbf{p}(t)$ can be calculated using integration, i.e. summing the values at each control iteration. This calculation can be done online in real-time on the used microcontroller (see Section 4.4). Thus, the key challenge is to calculate the different durations.

The durations are calculated in multiple steps similar to the approach of Lambrechts et al. (2005). First, $t_{\hat{j}}$ is calculated ignoring the limits on acceleration and velocity. Second, $t_{\hat{a}}$ and $t_{\hat{v}}$ are calculated reducing $t_{\hat{j}}$ in order to include these limits. Third, the maximal jerk $\hat{j}$ is slightly reduced in order for all durations to be multiples of the sensor and control update interval T. Although this slightly prolongs the trajectory duration, it is a necessary step because the robot movement can only changed once for each control iteration.

4.3.2 Latency compensation

The position tracking system as well as the open- and closed-loop control algorithms require a certain amount of execution time. Thus, there is a latency between the actual position measurement and the reaction to the new information by the robot. Fig. 4.11 shows the timing behavior of the system. The simple form of tracking a square-shaped pattern with a horizontal and a vertical line scan is used. The tracking operates at an update rate of 1 kHz.

The result of each line scan roughly represents the tracked object's position at the time when half the scan is completed (see Section 3.2.2). Thus, the x-scan reports the position of the robot at the time t_x whereas the y-scan reports the position at time t_y. After completing the line scan in the y-direction and the subsequent calculation to update the tracked object's position and estimated velocity, the data is transferred to the visual servoing controller using the CAN protocol. Then, the closed-loop controller calculates desired motion velocities which are transformed into actuation parameters by the open-loop controller. The actuation parameters are then applied to the signal generation architecture (see Section 4.4) using the serial peripheral interface (SPI) protocol. Once this transfer is complete, the new actuation parameters are finally applied to the robot at time t_r. Thus, there is a latency for the x-coordinate amounting to $T_x = t_r - t_x$ and a latency for the y-coordinate amounting to $T_y = t_r - t_y$.

As described in Section 2.5.4, such latencies have a significant influence on the control performance and stability. At the time, the robot actually reaches its destination, the delayed sensor information misleads the controller to move the robot further, i.e. away from the target position. In order to achieve a high closed-loop control performance, this latency needs to be compensated for (see Section 2.5.4). This is possible, because all components are implemented in hardware leading to a well-defined latency with minimal jitter. To this end, the robot's movement during the latency intervals T_x and T_y is estimated based on the open-loop model. The estimated motion is added to the current tracking result and is thus an approximation of the current position. This approach is a variant of the Smith predictor (see Section 2.5.4) and can deliver a high performance with the downside of strong dependency of

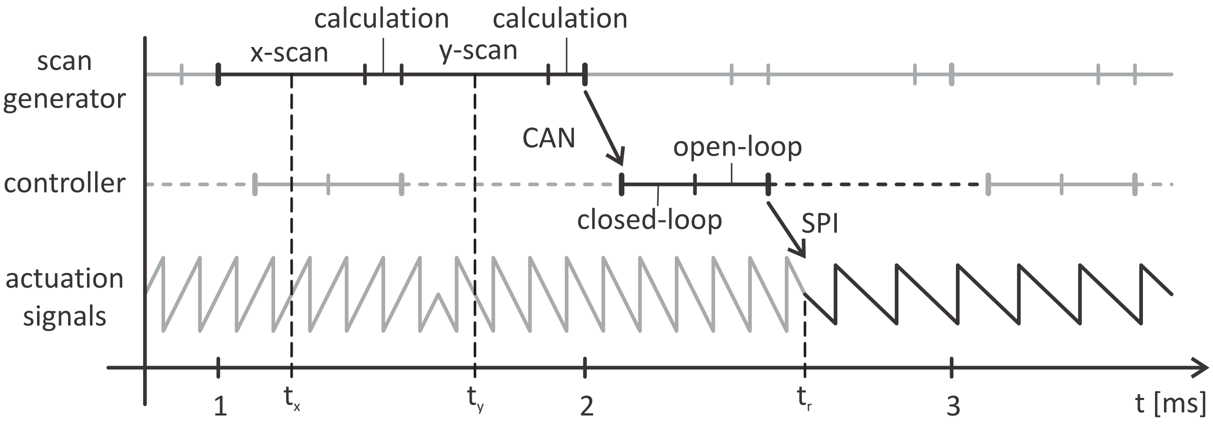


Figure 4.11: Timing diagram for visual servoing with line-scan based tracking.

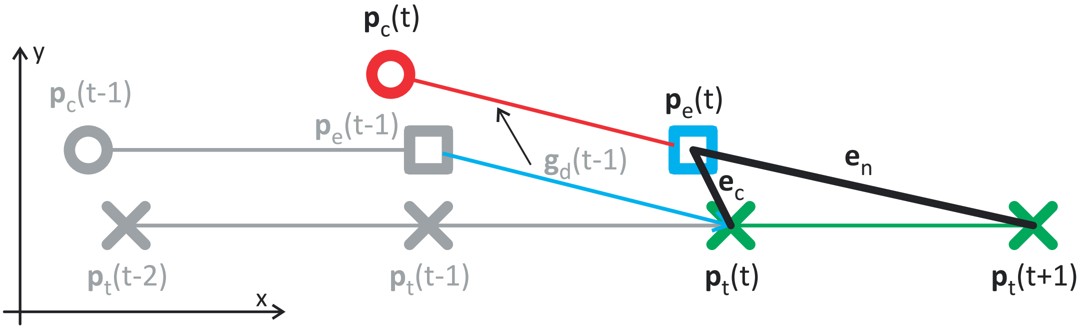


Figure 4.12: Latency compensation during trajectory control.

the reliability of the latency and the open-loop model. If the latency or the open-loop behavior changes, the control loop can become inefficient and even unstable. However, in a well-controlled nanohandling architecture, both characteristics can be assumed to be constant.

For each control iteration a desired motion vector $\mathbf{g}_d(t)$ is to be calculated and applied at time t_r. To compensate for the latency, the previous $\mathbf{g}_d(t)$ are saved. During the control iteration, an estimated pose $\mathbf{p}_e(t)$ is calculated based on the updated pose measurement $\mathbf{p}_c(t)$. For the example shown in Fig. 4.11, the following equations compensate for the latency:

$$\mathbf{g}_d(t) = \begin{pmatrix} v_x(t) \\ v_y(t) \end{pmatrix} \tag{4.26}$$

$$\mathbf{p}_c(t) = \begin{pmatrix} x_c(t) \\ y_c(t) \end{pmatrix} \tag{4.27}$$

$$\mathbf{p}_e(t) = \begin{pmatrix} x_e(t) \\ y_e(t) \end{pmatrix} = \begin{pmatrix} x_c(t) + T \cdot v_x(t-1) + (t_r - T - t_x) \cdot v_x(t-2) \\ y_e(t) + (t_r - t_y) \cdot v_y(t-1) \end{pmatrix} \tag{4.28}$$

The calculation of $x_e(t)$ is more complex, because $t_x - t_r > T$, i.e. the latency for the x-coordinate is longer than the sensor's update interval. Note, that the rotational movement is omitted as it cannot be measured with just the two line scans. Fig. 4.12 shows a simplified schematic of this approach. At time t, the measured pose $\mathbf{p}_c(t)$ is received, which is outdated because of the latency. The algorithm then calculates the robot's estimated pose $\mathbf{p}_e(t)$ by estimating the robot's movement during the latency interval. This estimated pose closely matches the real pose of the robot with only a small error introduced by the inaccuracy of the movement model. As the estimation is updated with each new pose measurement during each update, this error does not accumulate.

4.3.3 Computation of the desired movement

The goal of the closed-loop controller is to compute a motion vector that keeps the robot on the calculated trajectory. Building on the open-loop controller described above, the closed-loop controller needs to calculate a vector $\mathbf{g}_d(t)$ of desired velocities. The open-loop controller will execute this vector for the duration $\Delta t = T$ until the next control iteration. Using the estimated pose $\mathbf{p}_e(t)$, two control deviations can be calculated (see Fig. 4.12). The distance $\mathbf{e}_n$ between $\mathbf{p}_e(t)$ and the following trajectory position $\mathbf{p}_t(t+1)$ is the movement that the robot should execute in order to move along the trajectory. The corresponding velocities are calculated by dividing $\mathbf{e}_n$ by the update interval Δt. Additionally, the distance $\mathbf{e}_c$ between $\mathbf{p}_e(t)$ and $\mathbf{p}_t(t)$ can be used as it is represents movement errors of the open-loop control. The integral of this distance divided by Δt is added to the control values forming an integral controller part. This leads to the PI-controller with the control parameters K_P and K_I:

$$\mathbf{g}_d(t) = \begin{pmatrix} v_x \\ v_y \\ \omega_\varphi \end{pmatrix} = \frac{K_P}{\Delta t} \cdot [\mathbf{p}_t(t+1) - \mathbf{p}_e(t)] + \frac{K_I}{\Delta t} \cdot \sum_{i=0}^{n} [\mathbf{p}_t(t) - \mathbf{p}_e(t)] \tag{4.29}$$

As the open-loop controller is assumed to correctly model the motion behavior of the robot, $K_P = 1$ leads to a reliable motion along the trajectory. For K_I, a value of 0.02 has proven to quickly remove the errors generated by inaccuracies of the open-loop model (see Section 5.2). The validation measurements in Section 5.2 will show, that a high control performance and robustness is achieved based on the open-loop model, latency compensation and trajectory calculation.

4.3.4 Auto-configuration

The mechanical integration of nanorobotic systems into electron or other microscopes is straightforward. Mobile nanorobots in particular are fast to setup, as they simply need to be put onto a flat working surface. This effortless integration, however, only extends to teleoperation and not to the trajectory-controlled closed-loop positioning. For the closed-loop control described in this chapter to work, several properties of the tracking system have to be configured. This includes resolution, noise, update rate, latency, scale and orientation. The goal of auto-configuration is the automated determination of these parameters so that a robot can be integrated into a microscope without the need of manual configuration.

Automated characterization of sensor system

The first step of the auto-configuration is the characterization of the sensor system itself, i.e. the sensor's noise, resolution, update rate, and latency. As the robot is supposed to work with a multitude of sensor systems, few assumptions about the sensors can be made. The **update rate** is measured with a timer while capturing several sensor updates. The update rate is called f whereas the update interval is called $T = 1/f$. The standard deviation of f, also called **jitter**, can also be determined during this measurement. The **noise** is determined by recording a sequence of sensor updates and evaluating the maximum and standard deviations. After that, the robot performs linear movements of increasing length starting with single steps until a significant change of the sensor value is detected. The smallest measurable movement is used as **resolution** of the sensor.

The precise determination of the sensor's **latency** T_0 is done using two steps. First, the latency is measured in multiples of T. The robot starts a movement synchronously to a sensor update. The movement's duration is chosen as T and the distance is long enough so that it can be reliably determined with the sensor's resolution, e.g. 20 times the resolution. This is shown in Fig. 4.13. Starting with 0 for the next update, sensor updates are counted, until the final position is reached, i.e. sensor data remains constant. If n sensor updates are counted, then:

$$(n-1)T < T_0 \leq nT. \tag{4.30}$$

The movement conducted by the robot is highly dynamic, as a significant displacement has to be achieved within one update interval, which can be as short as 1 ms. Thus, the motion behavior cannot be assumed to be linear as depicted in Fig. 4.13. Furthermore, each sensor update is affected with noise. Thus, the position at the intermediate sensor update A cannot be reliably used to calculate the exact sensor latency. Instead, an approach based on a delayed movement and a binary search is used. With this approach, the latency is determined by delaying the start of the

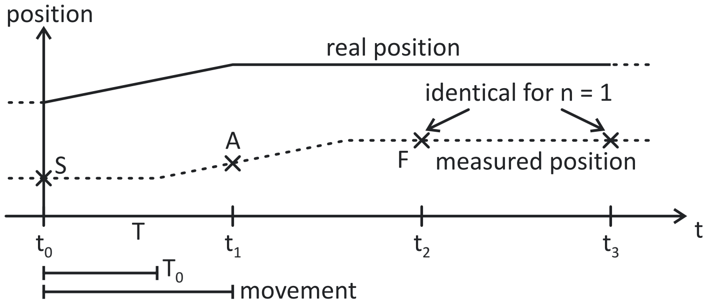


Figure 4.13: Measuring the sensor latency in multiples of T. Movement is completed with sensor update F.

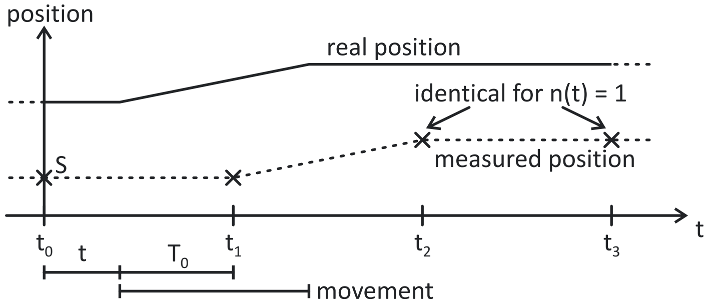


Figure 4.14: Delaying the movement by t in order to determine the latency precisely.

movement by the time t, with:

$$0 < t < T. \tag{4.31}$$

This added delay introduces additional, artificial latency (see Fig. 4.14). For any delay, the procedure to calculate the latency in multiples of T can be repeated, i.e. a delay dependent $n(t)$ can be calculated. If $T_0 + t < n(0)T$, this repeated calculation will yield the same result ($n(t) = n(0)$). The goal is finding a t that acts as a threshold for $n(t)$:

$$n(t) + 1 = n(t + \epsilon) \text{ for } \epsilon \to 0, \tag{4.32}$$

as shown in Fig. 4.14. With such a t, the precise determination of the sensors latency can be done:

$$T_0 = n(0)T - t. \tag{4.33}$$

To find this t efficiently, binary search is used. Binary search iteratively finds t in an interval $[a, b]$. Initially, $a = 0$ and $b = T$. During each iteration, the middle of the interval $c = (a + b)/2$ is tested. If $n(c) = n(0)$, then $t > c$ and the search can be continued in $[c, b]$. Otherwise, $t < c$ and the search can be continued in $[a, c]$. Hence, each iteration cuts the interval in half. For the employed robotic system, determining t with a resolution of $T/256$ is sufficient (see Section 5.2.7), as it results in a maximum error of approx. 0.4% which is significantly lower than the accuracy of the robot's open-loop model. Thus, eight iterations are required.

Auto-configuration

In a robotic system like the one in Fig. 4.2, a mobile robot is tracked by one or more cameras as well as the SEM. For each such tracking, the robot needs to find a basic mapping between its local coordinate system C_R and the tracking coordinate system C_C (see Fig. 4.15). In the tracking coordinate system, a specific point on the robot is tracked, e.g. the LEDs mounted to the robot or the tip of the tool. The result is a pose (x, y, φ) describing the location of the tracked point in C_C. The coordinate

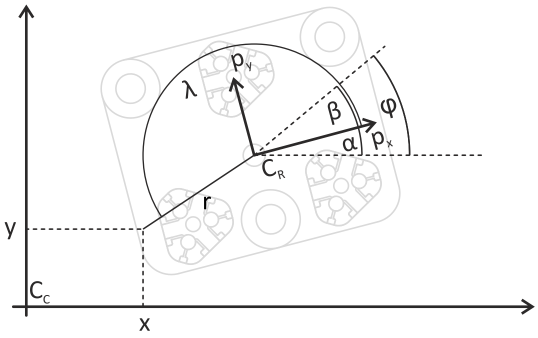


Figure 4.15: Relations between the coordinate system C_C of a vision sensor tracking the pose (x, y, φ) and the robot's coordinate system C_R.

system C_R is local to the robot. In this local coordinate system, movements are calculated along the vectors $\mathbf{p}_x$ and $\mathbf{p}_y$ for pure x- and pure y-movements.

In general, the pose of the tracked object does not correspond to the origin of C_R. Instead, there is an offset r into a direction λ between the origin of C_R and (x, y) as well as an offset β between φ and α. Additionally, dependent on whether the tracking observes the robot from above or from below, C_C might be flipped in relation to C_R.

