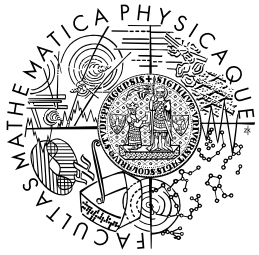


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Bc. Tomáš Iser

Dimensional measurements from a limited set of X-ray projections

Department of Software and Computer Science Education

Supervisor of the master thesis: RNDr. Jan Horáček, Ph.D.
Study programme: Computer Science
Study branch: Computer Graphics and Game Development

Prague 2019

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ..... signature of the author

I would like to thank my supervisor, RNDr. Jan Horáček, Ph.D., for a very interesting research topic and all the valuable information, advice, and feedback he offered during my research. I also thank the Computer Graphics Group of Charles University, especially for providing a modern GPU for evaluating the method. Finally, I would love to thank my dear family, friends, and Sonka for their incredible support in my life and in my studies.

Title: Dimensional measurements from a limited set of X-ray projections

Author: Bc. Tomáš Iser

Department: Department of Software and Computer Science Education

Supervisor: RNDr. Jan Horáček, Ph.D., Department of Software and Computer Science Education

Abstract: Modern non-destructive approaches for quality control in manufacturing often rely on X-ray computed tomography to measure even difficult-to-reach features. Unfortunately, such measurements require hundreds or thousands of calibrated X-ray projections, which is a time-consuming process and may cause bottlenecks. In the recent state-of-the-art research, tens and hundreds of projections are still required. In this thesis, we examine the radiography physics, technologies, and existing solutions, and we propose a novel approach for non-destructive dimensional measurements from a limited number of projections. Instead of relying on computed tomography, we formulate the measurements as a minimization problem in which we compare our parametric model to reference radiographs. We propose the whole dimensional measurements pipeline, including object parametrizations, material calibrations, simulations, and hierarchical optimizations. We fully implemented the method and evaluated its accuracy and repeatability using real radiographs of real physical objects. We achieved accuracy in the range of tens or hundreds of micrometers, which is almost comparable to industrial computed tomography, but we only used two or three reference radiographs. These results are significant for industrial quality control. Acquiring two or three radiographs only takes a couple of seconds, so we significantly reduce the X-ray machine time and the time required to detect manufacturing errors.

Keywords: X-ray, radiography, metrology, optimization

Contents

Introduction

"All is number" or "all things are numbers". These are said to be the fundamental concepts of the ancient school founded by the Greek philosopher Pythagoras in the sixth century BC [Zhmud, 1989, Russell, 2004]. This perhaps a little mystical conclusion was coming from their early discoveries about numbers and their relation to astronomy, geometry, or even music. Nowadays, numbers are an every-day concept used for describing the world around us. And to assign meaningful numbers to the physical objects, we need *measurements*.

Measurement is probably one of the most essential tools in science. It connects mathematics with the matter, enables conducting experiments and objectively describing physical phenomena [Finkelstein, 2014]. But it was not until 1875 that the *International Bureau of Weights and Measures* (BIPM) was set up with the goal of worldwide measurements unification. And in 1960, their *metric system* adopted the name *Système International d'Unités*, abbreviated *SI* [BIPM, 2019].

Measurement is not only about numbers and their units. In order to get to these numbers, we need *measurement technology and instruments* for measuring the physical entities around us. And with the development of modern electrical sensors and computer science, we significantly extend the range of *what* can be measured and *how accurately* we can do it.

Background and motivation

This thesis aims at measuring objects in the *manufacturing* context (Figure 1). Typically, products are designed with a set of requirements and tolerances defining the intended quality. Manufacturing is never hundred percent accurate[1], so when physical products are made, their real quality has to be verified. For example, an automobile typically consists of several thousand components and if any of them is too big or too small, the parts may not fit together well, and the car may not perform as intended by the designers and engineers [Montgomery, 2009].

In case of automobiles or airplanes, it is also an important safety aspect: imagine if bolts holding an airplane engine were too small and engine vibrations would cause them to fall off the plane. That is, of course, unacceptable, so the designers of the individual parts and components need to specify *dimensional tolerances*, i.e., the minimum and maximum permissible dimensions.

The tolerances depend on how exactly the products are going to be used. LEGO® bricks are an interesting example: to ensure their perfect fit, tolerances as low as 0.02 mm are used for their manufacturing [Nguyen et al., 2019]. But such low tolerances are not possible with all technologies. For example, with stereolithography 3D printing, it is generally not possible to achieve accuracies better than ±0.1 percent plus 1 mm, whereas with computer numerical control (CNC) machining, 0.01 mm accuracies are practically achievable [Lefteri, 2012].

The lower our tolerances are, the more accurate measurements we need. One of the oldest way to measure objects with good precisions is using *contact methods*.

[1] A review of various manufacturing technologies and materials, including their achievable accuracies, was written for example by Lefteri [2012].

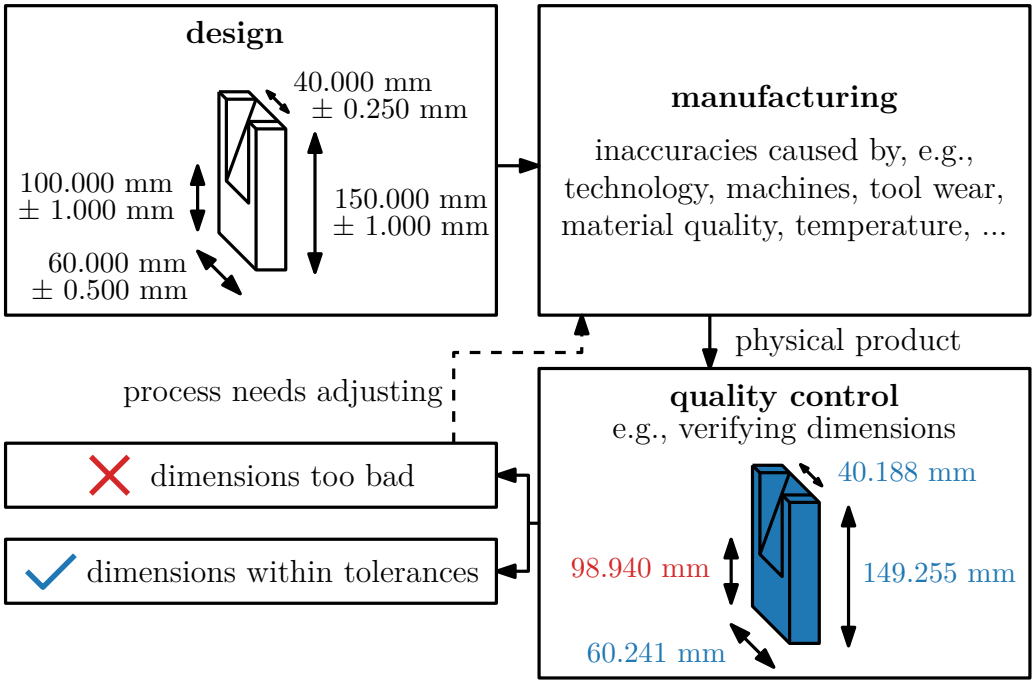


Figure 1: Illustration of a typical manufacturing workflow. An object is designed with certain parameters. When we manufacture the object, we must verify that the actual parameters conform to the design. Based on the results, we either accept the object, or if it does not meet the requirements, we may need to adjust the manufacturing process to prevent the errors in the future.

Contact methods Dimensional measurements with contact methods such as *rulers* and *calipers* date back to ancient Rome [Ulrich, 2007]. Modern calipers with vernier scales easily provide a precision to 0.02 mm, so-called *micrometers* even 0.004 mm [Bewoor and Kulkarni, 2009]. With the development of electronics, these devices are available with linear encoders and digital displays.

More advanced and modern contact inspection devices include *coordinate measurement machines* (CMM). These have a moveable sensing probe whose three-dimensional coordinates are tracked in space with resolutions as high as 0.0001 mm [Bewoor and Kulkarni, 2009]. By touching the measured object with the probe at various positions, dimensions can be computed from the coordinates.

Complex geometries and non-destructive evaluation Despite the great accuracies of the contact methods, it is not always possible to use them. Nowadays, modern manufacturing processes are getting faster and more complex, which requires more sophisticated solutions for measurements and quality control. Especially the recent research in *additive manufacturing* (3D printing) brings increasing complexity of the geometries that can be created from various materials.

This presents very demanding challenges for product inspections [Du Plessis et al., 2018, Villarraga-Gómez et al., 2018, Xu et al., 2019]. First, how can we accurately measure dimensions of easy-to-deform or flexible materials if they deform under the pressure of contact instruments? And second, how to measure dimensions of difficult-to-reach or even fully internal geometrical features without cutting or otherwise destroying the object so that we can still use it after?

This is a sub-problem of a more general field called *non-destructive evaluation* which is not only about dimensional measurements but also other kinds of defects and there is a panorama of optical, thermal, ultrasonic, electromagnetics, or radiography methods that can be used for that purpose. But that is way beyond the scope of this thesis as we are only interested in dimensional measurements.

Optical methods Let us go back to the two questions above. The first problem, measuring dimensions without any contact, can be solved by various optical methods. These include *optical interferometry*, *time-of-flight technique*, *stereo vision*, *shape from focus*, or *structured light* [Zuo et al., 2018]. They have various accuracies from 0.02 nm (nanometers) in micrometer ranges up to 1 cm accuracies in the range of tens of meters [Blais, 2004].

Among the non-contact optical techniques for dimensional measurements, a very popular one is the *structured light*, typically done with fringe patterns and called *fringe projection profilometry* [Budianto and Lun, 2016, Zuo et al., 2018, Feng et al., 2018]. In its basic setup, it has a simple hardware configuration, high accuracy, high speed, and low cost. It is based on *projecting* fringe patterns on object surfaces and then recording the *reflected* patterns with a camera at a different position. The basic idea is that the surface geometry changes the phase of the reflected pattern based on the surface height. One of the major fringe projections problem is reflections from shiny (specular) surfaces, which can be solved by spraying the object with diffuse powders (but that changes the thickness that we are measuring), or by using high dynamic range (HDR) projection techniques [Feng et al., 2018]. For example, Feng et al. [2018] report accuracies roughly in the range of 0.049 mm to 0.34 mm for various test objects.

X-rays for dimensional measurements The optical methods mainly solve our first problem, i.e., they are non-contact. But they cannot penetrate the objects to simultaneously measure features on both sides, measure internal geometry, etc. For measuring both external and internal features, we can use one of the few capable technologies: *X-rays* (Chapter 1).

X-rays, discovered in 1895 by Röntgen, awarded by the first Nobel prize in Physics in 1901 [Mery, 2015], are electromagnetic radiation just like the visible light. However, they are capable of passing through even thick objects including metals that are otherwise opaque for the visible light. Hence, when we illuminate objects by an X-ray source and detect the intensities that passed through the object to an X-ray detector, we can acquire images called *radiographs*. Unlike regular photographs, the radiographs do not explicitly show the external shape but rather the object *thickness* at varying positions. That is because the X-rays intensity decreases with the object thickness (Section 1.2).

The radiographs are successfully used for dimensional measurements as we discuss in Chapter 2. In specific cases, it is possible to use only a single radiograph [Wawrzinek et al., 1997, Zscherpel et al., 2007] (Section 2.1). But a more advanced technique exists that relies on hundreds or even thousands of these radiographs taken from different calibrated angles. It is called *computed tomography* (CT) from the Greek "tomos", meaning "to slice", and it was invented by Hounsfield [1973] for usage in medicine. With CT, it is possible to *reconstruct* how the scanned object looks like in sections, or *slices*. And with modern accurately

calibrated devices, we can use CT not only for visual inspections, but also for dimensional measurements of both external and internal features with accuracies as good as 0.005 mm and for positional information [Bossi et al., 1991, Kruth et al., 2011, Du Plessis et al., 2018, Villarraga-Gómez et al., 2018, Jones and Huthwaite, 2018, Xu et al., 2019, Butzhammer and Hausotte, 2019] (Sections 2.1, 2.2, 2.3).

Villarraga-Gómez et al. [2018] state that X-ray CT is becoming accepted as a metrology tool and is *"uniquely suited for dimensional measurements of components having internal geometry, difficult-to-reach part features, and easy-to-deform or flexible structures,"* which is exactly our motivation.

Limited-view projections One of the main problems of CT is the necessity to take so many projections. Not only it requires perfect calibrations of the projection angles, not only it produces tens of gigabytes of data that need to be processed, but taking the projections is also a huge time-consuming bottleneck.

Recently, several researchers have aimed at tackling the problem by limiting the number of projections [Fischer et al., 2016, Jones and Huthwaite, 2018, Butzhammer and Hausotte, 2019]. This is called *limited-view* or *sparse-view* CT and accuracies ranging from roughly 0.007 mm to 0.230 mm were reported by various authors using from about 24 to 72 projections (see Section 2.2 for details).

These results are possible by using at least some *prior knowledge* about the inspected objects. This makes sense in the manufacturing context because when we are measuring an object that we have manufactured ourselves, we have a lot of prior knowledge available about how the object *should* look like. Essentially, the real shape should not be very far from the intended design, otherwise we would have a far bigger problem than dimensions being off a few millimeters.

Thesis goals and structure

In this thesis, we partially follow the previously mentioned state-of-the-art research in X-ray dimensional measurements. That means to use as few projections as possible to still estimate reasonably accurate dimensions in non-destructive non-contact quality control. We propose a novel approach (Figure 2) that uses prior knowledge of the objects but does not use CT reconstructions, so we only require a couple of radiographs acquired using standard industrial X-ray setups.

The overall objectives and structure of this thesis are the following.

1. **Fundamentals** We briefly examine the fundamentals of X-rays and radiographs. We first focus on the physics and mathematics behind the process. Then, we explain the technical details of generating and detecting X-rays. Finally, we discuss the radiograph qualities and what errors and difficulties we can expect. After reading Chapter 1, the reader should understand how radiography works.

2. **Related works** We devote Chapter 2 to briefly explaining the research, including state-of-the-art, in using X-rays for dimensional measurements. We mention techniques that only use a single projection, then methods that rely on CT reconstructions, and then the state-of-the-art in limiting the number

of projections. Finally, we mention related research in X-ray simulations (3D rendering) that are important for our own approach.

3. **Our approach** In Chapter 3, we propose the novel approach. We formulate the problem, explain the high-level concept and its components, and then we show the details of each part of the method. Mainly, we parametrize the inspected object and calibrate its material properties, which is our prior knowledge. Then, using a small set of X-ray projections, we find the positional and dimensional parameters of the measured object using optimization methods and 3D rendering.

4. **Implementation** We implemented our approach in a prototype demo application for dimensional measurements from real radiographs of real physical objects. In Chapter 4, we briefly show our implementation using dataflow graphs and GPU-accelerated components such as ray-tracing.

5. **Results and evaluation** Finally, in Chapter 5, we verify that our method is indeed capable of measuring object dimensions on *real data* coming from a real X-ray machine. Our results are compared to dimensional measurements that we performed using other approaches such as contact methods (calipers) and X-ray tomography. We discuss our accuracy, performance, and theoretical method limits.

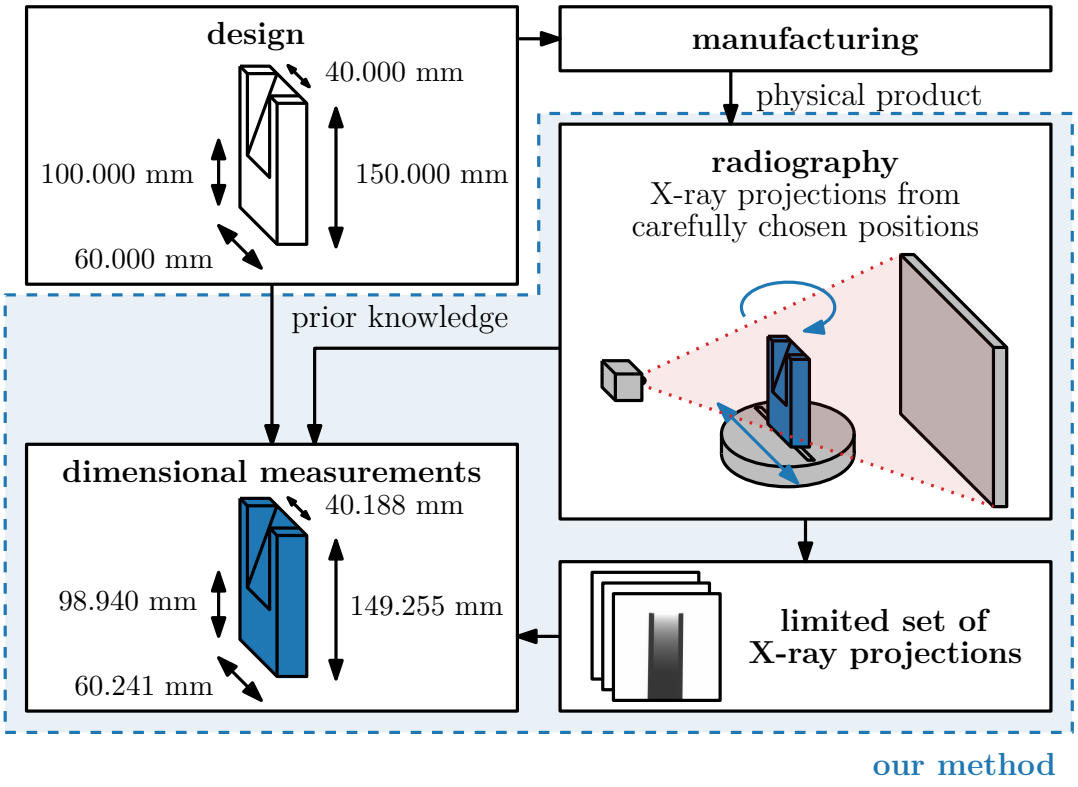


Figure 2: Very high-level overview of our proposed method.

1. Fundamentals

As introduced in the previous chapter, our main topic is *dimensional measurements* using *X-ray projections*. While the concept of measuring object dimensions is fairly simple and was already explained in our Introduction, the concept of X-rays is much more complicated and deserves its own chapter. That is because it requires more detailed understanding of the physics and the technology behind radiography. In this chapter, we introduce the fundamentals of X-rays used for non-destructive evaluation.

A typical X-ray setup for non-destructive evaluation is designed in the following way (Figure 1.1) [Mery, 2015, Villarraga-Gómez et al., 2018, Du Plessis et al., 2018]. An evaluated object is placed in a *manipulator*, which is a device designed for holding the object in a fixed position while also being able to move and/or rotate it. When the object is positioned, we irradiate it with a beam of X-rays, whose elementary properties are discussed in Section 1.1. As the X-rays partially pass through the object, their intensity attenuates due to *interactions* between the X-ray photons and the object atoms described in Section 1.2. Generating the X-ray beam with an *X-ray source* is explained later in Section 1.3. Detecting the rays that passed through the object is described in Section 1.4 followed by a discussion about the quality of the acquired radiographs in Section 1.5.

We refer all readers interested in even more theory that is beyond the scope of this thesis to the following literature. More details about X-ray physics can be found in *Elements of Modern X-ray Physics* by Als-Nielsen and McMorrow [2011]. More details specifically about non-destructive evaluation and visual computing with X-rays can be found in *Computer Vision for X-Ray Testing* by Mery [2015]. Recently, the second edition of *Digital Radiology* by Seeram [2019] was released and includes detailed illustrations of X-ray detectors and image quality.

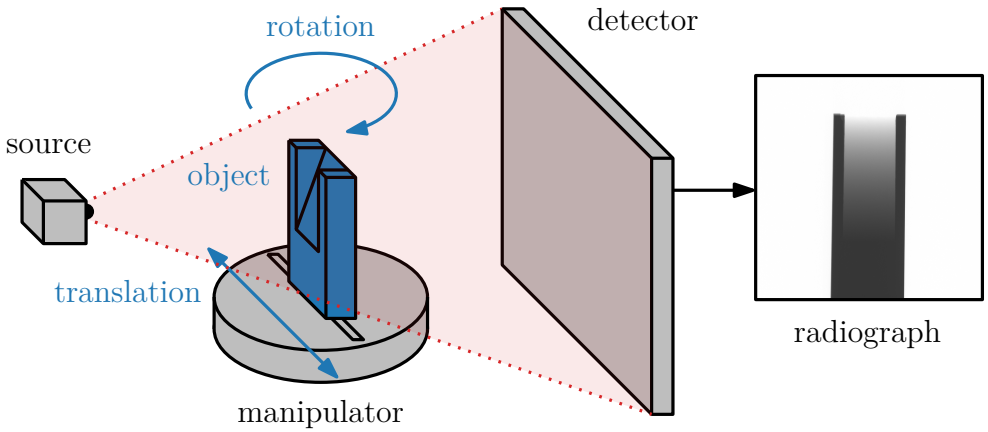


Figure 1.1: Typical setup for X-ray non-destructive evaluation. An evaluated object is placed in a manipulator between an X-ray source and a detector. The object is irradiated and a digital radiograph is formed from the detected beam.

1.1 X-rays as electromagnetic radiation

X-rays are *electromagnetic waves* just like the visible light, radio waves, or microwaves that we use for Wi-Fi and heating our food in microwave ovens. The electromagnetic waves are oscillations of an electric and a magnetic field propagating through a medium. These waves can be characterized by several properties, some of which we now explain. For comparing the properties of X-rays to the other types of electromagnetic radiation, see Figure 1.2 [Mery, 2015].

Wavelength Typically, we characterize waves with *wavelength* λ with units of meters [m]. The wavelengths of X-rays are very short with a very wide range of about 10^{-12} m to 10^{-8} m. Compare that to the visible light ranging from about $4 \cdot 10^{-7}$ m = 400 nm for a blue color to $7 \cdot 10^{-7}$ m = 700 nm for a red color.

Frequency Wavelength λ is inversely proportional to *frequency* $f = c/\lambda$ [s^{-1}], where c is the speed of light. Because of this relation, we can see that the X-rays have much *higher frequencies* than the visible light.

Photon energy The last important property that we mention is *photon energy*, which is directly proportional to frequency f. For this purpose, we look at electromagnetic radiation as if it consisted of *photons*, uncharged particles representing a quantum of an electromagnetic field. A single photon has *photon energy* $\varepsilon = hf$ [J], where $h \doteq 6.626 \cdot 10^{-34}$ J · s is the Planck constant[1].