The goal of the auto-configuration is the automatic determination of r, λ and β for each sensor. To do so, the robot has to perform a series of movements. Each of these movements needs to be long enough so that it can be precisely measured taking into account the previously measured resolution of the sensor system but needs to remain within the sensor's working range. The first movement is performed along the axis $\mathbf{p}_x$ of the robot's internal coordinate system C_R. With this movement, α can be determined and β can be calculated as φ is known. Furthermore, the scaling

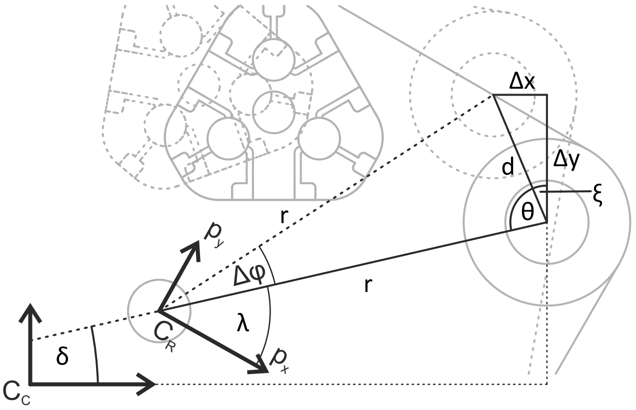


Figure 4.16: Determination of the offset r and angle λ of the tracked point to the robot's rotational center point.

of C_C can be approximated, as the step length of the robot is known. A second movement along $\mathbf{p}_y$ is used to determine whether the coordinate system is flipped. A third movement is then conducted in order to measure r and λ. This movement is purely rotational around the origin of C_R. It should be as big as possible so that the sensor's noise becomes insignificant. As a result of this movement, the pose measured by the sensor changes by (Δx, Δy, $\Delta\varphi$) as shown in Fig. 4.16. The measured poses and the origin of C_R form an isosceles triangle and the distance r can be calculated by:

$$\begin{aligned} d/2 &= r \cdot \sin \frac{|\Delta\varphi|}{2} && (4.34) \\ \Leftrightarrow \sqrt{\Delta x^2 + \Delta y^2} &= 2r \cdot \sin \frac{|\Delta\varphi|}{2} && (4.35) \\ \Leftrightarrow r &= \frac{\sqrt{\Delta x^2 + \Delta y^2}}{2 \cdot \sin(|\Delta\varphi|/2)}. && (4.36) \end{aligned}$$

The angle δ of the r in C_C can also be calculated from this measurement using the different angular relations:

$$\begin{aligned} \delta &= \frac{\pi}{2} - (\pi - \theta - \xi) && (4.37) \\ &= \frac{\pi}{2} - \left(\pi - \frac{\pi - \Delta\varphi}{2} - \tan^{-1}\frac{\Delta x}{\Delta y}\right) && (4.38) \\ &= \frac{\Delta\varphi}{2} + \tan^{-1}\frac{\Delta x}{\Delta y}. && (4.39) \end{aligned}$$

Using δ and β, the angle λ can be calculated (see Fig. 4.15). Using the `atan2` function that uses both parameters, Δx and Δy, δ can be calculated in the full range $[-\pi, \pi]$. If the coordinate system is flipped or if the rotation is performed clockwise, δ needs to be negated. With these measurements, the robotic system is fully configured and trajectory-controlled visual servoing can be used.

4.4 Microrobot control unit

In order to implement the motion control described above effectively, the robot's control algorithm as well as the signal generation need to have a short latency with limited jitter. Thus, a hardware-based real-time control unit needs to be developed. This control unit receives movement commands such as “move linearly to position X” and implements the open-loop control, closed-loop control, trajectory calculation and signal generation for the employed robot.

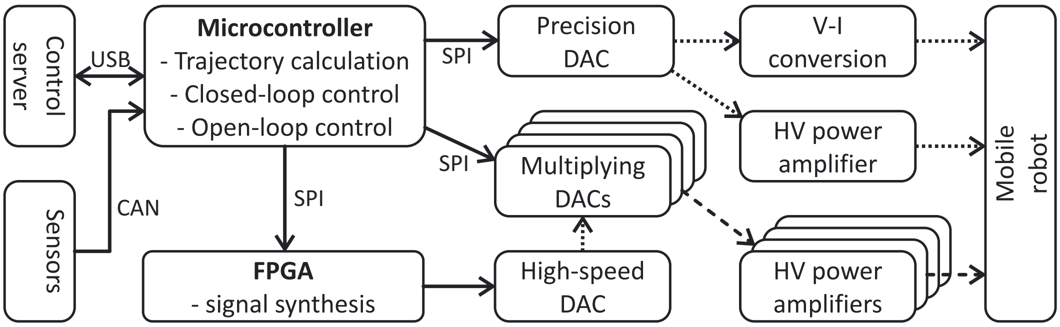


Figure 4.17: Architecture of the control unit.

The architecture of the robot control unit is shown in Fig. 4.17. The key component is a microcontroller that implements the communication to the control server, the reception of sensor signals as well as the motion control itself. The USB connection to the control server is fast although not real-time capable. Thus, the actuation commands can be assumed to be executed quickly, but a precise timing cannot be guaranteed. In contrast, the sensors signals are received via a real-time capable CAN bus. Using this bus, the maximum delay of a sensor update can be limited, which is important for a reliable control loop. The microcontroller controls two precision DACs, the output of which is amplified and used to operate possible tools attached to the robot, e.g. an electrostatic or electrothermal gripper. Furthermore, it configures an FPGA and a number of multiplying DACs that generate the required signals. The communication between the microcontroller, the DACs and the FPGA is done via the SPI protocol.

The robot itself requires six synchronous signals in order to move on the surface in three degrees of freedom (x, y and φ). As the robot is based on piezoceramic actuators, high voltages are necessary in order to create sufficient deformation. The stick-slip actuation principle demands high slew rates in order to generate enough acceleration to overcome the friction force. Table 4.1 summarizes the requirements applicable to each of the six signals.

The signals required for the robots are generated by multiple steps in different hardware components. The FPGA is programmed to perform a so-called direct digital synthesis. Thus, in addition to several control registers, it contains a 32 bit accumulator (see Fig. 4.18). The value stored in this accumulator stands for the phase of the generated signal. Value 0 represents a 0° phase, 2^{31} corresponds to 180° and $2^{32} - 1$ to almost 360°. The FPGA is clocked with a clock f of 48 MHz, which is close to the maximum frequency for the used FPGA. With each clock cycle, the phase accumulator is incremented by a variable 24 bit value v. Thus, any frequency

Parameter	Requirement
Amplitude	300 Vpp
Slew rate	300 Vpp/µs
Frequency	0-100 kHz
Current capability	100 mA
Signal shape	sawtooth-like

Table 4.1: Signal requirements for sawtooth-shaped actuation signals.

between 0 and $f_{\max}$ with

$$f_{\max} = 48\,\text{MHz} \cdot \frac{2^{24} - 1}{2^{32}} \approx 187\,\text{kHz} \tag{4.40}$$

and a frequency resolution Δf of

$$\Delta f = \frac{48\,\text{MHz}}{2^{32}} \approx 11.18\,\text{mHz} \tag{4.41}$$

can be generated. The combination of a 32 bit accumulator and a 24 bit increment enables the generation of signals in the required frequency range at a high resolution. The highest 12 Bits of the phase accumulator are converted to an analog sawtooth signal using a high-speed DAC. The output of a DAC is a signal with an amplitude of 3 Vpp and the set signal frequency f_s. For increased flexibility, the output of the FPGA can also be used for addressing a signal memory. The data output of this signal memory is then connected to the high-speed DAC. In this configuration,

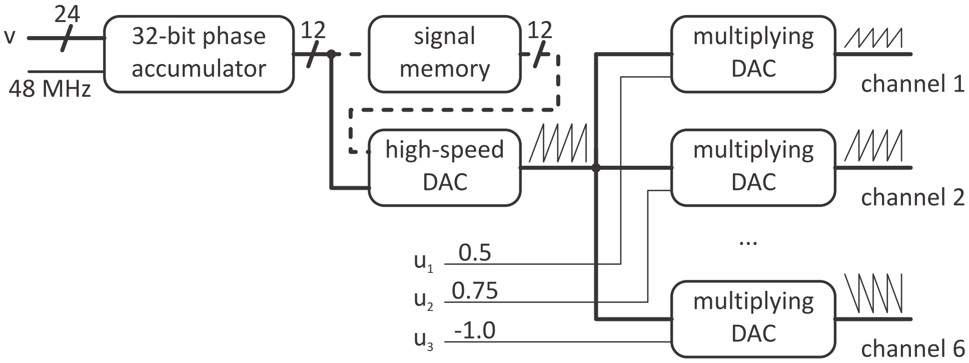


Figure 4.18: Generating synchronous periodic signals with an arbitrary frequency and different amplitudes.

the FPGA in conjunction with the memory and the DAC can generate arbitrary periodic signals.

In order to generate signals for all actuation channels of the robot, the generated signal is passed through six multiplying DACs. Each of these DACs performs a so-called four-quadrant multiplication, i.e. the input signal can be multiplied with any value between -1 and 1. This allows for changing the signal's amplitude as well as polarity individually for each channel. The DACs do not influence the signal's phase or frequency. Thus, all channels are synchronous.

The last step in the signal generation process is the amplification. Analog high-voltage amplifiers with with a gain of 100 are used to convert the signal into the required voltage and current range.

4.5 Conclusions

In this chapter, a new and flexible nanorobot as well as its control were designed. The nanorobot combines the long range of step-wise actuators with the smooth motion of scanning actuators by significantly reducing the step length and the accelerated masses. In addition, the mobile nanorobot can be effortlessly integrated into different microscope systems as the only requirement is a flat working surface. The robot's response to specific actuation amplitudes was modeled based on a combination of theoretic analysis and system identification. Inverting the developed model, an open-loop model was derived. The high accuracy of this model will be shown in the next chapter. Based on the accuracy of the open-loop control as well as the reliable latency of the line scan-based tracking, a closed-loop controller with latency compensation enables visual servoing along third-order trajectories. Thereby, the exact behavior of the robot including position, velocity, acceleration and jerk are precisely controlled during all movements. An auto-configuration sequence measures the tracking's configuration and characteristics. Thus, after the nanorobots are integrated into a variety of microscopes, visual servoing can be used without manual configuration.

5 Validation

This chapter contains validation measurements for both the position tracking based on SEM line scans and the automated motion control of the mobile nanorobots. All measurements were conducted using one of the two robotic setups shown in Fig. 5.1. Both setups are mounted on the motorized stage of a Zeiss LEO 1450 SEM. The motorized stage features five degrees of freedom and is used to coarsely position either setup under the electron gun. The first setup used for validating the line scan-based tracking features a scanning nanomanipulator developed by Physik Instrumente (PI), Germany. This manipulator positions a specimen stub in three degrees of freedom. Each degree of freedom has a range of 50 μm and is equipped with internal capacitive sensors which allows for motion with a 0.1 nm resolution. During short-term experiments, drift effects can be neglected and the actuators have an accuracy of 10 nm. The second setup uses two mobile nanorobots, one acting as a stage robot, the other carrying a microgripper. Specimen stubs can either be mounted to the fixed working platform or to the stage robot.

5.1 Line scan-based tracking

The first validation step proves the performance of the line-scan based tracking. This includes the accuracy and noise of the tracking as well as its robustness against changing focus, brightness and contrast. Furthermore, the tracking's working range, possibility to directly track objects as well as the capability of tracking objects moving at high velocities are verified. The tracking of additional degrees of freedom such as rotations and depth measurements is shown.

5.1.1 Directly tracking objects

To validate that objects can be directly tracked, line scan-based tracking was used with the particle shown in Fig. 5.2a. The particle is almost circular and about 2 μm in diameter. The particle does not exhibit straight edges that could be used for edge tracking. Instead, the center of the particle is tracked by two orthogonal line scans. The particle is roughly symmetrical but exhibits a certain amount of shape deviations. To evaluate the robustness of the line scan-based tracking, line

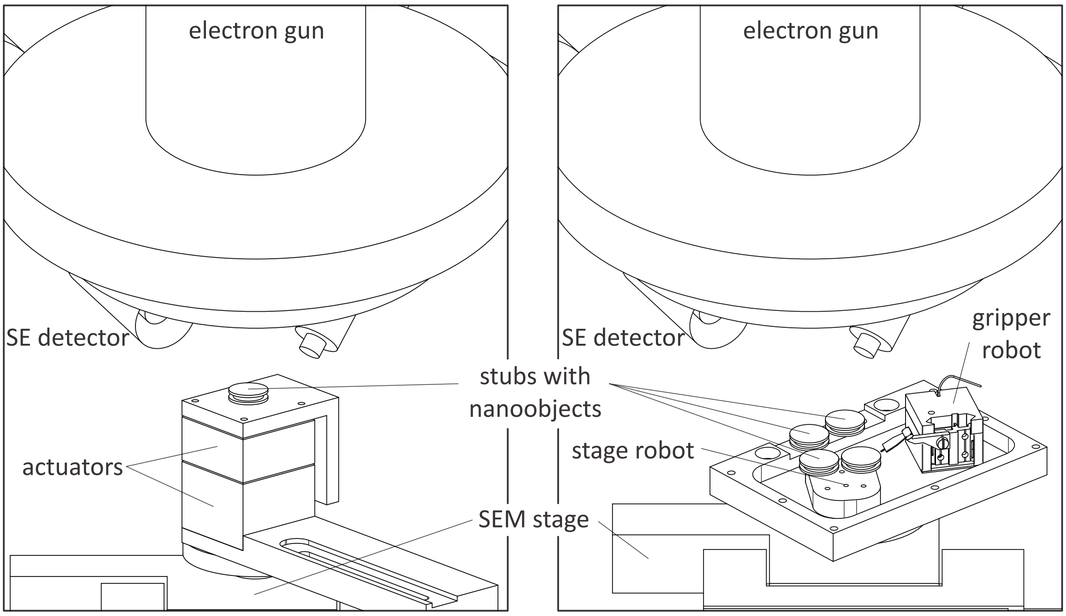


Figure 5.1: Test setups used for validation: Left) Setup with scanning actuators. Right) setup composed of two mobile nanorobots.

scan pairs are not only conducted over the particle's center (see Fig. 5.2b) but also over other positions of a 2×2 µm^2 test area. The CoG calculation is then used to determine the pattern's center from each line scan pair.

Fig. 5.3 shows the distances between the measured position and the real center of the particle, i.e. the measurement error. The x- and y-coordinates show the center of the line scans with respect to the particle's real center. For offsets in the range of ±0.6 µm, the particle's center is measured with an error of less than 50 nm. The

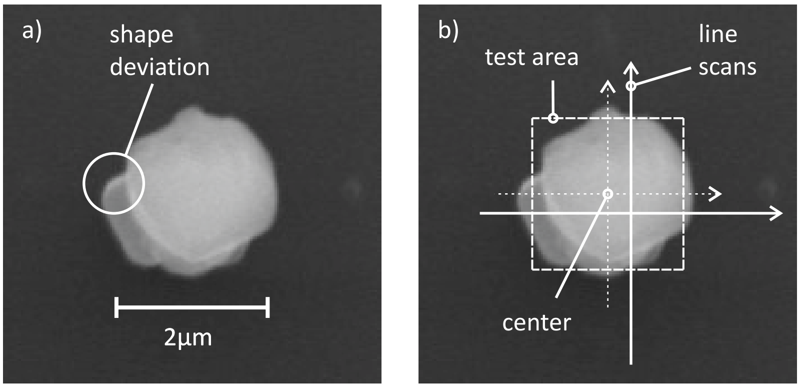


Figure 5.2: Tracking a particle with line scans: a) Particle with shape deviation. b) Line scans are conducted at different locations in a 2×2 µm^2 test area.

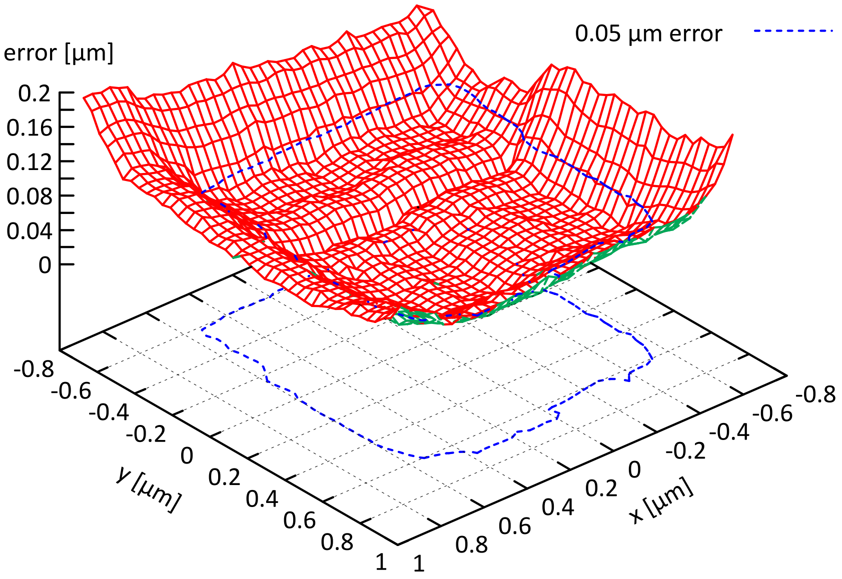


Figure 5.3: Measurement error when tracking a particle with line scans.

shape deviation in Fig. 5.2 is directly translated into an increased measurement error for $y < -0.1$ µm, virtually independent of the x-offset. The tracking error increases significantly for offsets outside the ±0.6µm-range. Thus, the particle should not move further than 0.6 µm between a line scan and the next orthogonal line scan (see Section 3.2.3). Therefore, the particle size s should be assumed to be 1.2 µm for the calculation of the maximum velocity and acceleration.