For convenience, as we will see in Section 1.3, it is common with X-rays to represent this energy with the *electronvolts* [eV] instead of the joules [J]. They are just a different unit where 1 eV $\doteq 1.602 \cdot 10^{-19}$ J. We can now compute that the energies of X-rays range from about 120 eV to 1.2 MeV. Again, compare this to the visible light of about 1.6 eV for red color and 3.2 eV for blue color.

In industry, for non-destructive evaluation, we typically work in the range of 30 keV to 400 keV, whereas in medicine it is usually below 70 keV as we want to penetrate different materials [Kingon et al., 2018].

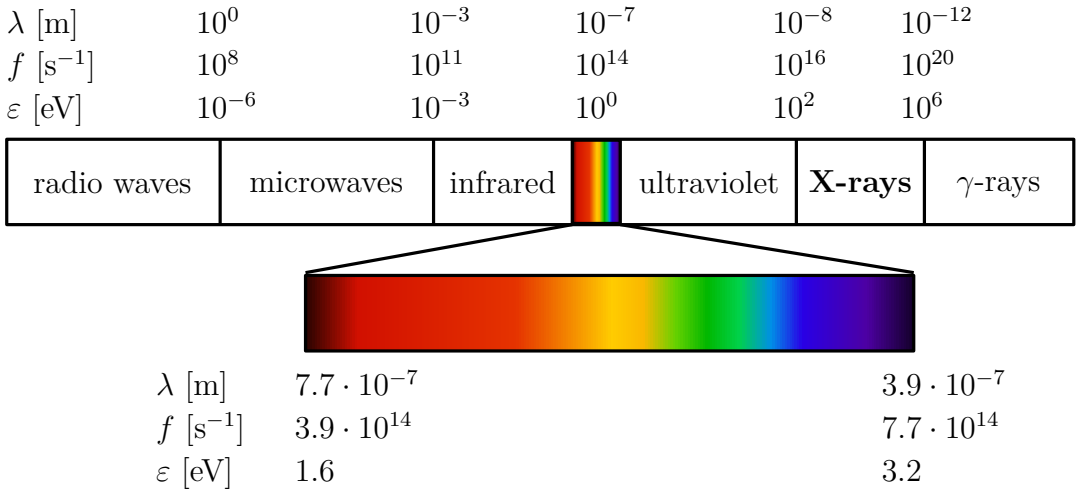


Figure 1.2: Illustration of a part of the electromagnetic spectrum. Symbols used: wavelength λ, frequency f, photon energy ε. Not to scale, values by Mery [2015].

[1] For Planck constant and electronvolt, see BIPM [2019].

1.2 Interactions of X-rays with the matter

We have seen that X-rays are electromagnetic waves similarly to the visible light. And just like light interacts with the objects around us, being reflected from surfaces or scattered in a dense morning fog which the author described in his previous thesis [Iser, 2017], the X-rays also interact with the matter. However, we will see that these interactions significantly depend on the photon energies, which is why X-rays or γ-rays interact differently than the visible light.

Suppose we have our setup from Figure 1.1 with an object being irradiated with an X-ray beam. For convenience, we will model the object as a collection of atoms with electrons in the shells of the atoms, and we will model the beam as a stream of photons. It is clear that the photons *have* to interact with the object in *some way*, because otherwise the detector would detect the same intensities in the whole image and all X-ray projections would be plain white. Then, radiography would not make any sense. So let us now have a look at how individual photons interact with individual atoms [Mery, 2015, Choppin et al., 2013].

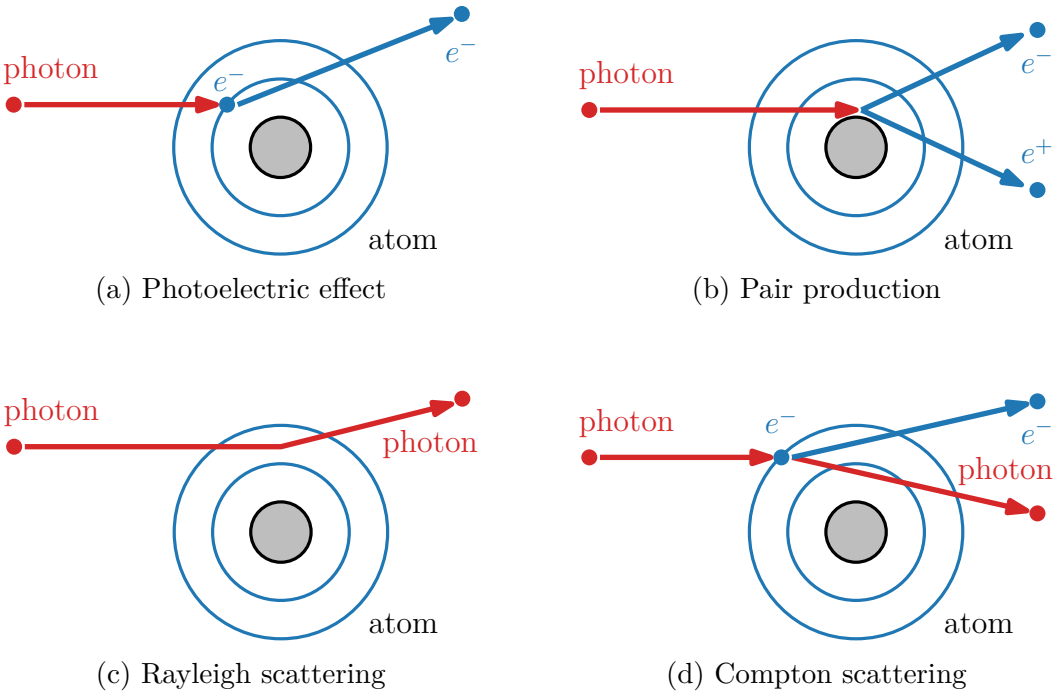
Just like with the visible light in participating media, there are two main types of interaction: absorption and scattering. **Absorption** means that a photon passing through an object is absorbed, the photon is annihilated and its energy is transformed into another form. This happens via the *photoelectric effect*, *pair production*, and partially *Compton scattering*. **Scattering** means that a photon originally flying in some direction is diverted into another direction. During this process, the energy of the diverted photon may remain the same as of the original photon (as in *Rayleigh scattering*) or it may decrease (as in *Compton scattering*).

Photoelectric effect The photoelectric effect (Figure 1.3a) may occur when the photon energy ε is greater than the binding energy of an electron in an atom. This binding energy depends on which chemical element it is and which shell of the atom the electron is in. When the photoelectric effect happens, the energy of the photon is completely transferred to one of the electrons e^- in the shell and the electron is emitted out. The original photon is annihilated.

Pair production Pair production (Figure 1.3b) may occur when a photon with a very large energy ε passes in the proximity of the nucleus of an atom. When pair production happens, the photon energy is converted into mass in the form of two particles: an electron e^- and a positron e^+. Because each of these particles has a non-zero mass of $511\,\mathrm{keV/c^2}$, the original photon needs to have an energy of at least double that amount $\varepsilon \geq 1.022\,\mathrm{MeV}$. The original photon is annihilated.

Rayleigh scattering Rayleigh scattering (Figure 1.3c) is an *elastic* scattering in the sense that no energy of the photon is lost during the process. The original photon is simply diverted into a new direction. In the X-ray range, photons with lower ε get diverted more than photons with higher ε. As an interesting fact, we note that Rayleigh scattering also occurs in the range of the visible light, where it is responsible for scattering phenomena such as the sky color [Elek, 2016].

Compton scattering Compton scattering (Figure 1.3d) is an *inelastic* scattering where the energy of the original photon is *not conserved*. It occurs when



(a) Photoelectric effect (b) Pair production

(c) Rayleigh scattering (d) Compton scattering

Figure 1.3: X-ray photons interacting with atoms and electrons.

the photon energy ε is much larger than the binding energy of an electron in an atom. A portion of the energy is used to strike the electron out from the atom, the remaining energy is reemitted in the form of a new photon in a possibly different direction. Because of that, Compton scattering can be understood as both scattering and absorption as the photon energy is partially lost.

Note: Thomson scattering Please note that in some literature about X-rays, for example in Als-Nielsen and McMorrow [2011], we can also find mentions about so-called Thomson scattering. Same as Compton scattering, it describes an event of a photon being scattered of an electron (Figure 1.3d). Historically, these two events have been divided. Thomson scattering is essentially a low-energy limit of Compton scattering for $\varepsilon \ll mc^2$, where m is electron mass [Moore, 1995]. The major difference is that with these low energies, the radiation energy does not change [Chen et al., 1998], so Thomson scattering is elastic.

We have now seen how individual X-ray photons interact with atoms and electrons. But when working with X-ray projections, we do not track every single photon separately and detect its individual behavior. Rather, our measurements are more global than that, because radiographs essentially correspond to intensities (without giving them any specific units for now) of the X-ray beams detected at different parts of our detector. Hence, we need to understand how the individual interactions influence the *intensity of a beam*.

Attenuations When photons of a beam travel a certain distance x in the matter (Figure 1.5), there is a certain probability that one of the interactions between the photons and the matter will occur. The average distance that a photon can travel until an interaction occurs is called the *mean free path* mea-

sured in meters [m]. The inverse of this value is called the *linear attenuation coefficient* μ [m^{-1}]. The exact number depends on the composition of the matter and the photon energy ε which might be too low or too high for some of the interactions to occur.

When μ is normalized against the volumetric mass density ϱ [$\mathrm{kg} \cdot \mathrm{m}^{-3}$] of a material, we get the so-called *mass attenuation coefficient* $\mu_m = \mu/\varrho$ [$\mathrm{m}^2 \cdot \mathrm{kg}^{-1}$] (sometimes reported as [$\mathrm{cm}^2 \cdot \mathrm{g}^{-1}$] as we usually refer to small quantities).

Note that for a fixed ε, we can simply add up the coefficients of different interaction types [Mery, 2015, Choppin et al., 2013]:

$$\mu = \mu_{\mathrm{phot}} + \mu_{\mathrm{pair}} + \mu_{\mathrm{Rayl}} + \mu_{\mathrm{Comp}}, \tag{1.1}$$

where the partial coefficients correspond to the photoelectric effect, pair production, Rayleigh scattering, and Compton scattering, respectively. An example of how these partial coefficients contribute to the total attenuation coefficient depending on ε can be seen in Figure 1.4 for aluminum.

Attenuation law Let I_0 be the initial intensity of a monochromatic[2] narrow beam travelling a distance x through a homogeneous material (Figure 1.5). The

[2]Monochromatic means that all photons in the beam should have the same initial energy ε.

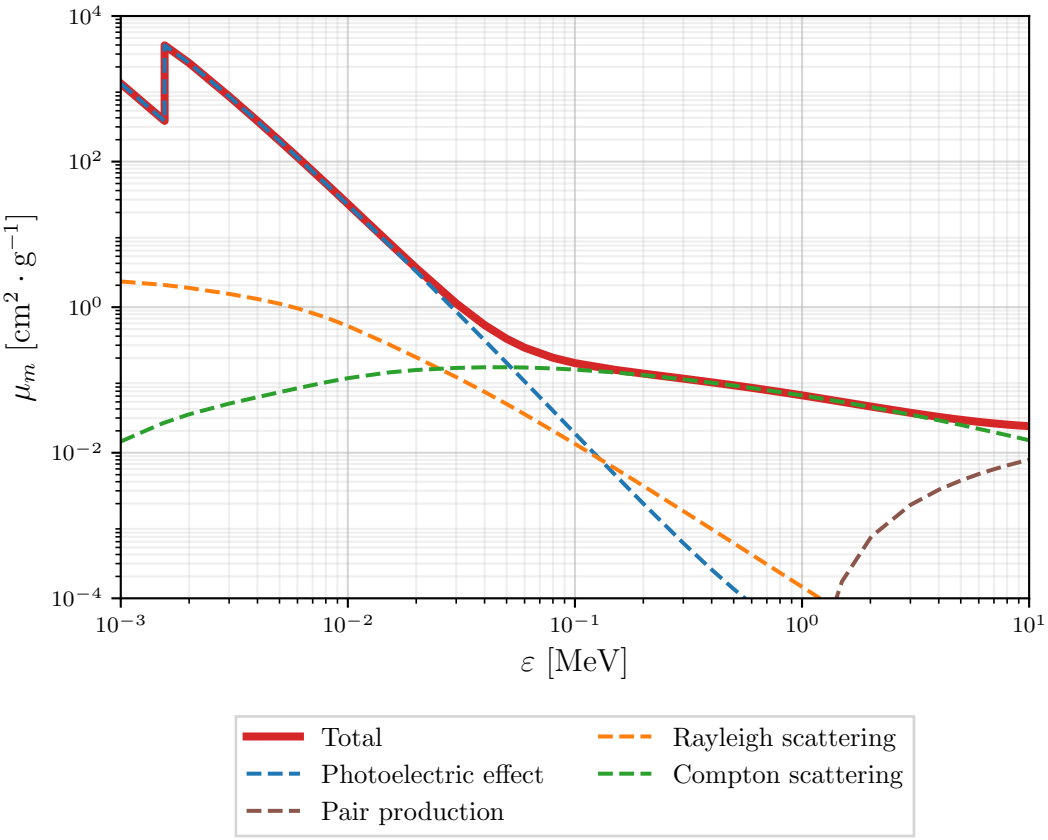


Figure 1.4: Partial and total mass attenuation coefficients for aluminum and photon energies ε from 1 keV to 10 MeV. Both axes are logarithmic. Notice the discontinuity (edge) in photoelectric effect when ε exceeds energy required to remove electrons from a higher atom shell. (Values from Seltzer [1987].)

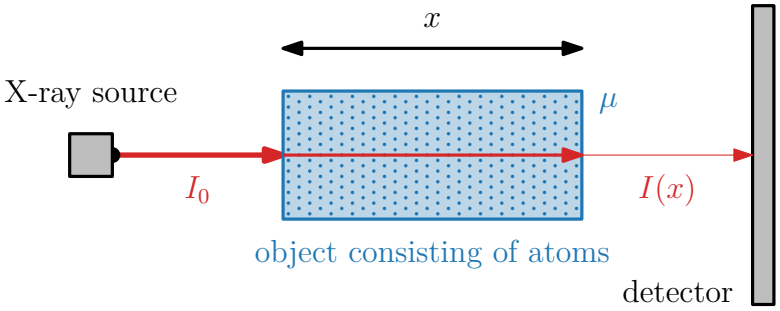


Figure 1.5: Photons in a narrow beam travelling through and interacting with a homogeneous material. Note that this model does not consider that scattered photons may reach the detector at different positions, nor does it model the fact that photon energies decrease during some interactions, which changes the received spectrum. Compare to full Monte Carlo simulations in Section 2.4.

intensity I after passing through the material decreases *exponentially* with x and the attenuation coefficient μ:

$$I(x) = I_0 \cdot \exp(-\mu x). \tag{1.2}$$

For X-rays, this equation is sometimes called the *attenuation law* or the *absorption law* [Vidal et al., 2009, Mery, 2015]. However, especially in the context of the visible light, where the exact same formula is used, it is sometimes referred to as the **Beer-Lambert law** [Marinovszki et al., 2018, Vidal et al., 2009, Vidal and Villard, 2016, Elek, 2016].

More generally, we can rewrite the formula as:

$$I(x) = I_0 \cdot T(x), \tag{1.3}$$

where T is the *transmittance*, *transmission*, or *throughput* of the path through the material, in our case $T(x) = \exp(-\mu x)$. The transmittance then corresponds to what we can see on radiographs, where thick and dense materials have very low transmittance (the image is dark) and the air or soft tissues have very high transmittance (the image is light).

1.3 X-ray sources and their spectrum

Now that we know what X-rays are and how they interact with the matter, it is time we explain *how to generate* them [McCollough, 1997, Als-Nielsen and McMorrow, 2011, Mery, 2015]. This knowledge is important for our work as the X-ray sources that we typically use have very wide *spectrum* of photon energies ε that they generate (Figure 1.8). And we already know that photons with different ε interact differently, so it is important to understand how the typical generated spectra look like.

The first experiment When Röntgen first discovered the X-rays in 1895, he was originally performing experiments with radiation coming from electrodes in evacuated glass tubes. He noticed that even when the tube was fully covered in a darkened laboratory, something caused a fluorescent screen placed far from the

tube to shine [Swindells, 1928]. During closer inspection, he noticed that he could see the bones inside of his own hand casting a shadow on the screen. It was clear that he must have discovered a new kind of radiation that could be incredibly useful in medicine and other fields. As the first tubes for generating X-rays were not very reliable, it was necessary to develop a new design.

Coolidge tube The standard X-ray tube was developed by Coolidge in 1912. This tube (Figure 1.6) consists of a *heated cathode* and a water-cooled *anode*. Between them, we apply a high voltage U. The cathode itself is made of a filament that we strongly heat by voltage U_h. Because of the heating, electrons are emitted from the filament and these electrons are immediately accelerated towards the anode thanks to the voltage U. As the accelerated electrons collide with the anode, *photons* are emitted out.

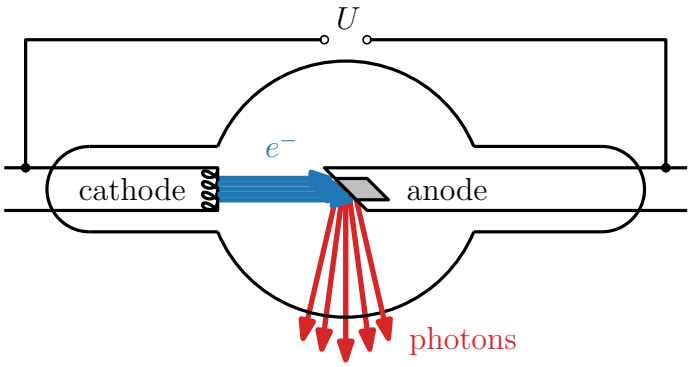


Figure 1.6: Diagram of a Coolidge tube.

There are two distinct ways *how*, or perhaps *why*, the photons are emitted from the anode. Some of the photons are emitted because of the so-called *Bremsstrahlung*, but these photons are not the only ones. When we look at the emitted electromagnetic spectrum (Figure 1.8), we can not only notice a *continuous* "hump", but also a few *discrete* spikes of a characteristic radiation.

Continuous X-rays (Bremsstrahlung) When a highly accelerated electron comes very close to a nucleus of one of the anode atoms, it becomes attracted to the nucleus and gets *deflected* with a Coulomb force. During the process, the electron is slowed down (hence the German name *Bremsstrahlung*, where *bremsen* means *brake* in German), so it loses some or all of its kinetic energy, which is emitted as a photon. Because the amount of energy depends on how close the electron happened to be to the nucleus, the energy spectrum is *continuous*.

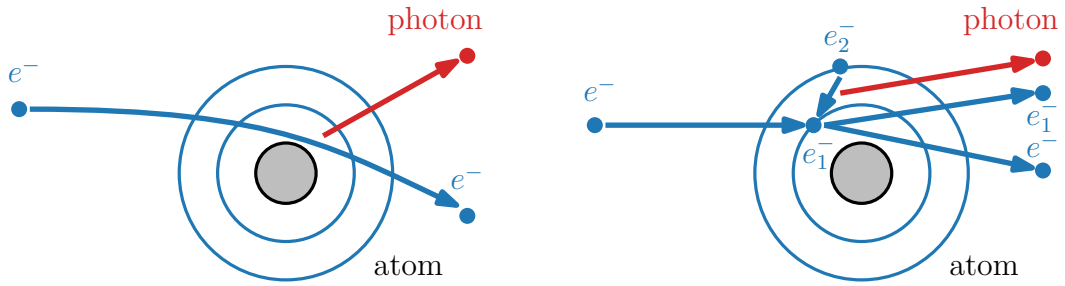
Notice that the emitted photon energy ε can never be higher than the kinetic energy E of a single accelerated electron. As the electrons are accelerated with the cathode-anode voltage U, we have $E = e \cdot U$, where $e \doteq 1.602 \cdot 10^{-19}\,\mathrm{C}$ is the elementary charge. So ε can never be larger than $e \cdot U$, which from the definition of electronvolts[3] is equal to exactly U electronvolts. For example, in a tube with voltage $U = 220\,\mathrm{kV}$, the maximum emitted photon energy is $\varepsilon = 220\,\mathrm{keV}$.

[3] One electronvolt is equal to the kinetic energy that an electron acquires by passing through a potential difference of one volt in vacuum. [BIPM, 2019]

Discrete (characteristic) X-rays In addition to Bremsstrahlung, there is a second way how photons are emitted from the tube anode. When an accelerated electron collides with an anode atom, it may knock an atomic electron out of one of the inner atom shells. As both electrons leave, there is a vacant spot in the atom shell, which is immediately filled with an electron of an outer shell. As this electron jumps from the outer to the inner shell, the energy difference between the energy levels is emitted as a photon. Because the energy levels are precisely specified for every element atom, the energies of photons emitted this way may only be *discrete* from a specific set of values.

We have seen a Coolidge tube and how exactly photons are emitted from it. It remains to note a few practical concerns about the cathode and anode. First of all, the hot cathode is usually made of tungsten (W, wolfram) as it has a high melting point of about 3380 °C. The same material is typically used also for the anode as tungsten additionally has a high atomic number and high thermal conductivity. That is important because about 99% of the incoming kinetic energy is converted into heat and only about 1% into X-ray photons [Mery, 2015].

This has a consequence for *X-ray projections*. In order to get sharp radiographs without geometric blur, we would need an infinitely small focal spot (Sec-



(a) Source of continuous X-rays (b) Source of discrete X-rays

Figure 1.7: Accelerated electrons interacting with the Coolidge tube anode atoms.

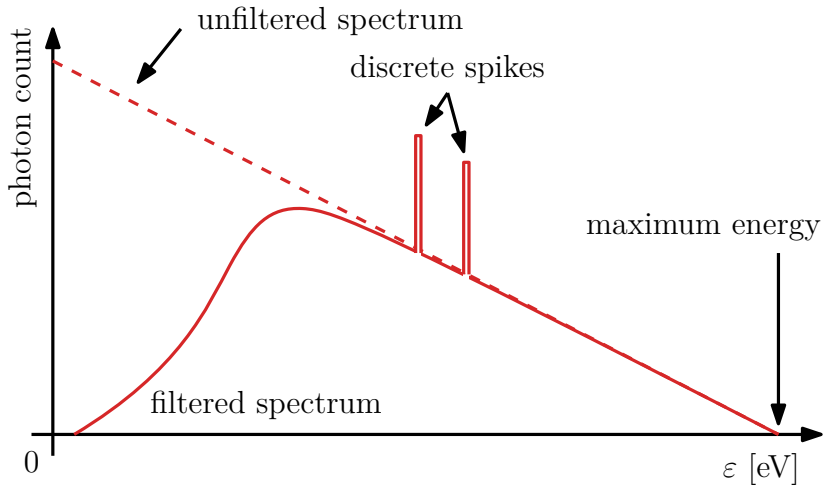


Figure 1.8: Illustration of an approximate spectrum of photon energies ε emitted from a Coolidge tube. Low-energy photons are removed by inherent or added filtration [McCollough, 1997].

tion 1.5, Figure 1.11). But then the heat would not be dissipated quickly enough and the anode would essentially melt, destroying the Coolidge tube. Hence, a few tricks need to be used such as a *rotating anode* (which helps distributing the heat) or inclining the electron beam in the focal spot. This is already beyond the scope of this thesis, but it is important to keep in mind that the technology that we use is never as simple as explained in theoretical backgrounds.

As a final note for this section, we mention that there is an ongoing research in developing new X-ray sources discussed for example by Als-Nielsen and McMorrow [2011]. Specifically, for certain physical experiments, it is necessary to use X-ray sources that emit a large number of photons per second with tuneable and narrow spectra, which is not possible with the standard tubes. To overcome the limitations, it is possible to use *synchrotron sources*, which are particle accelerators where radiation is emitted from charged particles flying at high relativistic speeds. Another option is to use a *free-electron laser*: the first one for hard X-rays was started in April 2009 [Als-Nielsen and McMorrow, 2011].

1.4 X-ray detectors

The last part of our setup from Figure 1.1 that remains to be explained is the *X-ray detector*, which is an essential part responsible for creating the radiographs. It is a complicated component with many possibilities of how exactly it can be constructed. In this thesis, we are not much interested in deep engineering details of various scintillation layers and photoconductors. Instead, we introduce the fundamental concepts of selected X-ray detectors, which will be relevant for the next Section 1.5, where we discuss image quality of digital radiographs.

Analog radiography When Röntgen discovered the X-rays in 1895, he used *fluorescent screens* that emit visible light when hit by X-rays. Such a screen can be placed on top of a *photographic film*. When hit by X-rays, the light from the screen acts on the film together with the X-rays themselves and a radiograph is captured [Swindells, 1928, Oborska-Kumaszyńska and Wiśniewska-Kubka, 2010, Seeram, 2019]. For convenience, both the screen and the film were placed in a *film-screen cassette* and the film was then *chemically processed* so that it could be displayed. This is known as *analog radiography* or *film-based radiography* and for a long time, it used to be the standard way of taking radiographs. Because of the film chemical structure, the parts hit by X-rays became *dark*. That is why conventional medical radiographs have a *black background* with *white (transparent) bones*.

Image intensifiers and CCD sensors Because fluorescent screens were very dim, a technology was needed for improving the brightness. In 1948, an X-ray *image intensifier* [Yaffe and Rowlands, 1997, Mery, 2015, Seeram, 2019] was developed for this purpose. It is a vacuum tube with a fluorescent screen and a *photocathode* on its input (Figure 1.9). As X-ray photons hit the screen, photons of visible light are emitted and they immediately hit the photocathode, which converts them into *electrons*. The electrons are then accelerated and focused towards an *output phosphor screen* which converts them to visible light again.

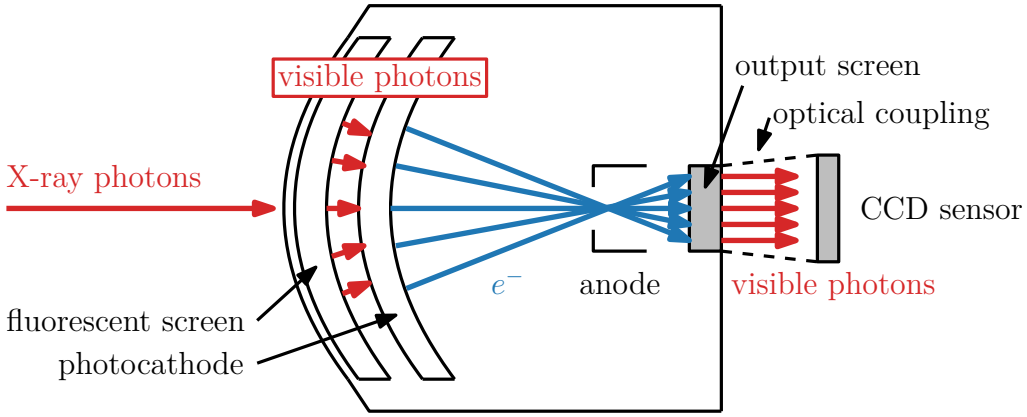


Figure 1.9: Image intensifier with a CCD camera.

The reason why we mention this old technology is the following. With the later development and expansion of *CCD sensors* in 1970s and 1980s, the intensified image from the vacuum tube could be *digitally captured in real time* as described by Mery [2015] or Yaffe and Rowlands [1997]. A CCD (charge-coupled device) sensor is based on a CCD array corresponding to individual pixels, where incident light is converted into an electrical charge. This charge can be rapidly read using a system of registers and a whole image can be formed.

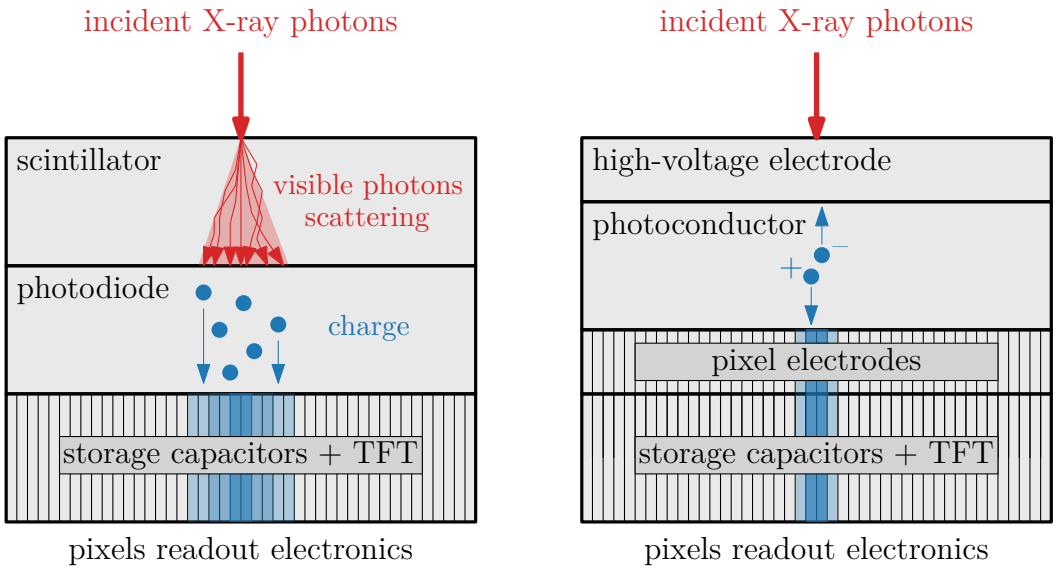
Nowadays, modern CCD sensors are available. But capturing the image from an intensifier results in *geometric distortion* as the surface is curved, limited resolution of intensities, too small image surface, and the whole setup is large and heavy. Several other possibilities exist to *couple* a CCD sensor directly to a screen, such as *optical lens coupling* or *fiber optic coupling*, but these have their own problems discussed for example by Yaffe and Rowlands [1997].

Computed radiography In 1980s, another technology relevant to digital radiography was born. The so-called *computed radiography* (not to be confused with *computed tomography*) [Seeram, 2019, Oborska-Kumaszyńska and Wiśniewska-Kubka, 2010] enables digitally reading an acquired radiograph. It is based on *imaging plates* with *photostimulable phosphors*. When an X-ray source shines a beam through an object to the imaging plate, a *latent image* is created in the plate. Later, the plate is placed in an image reader that reads the latent image by scanning the plate by a laser beam via *photostimulable luminescence*, which is measured, read-out, and digitized. After the digitization, the plate can be erased and the whole process repeated with another radiograph.

Obviously, this process has its own disadvantages [Seeram, 2019] like low efficiency, low spatial resolution, and the necessity to physically move the imaging plate to a separate device for reading and digitizing the image.

Flat-panel digital radiography Approximately in 1995, a better technology became available to overcome the shortcomings of image intensifiers and computed radiography and is becoming the standard way of capturing digital radiographs. The technology is called the *flat panel detector* [Seeram, 2019, Mery, 2015, Körner et al., 2007, Yaffe and Rowlands, 1997].

It is a very light and thin panel which directly contains everything required for



(a) Indirect flat panel (b) Direct flat panel

Figure 1.10: Illustration of the flat panel technology.

detecting and digitizing X-ray projections without any geometric distortions. Like CCD sensors, a flat panel is based on pixel arrays from which the corresponding pixel values can be read. In our flat panel used in Chapter 5, the pixel matrix is approximately 40×40 cm large with a resolution of 4096×4096 pixels. In a flat panel, each pixel has a *thin-film transistor* (TFT) and a *storage capacitor* which collects and stores an electric charge proportional to the incident radiation that it received. What remains to be answered is how the incident X-ray photons get converted into the electric charge inside the pixel.

a) Indirect flat panels One possibility is to use an idea similar to what we have already seen: use an *X-ray scintillator* which converts X-ray photons into visible light photons, and then detect these photons using a *photodiode*, which converts them into the electric charge (Figure 1.10a). This is called an *indirect flat panel* as the X-rays are detected indirectly via a scintillator. The main difference to the CCD sensors is that the scintillator is directly embedded in the pixel array so there is no need for any image intensifier, optical, or fiber-optic coupling.

There is two main possibilities how the flat-panel scintillator may be constructed: with a *powdered phosphor* or with a *structured phosphor*. The structured phosphor is organized in thin columns (needles), which reduces spatial spreading of the light inside the scintillator.

b) Direct flat panels Another technology of flat panels is called *direct flat panels* (Figure 1.10b). These do not have any scintillators at all, rather, they use *X-ray photoconductors* which can directly convert X-ray photons into the electric charge. The physics behind these photoconductors are complicated and various photoconductors may be used (for a detailed study, refer to Kabir and Kasap [2017] or Kasap et al. [2011]). The basic idea is the following: the photoconductor has electrodes, a top one (incident to X-rays) with a high voltage and then a bottom one, which is connected for each pixel separately. When an X-ray photon

hits the photoconductor, it is absorbed, and an electron-hole pair is generated. The pair travels along the electric field created by the high voltage and the holes accumulate on the storage capacitors.

1.5 Radiograph image quality

In the previous sections, we saw how a typical non-destructive evaluation setup from Figure 1.1 looks like and how its individual components work. The output of the whole setup is a *two-dimensional digital radiograph* consisting of individual pixels, whose intensity values would, ideally, be proportional to the transmittances along the optical paths through the projected objects (Section 1.2).

Unfortunately, when the setup is constructed in a real world, it is never perfect and all its imperfections, mainly caused by the detectors, manifest in the form of quality loss in the acquired image. Some of the issues may not be noticeable at first, but when we are developing an algorithm that takes these imperfect images on its input, any slight error may *propagate* to the quality of the algorithm outputs. When we evaluate our own algorithm in Chapter 5, some of the image imperfections will be noticeable in our results and we will often refer back to this section. So let us have a look at a few selected radiograph characteristics.

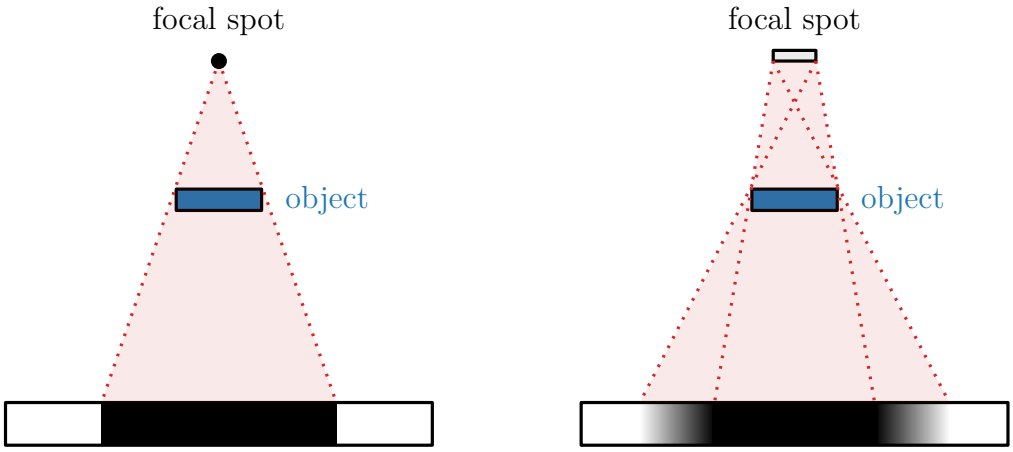
Focal size In a sense, taking radiographs (Figure 1.11) is the same as taking photos with a camera. The first difference is that instead of visible light photons, it is X-ray photons that we capture, although we already know that many detectors first convert them to a different spectrum anyway.

The second is that we do *not* use any *lenses or mirrors* for focusing the "light" because constructing them for X-rays is incredibly difficult: refractive index n of vacuum is by definition 1.0, for many materials transparent to the visible light it is typically between 1.2 and 2.0, but for X-rays it is only around 0.99999, which is even *below* 1.0 because of resonance [Als-Nielsen and McMorrow, 2011]. So taking radiographs is more like taking photos with an "old-school" *pinhole camera*.

The third difference is that with X-rays, the objects that we are "photographing" are *between* the "pinhole" and the detector, not outside. The "pinhole" itself acts as a source of the photons, in other words, the X-ray source *is* the "pinhole". Unfortunately, the X-ray source is not infinitely small, so the "pinhole" has a diameter which causes *geometric blur* of the image (Figure 1.11).

This *focal size* or *focal spot diameter* for medical X-rays can typically range from 0.3 mm to 2.0 mm [Vidal and Villard, 2016]. In our industrial setup that we use for evaluation in Chapter 5, we have a focal size as small as 0.002 mm to 0.006 mm, which of course significantly lowers the possible geometric blur.

Refraction and reflection So far, we have talked about the X-rays as individual photons interacting with atoms and electrons. However, there are two phenomena that strongly depend on the wave characteristics of X-rays: refraction and reflection. In the previous paragraphs, we already mentioned that the X-rays refractive index of many materials is around 0.99999, which is *below* one. When we plug these values to the Snell's law of refraction [Als-Nielsen and McMorrow, 2011], we can see that the refraction effect is negligible.



(a) Small focal spot (b) Large focal spot

Figure 1.11: Focal size and geometric blur.

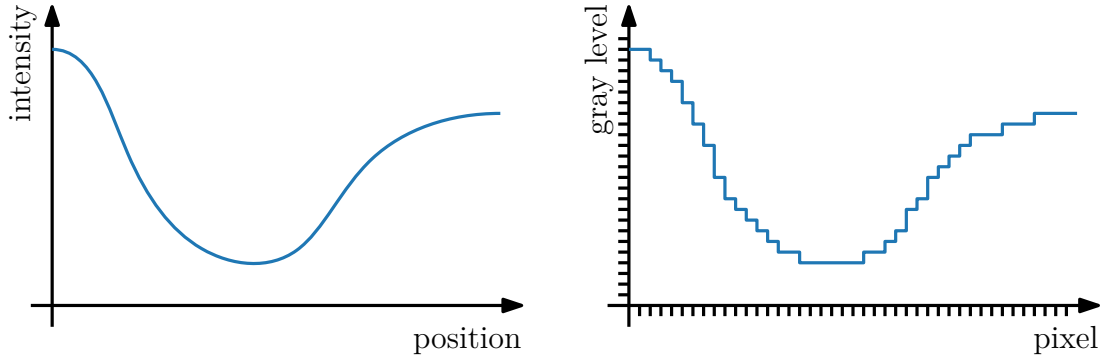
However, as the values are less than one, X-rays are prone to *total external reflections*, which allows constructing sophisticated focusing optics [Als-Nielsen and McMorrow, 2011]. Practically, these external reflections mean that when X-rays hit a surface in an extremely small angle in the range of milli-radians, their energy is reflected, which may be occasionally visible in the radiographs when objects are positioned and rotated in specific ways.

Gray-level resolution As we have seen, the pixels of digital detectors (Section 1.4) usually accumulate an electric charge which corresponds to the intensity of the incident X-rays. But the amount of the charge, which is a *continuous analog* value, has to be somehow measured and converted into a *discrete* value that can be represented in a computer (Figure 1.12). In signal processing, this is called *quantization* and we are interested in *how many gray levels*, or intensity values, a pixel can have [Mery, 2015, Seeram, 2019]. For 8-bit sensors, there are only $2^8 = 256$ possible intensities, so we can only recognize 256 different object thicknesses, which is obviously limiting any dimensional measurements.

Our sensor from Chapter 5 is 16-bit, so $2^{16} = 65536$ intensity values can be represented. In practice, this is not relevant by itself, because if the radiograph darkest pixel had intensity 3000 and the brightest one 5000, then we would still only have 2000 different values. So extra attention must be paid when taking the radiograph so that the covered portion of the interval is as large as possible.

Noise and detective quantum efficiency There is always *random noise* present in the image signal due to the random nature of image quanta (*quantum noise*) and various statistical fluctuations along the whole imaging chain [Seeram, 2019, Kabir and Kasap, 2017]. In direct flat panels, the photoconductors are responsible for additional noise by random charge-carrier trapping, when a charge is not completely collected Kabir and Kasap [2017].

The *detective quantum efficiency* (DQE) measures how *efficient* a detector is in converting the incident X-rays into a useful image signal taking noise into account. The ideal detector would have DQE equal to 100 %, but that is not the case with real detectors. Exact measurements and calculations are complicated,



(a) Original continuous signal (b) Sampled digital signal

Figure 1.12: The originally continuous analog signal is sampled at discrete positional and intensity intervals.

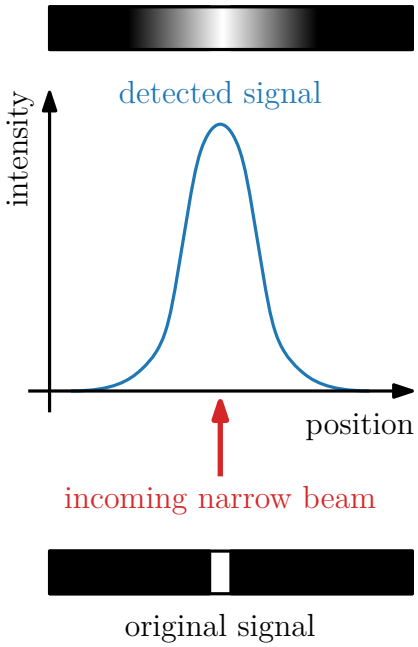
and a discussion is provided for example by Kabir and Kasap [2017] and data by Körner et al. [2007] show that indirect flat panels can achieve higher DQE than direct flat panels.

Spatial resolution *Spatial resolution* describes how well we can distinguish fine details in objects [Seeram, 2019, Kabir and Kasap, 2017, Körner et al., 2007]. Please note that this is not just about the number of pixels. Of course, if we had too few pixels, we would not be able to tell apart different parts of an object. But the problem is that the intensity values of *neighboring pixels* on a detector are more or less correlated. This has a lot to do with the way the detectors are constructed (Section 1.4 and mainly Figure 1.10).

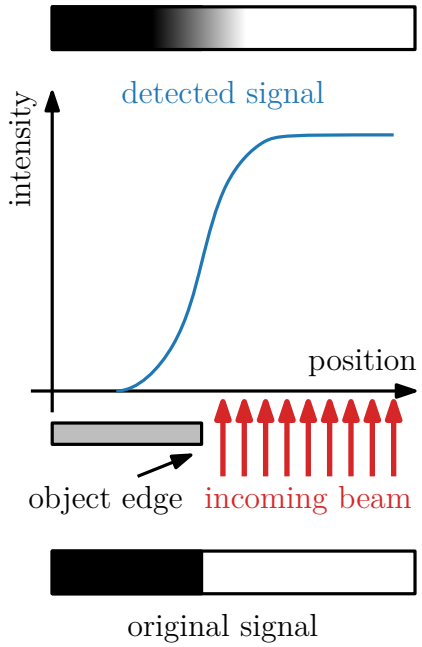
In *indirect flat panels* and other indirect solutions like CCD detectors, an X-ray scintillator is used to convert X-ray photons into light. The scintillator has a non-zero thickness, so the converted light photons can *scatter* inside the structure and may hit neighboring pixels photodiodes [Seeram, 2019, Yaffe and Rowlands, 1997]. This is called *lateral spread* and is especially problematic in *powdered* (*unstructured*) scintillators. The spreading can be reduced by *structuring* the phosphor into thin microscopic needles, which improves the spatial resolution.

In *direct flat panels*, one could think that no spreading is possible and the spatial resolution is perfect. Even though it is much better, spreading to neighboring pixels still occurs due to complicated reasons discussed for example by Kabir and Kasap [2017] or Kasap et al. [2011]. Inside the photoconductor between the two electrodes, a charge-carrier may be trapped and induce charges not only to the central pixel but also to its neighbors. If a hole is trapped, charge is added to the neighbors, but if an electron is trapped, a charge with an opposite sign is induced, so the charge in neighboring pixels is essentially decreased.

The results of spatial spreading are best seen on *line spread functions* and *edge spread functions* (Figure 1.13) [Konstantinidis et al., 2012, Kingon et al., 2018, Seeram, 2019]. Line spread function represents detected signal of a narrow slit (line). Edge spread function represents detected signal of an edged object. On both of them, we can see that sharp discontinuities are *smoothed* via the spreading to neighboring pixels.