For the rest of the tracking validation, this particle is used, unless otherwise noted.

5.1.2 Accuracy and noise

To verify the tracking's accuracy, measurement results are compared to the employed PI nanopositioner. The positioner's internal, calibrated, capacitive position sensor is used for closed-loop position control. As the entire measurement interval is shorter than 5 seconds, drift effects can be neglected (compare Section 2.3.4). The positioner is set to perform several 242 nm steps and the positions calculated with the internal sensor and the SEM-tracking are recorded simultaneously (see Fig. 5.4). A step length of 242 nm is used because it has an easy representation in the nanopositioner's coordinate system. Fig. 5.4a shows that all 10 steps are tracked precisely as the two curves are identical. The magnification of a single step (Fig. 5.4b) elucidates that the SEM-tracking has a slightly higher noise level but nevertheless precisely resolves the step, including the dynamic settling behavior. Fig. 5.4c shows the deviation between the two sensors which remains below 25 nm. Furthermore, if the noise is reduced by averaging, the deviation between tracking and internal sensors remains

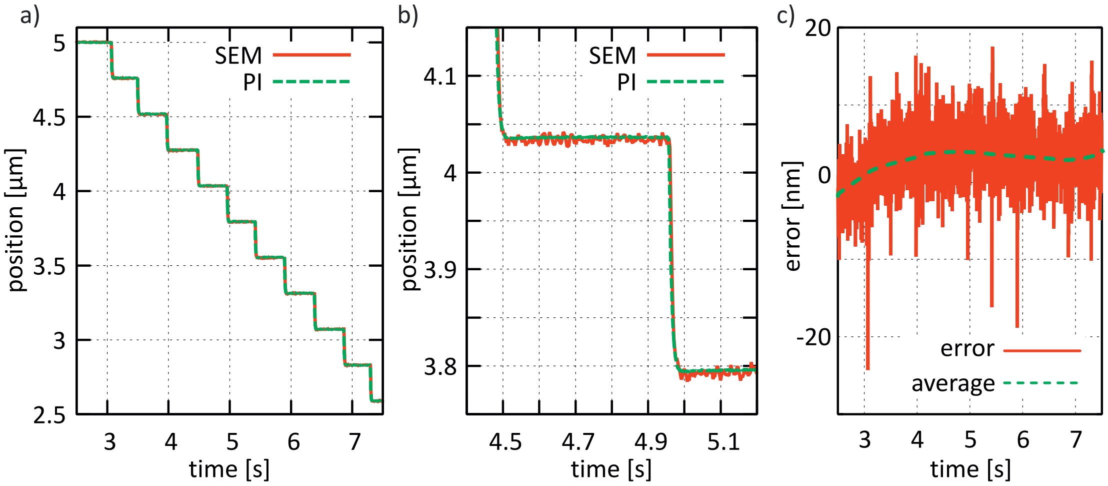


Figure 5.4: Reference measurement between SEM tracking and internal sensor of piezo-based nanopositioner developed by PI, Germany.

below 10 nm. Thus, the accuracy of the line scan-based tracking is better than 10 nm. High-frequency distortions as described in Section 2.2.1 are not present.

5.1.3 Working range

With the developed external scan generator, precise line scans can be conducted anywhere within the SEM's working area. The SEM's working area is determined by the magnification setting of the SEM. At different magnification settings, the ±10 V applied by the scan generator correspond to a different area. If set to a 500× magnification, the area is about 250×250 µm^2. Fig. 5.5 shows, that high-resolution measurements are possible anywhere within this area. Manually operating the

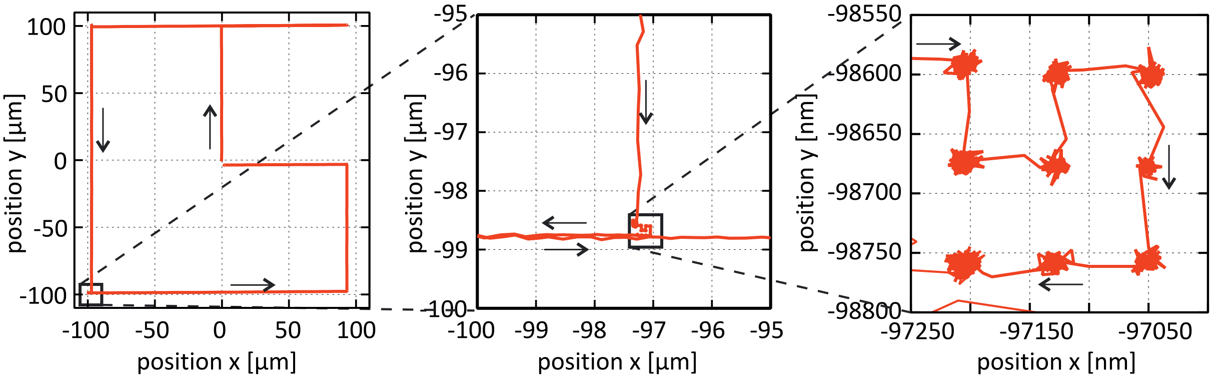


Figure 5.5: Tracking movements over long travel ranges.

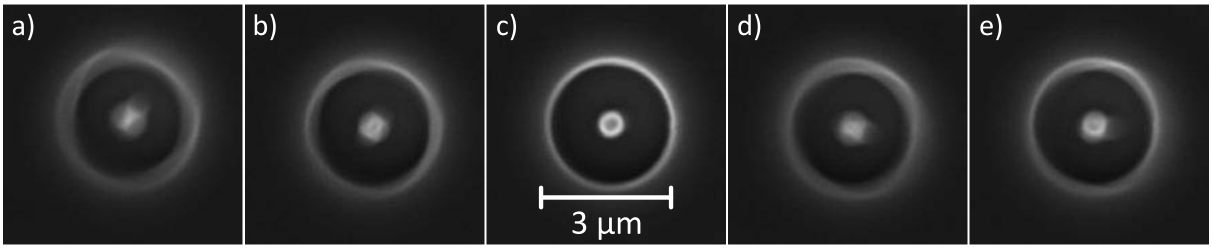


Figure 5.6: High-resolution images acquired in the corners of the SEM's working area. a) Top-left, b) bottom-left, c) center, d) bottom-right and e) top-right.

stage of the SEM, the particle is moved roughly to the corners of a 200 µm square. At each corner of this square, small movements (75 nm) are conducted with the nanopositioner based on its internal sensors. The small movements are measured correctly and with the same level of noise in each corner.

For high-resolution SEM imaging, objects have to be located in the center of the working area. The reason is that high deflections of the electron beam reduce the beam quality, i.e. the size and shape of the focus point. Fig. 5.6 shows high-resolution images of a pattern located in different corners of the 250×250 µm^2 working area. Especially in the top-left and bottom-right corners, the image reveal a significant amount of astigmatism.

5.1.4 Robustness to changing focus and astigmatism

The results from the tracking over long working ranges show that the tracking is robust against changing focus and astigmatism. Both conditions change substantially when the object is tracked in the different corners of the 200 µm square (see Fig. 5.6). Nevertheless, tracking is possible and the performance does not decrease noticeably.

5.1.5 Robustness to changing contrast and brightness

To show the approach's robustness to changing contrast and brightness, the corresponding SEM settings are changed while continuously tracking a pattern. Fig. 5.7a shows the tracking result during nine changes to the brightness and contrast setting. Changing the SEM's setting leads to an immediate change of either the gain or the offset of the detector signal. As the changes were not synchronized to the scanning, they were either performed during the x-scan or during the y-scan. The result of this scan shows a slight deviation of up to 50 nm. Afterwards, the threshold of the center of gravity calculation is automatically adapted to the measured medium

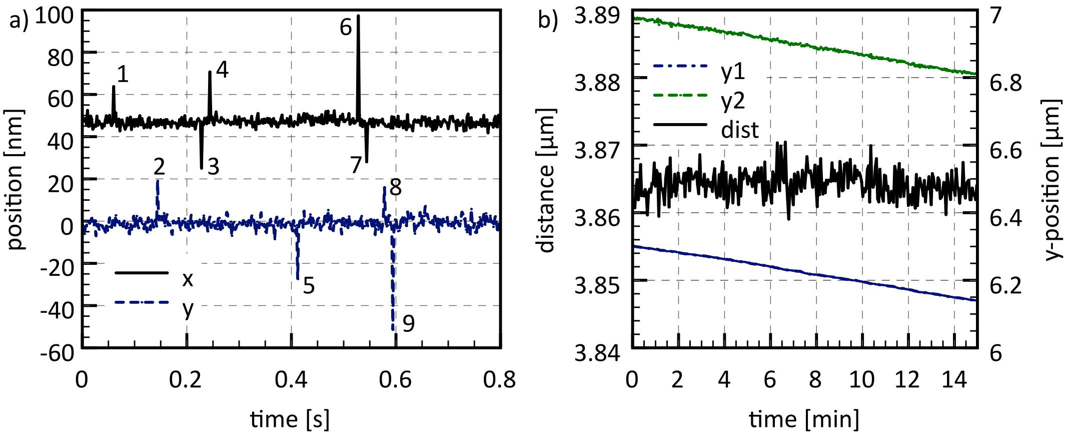


Figure 5.7: a) Tracking during step-wise changes to brightness and contrast, b) measuring robot drift and tracking drift.

gray value and the tracking result is correct again. The measurement shows that tracking is highly robust against changing brightness and contrast.

5.1.6 Drift

Two kinds of drifts can be distinguished. First, there is a certain amount of drift within the setup, e.g. induced by temperature variations. This drift leads to movements of the different objects relative to each other and to the electron column. The drift speed is between a few nm and a few µm per minute and does not pose a challenge for the line scan-based tracking. Second, the tracking itself can create drift, which is caused by unintentional electron beam-induced deposition on the tracked object. Fig. 5.7b shows a measurement of both drift types. Two patterns in close proximity, i.e with a distance of approx. 3.86 µm, are tracked. The close proximity makes thermally-induced drift between the two patterns negligible. One pattern is tracked with an update rate of 1 kHz, whereas the other is scanned only once per second. The second pattern can be used as reference because scanning once per second does not create measurable beam-induced depositions. Over the shown 15 min interval, both patterns drift by about 200 nm along the y-axis. This drift is caused by the setup itself. The measured distance between the two patterns, however, remains virtually constant. Thus, despite possible electron beam-induced depositions, the tracking does not cause measurable drift.

5.1.7 Maximum movement speed and acceleration

To validate the maximum movement speed, the scanning nanomanipulator in the validation setup is set to perform a periodic sine-shaped movement along the x-axis. Such a movement is well-suited for this validation, because speed and acceleration are precisely controlled by the sine-function and its derivations. For a given movement amplitude $\hat{x}$ at a frequency f, the position $x(t)$ is defined by:

$$\begin{aligned} x(t) &= \hat{x}\sin(2\pi f t) && (5.1) \\ \dot{x}(t) &= v(t) = 2\pi f \hat{x}\cos(2\pi f t) && (5.2) \\ \ddot{x}(t) &= -4\pi^2 f^2 \hat{x}\sin(2\pi f t) && (5.3) \end{aligned}$$

Fig. 5.8 shows the successful measurement of a 38 µm-long movement at a frequency of 25 Hz. Thus, with $\hat{x} = 19\,\mu\text{m}$ and $f = 25\,\text{Hz}$, the maximum velocity and acceleration can be calculated:

$$\begin{aligned} \hat{v} &= 2\pi f \hat{x} \approx 2.984\,\text{mm/s} && (5.4) \\ \hat{a} &= 4\pi^2 f^2 \hat{x} \approx 468.8\,\text{mm/s}^2. && (5.5) \end{aligned}$$

This corresponds to the measured maximum velocity of approx. 3 mm/s. The particle was tracked with an update rate of 1.08 kHz ($T_L \approx 0.4\,\text{ms}$ and $T_C \approx 0.1\,\text{ms}$). With $s = 1.2\,\mu\text{m}$ (see Section 5.1.1), the maximum trackable acceleration according to Section 3.2.4 is:

$$a_{\max} = \frac{s}{(1.5 \cdot T_L + T_C)^2} \approx 2.5\,\text{m/s}^2, \qquad (5.6)$$

which is significantly higher than $\hat{a}$. In fact, when positioning based on its internal sensors, the nanopositioner is not able to move fast enough so that tracking is inter-

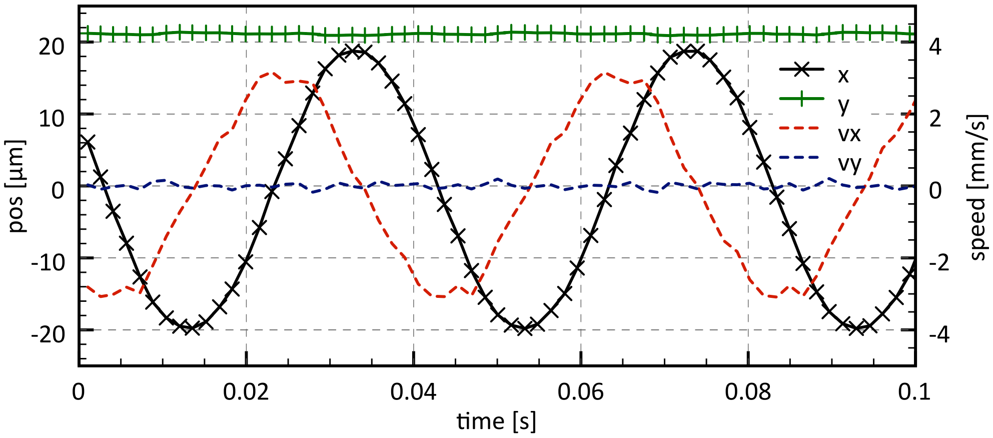


Figure 5.8: Tracking a particle at high velocities.

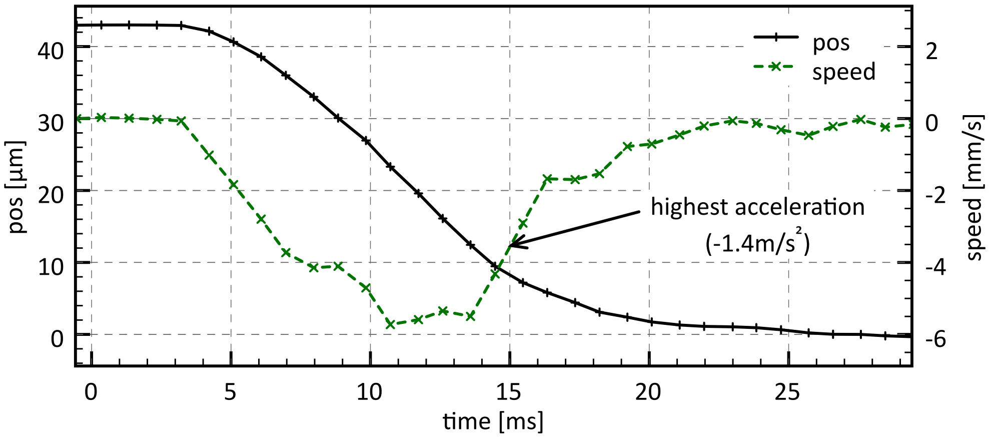


Figure 5.9: Tracking a step function of the scanning actuator.

rupted. This is validated by performing a full range step as fast as possible as shown in Fig. 5.9. The step was successfully tracked and the highest measured acceleration is approx. -1.4 m/s^2. Thus, any movement conducted with the nanopositioner can be tracked.

5.1.8 Tracking rotations

With the cross-shaped pattern shown in Fig. 3.21, it is possible to track rotations using a square-shaped scanning sequence. A corresponding measurement on a 3 µm-sized pattern is shown in Fig. 5.10. The pattern is scanned with a square-shaped beam movement with a 2 µm edge length. The scan generator's scanning rotation is changed in 1° steps. Each rotational step is clearly visible. Low-pass filtering is used to reduce the noise of the rotational measurement, leading to a resolution of approx. 0.05°. This is a high resolution considering that the opposite sides of the scanned square which are effectively used to determine the angle have a distance of only 2 µm. If a rotational measurement with higher resolution is required for a specific application, two patterns with a substantial distance between them can be tracked. The tracking's x- and y-coordinates are not influenced, as the pattern is located in the center of the SEM's coordinate system. Thus, the measurement proves the rotational measurement capabilities as well as the independence of translational and rotational tracking.