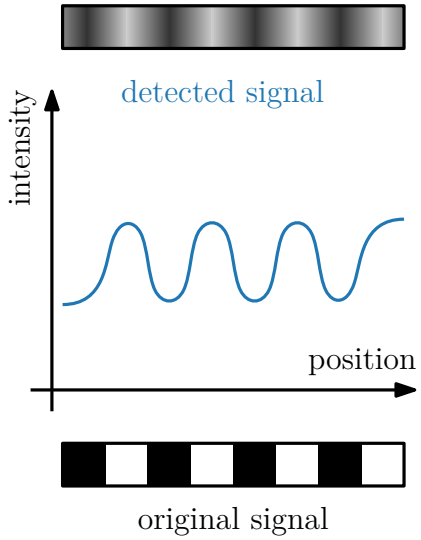


(a) Line spread function

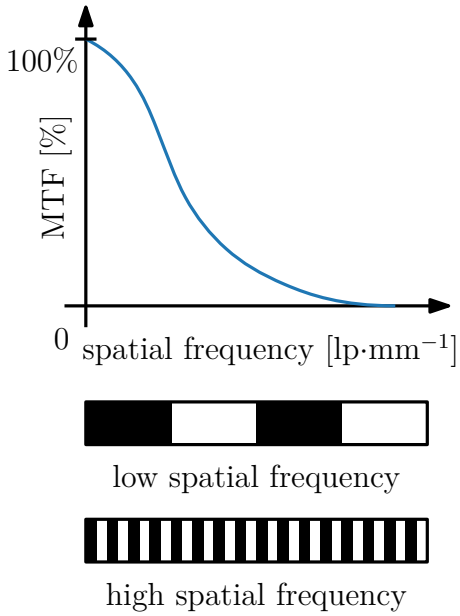


(b) Edge spread function

Figure 1.13: Illustrations of line spread and edge spread functions of an indirect X-ray detector (not to scale, see text for explanation).



(a) Line pairs blurring



(b) Modulation transfer function

Figure 1.14: Illustration of line pairs and their blurring which is expressed in a modulation transfer function (not to scale, see text for explanation).

Modulation transfer function The spatial resolution can be expressed in a single *modulation transfer function* (MTF) [Seeram, 2019, Kabir and Kasap, 2017, Kasap et al., 2011, Yaffe and Rowlands, 1997]. It expresses the ability of a detector to resolve very thin objects placed very close to each other. Imagine a black line placed next to a white line (Figure 1.14), this is called a *line pair* (lp). When these pairs are repeated many times within one millimeter, we measure this *spatial frequency* ξ in $[\mathrm{lp} \cdot \mathrm{mm}^{-1}]$. The value of MTF for a certain ξ says how well *contrast* is preserved, i.e., the ratio between the output contrast and the input contrast for a given ξ.

Ghosting *Ghosting*, *image lag*, or *memory effect* all refer to the fact that when an X-ray beam is turned off or when an object moves, the previous image can still be seen (superimposed on the new one) [Seeram, 2019, Kabir and Kasap, 2017] (see Figure 5.14 in Section 5.4 for an example). In case of *indirect* detectors, the fluorescence layer may keep shining for a longer time than the X-ray photons keep coming, but according to Seeram [2019], they are better than *direct flat panels*. In the direct panels, some of the drifting carriers are trapped in the photoconductor and are only released later, which for some photocunductors may take several hundreds or even thousands of seconds [Kabir and Kasap, 2017]. Hence, it influences not only real-time radiography, but also taking regular radiographs with a delay of a few seconds or even minutes.

Defects and flat fielding Finally, it is important to realize that the *raw data* coming from a detector, before being preprocessed, may contain various *artifacts* because of manufacturing defects [Seeram, 2019]. These include individual *dead pixels* as well as small defective pixel *clusters* or even *whole columns*. Furthermore, areas may have different sensitivities to the radiation. Because these defects are fixed (it is always the same pixels having the same artifact), the raw images can be corrected via calibration and preprocessing, usually referred to as *flat fielding*.

2. Related works

In Chapter 1, we presented the fundamental digital radiography concepts. Now, we discuss the existing research that we found relevant for the topic of non-destructive dimensional measurements from a single or multiple X-ray projections.

Before we introduce the related works, let us note that the area of measurements and non-destructive testing and evaluation is very broad and active. There are specialized journals like *NDT&E International* that has been publishing new peer-reviewed original research results since 1967, nowadays tens of articles eight times a year. Along radiography, these works include optical, thermal, ultrasonic, and electromagnetics methods. In the following sections, we focus on the most related works in radiography and we discuss their main contributions.

2.1 Dimensional measurements with X-rays

Measuring object dimensions from projected images is problematic. Notice that for a two-dimensional image, there is infinitely many objects that could have created that image (Figure 2.1) [Goldstein and Brockmole, 2017]. Even if we knew the shape of our object, for example, a rectangle, it could still have infinitely many sizes depending on the distance from the detector. Nevertheless, there are of course various approaches that simply go around this problem. In this section, we show how one can do that.

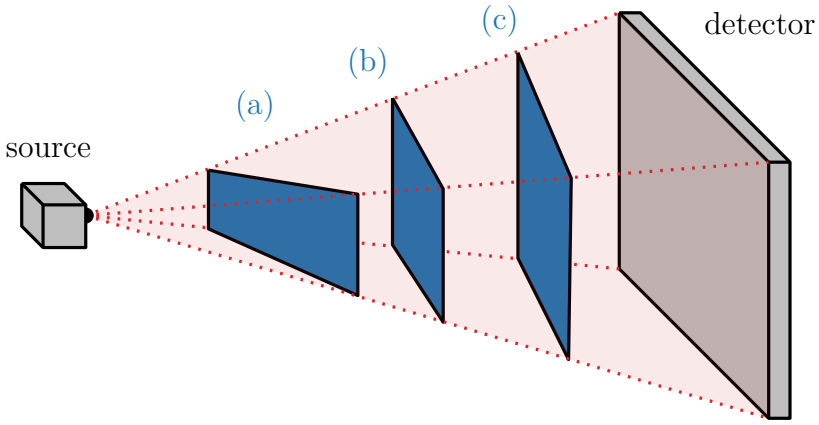


Figure 2.1: When projected, all of the objects (a), (b), and (c) would correspond to the same two-dimensional image even though they have different dimensions.

Single projection Measuring dimensions from a *single* radiograph may be possible in specific situations. We show two standard methods, one reportedly in use since 1950s, for measuring *wall thickness* of tubes (Figure 2.2), described for example by Wawrzinek et al. [1997] or Zscherpel et al. [2007]. Note that similar techniques could be derived for other shapes as well, but we only found references for measuring corrosion in tubes and pipes, so we show these.

Assume we have a single projection of a tube with unknown inner wall thickness w as in Figure 2.2. When projected on a detector, the inner wall has width w'

bounded by the *tangential* rays. With the use of software, the width w' (in pixels) can be automatically determined by detecting edges in the intensities, for example by analyzing the first derivative. The real width w (in metric units) is then calculated from w' and a priori known dimensions such as the tube radius r and source-detector distance f. This is called the *tangential technique* and can be derived even for more complicated tubes with multiple layers.

Yet another approach is called *double-wall technique*. Instead of measuring widths on a radiograph, we measure local changes in intensities accumulated through walls on both sides (hence double wall). We already know that intensities exponentially decrease with thickness along an X-ray beam (Equation 1.2). We calibrate the attenuation coefficient μ from reference points and then we can calculate *relative changes* in thickness (corrosion) from local changes in intensities.

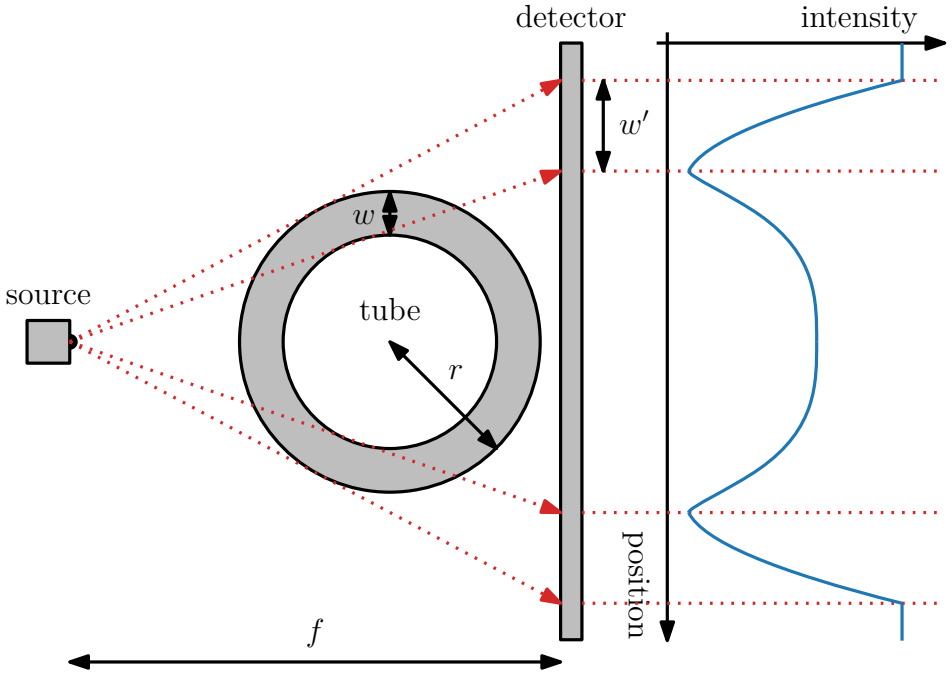


Figure 2.2: Radiograph projection of a tube and the corresponding intensities.

Computed tomography Another common possibility is to use *computed tomography* (CT) invented by Hounsfield [1973]. It allows reconstructing cross-sectional slices from X-ray projections taken from many different angles. Later, Feldkamp et al. [1984] and Feldkamp and Jesion [1986] published the first approximate algorithm for reconstructing the objects into volumetric 3D arrays consisting of voxels (3D pixels). In 1991, the company Boeing was already evaluating the usage of CT for dimensional measurements, reporting accuracies better than 0.1 mm for testing gaps in castings [Bossi et al., 1991].

The first CT machine dedicated to dimensional measurements was exhibited in Germany in 2005 and since then, CT systems are becoming accepted as a metrology tool [Villarraga-Gómez et al., 2018]. A very detailed survey of dimensional measurements with CT with 2011 state-of-the-art was written by Kruth et al. [2011]. We depicted a possible workflow in Figure 2.3.

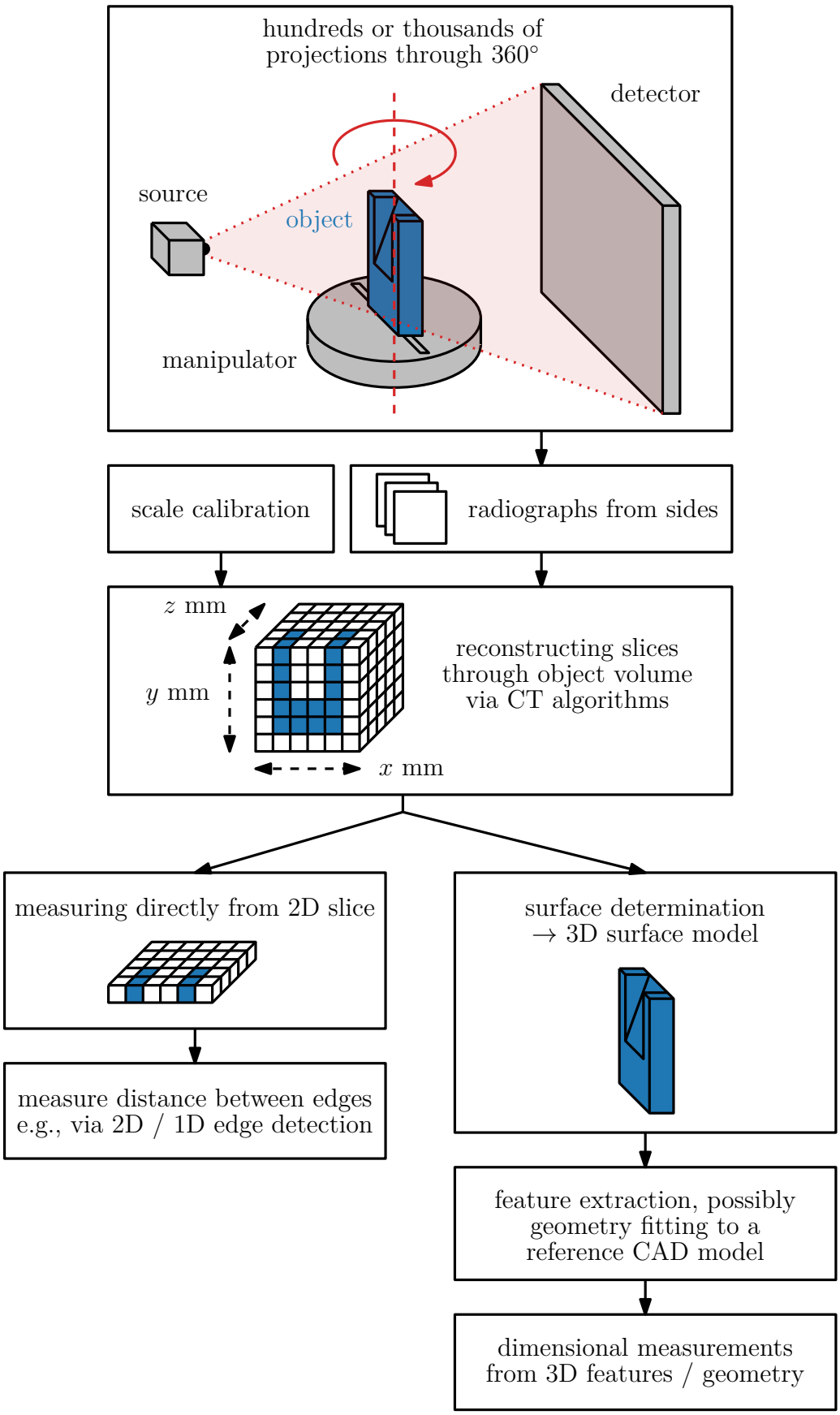


Figure 2.3: Typical computed tomography workflow for dimensional measurements. Projections from different angles are taken and slices through the object are reconstructed. Scale calibration is necessary to ensure correct dimensions of the reconstructed data. Then, we can do measurements directly from 2D slices or we can use surface determination to reconstruct a 3D mesh to measure afterwards.

State-of-the-art tomography measurements We now have a look at more recent state-of-the-art from 2018 and 2019. Detailed evaluations of CT dimensional measurements have recently been written several times by Villarraga-Gómez et al. [2018] (and previously [Villarraga-Gómez, 2016, Villarraga-Gómez et al., 2014]). They report deviations lower than 0.005 mm in comparison to precise *coordinate measurement machines* (CMM) but with the advantages of being contactless and accurately measuring soft surfaces.

Deviations under 0.005 mm for smooth surfaces were also reported recently by Xu et al. [2019] who evaluated CT dimensional measurements for *additive manufacturing*. Specifically, for that use case, they also measured rough surfaces, where the deviations were approximately ten times higher. This can be explained the following way: the CMM measures mostly the peaks of the rough surface, while the X-ray CT acquires an approximate curve between the peaks and valleys. We refer all readers interested in more details about X-ray CT and its usage in additive manufacturing to a recent very detailed review by Du Plessis et al. [2018].

2.2 State-of-the-art in limited projections

In the previous section, we saw that in very specific cases, dimensions can be calculated from a single projection. But the invention of computed tomography brought a much more robust tool for accurate measurements. Unfortunately, CT has its own disadvantages, some of which we already mentioned in Introduction. Its significant bottleneck is that hundreds or thousands of projections need to be acquired of the measured object, which is very time consuming and slows down quality inspections. In medicine, additionally, taking many projections means that a patient is exposed to X-rays for a longer time, which we obviously want to avoid. Hence, a lot of research is dedicated to *limiting* the amount of views.

In the context of computed tomography, this is called *limited-view* or *sparse-view* CT. Recently, evaluations of state-of-the-art reconstruction algorithms were published. Details about these algorithms and the mathematics behind them are beyond the scope of this thesis, but their *evaluations* are very relevant.

Jones and Huthwaite [2018] evaluate five reconstruction algorithms[1] suitable for limited-view dimensional measurements. Two objects were being measured: a turbine blade and a 70×70 mm additive-manufactured sample with 48 hexagonal holes. Instead of doing full 3D reconstruction, thicknesses were measured between edges detected on 2D reconstructed slices. For the turbine blade, in their best result, they managed to find the edges within ± 0.23 mm using only 24 projections compared to the refence measurement with 5000 projections (0.072° through 360°). For the additive-manufactured sample, one algorithm achieved thickness measurements within ± 0.05 mm from 72 projections, otherwise the measurements were within ± 0.20 mm compared to digital micrometers.

Recently, a different evaluation was published by Butzhammer and Hausotte [2019], who show that deviations in measured dimensions can be reduced by care-

[1] Algebraic Reconstruction Technique (ART), ART with positivity constraints, ART with total variation (TV), Simultaneous Iterative Reconstruction Technique (SIRT), and gradient descent with TV regularization, see Jones and Huthwaite [2018] for explanations and further references.

fully selecting the projection angles. Instead of taking projections from regularly-sampled angles through 360°, they use projections tangential to the object surfaces with linearly or quadratically increasing angle increments. Measurements are performed on full 3D reconstructions that are fitted to an a priori known 3D model shape. We mention their results achieved on an aluminum test part with 25 task-specific projection angles. They managed to lower their mean absolute deviation from roughly 0.058 mm to 0.007 mm and the maximum absolute deviation to 0.023 mm.

Investigating how to choose the best set of projections is a topic of other researchers as well. Fischer et al. [2016] propose an optimization algorithm for determining the best scan trajectory of an a priori known 3D CAD-model. This is reasonable in the context of manufacturing quality control, where the scanned objects only differ slightly (defects, dimensions) from the input model. Together with the 3D model, users are able to select multiple areas of interest where the reconstruction should have high quality. Their work reduces the number of necessary projections, which reduces the necessary scan times.

Recently, Brierley et al. [2018] published a work complementary to the one above. Instead of CT, they target two-dimensional radiography for detecting manufacturing defects. Their algorithm exploits prior knowledge about the inspected model and expected defects and performs a simulate-evaluate-optimize loop to find the best projections for detecting the expected defects. Their work presents a route to reducing the reliance on X-ray CT for quality inspections, which is also what we want to achieve with our method in this thesis.

2.3 Object fitting and pose estimation

The desire to accelerate X-ray CT or perhaps avoid it at all by using ideal projection angles could be seen in the previous section. We now show works that try to tackle the problem in a different way: by fitting parametrized 3D models into real acquired data, which is what we propose in our method (Chapter 3) for dimensional measurements.

Such an idea was recently presented by Dael et al. [2017] but instead of dimensional measurements, they use it to detect defects in horticultural products such as pears. As a first step, they acquire a dataset of pear models reconstructed via X-ray CT from real pears. The dataset is used to create a universal parametrized 3D model of a pear. When they evaluate a new pear, they first use an optical 3D vision system that scans the exterior shape. The parametrized 3D model is then fitted into the real shape and it is used to *simulate* a radiograph of the pear *as if* it did not have any internal defect. This simulated radiograph can then be subtracted from a real radiograph and the difference image will show all possible defects. A classifier is used to decide if the image is a good pear or not.

We show yet another approach, this time based on photogrammetry from 2D radiographs. Noble et al. [1998] measure positioning of holes drilled in a manufactured object. A few radiographs taken from different angles are registered against each other using calibrated spheres placed at precise locations. The projected holes are registered on the 2D projections via geometry-guided matching

against a 3D CAD model, the 3D parameters of the holes are reconstructed from the correspondences between the projections.

Pose estimation Fitting models into real data often consists of *pose estimation*, i.e., finding the position and rotation of the object on the radiograph. As the model is three-dimensional and the radiograph is two-dimensional, this is often referred to as *2D-3D registration* [Miao et al., 2016, Bui et al., 2017]. In a three-dimensional space, the pose has six *degrees of freedom* (DoF) in total (3 for translation, 3 for rotation). In the aforementioned methods, Dael et al. [2017] estimate the pose from the 3D vision system and Noble et al. [1998] put the scanned object inside a fixture with reference spheres at known locations.

However, there are a few recent specialized methods for very fast (even real-time) pose estimation from a priori known models. Miao et al. [2016] propose to pre-compute a library of canonical radiographs covering two geometry-relevant parameters. Then, the pose estimation is done via intensity-based registration, i.e., selecting the pose that maximizes the similarity between a real radiograph and a canonical model. For that, derivative-free optimization algorithms are used, and the pose estimation can be executed in real time, which is useful in medicine for interventional radiology.

Recently, Bui et al. [2017] proposed to estimate pose via machine learning. In their *X-ray PoseNet*, deep learning is used to learn the mapping between radiographs and object poses. Their dataset is acquired by simulating radiographs from a priori known 3D CAD models. For downsampled images, the trained network can predict a pose in a few milliseconds. The predictions were used for X-ray CT reconstruction to avoid calibrating the projections.

2.4 Simulating X-ray projections

Many of the mentioned works relied on X-ray simulations [Fischer et al., 2016, Brierley et al., 2018, Dael et al., 2017, Bui et al., 2017] and so does our proposed method (Chapter 3). Hence, in the last section of related works, we discuss how two-dimensional radiographs can be simulated.

Full Monte Carlo simulations If one wanted to precisely simulate the full radiography pipeline from Figure 1.1, they would need to simulate the X-ray source, mainly the spectrum of outgoing photons and the focal spot, then all possible interactions between the photons and the matter including scattering directions and energy losses, and then for all the photons that happened to arrive at the detector, simulate how the photons get converted into the electric charge in the detectors, including the scintillators for indirect detectors.

Such simulation frameworks do exist, for example *McXtrace*[2] [Knudsen et al., 2013, Busi et al., 2018] or *SINDBAD* [Tabary et al., 2007]. They rely on stochastic Monte Carlo simulations, where a very large number of numerical photons is generated and their paths are traced, everything handled stochastically with given probabilities. There are many implementation details that make these simulations possible, e.g., in McXtrace, one so-called numerical photon actually represents a

[2]http://www.mcxtrace.org/

bunch of real physical photons, whose amount is traced in a weight factor, and much more details explained by Knudsen et al. [2013]. As another example, in SINDBAD, to speed up the simulation, they propose a hybrid approach where the Monte Carlo simulation is *combined* with an analytical simulation that only takes attenuation into account without the photons being scattered in new directions.

These complex simulations are important for physical experiments such as crystallography or pulsed X-ray scattering experiments [Knudsen et al., 2013] and advanced imaging methods such as *spectral CT* that provide measurements at isolated high-energy ranges [Busi et al., 2018]. A simulation is also available for the imaging chain of computed radiography (Section 1.4) by Yao et al. [2016].

Attenuation-only ray tracing The aforementioned complex simulations can be avoided for the price of lower accuracy. For simplicity, we can only take the attenuation law (Equation 1.2) into account, i.e., assume that if a photon flying from an X-ray source interacts with the atoms in the evaluated object, that photon disappears and never reaches the detector. In reality, this may not be true, because that photon may be scattered in a new direction and may fly towards a different pixel of the detector, perhaps with a different photon energy if Compton scattering occurred. However, in *attenuation-only* simulations, we assume that this is negligible (we offer further discussion in Section 5.5).