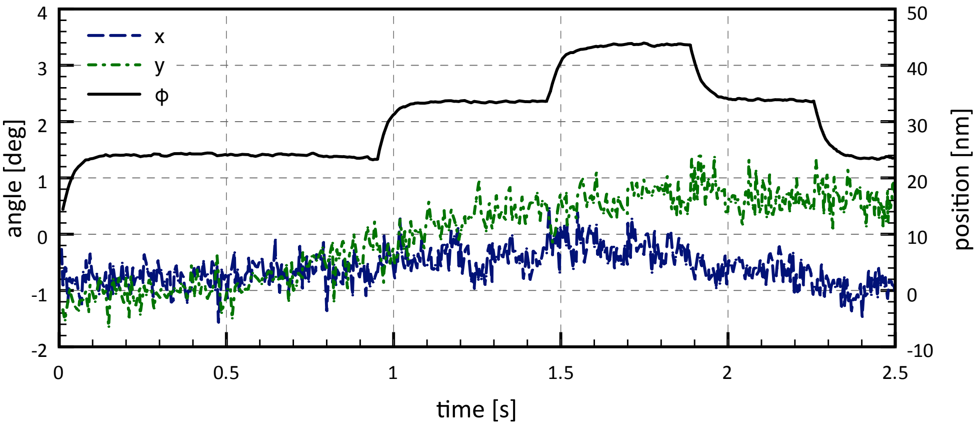


Figure 5.10: Tracking rotational movement.

5.1.9 Depth measurement

For the depth measurements, two basic techniques have been described: Depth from defocus and depth from focus. The depth from defocus is validated using the setup based on the scanning nanopositioner (see Fig. 5.1). An initial focus sweep is conducted and a linear interpolation done in the linear defocus area. Fig 5.11 shows two sets of ten 500 nm steps performed along the z-axis. After each step, the nanopositioner is stopped for about 1 s, so that the noise of the depth detection can be evaluated. After each set, the positioner is brought back to its initial position. All steps are reliably detected with minimal nonlinearity. The 5 µm step back

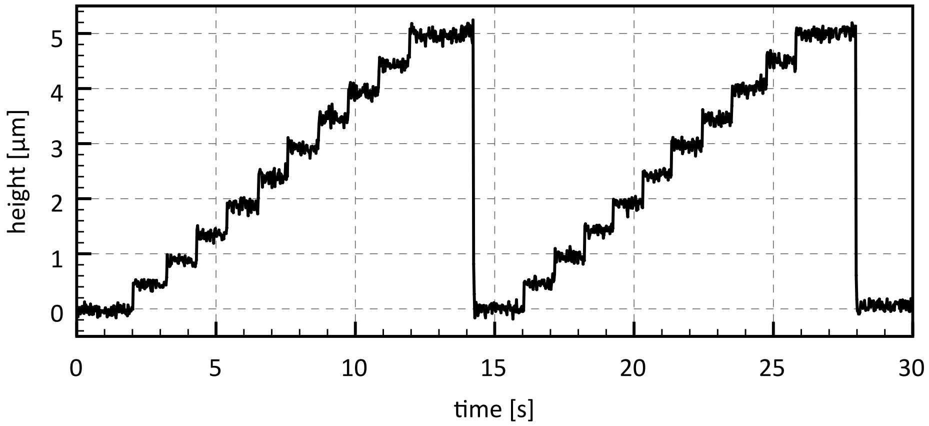


Figure 5.11: Height of the tracked pattern measured by depth from defocus when conducting two sets of ten 500 nm steps.

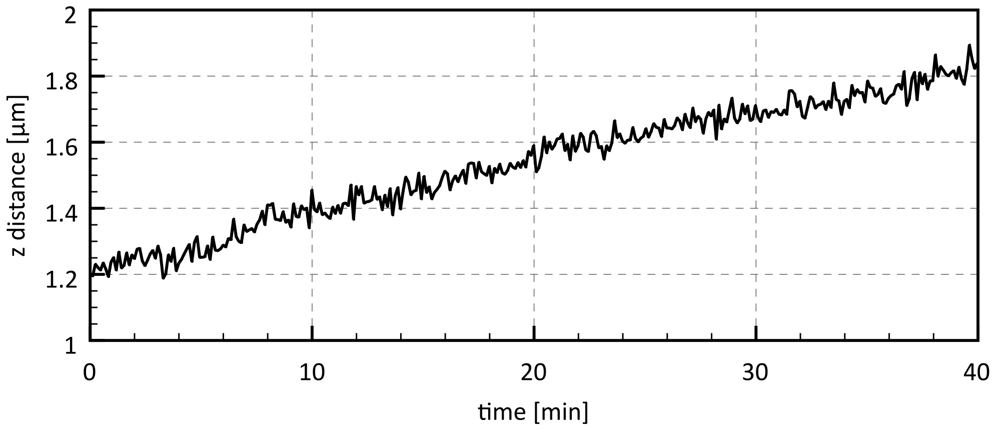


Figure 5.12: Z-distance between two objects repeatedly measured by depth from focus over a 40 min interval.

to the initial height as well as the second set of 500 nm steps show an excellent repeatability, despite the comparatively long 30 s measurement interval.

For validating the depth from focus approach, the validation setup with mobile nanorobots is used. A microgripper equipped with tracking patterns is brought into close proximity to a particle tracked on a specimen stub. The specimen stub is aligned at an angle, so that gripper and particle can be aligned on the same height. Fig. 5.12 shows the height difference measured by depth from focus every 6 s over a 40 min interval. The measurements clearly show that the two tracked objects drift at a constant rate of about 20 nm/h with respect to each other. The z-distance changes from an initial 1.2 µm to 1.8 µm. This behavior is expected from other drift measurements (see Fig. 2.5 and 5.7b). If the drift is removed using a linear background subtraction, the measurements show a standard deviation of only 27 nm, showing the high resolution of the depth from focus approach.

5.2 Automated motion control

The previous sections show that line scan-based tracking can act as a powerful position sensor for visual servoing. Now, the robot and its control need to be analyzed to show that they exploit the tracking's potential and can be used for high-speed positioning.

5.2.1 Motion behavior of robot

Several key characteristics of the robot's motion behavior are described in Section 4.2.2. The robot's step size increased linearly with the signal amplitude with a minimum amplitude of 50 Vpp. The step size is virtually independent of the step frequency. With the maximum step size of 170 nm along the robot's x-axis and the maximum frequency of 80 kHz, this leads to a velocity limit of about 13.6 mm/s.

Fig. 5.13a shows five stick-slip steps executed along the robot's y-axis at a frequency of 10 Hz. The system identification setup based on three laser interferometers (see Fig. 4.7a) was used to measure the robot's behavior. With the used maximum amplitude, the robot's step size along the y-axis is 150 nm. The actual displacement created during the stick phase is 200 nm, but a 50 nm backlash occurs during the slip phase. The backlash occurs because the acceleration generated by the piezo segments is insufficient to create a perfect slip motion with the small rotational inertia of the steel spheres. This, however, is the case for virtually all stick-slip actuators (see Section 2.3.2) and does not necessarily limit their performance. The vibrations created by this backlash subside quickly due to the small movement and the robot's high stiffness.

Fig 5.13b shows a scanning movement along a circle with a 100 nm diameter, also measured by the system identification setup. 100 nm were chosen, because it is the maximum displacement along the y-axis with the chosen ±150 V limit on each actuation channel. Note that this corresponds to the 200 nm movement observed during the stick phase in Fig. 5.13a, as the actuator moves from a displacement of

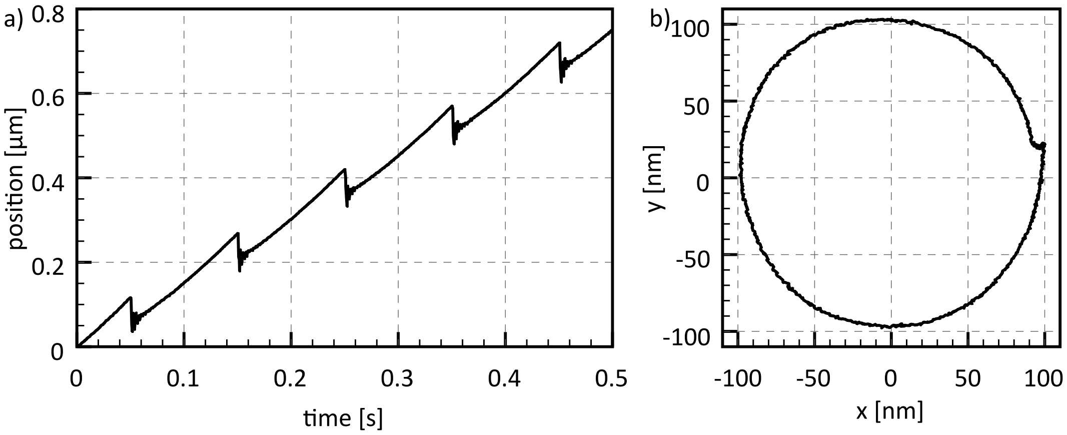


Figure 5.13: Robot's motion characteristics: a) 5 individual stick-slip steps, b) scanning motion along a 100 nm-diameter circle.

-100 nm to a displacement of +100 nm. This leads to a movement resolution r:

$$r = \frac{100\,\mathrm{nm}}{150\,\mathrm{V}} \approx 0.67\,\frac{\mathrm{nm}}{\mathrm{V}}. \tag{5.7}$$

The deviation from the circle is caused by hysteresis effects. After the clockwise circular motion, the platform slowly drifts back to its original position. Thus, although a sub-nm resolution is achieved, the movement accuracy without a sensor is limited for scanning motion.

The creation of smooth motions in spite of the step-wise motion principle is a design goal of the robot. Although each step creates a small backlash, which is clearly measurable at low step frequencies (see Fig. 5.13a), the mass of the robot is expected to act as a low pass filter at higher frequencies removing the backlash and the generated vibrations. This is a necessary prerequisite to move along smooth third-order trajectories. Fig. 5.14 shows the robot executing 100 steps at a step frequency of 500 Hz. The measurement shows the excellent repeatability of the steps, as the movement is highly linear. Furthermore, the movement appears smooth with no vibrations. Even the sudden stop after 100 steps does not create significant vibrations. The robot reaches the final position without overshooting. Thus, to move smoothly along third-order trajectories, actuation frequencies <500 Hz should be avoided. This is given for the line scan-based tracking, as the step frequency needs to be a multiple of the tracking frequency (see Eq. 4.14).

Another important property of the robot is the maximum acceleration that it can create. An upper bound for the robot's acceleration is the friction coefficient between the employed steel spheres and the working surface. If the robot operates on a glass surface, the friction coefficient is approx. 0.5 and thus, the robot can

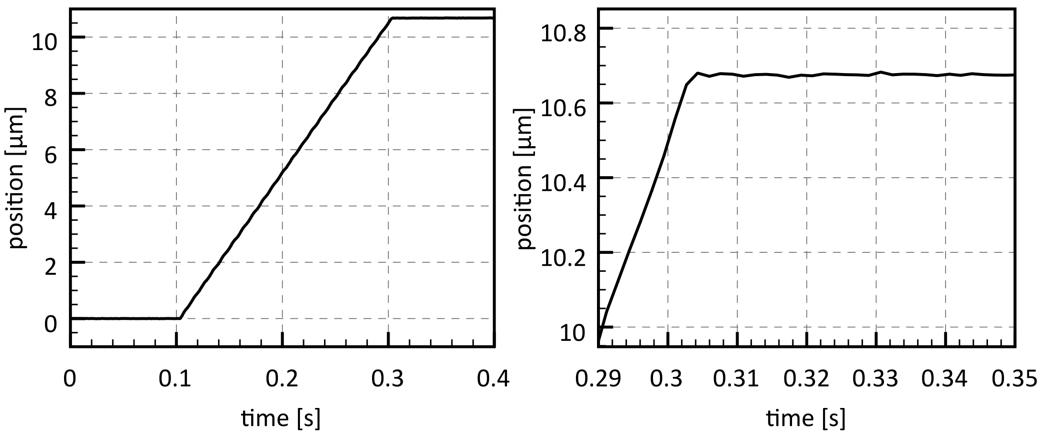


Figure 5.14: Smoothness of the robot's motion. Left) Motion is highly linear. Right) Robot reaches its destination without vibration or overshooting.

create an acceleration of $0.5 \cdot g \approx 4.9\,\mathrm{m/s^2}$. In practice, the acceleration is lower, as the stick-slip motion principle as well as the generated vibrations decrease the effective friction coefficient. Fig. 5.15a shows a robot executing 300 steps with the full 300 Vpp amplitude at a frequency of 25 kHz. The peak acceleration is approx. $2\,\mathrm{m/s^2}$ and the traveling velocity of 4.2 mm/s is quickly reached. Higher step frequencies lead to higher maximum velocities, but the measured maximum acceleration remains at about $2\,\mathrm{m/s^2}$.

5.2.2 Open-loop control

To verify the open-loop model of the nanorobot, several test motion vectors (see Fig. 4.6) have been selected and applied to the robot. The results of measuring single steps at a frequency $f = 100\,\mathrm{Hz}$ are shown in Fig. 5.16. 100 Hz are chosen as actuation frequency so that acceleration effects can be neglected. The measurements show an excellent correspondence to the model (compare Fig. 5.16 and Fig. 4.6). Thus, the model can be assumed to satisfactorily describe the robot's motion behavior. The remaining differences are caused by multiple factors. Important influences are manufacturing tolerances for the individual actuators and the force exerted by the robot's cabling.

To validate the reliability and accuracy of the open-loop model, the robot was open-loop controlled along a well-defined path. The robot's position was measured with the laser interferometer-based measurement station described in Section 4.2.2 and the result is shown in Fig. 5.15b. At a constant step frequency of 100 Hz and a constant step size of 130 nm, 50 steps were conducted along the edges a, b, c and d, $\sqrt{2} \cdot 50 = 70$ steps along the edges e and h, and 35 steps along the edges f and g. For each movement, the deviation from the expected position is less than 5%. Even after

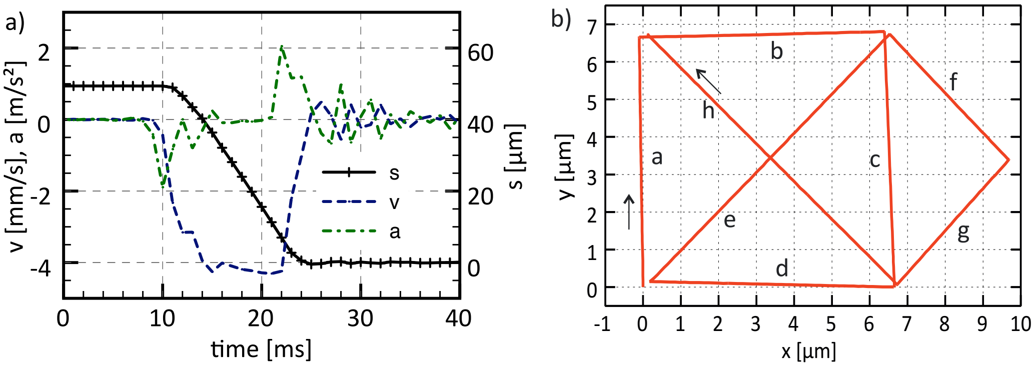


Figure 5.15: a) Position, velocity and acceleration during a 300 step movement at a frequency of 25 kHz. b) Accuracy of the open-loop control.

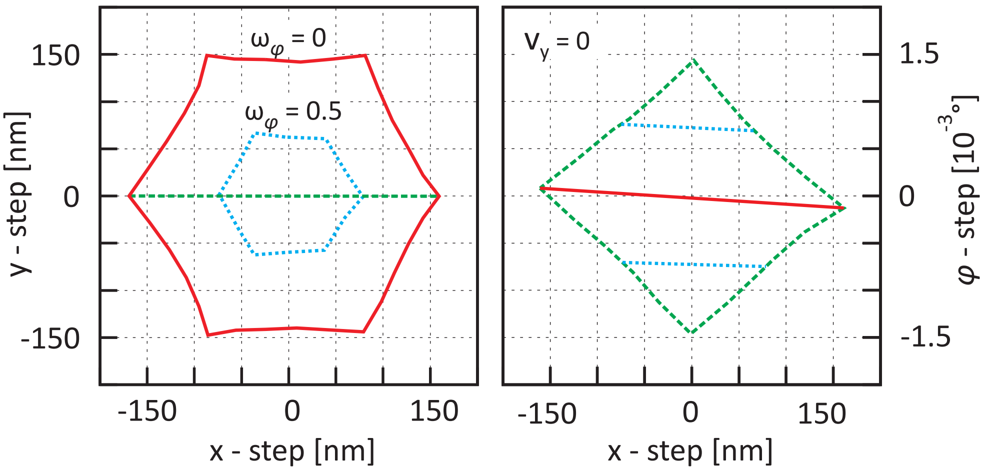


Figure 5.16: Validation of the open-loop model by measuring the response to characteristic test vectors.

all eight movements and a total travel of about 50 µm, the final position error is less than 300 nm. This shows that the open-loop control of the mobile robots is reliable and repeatable with only minor deviations due to manufacturing tolerances and environmental influences such as the robot's cable and dirt on the working surface. With such a reliable and repeatable behavior, the open-loop control facilitates a highly effective closed-loop control.

5.2.3 Latency compensation

Fig. 5.17 shows the effectiveness of the latency compensation for the closed-loop positioning to a target position. In contrast to a trajectory controller, a position controller operates with a fixed setpoint. According to Eq. 4.29, the proportional control part K_P can equal 1. Thereby, the robot quickly moves to the target position without overshooting. Without latency compensation, K_P has to be reduced to prevent a permanent oscillation, i.e. control instability, around the target position. For $K_P = 0.9$, oscillations are still visible but the robot settles to the correct target position. For $K_P = 0.5$, oscillations are reduced to a single 100 nm overshoot. To completely avoid overshooting, K_P has to be smaller than 0.2. In general, the settling behavior with latency compensation significantly outperforms all other settling behaviors.

In comparison to position control, where latency compensation is merely beneficial, trajectory control requires latency compensation. In a way, trajectory control can be viewed as a continuous settling to new target positions. Thus, with latency compensation, trajectories can be precisely followed as shown in the following section.

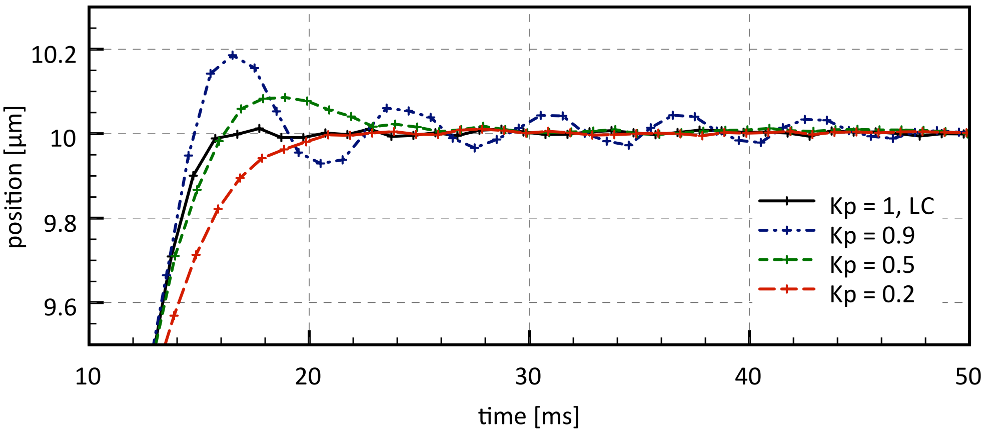


Figure 5.17: Effect of latency compensation (LC) and proportional control coefficient K_P on settling behavior.

Without latency compensation, the robot can either significantly lack behind the trajectory with $K_P \approx 0.2$ or oscillate around the trajectory with $K_P > 0.5$. Thus, accurate trajectory control without latency compensation is impossible.

5.2.4 Closed-loop motion along linear trajectories

With the robot's capability to generate smooth motion and the accurate open-loop model, closed-loop trajectories with third-order motion profiles are possible. Fig. 5.18 shows the movement along an 80 µm square at two different trajectory settings.

The movement along the square in Fig. 5.18a requires 1.2 s and has a very high accuracy. Fig. 5.19 shows the movement along a single edge and the control deviation, i.e. movement error. The movement along each edge requires about 300 ms and follows a smooth third-order profile. The error remains below 20 nm which is only slightly higher than the tracking's noise.

The movement along the square in Fig. 5.18b is completed in 200 ms, i.e. six times faster than Fig. 5.18a. Slight deviations from the square-shaped trajectory are visible. Fig. 5.20 shows the movement along a single edge. As the entire movement is completed in a little over 50 ms, only about 50 tracking updates are received at the 1 kHz update rate. The combination of higher movement speed and reduced number of tracking updates leads to an increased error of up to 500 nm. Considering that the robot moves along an 80 µm-long trajectory and reaches a velocity of 3 mm/s, this error can be regarded as small.

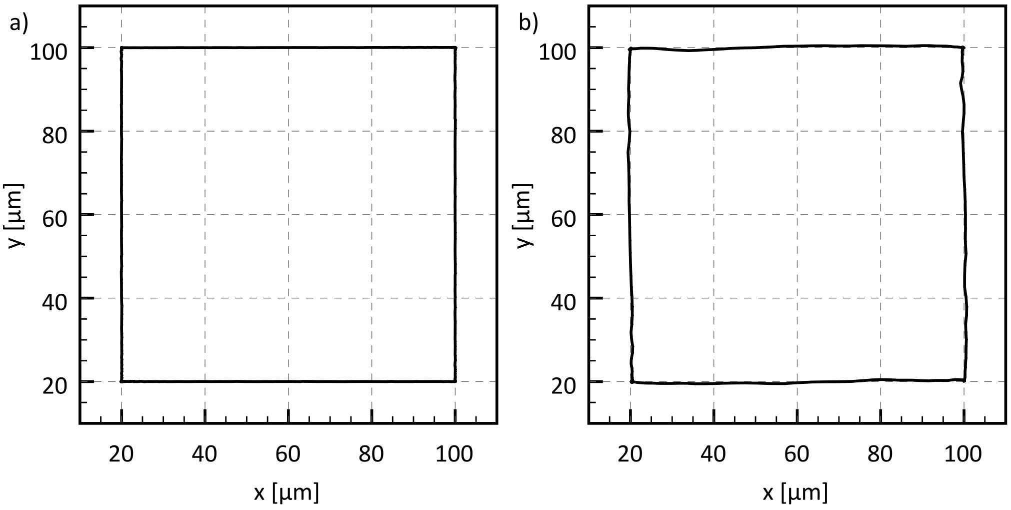


Figure 5.18: Movement along an 80 µm square: a) 1.2 s , b) 200 ms.

5.2.5 Robustness of closed-loop control

The closed-loop control as well as the latency compensation achieves high performance by relying on the open-loop model being accurate. In addition to performance, robustness is a key property of a control loop. Thus, the system should behave correctly, even if specific parameters change, e.g. the open-loop model is no longer accurate. An example of such a situation is an improper cabling of the robot so that the cable exerts a significant force altering the robot's movement. To simulate this condition, the scan generator is set to a scanning rotation of 30°. Thus, there is a 30° degree angular offset between the SEM coordinate system and the robot's

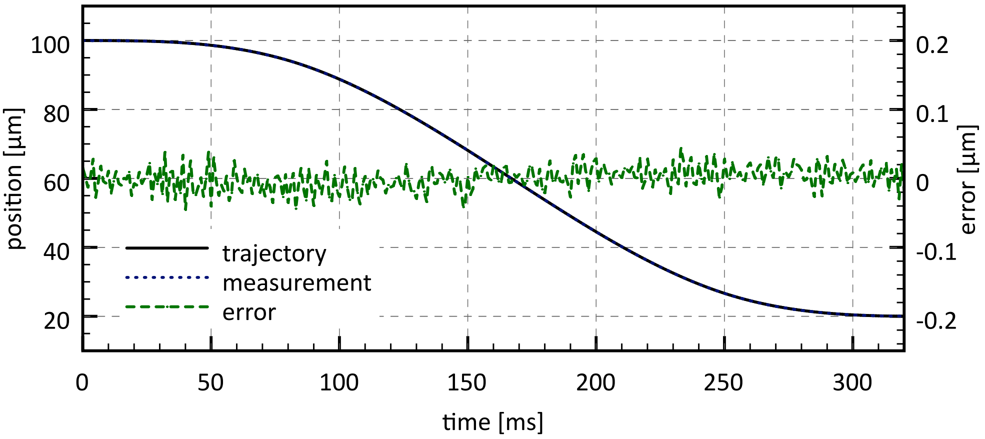


Figure 5.19: Precise movement along an 80 µm-long trajectory.

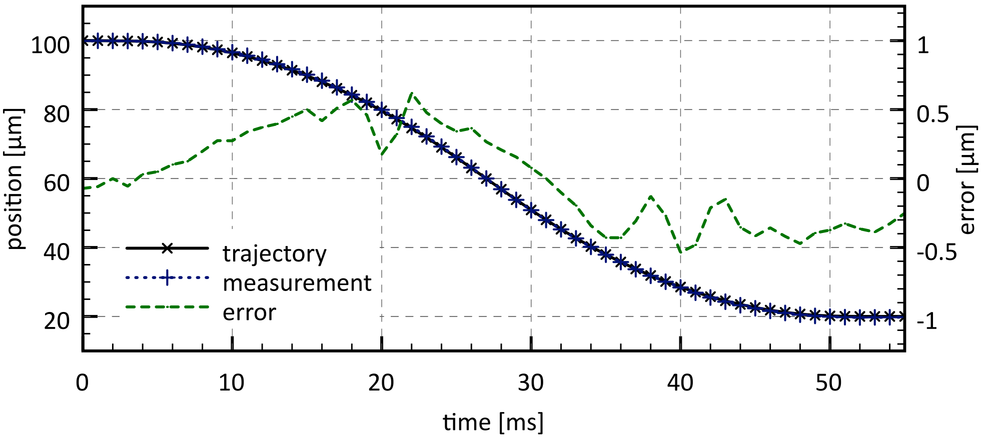


Figure 5.20: Fast movement along an 80 µm-long trajectory. The symbols on the graphs signify tracking updates and desired positions.

internal coordinate system. This offset is **not** incorporated into the open-loop model using auto-configuration.

Fig. 5.21 shows the robustness of position control while moving along the edges of a fourth-order Koch snowflake. The diameter of the snowflake is approx. 80 µm, i.e. similar to the diameter of a human hair. Fig. 5.21a shows the movement based on a correct model. All edges are reasonably straight with only marginal deviations. Fig. 5.21b shows the same measurement, but with the 30° artificial error introduced into the open-loop model. The magnification (Fig. 5.21c) shows that each movement no longer is a straight line but an arc towards the target position. However, in spite of the large error in the open-loop model, the robot still reliably navigates to the desired target position, albeit with a deviation from the expected path.

Fig. 5.22 shows the same measurement with trajectory control enabled. Here, the integral part of the controller quickly compensates for the error in the open-loop model and the robot remains on the planned linear trajectory even with the incorrect open-loop model (see Fig. 5.22b and c).

5.2.6 Long range movements

As shown in Section 5.1.3, line scan-based tracking implemented on a custom scan generator is capable of tracking objects over long travel ranges. It remains to be shown that the closed-loop motion control can exploit these long-range measurements for reliable closed-loop positioning. Fig. 5.23 shows trajectory-controlled movements along a seventh-order Koch snowflake. The entire fractal is about 80 µm in diameter, so that the robot has to move over long distances. Nevertheless, high accuracy is

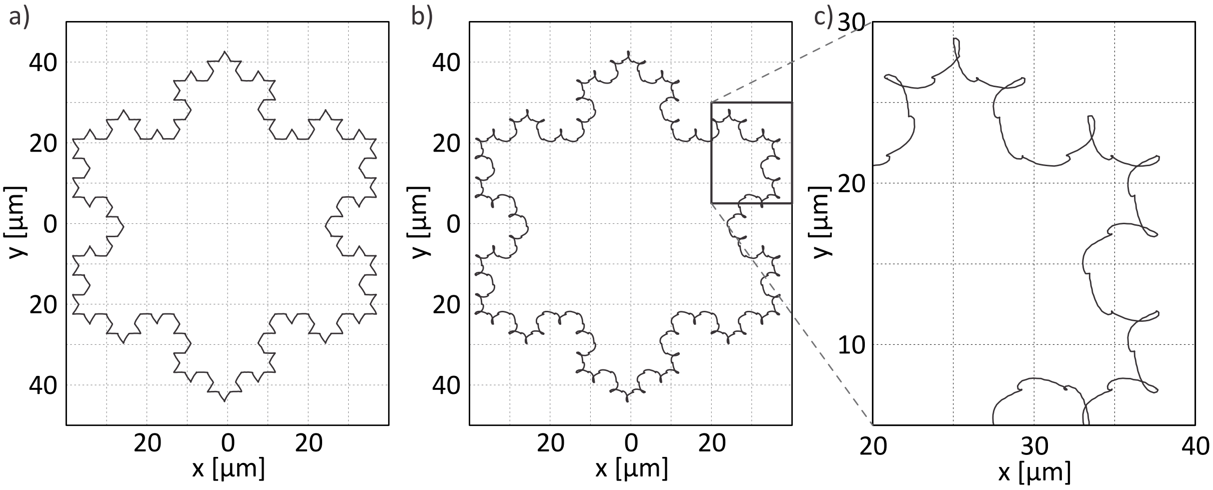


Figure 5.21: Position control along a Koch snowflake: a) With correct model, b) artificial model error by 30° scan rotation, c) zoom reveals movement deviation.

required at all positions, as each of the 2,187 individual movements is only 102 nm long. The magnifications show, that all of the small movements are executed correctly and appear linear. This measurement also shows the visual servoing's high performance, as the entire sequence is completed in 32 s, so that the average time required for each trajectory is only 15 ms.

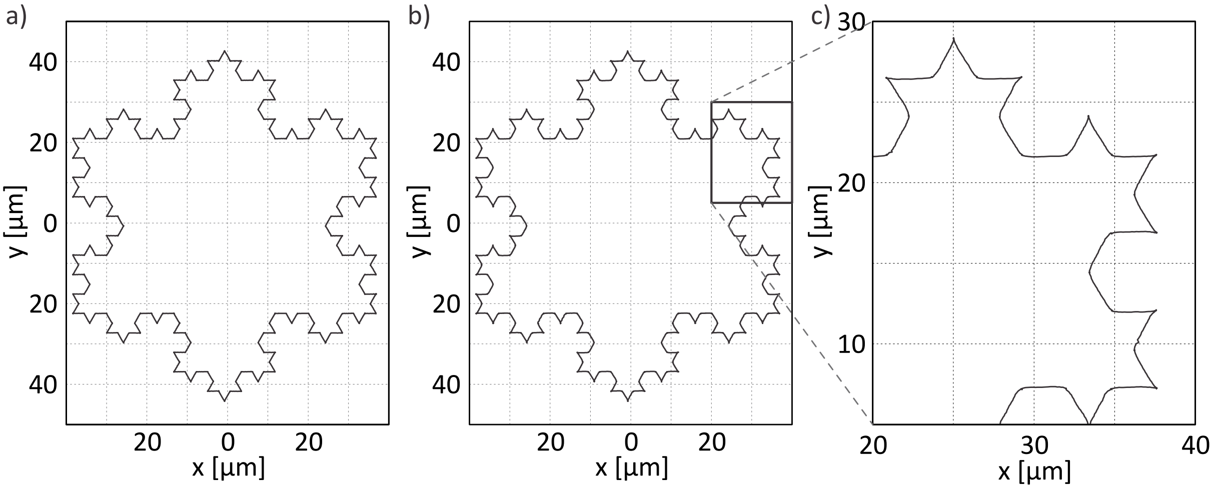


Figure 5.22: Trajectory control along a Koch snowflake: a) With correct model, b) artificial model error by 30° scan rotation, c) zoom reveals robustness of trajectory control.

5.2.7 Auto-Configuration

Using the latency measurement approach described in Section 4.3.4, the latency is individually characterized for the x-axis and the y-axis. Fig. 5.24a shows the results of 100 repeated measurements. As expected, the latency is longer for the x-axis as the horizontal line scan is conducted before the vertical line scan and the result is reported to the controller after the completion of the latter scan. The tracking's update rate is 1.08 kHz, so that the x-coordinate has a latency of a little under an update interval, whereas the y-coordinate has a latency of half an update interval.

To determine the latency measurement's resolution and linearity, an artificial delay is added between a sensor update and the start of the corresponding actuator movement. This delay is increased in 50 µs steps and ten measurements are conducted for each delay. Fig. 5.24b shows the corresponding results. All steps are clearly and accurately resolved showing that the latency measurement has an excellent linearity and a resolution higher than 50 µs.