Notice that in order to evaluate Equation 1.2, we need to know x, i.e., the distance that the X-ray beam passes through the object material. The distance can be acquired by *ray tracing* (Figure 2.4): a ray is generated from the X-ray source towards each pixel individually and on this ray, intersection points with the geometry are found, and the distances inside the geometry are accumulated.

One example of such simulation software is *aRTist*[3] [Bellon et al., 2012], which was used by Fischer et al. [2016] and Brierley et al. [2018] from the previous sections. By default, it only simulates attenuations, but there is an optional integration of a Monte Carlo program.

Recently, Marinovszki et al. [2018] presented a GPU-accelerated simulation using the Nvidia OptiX general-purpose ray-tracing framework [Parker et al., 2010]. They used the simulation to estimate energy spectrum of an X-ray source by comparing simulated radiographs to real ones. First, they registered a 3D CAD model on real 2D radiographs using the already mentioned 2D-3D registration like Miao et al. [2016]. Then, the energy spectrum was estimated via optimization methods by minimizing the difference between a simulated and a real radiograph.

Attenuation-only rasterization Instead of ray tracing, Vidal et al. [2009] propose to generate radiographs via GPU rasterization of triangulated 3D meshes. The method is based on the observation that the distances x through the objects can be accumulated in any unspecified order. That is, the accumulated length x for a certain pixel is (follow Figure 2.4):

$$x = \sum_i -\operatorname{sgn}\left(\boldsymbol{\omega} \cdot \mathbf{n}_i\right) d_i, \tag{2.1}$$

where the sum is over all intersections hit by that pixel, $\mathbf{n}_i$ is the normal vector at an intersection, $\boldsymbol{\omega}$ is the direction from the X-ray source to that intersection,

[3] http://www.artist.bam.de/

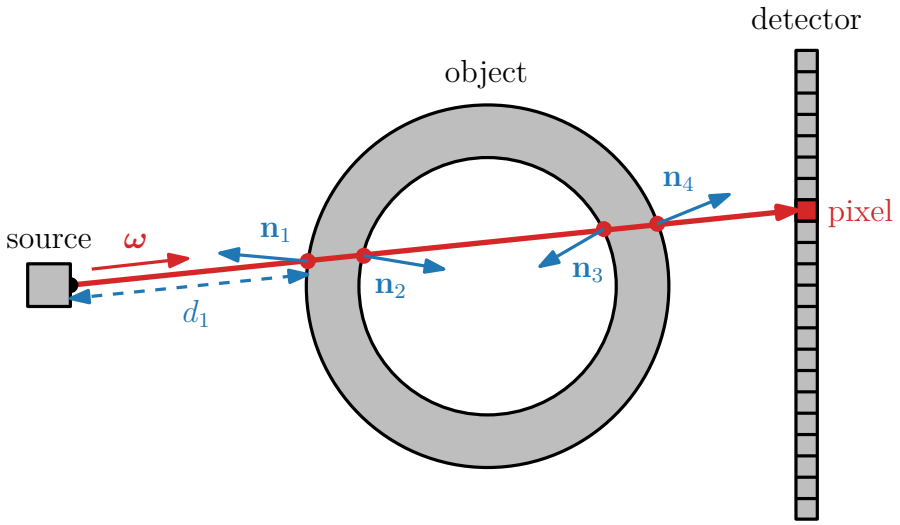


Figure 2.4: Ray tracing can be used to find intersections between the X-ray source and individual detector pixels. Our goal is to find the total distance that the ray travels through the hollow object, in this example, between intersections 1 and 2, and between 3 and 4.

and d_i is the distance between the source and that intersection.

A GPU framebuffer is allocated to hold the accumulated lengths and is called the *length buffer*. Its values are accumulated in a GPU fragment shader in an unspecified order via Equation 2.1. Unfortunately, due to the nature of GPU rasterization, some intersections may be duplicated when a pixel is at a triangle edge or vertex, which causes black or white artefacts as the numbers of enters and exits from an object do not match. Another problem occurs when the dot product $\boldsymbol{\omega} \cdot \mathbf{n}_i$ equals zero as such an intersection cannot be reliably counted. Because these situations can be detected, e.g., by comparing the number of enters and exits, Vidal et al. [2009] propose to replace these pixels by the average value of valid neighboring pixels.

Later, Vidal et al. [2010] improved the method by simulating focal size geometric blur and polychromatic sources via additional rendering passes in a loop. Essentially, they model a focal spot by multiple point sources and polychromatic radiation is modeled by splitting the spectrum into discrete energy channels.

Even later, Vidal and Villard [2016] used their method for real-time simulation of medical imaging with an animated object representing respiratory motion. In this publication, they compared their real-time GPU rasterizations to Monte Carlo simulations that took more than 12 days of computation. They show that the normalized cross-correlation between the results were higher than 0.996, which demonstrates the validity of their approach.

3. Our approach

In the previous chapter, we saw many related radiography methods, some of which were directly used for dimensional measurements and some for different purposes such as defect detection in horticultural products.

In this chapter, we present our novel approach for dimensional measurements from only a few radiographs. We build on and further extend some of the great ideas that we saw in the previous chapter. In some ways, our concept is similar to these works that tried fitting parametrized models to real acquired data (Section 2.3). We saw that these methods successfully used only a few projections, which is exactly what we want. But none of the works we saw so far pursued and extended the general idea so far that it could be used for robust dimensional measurements. So here, in our novel approach, we propose and do exactly that.

3.1 Problem formulation

Let us begin with formulating what exactly we aim to solve in our approach.

Problem statement Assume we have a real *object* made of a single solid homogeneous *material* with otherwise unknown exact material properties. The object has a certain more or less complicated *shape*. We know how the general shape looks like, which is expected in the context of manufacturing, but we do not know its exact *dimensional properties*, i.e., sizes of the shape and its features, some of which may be difficult-to-reach or internal.

We are able to take a limited set of two-dimensional *radiographs* of this object from certain positions and angles. For that, we use a typical industrial X-ray setup as in Figure 1.1, which enables contactless non-destructive evaluation. Some general properties of the setup are expected to be known, e.g., resolution of the detector in pixels and millimeters, but we are not required to know some complicated properties such as the exact spectrum of the source.

Our ultimate goal is to use this limited set of radiographs and all the mentioned prior knowledge to *estimate the dimensional properties* of the real object. We propose to achieve this via optimization, i.e., finding the values of the dimensional properties that best fit the real radiographs based on our simulations. We carefully designed the whole process in such a way that the optimization can automatically converge to accurate dimensions that the object really has in a reasonable timeframe within a few minutes.

Rationale Our approach is designed in the context of manufacturing a large number of objects with the same general shape but uncertain dimensional properties. We need to measure these dimensions for individual quality evaluation. Some of our input parameters, such as the parametrization of the shape, are only prepared *once* for a given shape and X-ray setup. Then, for every single object of the same material and general shape, we *only* repeat the process of taking new radiographs and manually setting a small number of initial parameters. The final dimensional estimation of each one specific object runs fully *automated*.

Formal look Let us have a more formal look at the problem and our approach. Let the word *hyperparameters* denote all the prior knowledge that we have about the object shape, the X-ray setup, the material, and other parameters that are fixed during our algorithm. Then, let $\boldsymbol{\theta} \in \mathbb{R}^n$ denote a vector encoding all the actual *parameters* with unknown values that need to be estimated, such as the object dimensions. And let $\mathsf{R}\colon \mathbb{N}^2 \to \mathbb{R}$ denote the real two-dimensional *radiograph* of the measured object. Formally, the radiograph R is a function that for two-dimensional pixel coordinates gives us the intensity of that pixel.

Our goal is to estimate the parameters $\boldsymbol{\theta}_{\text{estimate}}$ that best correspond to the reference radiograph R. We propose to do so by minimizing the difference between R and a *simulated* radiograph. Let $f : \mathbb{R}^n \to (\mathbb{N}^2 \to \mathbb{R})$ denote a simulation function that for given parameters $\boldsymbol{\theta}$ returns the simulated radiograph $f(\boldsymbol{\theta})\colon \mathbb{N}^2 \to \mathbb{R}$. Estimating the parameters can now be formally written as (Figure 3.1):

$$\boldsymbol{\theta}_{\text{estimate}} = \arg\min_{\boldsymbol{\theta}} \|f(\boldsymbol{\theta}) - \mathsf{R}\|\,, \tag{3.1}$$

where technically, $\|f(\boldsymbol{\theta}) - \mathsf{R}\|$ can be any function that computes the difference between the two radiographs. For simplicity, we can now think of it as the sum of absolute values of per-pixel differences:

$$\|f(\boldsymbol{\theta}) - \mathsf{R}\| = \sum_{(i,j)\in\mathbb{N}^2} |f(\boldsymbol{\theta})(i,j) - \mathsf{R}(i,j)|\,. \tag{3.2}$$

Solving Equation 3.1 Generally, for equations in the form $\mathbf{x} = \arg\min_{\mathbf{x}} f(\mathbf{x})$, where f is a nonlinear scalar *objective function* and $\mathbf{x}$ is a vector, there is a panorama of methods called *(nonlinear) optimization algorithms* [Nash, 2014]. They iteratively evaluate $f(\mathbf{x})$ with different $\mathbf{x}$ chosen in a clever way (specific to the algorithm), trying to find where $f(\mathbf{x})$ is either a *local minimum* (which is easier to find), or even the *global minimum* (which is often approached stochastically).

But the idea from Equation 3.1 and Figure 3.1 is very high-level. We still need to design the whole method (Section 3.2 and onward), especially how $\boldsymbol{\theta}$ should be practically constructed, how $f(\boldsymbol{\theta})$ can be calibrated and evaluated as fast and as accurately as possible, and how the references R are acquired so that we can use the method in real-world situations to get meaningful and accurate results.

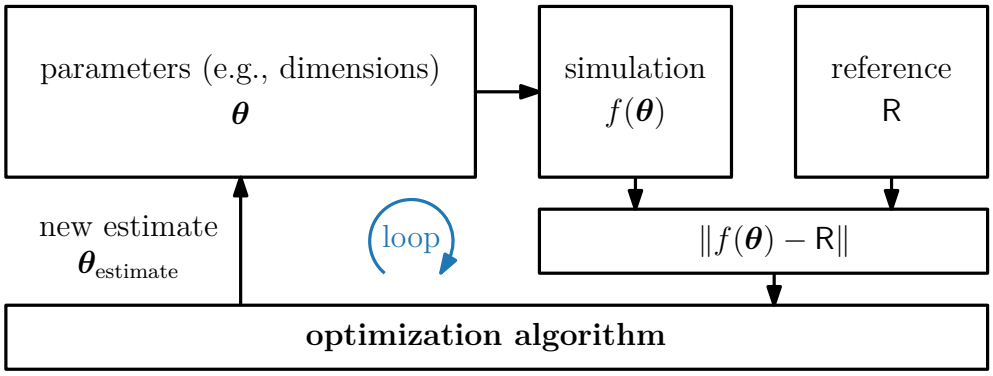


Figure 3.1: Illustration of Equation 3.1. One way of finding its solution is to use an iterative optimization algorithm, which tries to iteratively evaluate the target function value and estimate new parameters that lower the value.

3.2 Method outline

In Section 3.1, we formulated the problem and how we can look at it in a more formal way. This section presents an outline of our method that we carefully designed to solve the problem at hand. The outline gives an overview of the individual components that our method consists of. One by one, they are explained in more details in Sections 3.3 to 3.8.

Iterative workflow We are solving an optimization problem, so we designed our method in an iterative way as depicted in our dataflow diagram in Figure 3.2. There are multiple individual components, each has its own inputs connected to outputs of some other components. Notice that there is no loop, except the one that begins a new iteration, so the components can be topologically ordered, which simplifies the implementation (Chapter 4).

Each iteration i begins with a specific 3D model and its dimensional parameters, its material, and its poses on $N \geq 1$ reference radiographs. Some of these values are hyperparameters, some are parameters $\boldsymbol{\theta}_i$, as we will see later. Using them all as input, radiographs $f_1(\boldsymbol{\theta}_i), \ldots, f_N(\boldsymbol{\theta}_i)$ with different poses are simulated. Then, these are compared to the real reference radiographs $R_1, \ldots, R_N$ and we get a single scalar value representing the total difference among all references. This value is the objective that the optimization algorithm is trying to minimize. At the end of the iteration, if we are not yet satisfied with the result, the optimization algorithm estimates a new set of parameters $\boldsymbol{\theta}_{i+1}$ that are feedback-looped back to the beginning. A new iteration can start.

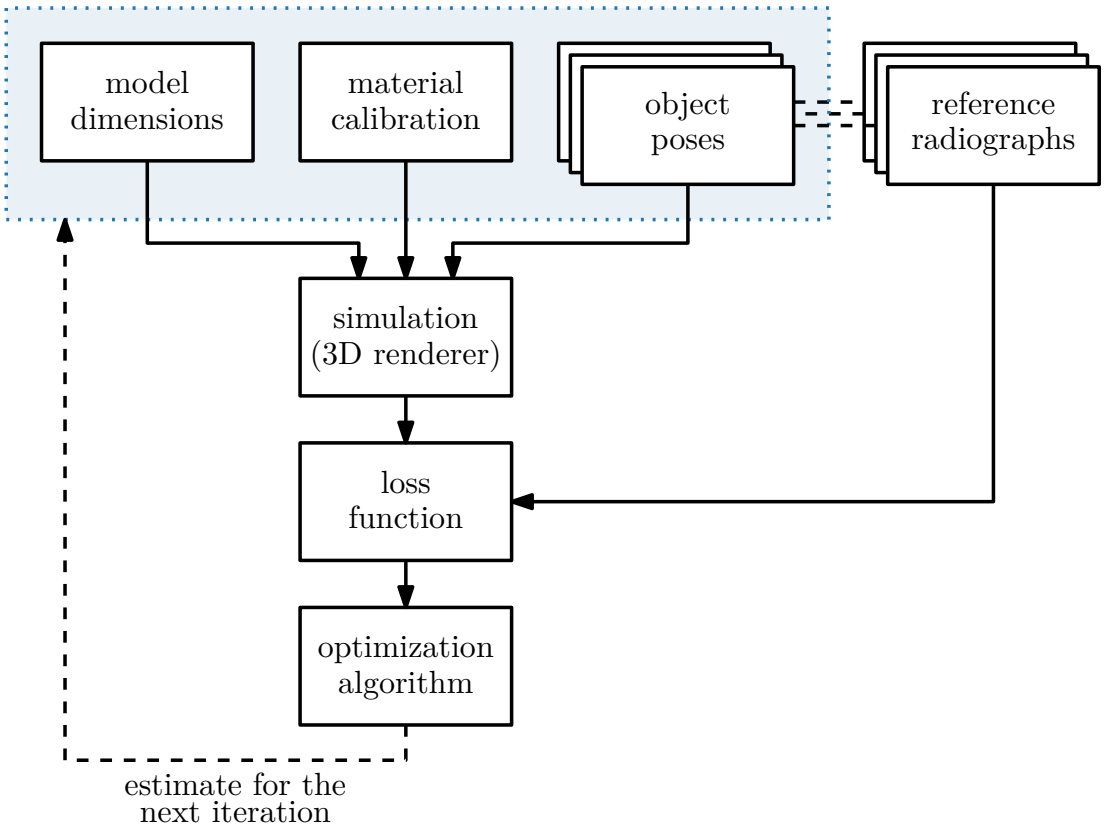


Figure 3.2: Diagram of our method and its iterative workflow (see text).

Components The individual components are the following:

- **Model and its parametrization** (Section 3.3)
  The object that we want to measure is first modeled in 3D with inaccurate dimensions. By triangulating the shape, we get a set of vertices and triangles connecting them. We need to carefully choose dimensions to be measured and define the relations between them and vertex coordinates.

- **Material and its calibration** (Section 3.4)
  The measured object is made of a single material with certain attenuation properties (Section 1.2). The properties depend on photon energies generated by the X-ray source. To avoid complicated measurements of the source, we propose a way of calibrating the material together with the source spectrum and detector response. We propose two ways of calibration: from a wedge reference object or from a precisely measured arbitrary object.

- **Reference radiographs and object poses** (Section 3.5)
  We have already seen in Figure 2.1 that infinitely many objects with different poses can create the same projection. It is perfectly possible that different $\boldsymbol{\theta}_1 \neq \boldsymbol{\theta}_2$ give the same radiograph $f(\boldsymbol{\theta}_1) = f(\boldsymbol{\theta}_2)$. Hence, we give special care to the selection of our reference images so that the optimization does not converge to a wrong minimum. We propose to take two radiographs with precisely known movement magnitude in an arbitrary axis together with another radiograph with a different rotation. Calibration is not necessary, only a precise movement magnitude needs to be measured.

- **X-ray simulation** (Section 3.6)
  The goal of the simulation function f is to generate radiographs as close to the references as possible. To avoid slow Monte Carlo simulations that require precisely measured X-ray sources and detector responses, we instead rely on a GPU-accelerated ray tracer that we designed to use our calibrated material data (see above). This allows fast evaluations of f and relatively accurate radiographs.

- **Loss function** (Section 3.7)
  The function computing the differences between images should be fast to evaluate and needs to be sensitive enough to report even the slightest differences between pixels. We settled with the simple sum of absolute values of per-pixel differences as in Equation 3.2. Note that because of the random noise present in the reference radiographs (Section 1.5), the difference will never be zero even if we had a 100 % accurate simulation.

- **Optimization algorithm** (Section 3.8)
  The last component is the optimization algorithm itself. We propose a hierarchical optimization process that first roughly estimates object poses within downscaled radiographs, then refines the pose estimations via a local optimization, and finally performs a full optimization of all parameters. For simplicity, we use gradient-free optimization methods as we do not know the gradients of our X-ray simulation f with regards to the parameters $\boldsymbol{\theta}$. Gradient-based optimization is proposed as future work.

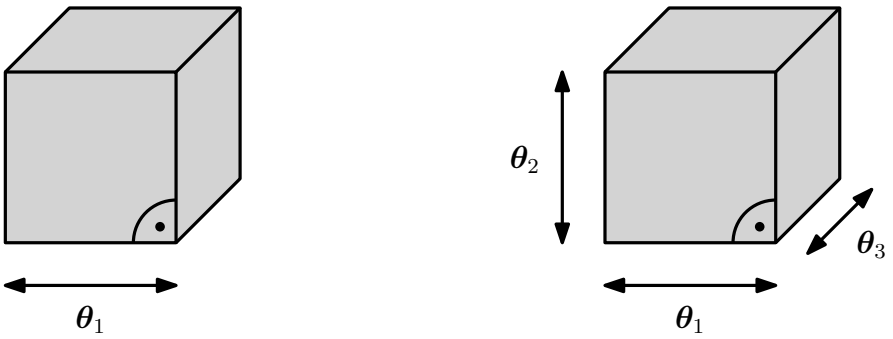
3.3 Object representation and parametrization

Our method is intended for measuring objects with an a priori known shape with unknown dimensional properties. Let us show how we represent these shapes and how we propose to parametrize their dimensions.

Simple example Suppose we are manufacturing *cubes* (Figure 3.3) and we want to measure if their sizes are within tolerances.

Our prior knowledge would be the general shape of a cube. A cube has six square faces, the neighboring always perpendicular to each other. The only dimensional parameter of a cube is the size of any of its sides, because the remaining sides have the same size by definition (Figure 3.3a).

But in a more realistic scenario, perhaps our manufacturing process cannot be trusted to make the sides perfect squares. Hence, we would add other dimensional parameters such as sizes of different sides (Figure 3.3b). Then, in our quality control process, we could be measuring if our cubes are still within our tolerances of "being a cube" or they are just arbitrary cuboids that cannot be used.



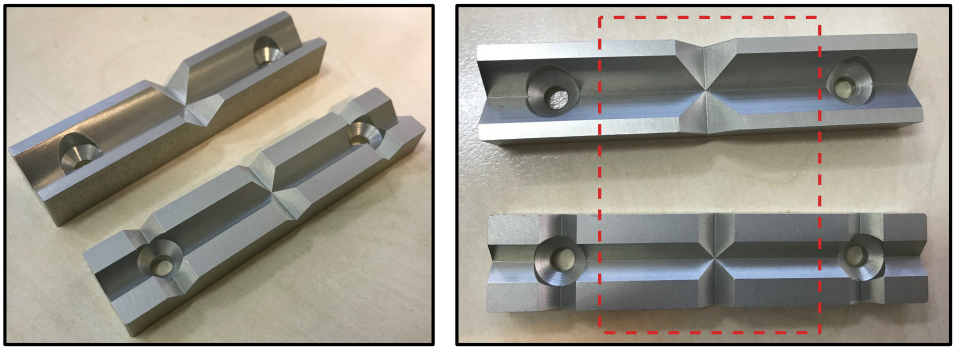
(a) In ideal conditions, if we trust that the measured object is really a cube, we can only parametrize one side and let the remaining have the same size.

(b) In a real scenario, we would probably parametrize at least three sides as we could not trust the manufacturing process to create a perfect cube with square sides.

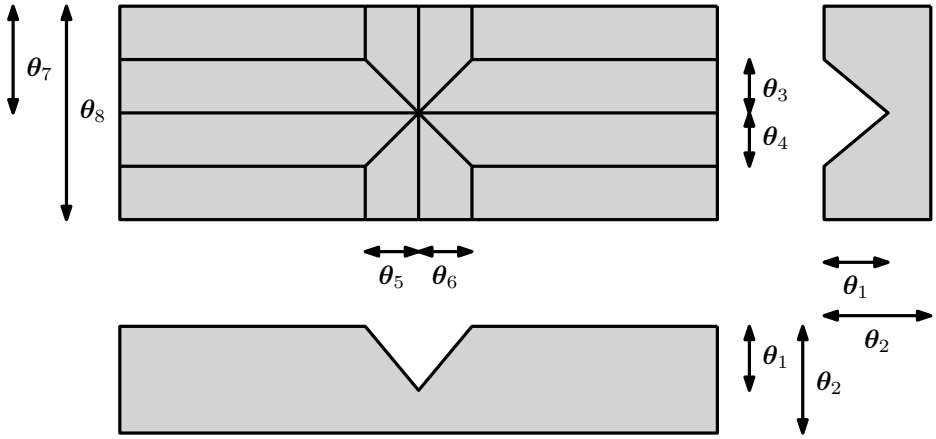
Figure 3.3: Example parametrization of a simple cube.