As described in Section 4.3.4, the robot can automatically find its own rotational center point, i.e. the origin of the coordinate system C_R in the current sensor's coordinate system C_C. To do so, the robot needs to perform a rotational movement. Such a rotational movement is dangerous as it creates a significant movement of the tool which might lead to collisions. Thus, it is desirable to determine the center point exactly with as little a rotation as necessary, i.e. $\Delta\varphi$ should be as small as possible. If the rotation is too small however, noise on the measured values (Δx, Δy and $\Delta\varphi$) might significantly affect the measurement's accuracy.

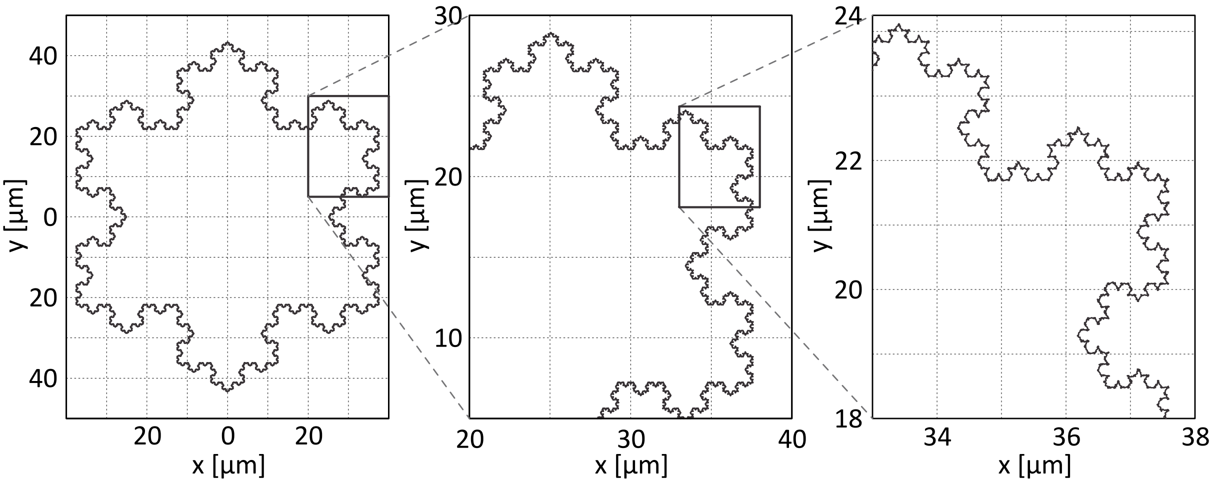


Figure 5.23: Visual servoing along a seventh-order Koch snowflake with 80 µm diameter. A total of 2,187 movements with a length of 102 nm each.

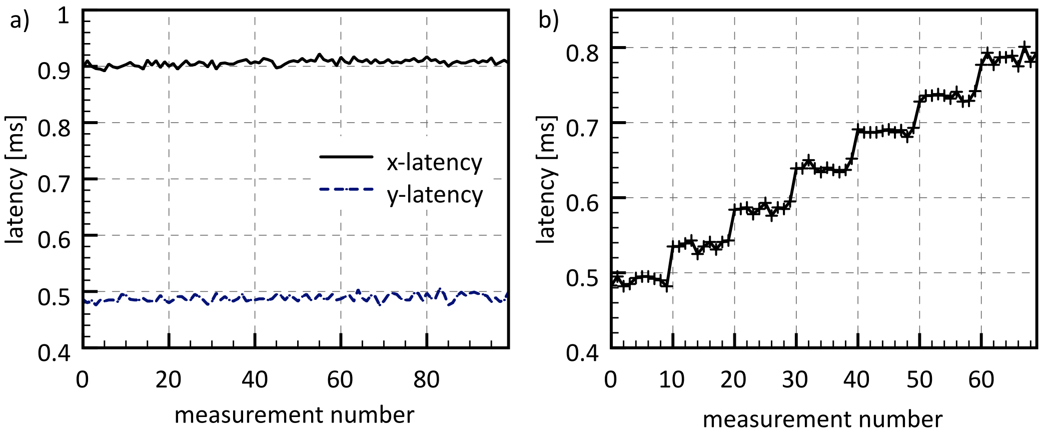


Figure 5.24: Results of latency measurements. a) Latency is 0.9 ms for the x-axis and 0.5 ms for the y-axis. b) Latency for the y-axis is artificially incremented in steps of 50 µs and 10 measurements are conducted for each step.

Fig. 5.25 shows the results for determining the rotational center with different $\Delta\varphi$ based on a pattern tracked on a specimen stub. For very small rotations ($\Delta\varphi < 0.06°$), there is a significant fluctuation in the measurement results. For rotations larger than 0.06°, the results become stable and reliable. The auto-configuration measures that the tracked pattern has a distance of approx. 14.1 mm from the robot's rotational center in the direction $\lambda = 89.8°$.

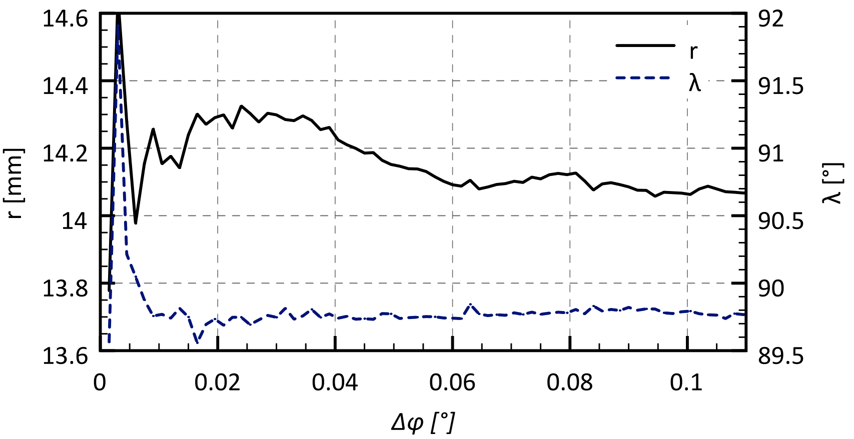


Figure 5.25: Result of measuring the robot's rotational center.

5.3 Conclusions

In this chapter, the principles of both the line scan-based tracking and the developed motion control are validated using two robotic setups. As expected, the tracking is robust against changing brightness, contrast and focus. Measurement results for a pattern do not drift, i.e. there is no time-variant offset between the tracking result and the location of a pattern in the SEM's coordinate system. Measurements with a reference sensor show that the tracking is accurate to about 10 nm and the tracking's noise also has a level of approx. 10 nm. For the chosen parameters, the tracking has an update rate of 1 kHz. Accurate tracking can be performed anywhere within a large working area of 200×200 μm^2. Velocities as high as 6 mm/s and accelerations of 2 m/s^2 are successfully tracked. With these characteristics, line scan-based tracking outperforms image-based tracking in all of the categories robustness, resolution, update rate and working range. Additionally, the combination with depth from focus and depth from defocus facilitates depth measurement with a resolution in the nm-range, with has not been achieved with image-based focus measurements.

In addition to the characterization done during system identification, several key motion properties of the developed robot are measured. Despite the step-wise motion principle, the robot is capable of executing a smooth vibration-free motion similar to scanning actuators. It has a virtually unlimited working range and can reach velocities of up to 13.6 mm/s at accelerations of up to 2 m/s^2. The open-loop model adequately describes the motion behavior of the robot and the deviation between desired and actual motion is as low as 5% in the characterization setup. The latency compensation significantly improves the effectiveness of the closed-loop control, e.g. the settling behavior to a target position. The robot is capable of accurately moving along trajectories with a third-order motion profile at high speeds. An 80 μm-long trajectory can be completed in 50 ms. The control is highly robust and even a significant error introduced into the robot's motion behavior does not prevent successful control. Control is possible anywhere within the tracking's large working range and an auto-configuration sequence makes manual configuration for a specific setup unnecessary. The described robot and its control make full use of the tracking's potential and facilitate fast and accurate movements.

6 Automated nanohandling

The motion control system described in this thesis provides the means for fast and accurate nanopositioning. This chapter shows its integration into a software architecture that can be used for automating a variety of handling and assembly operations. Based on this architecture, automated nanohandling with high throughput becomes feasible.

6.1 Software and system architecture

Fig. 4.2 shows a robotic setup that can be used for automated nanohandling. Two mobile nanorobots are integrated into an SEM as shown in Fig. 5.1. The SEM is controlled by the developed scan generator and, in addition to delivering images, it can track the position of nanoobjects and tools using line scan-based tracking for fine positioning. The robots operate on a glass working surface. LEDs mounted to each robot's bottom are observed by a camera underneath this surface. A hardware-based LED tracker evaluates the camera's images to extract the robots' positions for coarse positioning. A hardware control unit implements visual servoing and signal generation for each robot as described in Section 4.4.

Fig. 6.1 shows the software architecture developed for the automation of nanohandling tasks based on the described hardware. The core component is an automation server which executes automation sequences implemented in the programming language Python. A graphical user interface (GUI) is used for the development and execution of these sequences as well as the monitoring of the automation. An image processing software is used to analyze the images obtained with the SEM, e.g. for the initialization of the line-scan based tracking. The acquired images as well as the derived information can be displayed via the GUI. The automation server communicates with low-level control servers (LoLeCs) that encapsulate individual hardware components and provide a unified interface for automation sequence development. In Fig. 6.1, two LoLeCs encapsulate the robot control units and a third LoLeC controls the SEM's stage. All software components are implemented as individual programs communicating using the common object request broker architecture (CORBA). Thereby, the software system can be distributed to multiple computers if necessary.

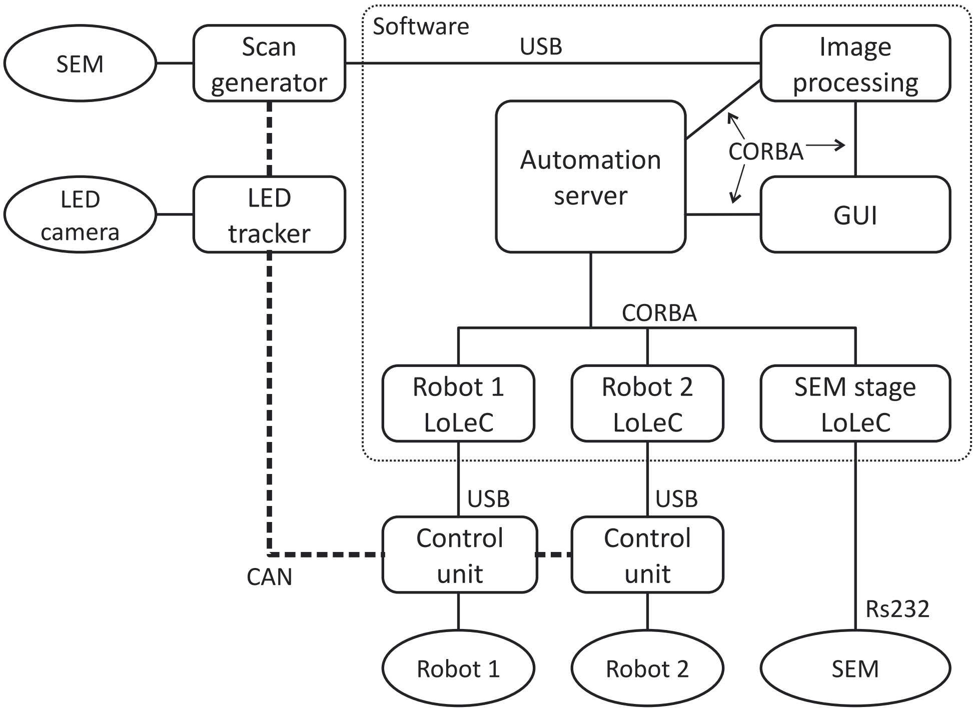


Figure 6.1: Software architecture for automated nanohandling.

Positions measured by the LED tracking as well as the line scan-based tracking need to be transmitted to the robot control units with a precise timing to enable the visual servoing as described above. Thus, the real-time capable controller area network (CAN) bus is used for this connection.

6.2 Automated microsphere handling

As a benchmark application, the automated pick-and-place handling of 10 µm spheres is chosen. These spheres are placed on a silicon surface in an initially random order. The automation task is to select seven spheres and place them alternately in two different lines (see Fig. 6.2). For the manipulation, the electrostatic gripper developed by Chen et al. (2009) is used. In addition to two gripper jaws, it features a central plunger. With this plunger, a sphere can be pushed out of the gripper to overcome surface forces (see Section 2.6). The gripper is mounted in a 30° angle towards the surface. For this automation, the square-shaped pattern milled into the central plunger with a focused ion beam is used to track the gripper's position with line scans.

The only initialization required for this automation is the measurement of the distance between the pattern tracked on the gripper's plunger and the exact location

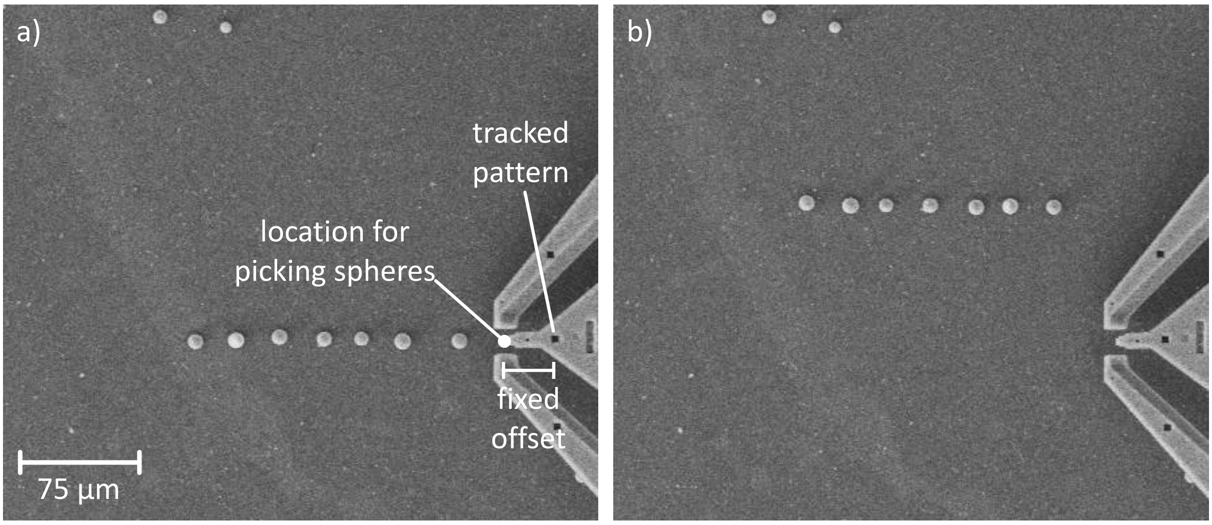


Figure 6.2: Automated sorting of 10 µm spheres: a) Seven spheres placed in lower row. b) Spheres placed in upper row.

between the gripper jaws where spheres should be picked up (see Fig. 6.2a). This location is a priori knowledge of the gripper which needs to be available. The distance to the tracked pattern is measured in a high-resolution SEM image of the gripper and stored as a fixed offset. Thus, if the gripper needs to be moved to a certain location, the pattern needs to be moved to this location plus the fixed offset.

The handling task can be subdivided into the following steps for each of the seven spheres in a row:

1. Determine the exact location of a sphere
2. Align gripper to sphere and pick up sphere
3. Align gripper to target location and release sphere

The determination of the location of a sphere is done in three steps. First an image of the entire scene is acquired similar to Fig. 6.2 but with a faster scan speed, i.e. more image noise. The location of all spheres is detected by finding bright regions of a specific size. Second, a sphere is selected that is not already in the target line. Third, the position of the sphere, which is coarsely known from the image, is precisely determined by performing a horizontal and a vertical line scan.

For steps 2 and 3 of the handling task, the gripper needs to be aligned to a target. The open gripper needs to be aligned to a sphere for picking and the closed gripper with sphere needs to be aligned to a target location for placing. For the given automation task, it is desirable to separate the motion along the x- and y-axes from the motion along the z-axis during this alignment. The gripper should move to a location above the target, move down, pick or place the sphere and then move up before moving to another location. Thereby, a collision with other spheres can be avoided. To safely move above the spheres, a safety distance of 30 µm between gripper and surface is chosen. All movements along the x- and y-axes are performed using the visual servoing described in Chapter 4 tracking the pattern on the plunger. After arriving at the respective destination, the gripper is brought into contact with the surface (see Section 6.3). The picking of a sphere is accomplished by first closing the gripper and then moving the gripper up by 30 µm. An internal linear encoder is used for the upward movement, as high accuracy is not required. To place a sphere, a combination of plunger and jaw movements is used before moving the gripper upwards.