Preparing a new model Now we show the individual steps that we take to model and parametrize an object. The steps are general, but we illustrate them with a specific real-world example in Figure 3.4. Note that some of the steps (marked with an asterisk *) may be redundant if we already have technical drawings and a 3D CAD model available for the manufacturing process.

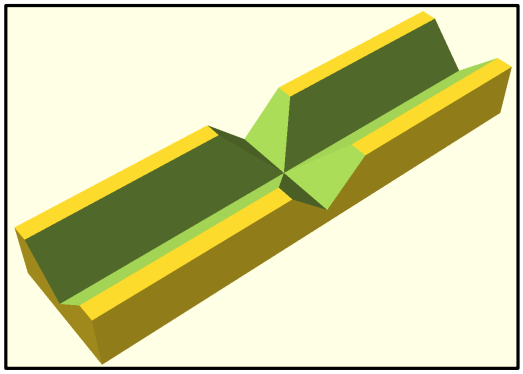
1. **Shape and dimensions*** The first step is to know which shape we are going to be measuring and how the individual objects of the same shape can differ from each other (Figure 3.4a). We choose if we want to measure the whole shape or just a cropped part that is interesting for us. Then, we decide which dimensions define the shape, for which it is useful to prepare a technical drawing (Figure 3.4b). Note that the drawing must be as complex as our intended measurements, so perhaps more complex than a typical drawing, because the dimensions are a part of our $\boldsymbol{\theta}$.



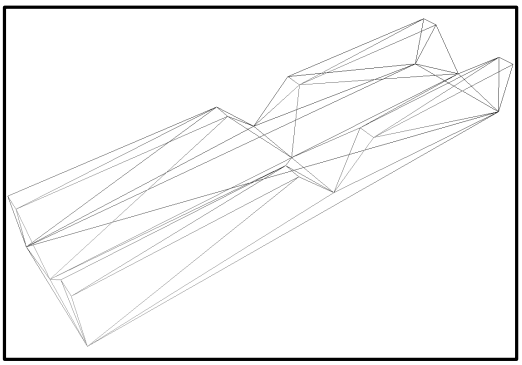
(a) Photos of prism jaws of different sizes. We are interested in dimensional measurements inside the red highlighted area without the circular holes.



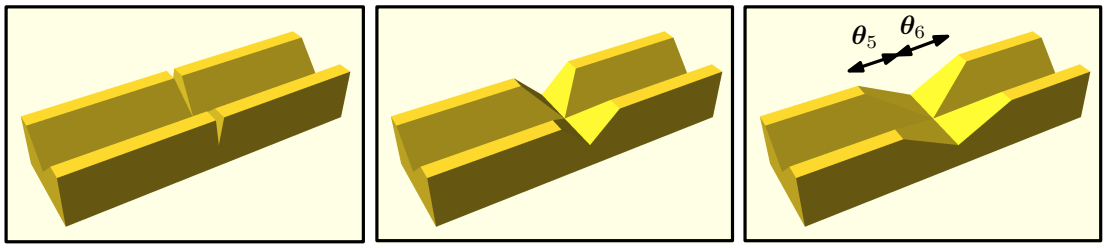
(b) Dimensional parametrization $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_8$ of a generic prism jaw from (a).



(c) Virtual 3D model of the prism jaw.



(d) Triangulated surfaces from (c).



small $\boldsymbol{\theta}_5, \boldsymbol{\theta}_6$ medium $\boldsymbol{\theta}_5, \boldsymbol{\theta}_6$ large $\boldsymbol{\theta}_5, \boldsymbol{\theta}_6$

(e) Example of how the dimensional parameters influence the 3D model.

Figure 3.4: Parametrizing and modeling a real object: prism jaws.

2. **3D modeling and triangulation*** Based on the technical drawings, we create a 3D model in a 3D editor (Figure 3.4c). We used OpenSCAD[1], an open-source project for creating solid 3D CAD models. For simplicity, to avoid parsing complicated file formats, we decided to *triangulate* the models (Figure 3.4d), e.g., to `.stl` files. Triangulation converts the shape surface into individual triangles represented by *vertices*, their 3D coordinates, and their connections. In theory, any parametric surface representation could be used if supported by the X-ray simulation (Section 3.6).

3. **Dimensional parameters to vertex coordinates**
Since we have triangulated surfaces, we need to decide how to reflect the dimensional parameters from $\boldsymbol{\theta}$ in the triangulation (Figures 3.4e and 3.5). Note that the triangulated topology does not change by changing $\boldsymbol{\theta}$, only the vertex coordinates change. The naïve approach would be to put all these coordinates directly to $\boldsymbol{\theta}$, but that would significantly increase the amount of optimized parameters, three times per each vertex. Hence, we propose to keep only the few $\boldsymbol{\theta}$ from the technical drawings and instead write all 3D vertex coordinates as a function of $\boldsymbol{\theta}$ as in Figure 3.5.

Algorithm steps in each iteration Whenever the optimization algorithm proposes a new estimate $\boldsymbol{\theta}$, we only recalculate the vertex coordinates of our triangulated model (see above and Figure 3.5). The new coordinates are then used by the 3D renderer as shown in the diagram (Figure 3.2). As our initial estimates in the first iteration, we can use the expected dimensions of a correctly manufactured object.

[1]https://www.openscad.org/

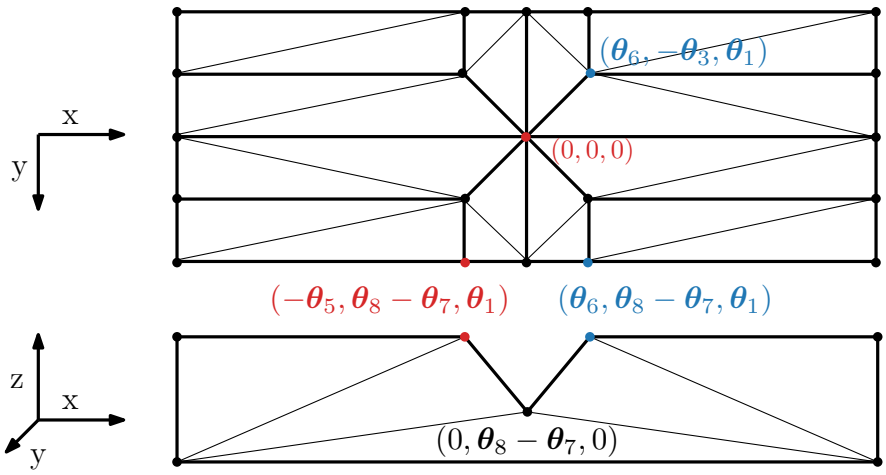


Figure 3.5: Triangulated sides of the prism jaw. We express the triangle vertices 3D coordinates using the dimensional parameters. In the figure, examples are shown of how the coordinates may be expressed in the given coordinate system.

3.4 Material

Even though this section is called "Material", as we will soon see, the situation is more complex as we actually calibrate the whole source-material-detector chain.

Polychromatic intensities In Section 1.2, we saw that the interactions with the matter depend on photon energies. As a typical X-ray source is polychromatic (Section 1.3), we want to rewrite the Beer-Lambert law (Equation 1.2) for polychromatic spectrum in the energy interval $(\varepsilon_{\min}, \varepsilon_{\max})$. Let $s(\varepsilon) \geq 0$ denote a normalized source spectrum, i.e., the distribution of photon energies where $\int_{\varepsilon_{\min}}^{\varepsilon_{\max}} s(\varepsilon)\,\mathrm{d}\varepsilon = 1$. And let $0 \leq d(\varepsilon) \leq 1$ denote a detector sensitivity, i.e., how efficient the detector is at detecting such photon energies, including a filter in front of the detector to filter out low energies. We can then write [Marinovszki et al., 2018]:

$$I(x) = I_0 \int_{\varepsilon_{\min}}^{\varepsilon_{\max}} d(\varepsilon)\, s(\varepsilon)\, \exp\left(-\mu(\varepsilon)x\right)\, \mathrm{d}\varepsilon, \tag{3.3}$$

where $\mu(\varepsilon)$ is the material linear attenuation coefficient at a certain photon energy (Section 1.2), x is the distance travelled through the matter, and I_0 is the source intensity as in Equation 1.2.

For convenience, in Section 3.5, we propose to *normalize* all reference radiographs to ensure that it holds $I(0) = 1$ for all pixels (with the exception of random noise), and further that it holds $0 \leq I(x) \leq 1$ for all $x \geq 0$. This allows us to rewrite Equation 3.3 this way:

$$I(x) = \int_{\varepsilon_{\min}}^{\varepsilon_{\max}} a(\varepsilon)\, \exp\left(-\mu(\varepsilon)x\right)\, \mathrm{d}\varepsilon, \tag{3.4}$$

where $a(\varepsilon)$ is a normalized source-detector spectrum:

$$\int_{\varepsilon_{\min}}^{\varepsilon_{\max}} a(\varepsilon) = 1, \quad \forall \varepsilon \in (\varepsilon_{\min}, \varepsilon_{\max}) : a(\varepsilon) \geq 0. \tag{3.5}$$

Discretization What we propose next is to discretize Equation 3.4 to a sum of $M \geq 1$ monochromatic fractions approximating the original polychromatic integral. This is a very old idea used for exponential-sum fitting of radiative transmission functions discussed for example by Wiscombe and Evans [1977]:

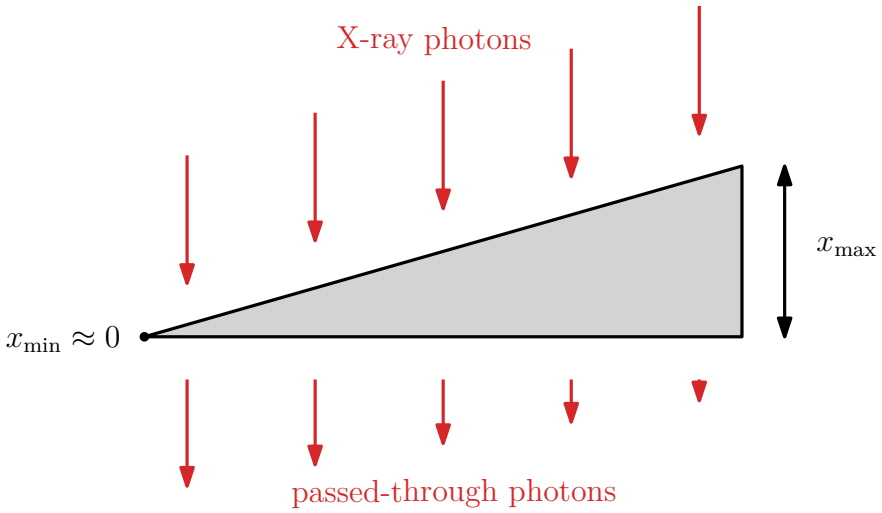
$$I(x) \approx \sum_{k=1}^{M} a_k \exp\left(-\mu_k x\right), \tag{3.6}$$

where $a_k > 0$ and $\mu_k \geq 0$ represent the behavior of the source, detector, and material in the discretized spectrum. In our case, additionally, we require $\sum_{k=1}^{M} a_k = 1$ because of the normalization ensuring $I(0) = 1$.

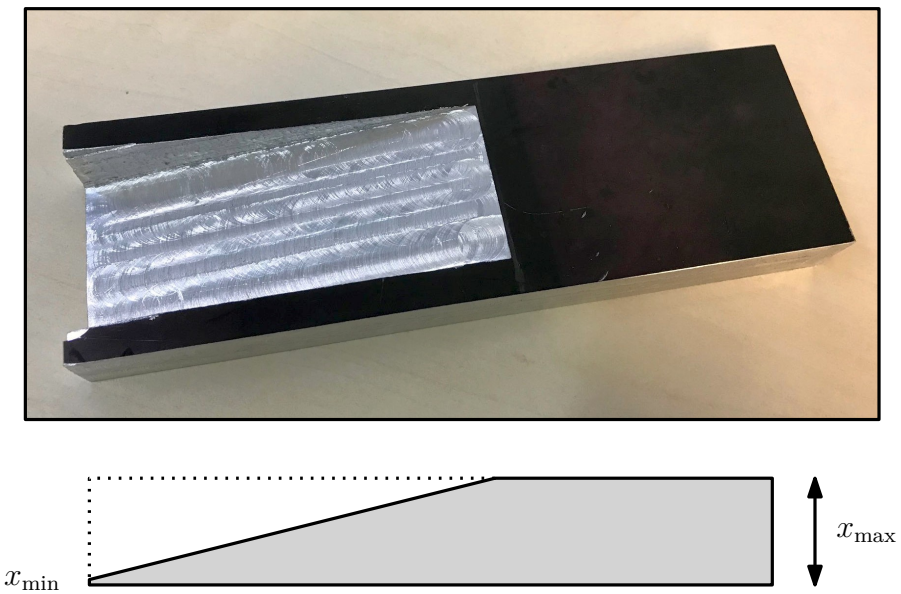
Note Our approximation is only based on the Beer-Lambert law (Equation 1.2). That is, it does not take into account any scattering inside the detector itself (Section 1.5, Spatial resolution), nor any X-ray photons that are mid-flight scattered towards a different pixel. This would require a full Monte Carlo simulation (Section 2.4), which we are not doing for performance reasons (Section 3.6).

Calibration What remains to be explained is how we can calibrate the parameters. We propose to calibrate the a_k and μ_k together in a single process. We use reference radiographs of an object made of the material that we are calibrating. We propose two different ways, each having its own advantages.

a) Calibration from a wedge The first option is calibrating against a special reference object. Ideally, we would use an object with a continuously increasing thickness, which would cause increasing attenuations of X-ray photons. Because of the Beer-Lambert law (Equation 1.2), the X-ray attenuations through such an object would increase exponentially with the thickness. The simplest object with such a property is a *wedge* as in Figure 3.6a and it was used for example by Scott and Krastev [2016] for learning material identifications.



(a) Wedge. Notice that the intensities of the passed-through X-ray photons decrease with the increasing wedge thickness, which is why this is a perfect shape for calibrations.



(b) Photo of our manufactured wedge-shaped object. Instead of manufacturing a clean wedge like in (a), we instead cut out a wedge hole in an aluminum block, which was easier to manufacture. The thickness of the whole block is $x_{\text{max}} = 20\,\text{mm}$ precisely, the minimum thickness on the left side is roughly $x_{\text{min}} \approx 0.2\,\text{mm}$ measured with calipers.

Figure 3.6: Wedge objects can be used for calibrations.

We propose to perform the calibration via the following steps.

1. **Manufacturing a wedge** First, we manufacture a *wedge* from the calibrated material (Figure 3.6b). A wedge is an ideal shape as it represents thicknesses smoothly ranging from $x_{\text{min}} \approx 0$ to a given x_{max} (Figure 3.6). Obviously, the smoothness, continuity, and how close x_{min} is to zero depend on the manufacturing quality. But the huge advantage is that we only need an a priori knowledge of x_{max}. All other dimensions can be arbitrary.

2. **Radiograph** We take a single reference radiograph of the wedge as in Figure 3.7a. We normalize the radiograph (Section 3.5) to ensure $I(0) = 1$. We manually select a region that contains the wedge with a vertical overlap.

3. **Intensity table** Now we sample the intensities in the radiograph in the vertical direction. To reduce the amount of random noise, we average the pixel intensities in each individual row. This gives us a table (array) containing average intensities for varying Y-coordinates in the 2D radiograph. If we plot these intensities (Figure 3.7b), we can see that they form an exponential shape, which corresponds to the attenuation law.

4. **Finding the beginning and end** The next step is to find which Y-coordinates correspond to the minimum and maximum thicknesses x_{min} and x_{max} (Figure 3.7cd). We perform this step manually by examining the intensity values and their first derivative.

5. **Exponential fitting** Between the first and last Y-coordinate found in the previous step, the intensities correspond to $I(x)$ for $x \in \langle x_{\text{min}}, x_{\text{max}} \rangle$. We need to fit the a_k and μ_k from Equation 3.6 into this discrete dataset (Figure 3.7e). Various fitting approaches are discussed by Wiscombe and Evans [1977], we tried fitting using `NonlinearModelFit` in Wolfram Mathematica[2] with satisfying results. Refer to Section 5.5 for evaluation.

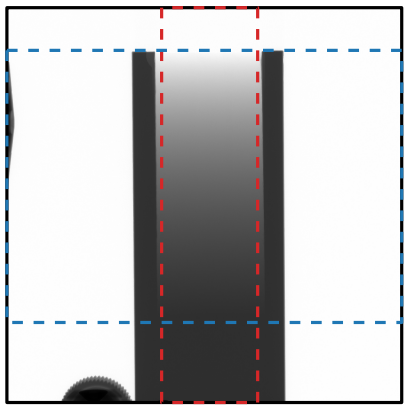
6. **Shift and scale** As a last step, we propose a way to fix two problems that our fitted $I(x)$ might have. First, we want to ensure that $I(0) = 1$. Second, as it is impossible to manufacture a wedge with an infinitely sharp beginning, it is highly likely that $x_{\text{min}} \neq 0$ even though we pretend it is. This causes discontinuities at the beginning of the wedge for x close to 0.

   We propose to find for which x_0 it holds $I(x_0) = 1$ in the fitted model, e.g., by using `FindInstance` in Wolfram Mathematica. It is highly likely that $x_0 < 0$. We assume that this fitted x_0 is closer to how the material really behaves for small x, but we want to retain how $I(x)$ looks like for higher x. Hence, we shift and scale the original fitted $I(x)$ and get $I^*(x)$:

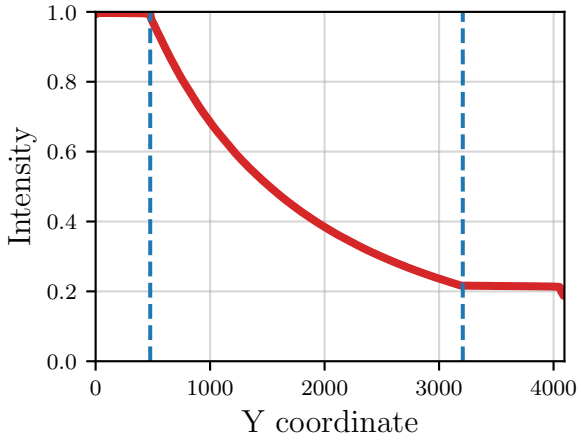
$$I^*(x) = I\left(x \cdot \frac{x_{\text{max}} - x_0}{x_{\text{max}}} + x_0\right), \tag{3.7}$$

   which ensures $I^*(0) = 1$ and $I^*(x_{\text{max}}) = I(x_{\text{max}})$.

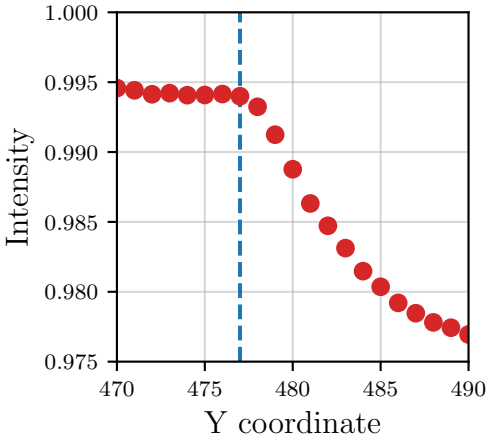
[2] http://www.wolfram.com/mathematica/



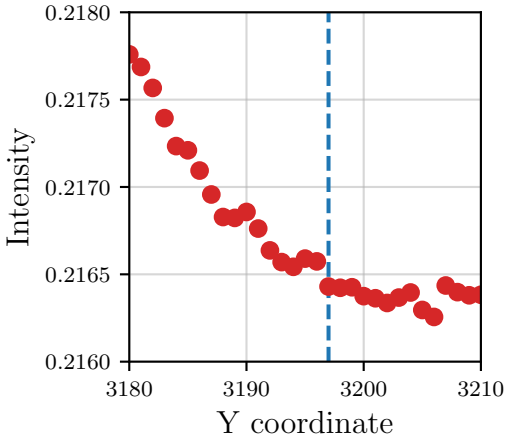
(a) Normalized radiograph of our wedge from Figure 3.6b.



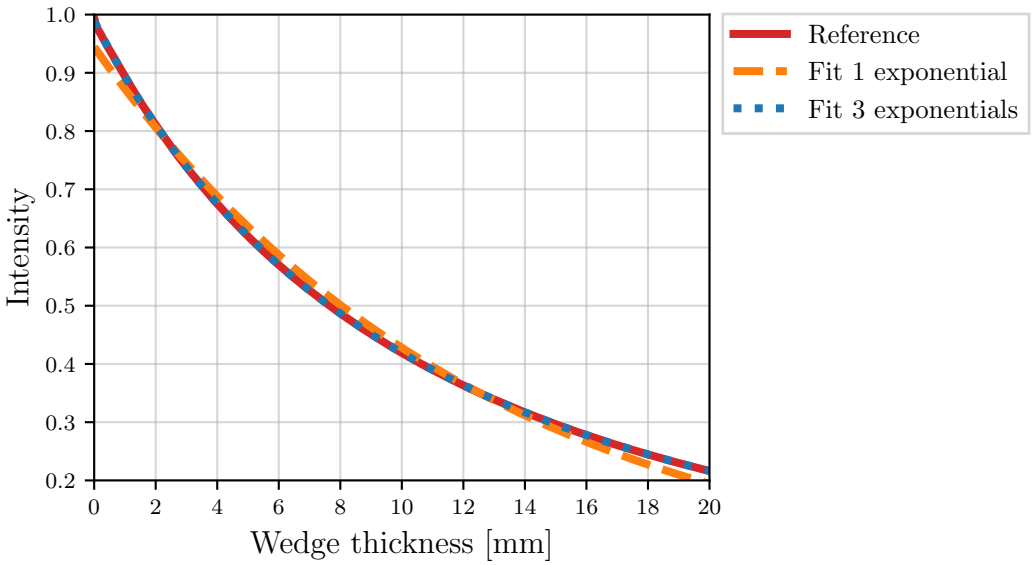
(b) Intensities averaged along the red area in (a). Blue lines show approximate wedge boundaries.



(c) Zoomed-in part of (b) showing the approximate beginning of the wedge.



(d) Zoomed-in part of (b) showing the approximate end of the wedge.



(e) Fitting the intensities w.r.t. wedge thickness with exponentials.

Figure 3.7: Analyzing wedge radiograph for material exponential fitting.

b) Calibration via optimization The second possibility of calibrating the parameters does not require manufacturing any reference wedge, instead, we can use any arbitrary physical object. We use the same principle as our whole method itself: parameter optimization, i.e., optimizing the parameters a_k, μ_k that best match the radiographs of the physical object. This is a kind of *bootstrapping* as we can use the same algorithm to learn the parameters for itself.