The implemented automation sequence contains a loop that repeats steps 1 through 3 until either the line is completed with the seventh sphere or no further spheres can be found. After step 3, the robot is moved to the right side of the working area and an image is acquired for the following step 1. Additionally, an SEM image is acquired after step 2 for human observation of the handling process.

6.3 Touchdown detection

For the z-alignment, the focus-based depth detection could be used. However, for the given handling task there is a faster method that simply detects the contact between the gripper and the surface. As the gripper is mounted on a 30° angle, a mechanical contact between the surface and the gripper can be recognized as a horizontal displacement. If the gripper is moved straight down, there is no x- or y-displacement. However, as soon as it touches the surface, it bends slightly and its tip moves horizontally as shown in Fig. 6.3. When the gripper is located above the substrate, the patterns on the gripper jaws are slightly right of the pattern on the plunger (see Fig. 6.3a). When the gripper bends slightly due to mechanical contact with the surface, the patterns on the jaws move slightly left whereas the pattern on the plunger remains stationary (see Fig. 6.3b). With the high resolution and update rate of the line scan-based tracking, this condition can be detected quickly even for high vertical movement speeds.

A priori tests with the gripper have shown that a deformation of up to 1 µm does not damage either the gripper or the surface. Assuming that the contact is detected by the first subsequent tracking update, the gripper can move up to 1 µm between two updates. As the tracking operates at a frequency of 1 kHz, this leads to a movement speed of 1 mm/s. Thus, the gripper can move downwards with a speed of 1 mm/s and the contact is detected without exceeding the 1 µm limit. As the gripper is located approx. 30 µm above the surface during the x- and y-alignment, the touchdown is completed in approx. 30 ms. Fig. 6.4 shows the result of 600 touchdown operations.

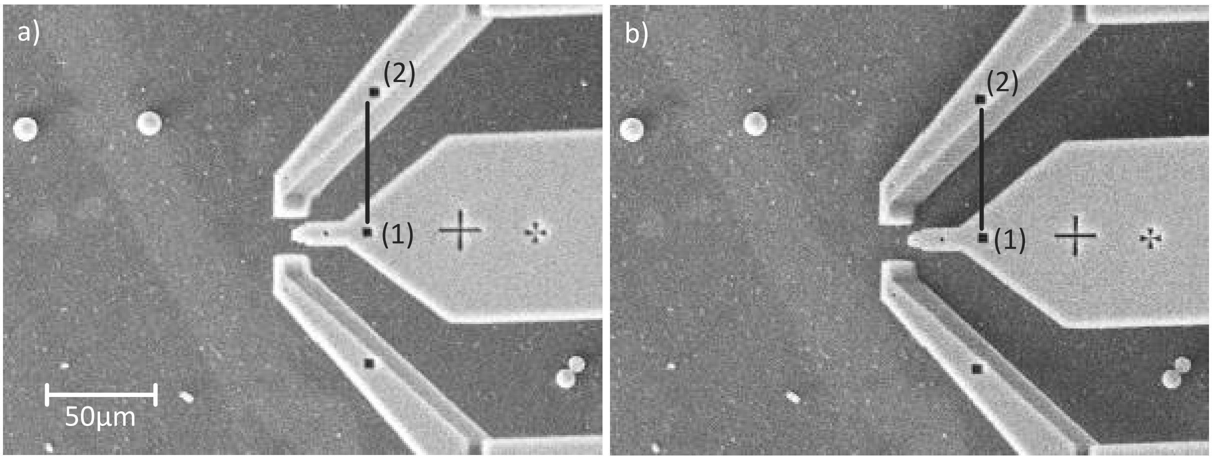


Figure 6.3: Gripper deformation on touching a surface: a) Above the surface, the jaw tracking pattern (2) is located right of the plunger tracking pattern (1). b) Upon touching the surface, (2) moves left.

The maximum deviation between any two touchdown operations was 2 µm. This is expected as the touchdown operation itself was set to accept deviations of up to 1 µm and the internal sensor used as reference has an uncertainty of about 1 µm.

6.4 Results and conclusions

The automation sequence described above repeatably solves the described handling task. The achieved average handling speed is 580 ms per sphere. Table 6.1 shows the durations of the average durations of the individual process steps. The determination of the position of a sphere takes about 103 ms. Most of this time is required for

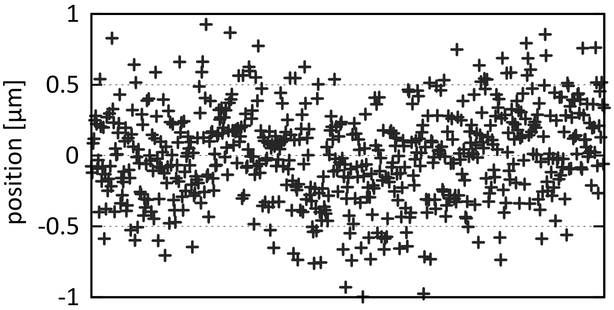


Figure 6.4: Repeatability of the touchdown operation demonstrated by 600 consecutive movements.

acquiring and processing an SEM image. Three visual servoing operations are used with a total duration of 310 ms. As expected, the touchdown takes about 30 ms and moving the gripper up is slightly faster. The other operations include all movements of the gripper jaws and plunger as well as computational overhead of the automation sequence execution.

Table 6.1: Average execution times of the different process steps.

Name	**Duration**
Determine position of sphere	103.1 ms
Visual servoing (3×)	310.7 ms
Touchdown (2×)	60.8 ms
Move gripper up (2×)	50.7 ms
Other	55.1 ms
Total	580.4 ms

This section shows the integration of the new tracking and motion control approaches into a system architecture designed to implement automated nanohandling. The architecture allows for the flexible automation of different handling and assembly tasks. The implemented handling of 10 µm spheres with the achieved throughput of almost two spheres per second shows that the high speed of the visual servoing can be effectively utilized. All alignment operations are conducted solely relying on feedback obtained from the SEM. Thus, the alignments can be assumed to have an accuracy of about 10 nm and time-variant effects such as thermal drift do no affect the operation. Thus, although the handling of 10 µm spheres is a microhandling operation, the results are transferrable to nanohandling.

7 Conclusions and outlook

In this thesis, a new approach to nanorobot motion control in scanning electron microscopes is devised. An analysis shows that existing nanorobot control methodologies are a bottleneck for automated nanohandling and nanoassembly with high throughput. Motion control using internal sensors does not deliver the accuracy required for nanohandling, whereas existing vision-based control approaches are unsuitable for high throughput. To overcome this limitation, a new method for tracking the position of nanoobjects by conducting line scans with the electron microscope is devised. The tracking is robust against changing brightness, contrast and focus. Measurement results for a tracked object do not drift, i.e. there is no time-variant offset between the tracking result and the location of a pattern in the SEM's coordinate system. Measurements with a reference sensor show that the tracking is accurate to about 10 nm. The tracking can have an update rate of up to 1 kHz and can be performed accurately anywhere within a large working area of 200×200 μm^2. Velocities as high as 6 mm/s and accelerations of 2 m/s^2 can be tracked. The combination of line scan-based tracking with depth from focus and depth from defocus facilitates depth measurement with a resolution in the nm-range.

A new and flexible nanorobot as well as its control system are designed. The robot combines multiple degrees of freedom in a compact design and can reach velocities of up to 13.6 mm/s at accelerations of up to 2 m/s^2. The resolution in scanning mode is 0.67 nm/V making sub-nanometer positioning possible. The robot's motion behavior is modeled based on a combination of theoretic analysis and system identification with deviations as low as 5% revealed in characterization measurements. Based on the accuracy of the open-loop control as well as the reliable latency of the line scan-based tracking, a closed-loop controller with latency compensation enables visual servoing along third-order trajectories. Thereby, the exact behavior of the robot including position, velocity, acceleration and jerk are precisely controlled during all movements. Even trajectories covering 80 μm are completed within a few tens of microseconds. The control is highly robust and even a significant error introduced into the robot's motion behavior does not prevent successful control. An auto-configuration sequence is used to measure the tracking's configuration and characteristics. Integrating the motion control into a software and system architecture, the automatic handling of 10 μm spheres with a throughput of almost two spheres per second is implemented.

7.1 Conclusions

For image-based tracking approaches, the time-consuming image acquisition is a bottleneck. To allow for movement in all directions, a significant amount of pixels has to be acquired that does not deliver information used for tracking leading to an inherent inefficiency. The developed tracking approach bypasses this bottleneck by relying on SEM line scans. With the measured characteristics, the new tracking method substantially outperforms existing tracking approaches in terms of resolution, update rate, working range and robustness. For the first time, depth measurements solely based on the SEM's focus deliver a resolution in the nm-range. Due to the large working range, alignment techniques such as zoom-and-center steps (see Section 2.6) become unnecessary during most nanohandling operations.

The presented approach is the first approach taking the dynamic characteristics of a moving object and the corresponding distortions in the SEM signal into account. Thus, objects moving at high speeds can be tracked as opposed to the tracking of quasi-static objects possible with image-based tracking. By incorporating a model of the tracked object's velocity into the tracking, even nanoscopic objects can be tracked at high velocities. Only the acceleration remains limited, but considerably exceeds accelerations usable in existing nanohandling operation. Higher accelerations would create unfavorable vibrations. With the possibility to track significantly moving objects as well as the high update rate, visual servoing does not have to be treated in a quasi-static fashion. Instead, the line scan-based tracking can be employed as an element in a robot's low-level control loop. Thus, the speed of positioning based on internal sensors is combined with the accuracy of visual servoing.

The described robot and its control make full use of the tracking's potential and make fast and accurate movements possible. It combines the smooth, vibration-free motion of scanning actuators with the long range of step-wise actuators by significantly reducing the step length and the accelerated masses. The accuracy of the open-loop model shows that the robot's motion behavior is well-understood and correctly modeled. The robot's performance in terms of velocity, acceleration, resolution and working range exceeds the requirements of existing nanohandling operations. The possibility to precisely move along jerk-limited trajectories is new for nanorobots and enables complex handling and assembly operations. The robot can be effortlessly integrated into different microscope systems as the only requirement is a flat working surface. The auto-configuration sequence extends this effortless integration to automated positioning. After integration into a specific microscope, visual servoing can be used without the need for manual configuration.

All goals presented in Section 1.1 have been reached and this work represents an important step on the way to achieve high-throughput nanohandling viable for industrial applications as shown by the implemented high-speed handling example.

7.2 Outlook

The line scan-based position tracking, the new nanorobot and the motion control approach offer a variety of possibilities for future research.

In this thesis, a basic approach for tracking arbitrary objects is described based on line scans over specific edges of the object (see Section 3.3.3). This process would be simplified if the edges used for tracking a specific object could be extracted automatically. Such an **automatic edge select** could be done in four steps. First, an image of the object could be acquired with as little noise as possible, i.e. with a long acquisition time. Second, a Canny edge detection filter could highlight the object's edges (Canny, 1986). Third, a Hough transformation-based line detection (Duda and Hart, 1972) could determine straight edges. Fourth, the detected edges could be analyzed for the free area around them and at least two good edges of a significant angular difference could be selected.

An additional application of the line scan-based tracking is **drift compensation** during nanomachining operations such as FIB milling, EBiD or electron beam lithography. These processes require durations on the order of minutes or hours and are substantially affected by the significant drift on the nanoscale (see Section 2.3.4). To compensate for drift, the machining operations can be done in close proximity to a reference pattern. As a single tracking update based on line scans only requires 1 ms, the position of the reference pattern can be measured repeatedly without noticeably increasing the process time. Scanning as often as once per second prolongs the process time by only 0.1%. Thus, even if objects drift quickly, e.g. because of improper grounding, they can be machined precisely.

The touchdown detection described in Section 6.2 is a basic form of **vision-based force measurement** (Greminger and Nelson, 2004). The concept is that a force can be determined by measuring the deflection of a flexible component using visual feedback, e.g. SEM images. In contrast to other force sensors, no cabling or other equipment is required. Due to its high resolution and update rate, the line scan-based tracking is especially well-suited for this application. Fig. 7.1a shows the prototype of an electrothermal gripper (Andersen et al., 2009) designed to deliver vision-based force feedback. Two patterns (1) and (2) are created on the force-sensing arm and a fixed structure, respectively, using EBiD. To measure the force applied during gripping, patterns (1) and (2) are both tracked so that the displacement of the force-sensing arm is measured independent of the gripper movement and drift. Based on its geometry and material composition, the arm has an approximated stiffness of 4 N/m. Fig. 7.1b shows the measured force, when a CNT is pushed against the gripper in five steps from two sides. The force exerted by the CNT bends the force-sensing arm either towards or away from the fixed structure. With the sixth step in either direction, the CNT slips to the other side and the force-sensing arm

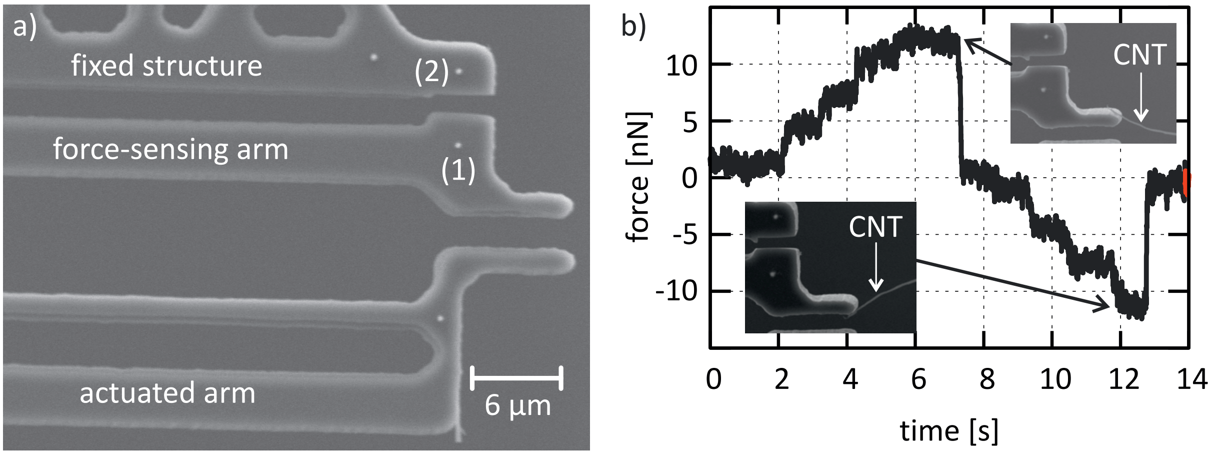


Figure 7.1: a) Gripper (Andersen et al., 2009) with patterns for vision-based force feedback. b) Measured force when pushing a CNT against the force-sensing arm from different sides.

returns to its original position. The maximum force of 12 nN corresponds to values found by Eichhorn et al. (2010) and the 2.5 nN resolution shows that vision-based force measurements are suitable for the nanoscale.

In principle, the tracking approach can be applied to **other scanning sensors** such as the AFM or the scanning tunneling microscope (STM). Similar to nanomachining operations, drift compensation is a major challenge for these microscopes due to the long image acquisition time and high resolution. Drift compensation can be achieved by using a reference pattern in AFMs or even a reference atom for STM imaging with atomic resolution. After a specific number of scanned lines, the position of the reference can be measured using two short line scans and the measured drift can be compensated for.

The movement model developed for the new nanorobot closely matches the measured movement. However, differences remain, e.g. comparing Fig. 4.6 and Fig. 5.16. In order to improve the control performance, the model of the robot's behavior can be replaced by an **adaptive model** to improve model accuracy. Such an adaptive model can be trained after a robot is placed into a specific setup to compensate for differences caused by manufacturing differences and working conditions. Furthermore, during the operation, training vectors consisting of applied actuation parameters and measured movements are acquired and the model could adapt to a time-variant robot behavior, e.g. create by wear or changing environmental conditions.

This outlook is just a short overview of possible future works. The described tracking and servoing approaches enable a variety of extensions and applications.

Bibliography

Jake J. Abbott, Zoltán Nagy, Felix Beyeler, and Bradley J. Nelson. Robotics in the Small, Part I: Microrobotics. *IEEE Robotics & Automation Magazine*, 14(2): 92–103, 2007.

Karin N. Andersen, D. H. Petersen, K. Carlson, K. Molhave, Ozlem Sardan, A. Horsewell, Volkmar Eichhorn, Sergej Fatikow, and Peter Bøggild. Multimodal electrothermal silicon microgrippers for nanotube manipulation. *IEEE Transactions on Nanotechnology*, 8(1):76–85, January 2009.

Daniel J. Arbuckle, Jonathan Kelly, and Aristides A. G. Requicha. A High-Level Nanomanipulation Control Framework. In *Proc. of International Advanced Robotics Programme (IARP) Workshop on Micro and Nano Robotics*, Paris, France, October 2006.

Karl J. Åström, Chang C. Hang, and B. C. Lim. A new Smith predictor for controlling a process with an integrator and long dead-time. *IEEE Transactions on Automatic Control*, 39(2):343–345, 2002.

Arvid Bergander and Jean-Marc Breguet. Performance Improvements for Stick-Slip Positioners. In *Proc. of Intl. Symposium on Micromechatronics and Human Science (MHS'03)*, pages 59–66, Nagoya, Japan, October 2003.

Andrew Blake and Michael Isard. *Active Contours*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition, 1998.

Jean-Marc Breguet and Reymond Clavel. Stick and slip actuators: design, control, performances and applications. In *Proc. of Intl. Symposium on Micromechatronics and Human Science (MHS)*, pages 89–95, Nagoya, Japan, November 1998.

Jean-Marc Breguet, Eric Pernette, and Reymond Clavel. Stick and slip actuators and parallel architectures dedicated to microrobotics. *Microrobotics: Components and Applications*, 2906(1):13–24, 1996.

Axel Buerkle and Sergej Fatikow. Laser measuring system for a flexible microrobot-based micromanipulation station. In *Proc. of Intl. Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 799–804, Takamatsu, Japan, October 2000. IEEE.

Arne Burisch, Jan Wrege, Annika Raatz, and Jürgen Hesselbach. PARVUS–miniaturised robot for improved flexibility in micro production. *Journal of Assembly Automation*, 27(1):65–73, 2007.

James M. Bustillo, Roger T. Howe, and Richard S. Muller. Surface Micromachining for Microelectromechanical Systems. *Proceedings of the IEEE*, 86(8):1552–1574, August 1998.

John Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8:679–698, November 1986.

Kenneth Carlson, Karin N. Andersen, Volkmar Eichhorn, D. H. Petersen, K. Mølhave, I. Y. Y. Bu, K. B. K. Teo, W. I. Milne, Sergej Fatikow, and Peter Bøggild. A carbon nanofibre scanning probe assembled using an electrothermal microgripper. *Nanotechnology*, 18:345501, 2007.

Brandon K. Chen, Yong Zhang, and Yu Sun. Overcoming adhesion forces: Active release of micro objects in micromanipulation. In *Proc. of IEEE Intl. Conference on Robotics and Automation (ICRA)*, pages 2611–2616, Kobe, Japan, May 2009.

John H. Comtois and Victor M. Bright. Applications for surface-micromachined polysilicon thermal actuators and arrays. *Sensors and Actuators A: Physical*, 58 (1):19–25, 1997.

Peter I. Corke and Malcolm C. Good. Dynamic Effects in Visual Closed-Loop Systems. *IEEE Transactions on Robotics and Automation*, 12(5):671–683, 1996.

Nicolas Cornille, Dorian Garcia, Michael A. Sutton, and Stephen R. McNeill. Automated 3-D Reconstruction Using a Scanning Electron Microscope. In *Proc. of the SEM Conference on Experimental and Applied Mechanics*, 2003.

Donald E. Croft, G. Shed, and Santosh Devasia. Creep, Hysteresis, and Vibration Compensation for Piezoactuators: Atomic Force Microscopy Application. *Journal of Dynamic Systems, Measurement, and Control*, 123:35–43, 2001.

Christian Dahmen. Focus-based depth estimation in the SEM. In *Proc. of Intl. Symposium on Optomechatronic Technologies*, San Diego, CA, USA, November 2008.

Christian Dahmen. Threedimensional Tracking using Object Defocus In Twodimensional Scanning Electron Microscope Images. In *Proc. of 6th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, Milan, Italy, 2009.

Bhaskar Dasgupta and T. S. Mruthyunjaya. The stewart platform manipulator: a review. *Mechanism and Machine Theory*, 35(1):15–40, 2000.

Reinhard Degen and Rolf Slatter. Hollow shaft micro servo actuators realized with the Micro Harmonic Drive®. In *Proc. of Actuator*, pages 205–212, Bremen, Germany, June 2002.

Claas Diederichs. Hardware-Software Co-Design Tracking System for Predictable High-Speed Mobile Microrobot Position Control. In *Proc. of IFAC Symposium on Mechatronic Systems*, 2010.

Lixin Dong and Bradley J. Nelson. Robotics in the Small, Part II: Nanorobotics. *IEEE Robotics & Automation Magazine*, 14(3):111–121, 2007.

Walter Driesen, Thierry Varidel, Stephane Régnier, and Jean-Marc Breguet. Micro manipulation by adhesion with two collaborating mobile micro robots. *Journal of Micromechanics and Microengineering*, 15(10):259–267, 2005.

Richard O. Duda and Peter E. Hart. Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, Januar 1972.

Volkmar Eichhorn, Sergej Fatikow, Thomas Wich, Christian Dahmen, Torsten Sievers, Karin N. Andersen, Kenneth Carlson, and Peter Bøggild. Depth-Detection Methods for Microgripper based CNT Manipulation in a Scanning Electron Microscope. *Journal of Micro-Nano Mechatronics*, 4(1):27–36, 2008.

Volkmar Eichhorn, Sergej Fatikow, Tim Wortmann, Christian Stolle, Christoph Edeler, Daniel Jasper, Ozlem Sardan, Peter Bøggild, Guillaume Boetsch, Christophe Canales, and Reymond Clavel. NanoLab: A Nanorobotic System for Automated Pick-and-Place Handling and Characterization of CNTs. In *Proc. of IEEE Int. Conference on Robotics and Automation (ICRA)*, pages 1826–1831, Kobe, Japan, May 2009.

Volkmar Eichhorn, Malte Bartenwerfer, and Sergej Fatikow. Nanorobotic Strategy for Nondestructive Mechanical Characterization of Carbon Nanotubes. *Bentham Journal Micro and Nanosystems*, 2(1):32–37, 2010.

Stephan Fahlbusch, Sébastien Mazerolle, Jean-Marc Breguet, Alexander Steinecker, Joël Agnus, Ricardo Pérez, and Johann Michler. Nanomanipulation in a scanning electron microscope. *Journal of Materials Processing Technology*, 167(2-3):371–382, 2005.

Sergej Fatikow, Thomas Wich, Helge Hülsen, Torsten Sievers, and March Jähnisch. Microrobot System for Automatic Nanohandling inside a Scanning Electron Microscope. *IEEE-ASME Transactions on Mechatronics*, 12(3):244–252, 2007.

Sergej Fatikow, Thomas Wich, and Torsten Sievers. Automatic Nanohandling Station inside a Scanning Electron Microscope. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 222(1): 117–128, 2008.

Ronald S. Fearing. Survey of Sticking Effects for Micro Parts Handling. In *Proc. of IEEE/RSJ Intl. Conference on Intelligent Robots and Systems*, pages 212–217, Pittsburgh, PA, USA, 1995.

Toshio Fukuda, Masahiro Nakajima, Pou Liu, and Haitham ElShimy. Nanofabrication, Nanoinstrumentation and Nanoassembly by Nanorobotic Manipulation. *Journal of Robotis Research*, 28(4):537–547, 2009.

M. Gharavi-Alkhansari. A fast globally optimal algorithm for template matching using low-resolution pruning. *IEEE Transactions on Image Processing*, 10(4): 526–533, 2002.

Michael A. Greminger and Bradley J. Nelson. Vision-based force measurement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):290–298, March 2004.

T. Hägglund. An industrial dead-time compensating pi controller. *Control Engineering Practice*, 4(6):749–756, 1996.

Robert Haschke, Erik Weitnauer, and Helge Ritter. On-line planning of time-optimal, jerk-limited trajectories. In *Proc. of IEEE/RSJ Intl. Conference on Intelligent Robots and Systems (IROS)*, pages 3248–3253, Nice, France, 2008.

Jürgen Hesselbach, Annika Raatz, Jan Wrege, and Sven Soetebier. Design and analysis of a macro parallel robot with flexure hinges for micro assembly tasks. In *Proc. of 35th International Symposium on Robotics (ISR)*, pages 23–26, Nord Villipinte, France, March 2004.

Helge Hülsen. *Self-organising locally interpolating maps in control engineering*. PhD thesis, University of Oldenburg, 2007.

Seth Hutchinson, Gregory D. Hager, and Peter I. Corke. A Tutorial on Visual Servo Control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, 1996.

Koji Ikuta. Micro/miniature shape memory alloy actuators. In *Proc. of IEEE Intl. Conference on Robotics and Automation (ICRA)*, volume 3, pages 2156–2161, Cincinnati, OH, USA, May 1990.

March Jähnisch and Sergej Fatikow. 3D Vision Feedback for Nanohandling Monitoring in a Scanning Electron Microscope. *International Journal of Optomechatronics*, 1(1):4–26, 2007.

Daniel Jasper and Christoph Edeler. Characterization, Optimization and Control of a Mobile Platform. In *Proc. of 6th Int. Workshop on Microfactories (IWMF)*, pages 143–148, Evanston, IL, USA, October 2008.

Daniel Jasper, Christoph Edeler, Claas Diederichs, Mirko Naroska, Christian Stolle, and Sergej Fatikow. Towards Automated Robotic Nanomanipulation Systems. In *Prof. of IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 94–99, Singapore, July 2009.

Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

Axel Kortschack and Sergej Fatikow. Development of a mobile nanohandling robot. *Journal of Micromechatronics*, 2(3-4):249–269, 2004.

Bradley E. Kratochvil, Lixin Dong, and Bradley J. Nelson. Real-time Rigid-body Visual Tracking in a Scanning Electron Microscope. *The International Journal of Robotics Research*, 28(4):498–511, 2009.

Konstantinos J. Kyriakopoulos and George N. Saridis. Minimum jerk path generation. In *Proc. of IEEE Intl. Conference on Robotics and Automation (ICRA)*, pages 364–369, Philadelphia, PA, USA, April 1988.

Paul Lambrechts, Matthijs Boerlage, and Maarten Steinbuch. Trajectory planning and feedforward design for electromechanical motion systems. *Control Engineering Practice*, 13(2):145–157, 2005.

Byeung-leul Lee, Chang-hoon Oh, Soo Lee, Yong-soo Oh, and Kuk-jin Chun. A vacuum packaged differential resonant accelerometer using gap sensitive electrostatic stiffness changing effect. In *The Thirteenth Annual International Conference on Micro Electro Mechanical Systems (MEMS)*, pages 352–357, Miyazaki, Japan, January 2000.

Yang Li and John Bechhoefer. Feedforward control of a closed-loop piezoelectric translation stage for atomic force microscope. *Review of Scientific Instruments*, 78(1):013702, 2007.

Chun-Shin Lin, Po-Rong Chang, and J. Y. S. Luh. Formulation and optimization of cubic polynomial joint trajectories for industrial robots. *IEEE Transactions on Automatic Control*, 28(12):1066–1074, December 1983.

Lennart Ljung. *System identification: theory for the user*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1986.

Sonja Macfarlane and Elizabeth A. Croft. Jerk-bounded manipulator trajectory planning: design for real-time applications. *IEEE Transactions on Robotics and Automation*, 19(1):42–52, February 2003.

Sylvain Martel and Ian Hunter. Nanofactories based on a fleet of scientific instruments configured as miniature autonomous robots. *Jornal of Micromechatronics*, 2(3-4):201–214, 2002.

Mark B. Moffett, Arthur E. Clark, Marilyn Wun-Fogle, Jan Linberg, Joseph P. Teter, and Elizabeth A. McLaughlin. Characterization of Terfenol-D for magnetostrictive transducers. *The Journal of the Acoustical Society of America*, 89(3):1448–1455, 1991.

Babak Mokaberi and Aristides A. G. Requicha. Drift Compensation for Automatic Nanomanipulation with Scanning Probe Microscopes. *IEEE Transactions on Automation Science and Engineering*, 3(3):199–207, July 2006.

Babak Mokaberi and Aristides A. G. Requicha. Compensation of Scanner Creep and Hysteresis for AFM Nanomanipulation. *IEEE Transactions on Automation Science and Engineering*, 5(2):197–206, July 2008.

Rakesh Murthy, Aditya N. Das, and Dan O. Popa. ARRIpede: An Assembled Micro Crawler. In *Proc. of 8th IEEE Conference on Nanotechnology (NANO'08)*, pages 833–836, Arlington, TX, USA, August 2008.

Cagdas D. Onal, Onur Ozcan, and Metin Sitti. Automated 2-D Nanoparticle Manipulation with an Atomic Force Microscope. In *Proc. of IEEE Intl. Conference on Robotics and Automation (ICRA)*, pages 1814–1819, Kobe, Japan, May 2009.

Aurelio Piazzi and Antonio Visioli. Global minimum-time trajectory planning of mechanical manipulators using interval analysis. *International Journal of Control*, 71(4):631–652, 1998.

Aurelio Piazzi and Antonio Visioli. Global minimum-jerk trajectory planning of robot manipulators. *IEEE Transactions on Industrial Electronics*, 47(1):140–149, February 2000.

Ludwig Reimer. *Scanning Electron Microscopy: Physics of Image Formation and Microanalysis*, volume 45 of *Springer Series in Optical Sciences*. Springer, 2nd edition, 1998.

Rémi Ronfard. Region-Based Strategies for Active Contour Models. *International Journal of Computer Vision*, 13(2):229–251, 1994.

Azriel Rosenfeld. Coarse-fine template matching. *IEEE Transactions on Systems, Man and Cybernetics*, 7(2):104–107, 1977.

Srinivasa M. Salapaka, Abu Sebastian, Jason P. Cleveland, and Murti V. Salapaka. High bandwidth nano-positioner: A robust control approach. *Review of Scientific Instruments*, 73(9):3232–3241, 2002.

Ferdinand Schmoeckel, Heinz Wörn, and Matthias Kiefer. The Scanning Electron Microscope as Sensor System for Mobile Microrobots. In *Proc. of IEEE International Conference on Emerging Technologies and Factory Automation*, Antibes-Juan les Pins, France, 2001.

Jianbo Shi and Carlo Tomasi. Good Features to Track. In *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

Yoshiaki Shirai and Hirochika Inoue. Guiding a robot by visual feedback in assembling tasks. *Pattern Recognition*, 5(2):99–106, 1973.

Torsten Sievers. Global sensor feedback for automatic nanohandling inside a scanning electron microscope. In *Proc. of 2nd I*PROMS NoE Virtual International Conference on Intelligent Production Machines and Systems*, pages 289–294, 2006.

Torsten Sievers and Sergej Fatikow. Real-time object tracking for the robot-based nanohandling in a scanning electron microscope. *Journal of Micromechatronics - Special Issue on Micro/Nanohandling*, 3(3-4):267–284(18), 2006.

Otto J. M. Smith. *Feedback control systems.* McGraw Hill, New York, NY, USA, 1958.

Steven W. Smith. *The scientist and engineer's guide to digital signal processing.* California Technical Publishing, San Diego, CA, USA, 1997.

Yu Sun, Stefan Duthaler, and Bradley J. Nelson. Autofocusing in computer microscopy: selecting the optimal focus algorithm. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Edmonton, Canada, 2005.

Michael A. Sutton, Ning Li, Dorian Garcia, Nicolas Cornille, Jean J. Orteu, Stephen R. McNeill, Hubert W. Schreier, and Xiaodong Li. Metrology in a scanning electron microscope: theoretical developments and experimental validation. *Measurement Science and Technology*, 17:2613–2622, 2006.

William C. Tang, Tu-Cuong H. Nguyen, M.W. Michael W. Judy, and Roger T. Howe. Electrostatic-comb drive of lateral polysilicon resonators. *Sensors and Actuators A: Physical*, 21(1-3):328–331, 1990.

Keiji Watanabe and Masami Ito. A process-model control for linear systems with delay. *IEEE Transactions on Automatic Control*, 26(6):1261–1269, December 1981.

Thomas Wich, Christian Stolle, Christian Dahmen, Tim Luttermann, Oliver Frick, Mirko Naroska, and Sergej Fatikow. ZuNaMi: Automated assembly processes on the nanoscale. In *Proc. of the 4M/ICOMM Conference*, Karlsruhe, Germany, 2009.

Wei Xu and Tim King. Flexure hinges for piezoactuator displacement amplifiers: flexibility, accuracy, and stress considerations. *Precision Engineering*, 19(1):4–10, 1996.

Kemal B. Yesin and Bradley J. Nelson. A CAD model based tracking system for visually guided microassembly. *Robotica*, 23:409–418, 2005.