We propose the following steps.

1. **3D modeling** First, we need to measure the reference physical object with a very high accuracy, e.g., using X-ray computed tomography as we already discussed Section 2.1. Then, we need to prepare a 3D model of this object with the exact dimensions, for which we may use our model and parametrization from Section 3.3.

2. **Radiographs** We take reference radiographs of this object, preferably from different angles so that a wide range of thicknesses can be seen. Ideally, we would cover all thicknesses uniformly, but that is difficult to achieve with an arbitrary object that is not a wedge. Once we have the radiographs, we normalize them (Section 3.5) to ensure $I(0) = 1$.

3. **Initial estimate** We construct an initial estimate of $I(x)$. We already know that $I(0) = 1$ but we need at least one other reference point $I(x_1) = I_1$. We propose to select a region on one of the radiographs that has a uniform intensity with a known thickness x_1. The average intensity in that region is the value I_1. Then, we set the initial parameters to:

$$a_1 = 1, \quad \mu_1 = -\frac{\ln(I_1)}{x_1}, \tag{3.8}$$

   and all remaining $a_2 = \cdots = a_M = \mu_2 = \cdots = \mu_M = 0$, which ensures that $I(0) = 1$ and $I(x_1) = I_1$, which is exactly what we wanted.

4. **Hierarchical optimization** We use these initial parameters to estimate the pose of the object on the reference radiographs. We use the same hierarchical optimization that we describe later in Section 3.8 for the dimensional measurements. But notice that the dimensions of the 3D object are already known, so in the last hierarchical step, we only optimize the pose and the coefficients a_k, μ_k as parameters $\boldsymbol{\theta}$.

Comparison We proposed two ways of calibrating the material and source-detector parameters. During the evaluation of our method (Chapter 5), we successfully used both of these approaches. Let us now briefly discuss the advantages and disadvantages of the two different solutions.

The major advantage of the first method is that it does not require a precisely measured and parametrized object. Also, as the wedge is uniformly covering the whole thickness range, the parameters are fitted for all thicknesses with the same "priority". In the second method, we benefit from not having to manufacture a reference wedge, but we need a precisely measured reference object and we have no control over the optimization process for thicknesses that are not dominant in our reference radiographs as these do not contribute to the loss function.

3.5 Reference radiographs and poses

The reference radiographs are a very important part of our method. As we have already seen in Section 2.2, selecting the best projections can significantly influence how well we can measure an object. And it not true just for the computed tomography, but also for our approach, where the reference radiographs are essentially the *only* source of truth for our optimization process.

The radiographs represent the measured object, its dimensions, and its pose in the 3D space. Unfortunately, as we have seen in Figure 2.1, the dimensions and pose cannot be both determined from a single projection as there is infinitely many possibilities. Hence, we generally need some additional prior knowledge about the projection or the object dimensions. For example, if we have a reference measurement in an image, we can analyze the vanishing points and use cross-ratios to perform dimensional measurements even from a single projection [Criminisi et al., 2000]. But what if all object dimensions are unknown in the first place? What do we use as a reference?

Radiographs with known distance We propose to solve the problem by using at least two radiographs R_1 and R_2 taken in the following way (Figure 3.8).

In R_1, the object has an arbitrary pose $\boldsymbol{\theta}_{\text{pose1}} \in \mathbb{R}^6$, which has six degrees of freedom. In R_2, we physically move the object from $\boldsymbol{\theta}_{\text{pose1}}$ towards a new position using a precise manipulator (Chapter 1). Our manipulator shows the distance $m \in \mathbb{R}$ that the object was moved with a micrometer accuracy. The only parameter is the axis along which the object has moved which we parametrize as $\boldsymbol{\theta}_{\text{pose2}} \in \mathbb{R}^2$, which has two degrees of freedom (normalized 3D axis).

With this approach, we have added a prior knowledge about the distance m between two radiographs with a very high accuracy, in our case 0.001 mm. The movement is much more precise and much easier to achieve than any typical dimensional measurements we could perform with contact methods such as calipers. This prior knowledge enables the optimization algorithm to correctly estimate the overall size of the object in the projections.

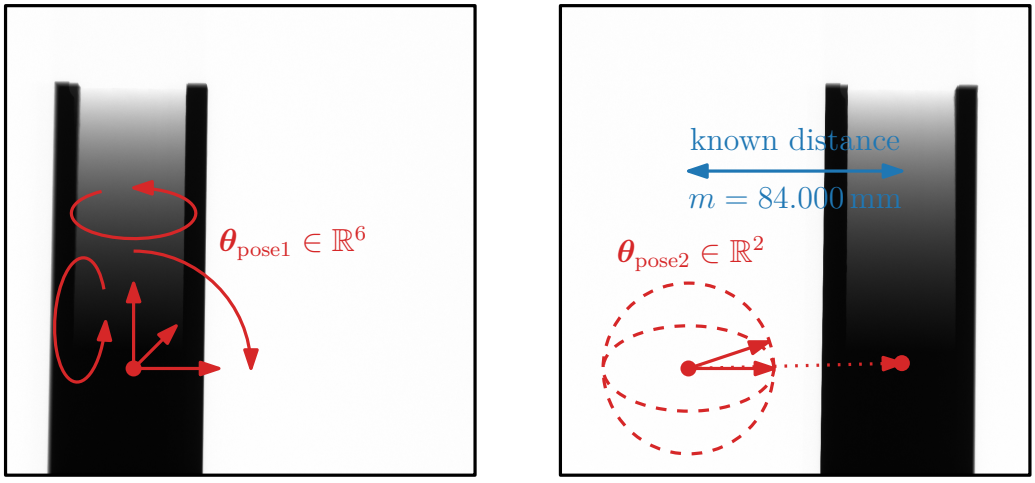


Figure 3.8: Two radiographs of an object moved a known distance. The original pose (left) has six degrees of freedom. The new pose (right) has only two, because we assume the rotation did not change and we know the movement distance.

Different rotations In addition to the radiographs mentioned above, we may also add at least one other radiograph with a different rotation (Figure 3.9). This may be helpful for objects where all dimensional features cannot be clearly seen from a single angle. In our case, the manipulator was not calibrated for rotations, so we always assumed new six degrees of freedom $\boldsymbol{\theta}_{\text{pose3}} \in \mathbb{R}^6$ for each rotation as we could not be sure about precise rotation angles or axis. Later in Chapter 5, we evaluate how these additional radiographs influence the accuracy.

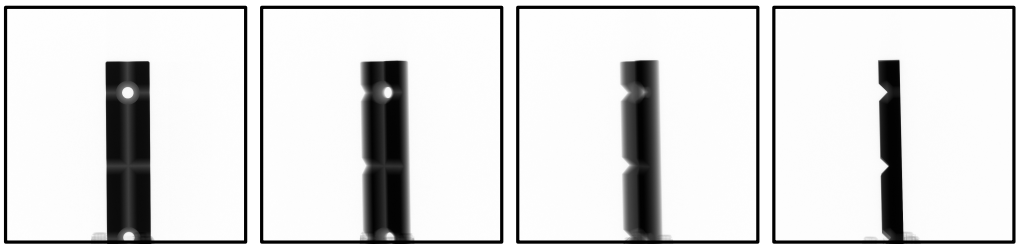


Figure 3.9: Four different rotations of prism jaws. Notice that some dimensional features, such as $\boldsymbol{\theta}_5$ and $\boldsymbol{\theta}_6$ from Figure 3.4, are much better visible from rotated radiographs. In practice, we never used more than one or two additional rotations.

Normalization We want to make sure that the individual radiographs are comparable to each other and that the intensities correspond to the material thicknesses as calibrated in Section 3.4. Hence, we want the maximum intensity $I = 1$ in areas with no material where $x = 0$. We propose to normalize the radiographs, each *individually*, by selecting an empty rectangular region and computing the average intensity I_0 in this region. The region should be large to average out any random noise, but it should *not* cover the edges of the radiograph as the intensities there might be higher due to the detector filters and artefacts.

Once we know the white average I_0, we normalize every single pixel separately. Let I denote a pixel intensity, we define the new intensity I^* as (Figure 3.10):

$$I^* = \min\left(\frac{I}{I_0}, 1\right). \tag{3.9}$$

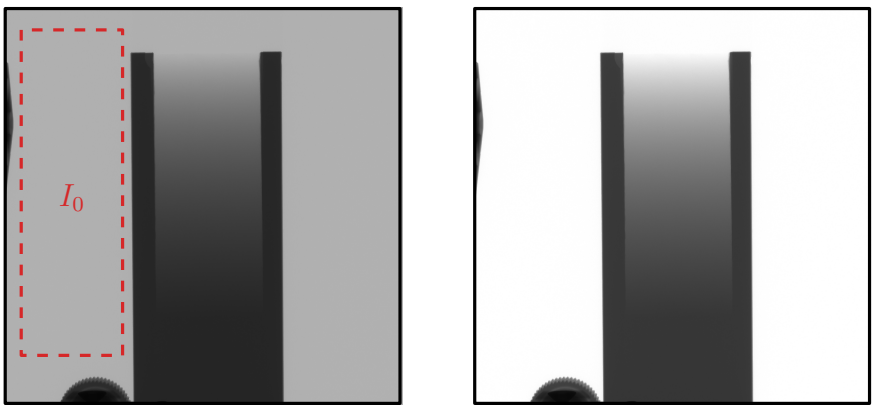


Figure 3.10: Radiograph before (left) and after (right) normalization. The red area in the original radiograph had intensity approximately $I_0 \approx 0.692$, after normalization it is 0.999. Note that typically we do not reach 1.000 exactly because we clamp the individual pixel intensities to 1 if they are above 1.

3.6 Renderer

In Section 2.4, we described several possibilities how radiographs can be simulated. Generally, the solutions are based either on full Monte Carlo simulations, attenuation-only ray tracing, attenuation-only rasterization, or some hybrid approaches that combine stochastic and deterministic simulations.

Discussion Full Monte Carlo simulations could provide very precise simulations, but it is highly likely that we would have to calibrate the individual attenuation coefficients, source spectrum, detector responses, and other parameters in order to get results comparable to references. Other major disadvantage is the performance of such stochastic simulations. In our approach, we need to simulate thousands of radiographs in very high resolutions as fast as possible.

On the other hand, attenuation-only rasterization is extremely fast as it benefits from GPU acceleration. Unfortunately, it is prone to artefacts that need to be filtered, which lowers the image quality. Furthermore, it is not straightforward to simulate off-center projections or effects such as geometry blurring and the geometry must always be triangulated, which we originally wanted to avoid.

Hence, we decided to use attenuation-only ray tracing, which we think provides the best ratio between cost and quality and allows us to easily implement the aforementioned features. Our implementation makes use of the trick shown by Vidal et al. [2009] and discussed in Section 2.4 that enables accumulating distances without intersection sorting. Furthermore, our implementation is GPU-accelerated via the OptiX ray-tracing framework (see Chapter 4 for more details).

Projection parameters The simulated radiographs should match the reference radiographs as much as possible. Hence, our ray tracing should be based on the parameters of the real X-ray setup that was used to acquire the reference radiographs. It is necessary to know the values depicted in Figure 3.11, that is the detector resolution (in pixels), the detector physical dimensions so that we can compute the pixel size in millimeters, the coordinates of the projection center (which in our setup is not the middle pixel), and the source-detector distance D.

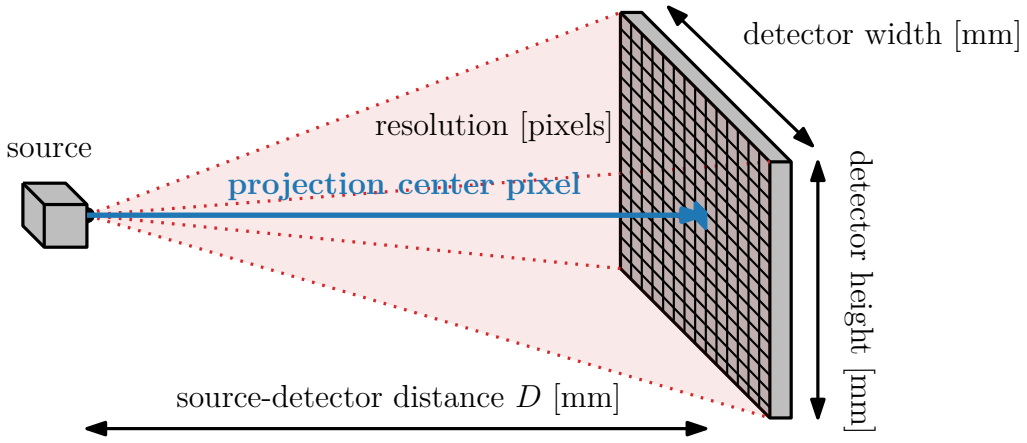


Figure 3.11: Projection parameters of the X-ray setup (see text).

Algorithm steps in each iteration To generate a virtual radiograph (render), the following steps are performed for each individual pixel. Assume that the current pixel is located at 3D coordinates $\mathbf{d} \in \mathbb{R}^3$ (in millimeters) calculated from the projection parameters and offset by the center. Note that the steps are independent for each pixel, so they can be spatially parallelized, which in our implementation is handled by the GPU-accelerated OptiX raytracing framework.

1. **Ray casting**
   First, we cast a ray $\mathbf{r}$ in the direction $\boldsymbol{\omega} \in \mathbb{R}^3$ from the X-ray source located at $\mathbf{s} = (0, 0, D) \in \mathbb{R}^3$ towards the detector pixel $\mathbf{d} \in \mathbb{R}^3$ (Figure 3.12):

   $$\mathbf{r}(t) = \mathbf{s} + t \cdot \boldsymbol{\omega}, \quad t \geq 0. \tag{3.10}$$

   We find if and where the ray $\mathbf{r}$ intersects surfaces of the 3D model. For n intersections, we get n ray parameters $t_1, \ldots, t_n$ that represent the distances along the ray in which the intersections occurred. These distances can be in any arbitrary order. For each intersection, we also find the normal vectors $\mathbf{n}_1, \ldots, \mathbf{n}_n$ of the surface geometry.

2. **Accumulating lengths**
   For each parameter $t_1, \ldots, t_n$, we accumulate the length x that the ray traveled through the object material as we already saw before in Equation 2.1 [Vidal et al., 2009]:

   $$x = \sum_{i=1}^{n} \Big(- \operatorname{sgn} (\boldsymbol{\omega} \cdot \mathbf{n}_i) \cdot t_i \Big). \tag{3.11}$$

3. **Calculating pixel intensities**
   With accumulated x, the pixel intensity I is computed as follows:

   $$I = \sum_{k=1}^{M} a_k \exp\left(-\mu_k x\right), \tag{3.12}$$

   where a_k and μ_k are the calibrated material parameters from Section 3.4.

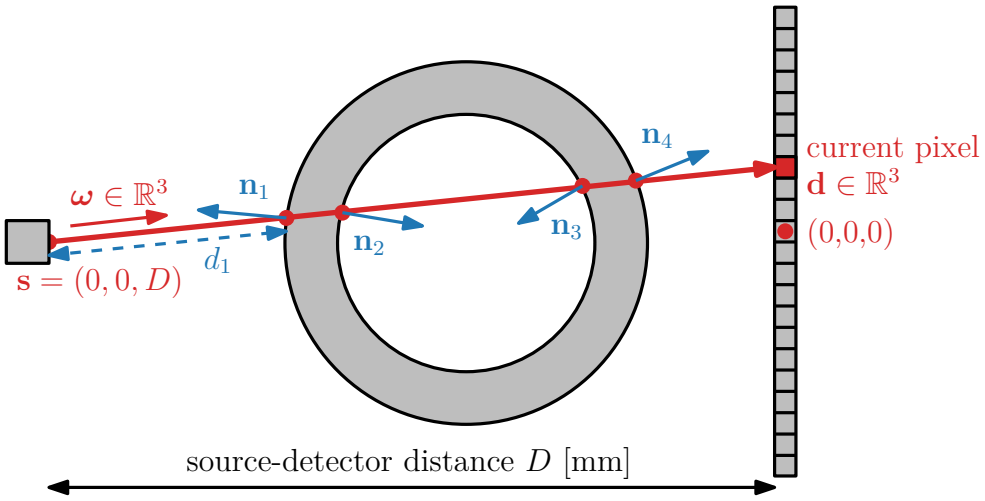


Figure 3.12: Our ray tracing coordinate system (see text and Figure 2.4).

3.7 Loss function

The reference radiographs (Section 3.5) and simulated radiographs (Section 3.6) are compared against each other using a *loss function* or *error function*. This is the function that we minimize in the optimization process.

Normalized measure In our approach, using the simplest error metric is sufficient. First, we compute per-pixel differences between the radiographs (Figures 3.13 and 3.14). To get a single number that we can optimize (Figure 3.15), we sum up the absolute per-pixel differences as already shown in Equation 3.2.

Sometimes, we are only interested in the difference in a specific *region of interest*, e.g., because the reference radiographs contain parts that we do not simulate. To ensure that regions of different sizes contribute the same, we also *normalize* the error metric. The loss function can be written as:

$$\sum_{n=0}^{N} \sum_{(i,j)} \frac{|f_n(\boldsymbol{\theta})(i,j) - \mathsf{R}_n(i,j)|}{(i_{n\max} - i_{n\min} + 1)(j_{n\max} - j_{n\min} + 1)}, \tag{3.13}$$

where N is the number of references and the (i, j) are coordinates inside the region of interest of the n-th radiograph with bounds $i_{n\min}, i_{n\max}, j_{n\min}, j_{n\max}$.

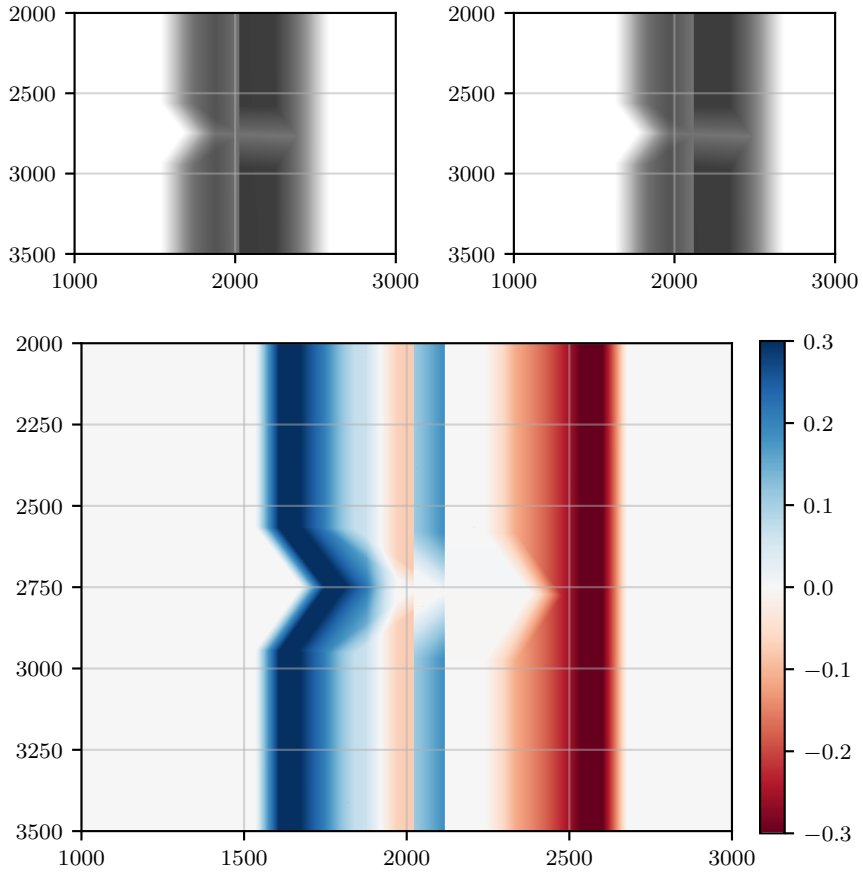


Figure 3.13: Simulations of two slightly misaligned prism jaws radiographs are shown at the top. Their per-pixel difference is shown at the bottom using a colormap. The blue parts are present in the top-left radiograph but missing in the top-right, the red parts are present in the top-right but missing in the top-left. Image axes are pixel coordinates.

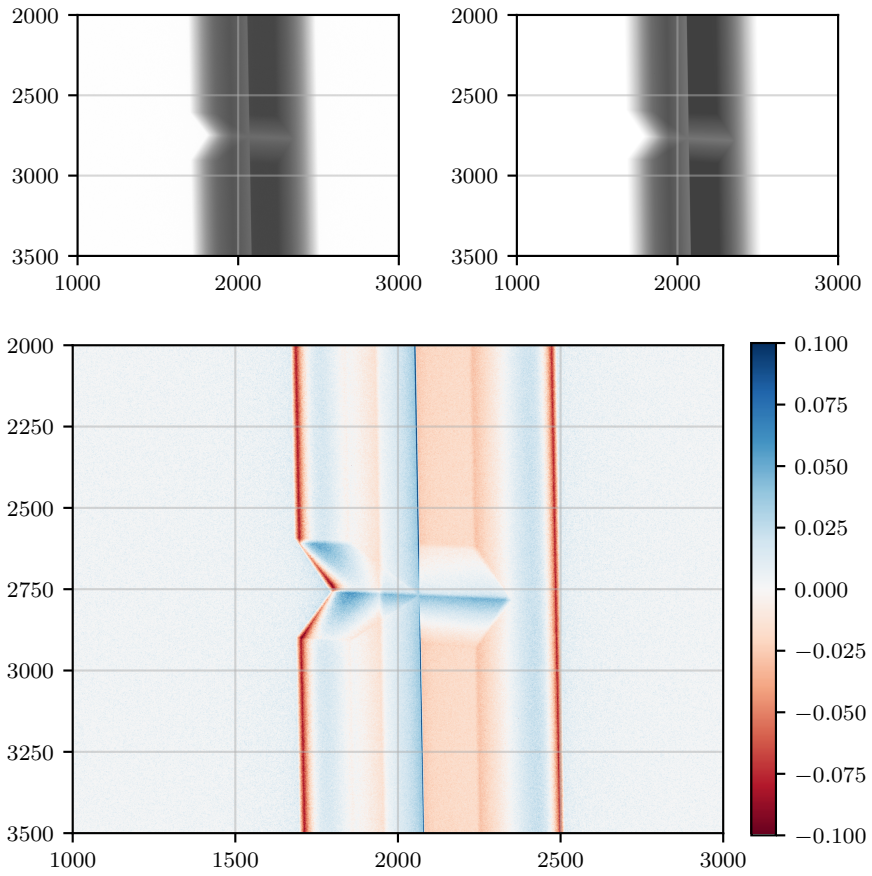


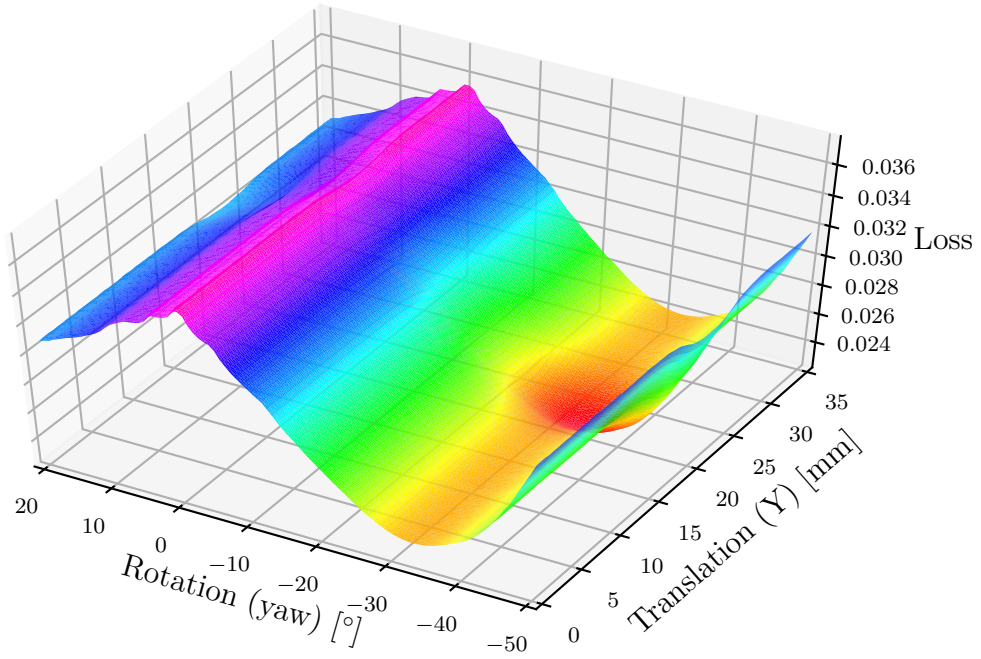
Figure 3.14: Visualizing the per-pixel loss can help spot slight differences that are otherwise not apparent when examining the radiographs side by side. Image axes are pixel coordinates. The optimization algorithm in our approach, however, does not see the individual per-pixel loss, rather, it minimizes the whole absolute sum as in Figure 3.15.

Loss function and parameters Generally, the optimization is performed for tens of parameters, both positional and dimensional, so the loss function is a $\|\boldsymbol{\theta}\|$-dimensional function. We would like to visualize how the loss function looks like with regards to various parameters. Unfortunately, visualizing more than two-dimensional functions is very complicated if not impossible. In Figure 3.15, we visualize the loss function on carefully selected examples with two parameters.

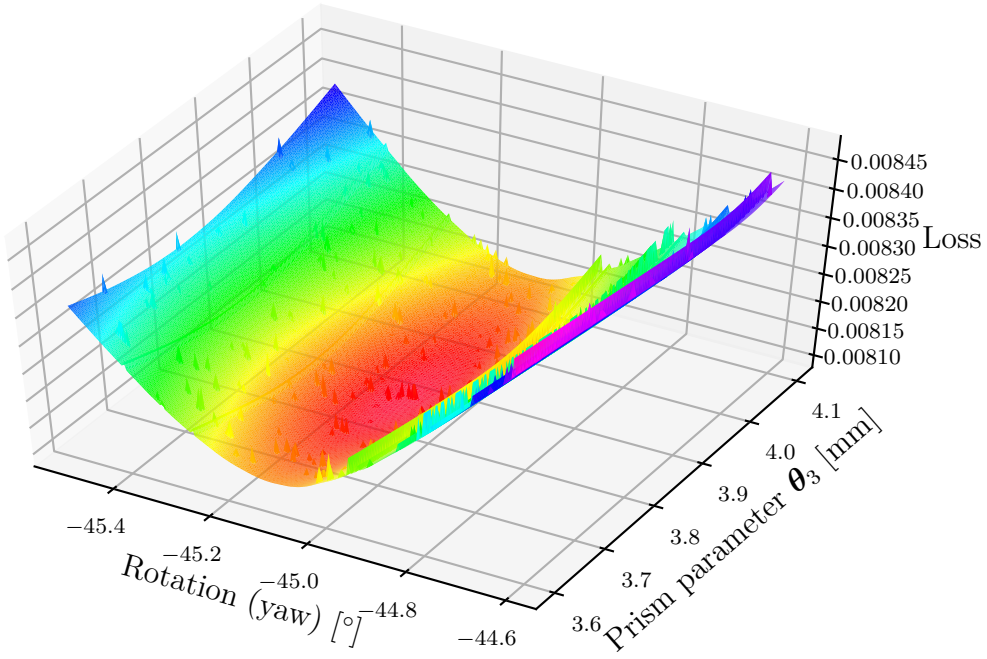
3.8 Optimization algorithm

The loss function (Equation 3.13) from the previous Section 3.7 now has to be minimized by an optimization algorithm. In other words, looking at Figure 3.15, we have to examine the loss values with regards to the parameters $\boldsymbol{\theta}$ and try to find the global minimum in the $\|\boldsymbol{\theta}\|$-dimensional space where the difference between the references and our $f(\boldsymbol{\theta})$ simulations is the smallest.

Motivation One of the major problems is that the loss function has many dimensions. Even if we wanted to only estimate poses in three reference radio-



(a) The impact of translation along the y-axis and rotation around the y-axis. There are various things to notice in the plot. First, there is a "barrier" for rotations around 0° which essentially divides the function into two parts. Second, there is a slight "pit" for translations around 20 mm where a global minimum can be expected. Third, the loss function value is almost constant for rotations around −35° except for the "pit".



(b) Here we can study the impact of a prism jaw dimensional parameter $\boldsymbol{\theta}_3$ (see Figure 3.4b) and the y-axis rotation. Notice that the plot is zoomed-in so we can see how noisy the function is. The spikes are most likely caused by rendering artefacts as the ray-triangle intersections are not watertight (see Section 5.5).

Figure 3.15: Example 3D plots of how our loss function behaves for the prism jaw radiographs with regards to two selected parameters.

graphs, two with a known distance (Section 3.5) and one from a different angle, that is $6+2+6=14$ parameters without even considering any dimensional measurements. Even if any of these parameters had only 1000 possible values, which is a very large understatement, we would still have $1000^{14}=10^{42}$ possible values of $\boldsymbol{\theta}$. The frequencies of modern processors are in the range of 10^9 Hz and we want our approach to finish in a few minutes at most. So, it is no surprise that a brute-force approach would be very inefficient, if not even impossible.

There is a better alternative to brute-forcing the best solution: to use *non-linear optimization* algorithms. There is a plethora of these algorithms [Nash, 2014, Johnson, 2019] that differ by their strategy and they usually try to follow the function gradient to get to a local function minimum.

In our case, unfortunately, we do not know the gradient of our function, because we would have to be able to analytically differentiate our X-ray simulation $f(\boldsymbol{\theta})$, which is beyond the scope of this thesis and we propose it as a possible future work. A possibility would be to numerically approximate the gradients ourselves, but this solution is feasible mainly for a small number of parameters [Nash, 2014]. Hence, in our case, we instead propose to use *gradient-free* optimization algorithms that do not require a gradient to be provided.

Gradient-free local optimization Local optimization algorithms specialize in finding *local* minima that may not necessarily be a global minimum in the whole parameter space. In this thesis, we show a few well-studied algorithms, some of which were also used in pose estimation by Miao et al. [2016] (Section 2.3):

- **Hill climbing** Hill climbing was described for example by Russell et al. [2010] for use in artificial intelligence, originally for maximization (*climbing*), but that is the same as minimization with a negated objective [Nash, 2014]. We describe the algorithm as Miao et al. [2016]. We start with initial parameters $\boldsymbol{\theta}$ and step sizes $\Delta\boldsymbol{\theta}$. We evaluate the objective function at $2\cdot\|\boldsymbol{\theta}\|$ neighbors of $\boldsymbol{\theta}$ by changing a single parameter of $\boldsymbol{\theta}$ by $\pm\Delta\boldsymbol{\theta}$. We select the neighbor with a higher (lower) value of the objective function. If no such neighbor exists, we reduce $\Delta\boldsymbol{\theta}$ by half. The process is iterated until certain stopping criteria are met.

- **Nelder-Mead simplex** The algorithm by Nelder and Mead [1965] [Miao et al., 2016, Johnson, 2019] uses a *simplex* with $\|\boldsymbol{\theta}\|+1$ vertices in the $\|\boldsymbol{\theta}\|$-dimensional parameter space. For example, the simplex is a line segment in one dimension, a triangle in two dimensions, a tetrahedron in three, etc. We start with an initial simplex whose vertices correspond to objective function values at that points. In each iteration, we replace the vertex with the highest objective function value by another point according to a set of rules. The simplex adapts itself and should contract to the final minimum.

- **BOBYQA** The BOBYQA algorithm (Bound Optimization By Quadratic Approximation) by Powell [2009] [Miao et al., 2016, Johnson, 2019] optimizes the parameters by iteratively constructing a quadratic approximation of the objective function, which it iteratively improves and minimizes. Because of the quadratic approximation, the method may perform poorly for objectives that are not twice differentiable [Johnson, 2019].

- **COBYLA** A previous algorithm by Powell [1994] [Johnson, 2019], called COBYLA (Constrained Optimization By Linear Approximations), used *linear* approximations instead of quadratic. It was not used by Miao et al. [2016], but we mention it as an alternative to BOBYQA that also provided very good results in our evaluation.

Initial estimate sensitivity One major problem with local optimizations is that they are very sensitive to initial estimates, i.e., the initial values that the algorithms start with. As an example, consider Figure 3.16. Why does it happen that a slightly off initial guess fails to converge to correct results?

Let us have a look again at Figure 3.15a. If we start a local optimization with rotations between 0° to 20°, the algorithms naïvely follow the steep descent in the left directions. But that is not where the global minimum is located. Similarly, finding the correct y-axis translations is also a huge problem: notice that the loss function is almost constant in the area around −35°, so the algorithms fail to find the correct y-axis region around 20 mm.

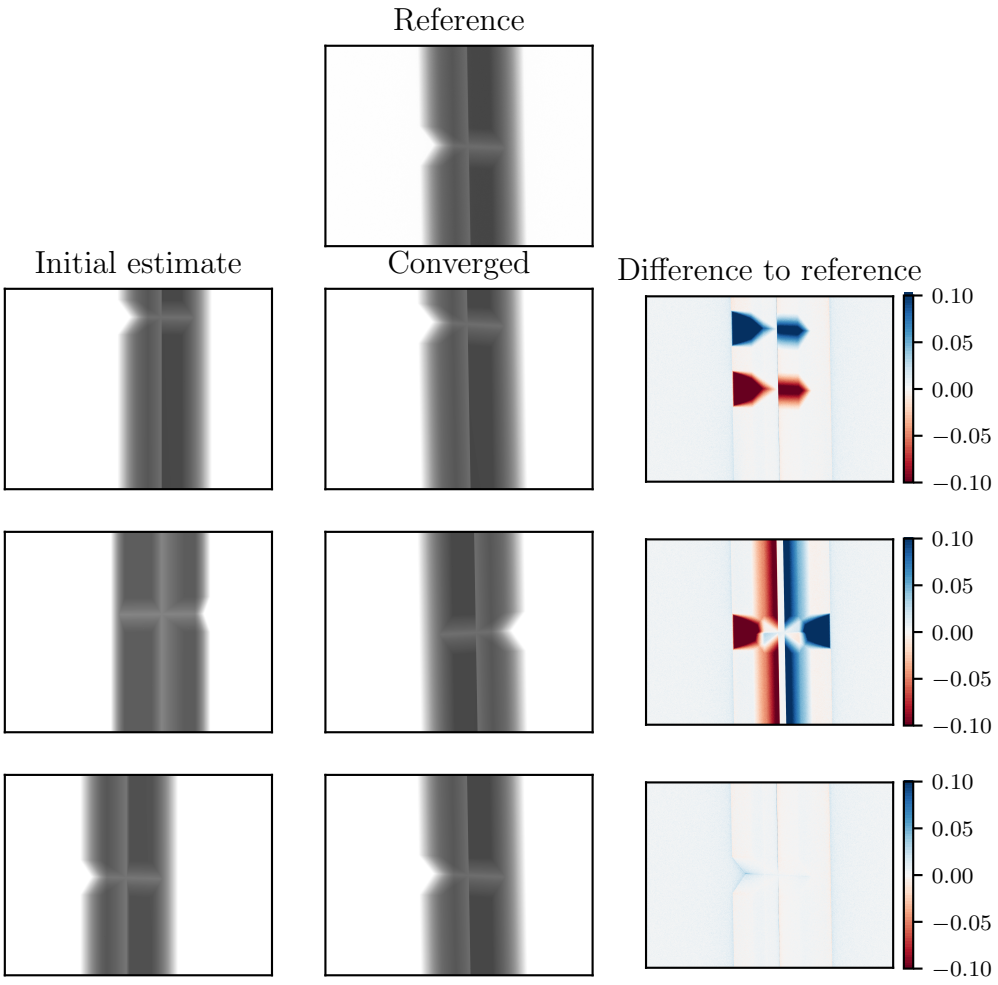


Figure 3.16: Convergence of Nelder-Mead simplex (similar to other algorithms) from different initial estimates towards the reference radiograph (top-right). The three rows depict different initial estimates (left column) and where the local optimization converges from these estimates (middle column). Notice the per-pixel differences from the reference (right column). See text for discussion.

Gradient-free global optimization A naïve solution to the initial estimate sensitivity would be to use *global* algorithms. These are designed to explore the whole parameter space in a clever way to heuristically find global minima. The problem is that our loss function has so many parameters that it becomes unfeasible to explore the space in only a few minutes, which is our target. Also, many of these algorithms are stochastic and inherently depend on random seeds that may give good results in some conditions and bad results in others.

In our approach, we explored two approaches that are *deterministic*, i.e., given the same inputs, they give the same results, which is important for repeatability and reproducibility. They are:

- **DIRECT** DIRECT, abbreviation for Dividing Rectangles, is an algorithm described by Jones et al. [1993]. It is based on systematically dividing the parameter space and finding area with a possible minimum, which is done in a heuristic and deterministic way. Unfortunately, we found that the algorithm does not explore the space as broadly as we would need, i.e., it usually gets stuck in a certain minimum in a low number of iterations and increasing the iterations does not seem to improve the minimum.

- **Uniform sampling** We found out that the best results can be achieved by uniformly sampling the parameter space. We wrote a simple algorithm which for a given maximum number of iterations m divides the N-dimensional space uniformly in $\sqrt[N]{m}$ steps in each dimension. We then evaluate the loss function in each of the points and select the parameters with the minimum corresponding loss. The obvious downside is that we need to know m beforehand, but that is no problem in our method as we can find and fix the best value for specific setups.

During our experiments, we found that the global approaches are more robust than local algorithms in the sense that they do not depend on the initial estimate, only on the lower and upper bounds of the parameters, and they can at least find an area where the local minimum is probably going to be. For example, the initial estimate in the last row in Figure 3.16, which gave the best results out of the three, was found using uniform sampling.

Unfortunately, global optimization becomes unsuitable for more than roughly 3 parameters. That is no surprise as the size of the problem increases exponentially. With M possible values in N dimensions, we have M^N options.

Hierarchical optimization To address the issues, we propose to combine the advantages of global and local optimizations and perform a *hierarchical* optimization that we now explain. We cannot start with a local optimization as that heavily depends on an initial estimate. Of course, we could use a human operator to define an initial estimate for every single radiograph, but that would be impractical and would limit the reproducibility of our method.

To solve this problem, we propose to start with a global optimization algorithm that can find a suitable initial estimate for a local algorithm as in the last row of Figure 3.16. However, we cannot use the global optimization on all parameters and all reference radiographs at the same time as we already know the complexity raises exponentially. Hence, we only use the global optimization for

pose estimation for each reference radiograph *individually*. But that would still be a six-dimensional problem, which we did not manage to solve successfully in our very limited time frames.

Hence, we propose to solve the pose estimation in a hierarchical way as well, inspired by Miao et al. [2016] (Section 2.3). We first only estimate the pose in three degrees of freedom instead of six. Miao et al. [2016] propose to only estimate the x and y positional coordinates (left-right and up-down) and a rotation around the z-axis (towards the radiograph) as these have the most influence on the image. In our case, we propose to instead optimize the rotation around the y-axis as that is the axis around which our manipulator can rotate the measured object. And unlike them, we propose to begin with a global optimization instead of local.

Schematically, our approach is the following:

1. **Global pose estimation (3DOF individually)** We begin with a global optimization of each radiograph separately, i.e., with the loss function only taking a single radiograph into account at a time. We only optimize three positional parameters (x, y, and rotation around y) in case of regular radiographs or two positional parameters (movement axis) in case of a radiograph with a known distance (Section 3.5). Best results were achieved with *uniform sampling* with $m = 5000$ per image.

2. **Local pose estimation (6DOF individually)** The rough results of the global pose estimation are then used as initial estimates improved using local optimization, now with all positional parameters, that is six degrees of freedom or two degrees of freedom for radiographs with a known distance. Again, the optimization is done individually for each radiograph and no more than 5000 iterations were necessary to obtain accurate results.

3. **Local optimization (all together)** The pose estimation results are then used as initial estimates for a final local optimization. Here, we optimize all parameters (including dimensions and/or material) together with all reference radiographs. This is important because the dimensions are shared across the radiographs and also depend on the poses, so we cannot optimize these values separately.

Level of detail In our case, the full resolution of the detector was 4096×4096 pixels, which is more than 10^7 pixels. Simulating such high resolutions and then computing the image differences may be slow even when GPU-accelerated. Furthermore, even a slight change in the parameters may cause a significant change of the objective function in comparison to lower resolutions. Downscaling the images may reduce the noisiness by averaging neighboring values and also make function evaluations faster.

Hence, we propose to begin the optimization with downscaled images, e.g., by constructing MIP maps [Iser, 2017], and dynamically increasing the resolution for higher accuracies. Attention must be paid not to optimize at too low resolutions where the changes in the parameters are not even visible, because then we might converge to non-sense values.

In our final evaluation (Chapter 5), we did the first two optimization steps (global and local pose estimation) with 2^3-times lower resolutions (that is the

third level of the pyramid) and only the final optimization was performed in the original resolution.

Stopping conditions Let us now briefly discuss the stopping conditions of our hierarchical optimization. We must somehow detect when the optimization finished and we converged to accurate results. The various optimization algorithms themselves have their own termination rules dependent on their specific implementation [Johnson, 2019]. But beware that such a termination may only mean that the algorithm simply lowered its step size (or simplex size) so low that it essentially cannot find any lower value anymore *in the local neighborhood*.

Unfortunately, because of the nature of our objective functions, stopping at a certain local minimum does not necessarily mean there is not a lower minimum a little bit further away. With many of the algorithms that we tried, it happens that running them again from their last estimate but with the original step size can converge to an even *lower* minimum.

Another problem is that we do not know the lowest possible error that we can achieve. The random noise and the differences between the real radiographs and our attenuation-only simulations mean that we typically cannot converge to a *zero* error. There will almost always be *some* difference between the reference and our estimate, even if we have 100 % accurate parameters.

During our experiments, we found out that the *iteration count* can be used as a valid stopping condition. For the first two hierarchical steps, using 5000 maximum iterations per radiograph gives accurate results and because of the lower level of detail, it is relatively fast. For the final optimization, we also used a 5000 iterations limit but this time for all the radiographs together as we do not perform the optimization individually anymore.

Please note that if the local optimization algorithm terminates before the maximum number of iterations with a certain estimate $\boldsymbol{\theta}^*$, we simply run the algorithm again starting from $\boldsymbol{\theta}^*$ but with the default step size again. When the total limit is reached, we terminate the currently running algorithm and as our final estimate, we use $\boldsymbol{\theta}_{\text{min}}$ corresponding to the lowest minimum found so far (which possibly is *not* the last estimate).

4. Implementation

In the previous chapter, we described our approach and its individual high-level components. In this chapter, we briefly explain how we implemented the method in our prototype GPU-accelerated application. In Chapter 5, we use it to evaluate how the method behaves in real situations with real radiographs.

We begin with an overview of our application design (Section 4.1) followed by a more technical Section 4.2, where we describe selected technical details about our individual components such as the renderer.

4.1 Overview

Our prototype application is intended for experimentally evaluating our method on real input data. It can be run in two modes: fully automatically in a command line, or with a user interface for live debugging (Figure 4.2).

It is no surprise that the application internal structure depicted in Figure 4.1 loosely follows the method diagram from Figure 3.2. The main differences are that the real application has to somehow load (and save) user inputs, which we represent as *projects*, and that our components are not abstract black boxes, but classes with real implementations (Section 4.2). Finally, the user interface is just a thin layer which reads and modifies the components directly.

Iterative dataflow model From the nature of optimization algorithms, our method is *iterative*. That means that the objective (loss) function is repeatedly evaluated with different parameters. However, a single evaluation depends on results of other components. Hence, application data flow from some components to other, until they reach the end, and then the whole process is repeated until a stopping condition is met. This is very similar to machine learning architectures like *TensorFlow*[1] that we got inspired by. The computation itself and the dependencies between components (*nodes*) are represented by a *dataflow graph*.

In such a graph, each node represents a computation that depends on the *input* nodes. Hence, for a node to be computed, we need to first compute its *predecessors* in the graph. Notice in Figures 3.2 and 4.1 that the nodes can be topologically ordered, i.e., there is no component whose input depends on its own outputs (within a single iteration). When the optimization algorithm wants to evaluate the loss function, we run all the predecessors one-by-one in their topological ordering, which guarantees that all inputs are known before the execution of each individual node.

User input Which specific nodes there are in the graph and how they are connected is defined in *user input*. For example, to create a graph containing a single reference image *node*, we run the application with a path to a TIFF image. Most of the nodes are configured with the JSON (JavaScript Object Notation) format, which contains human-readable attribute-value pairs. The configurations are inter-connected, so a single configuration file can link to a different file. For

[1] https://www.tensorflow.org/guide/graphs

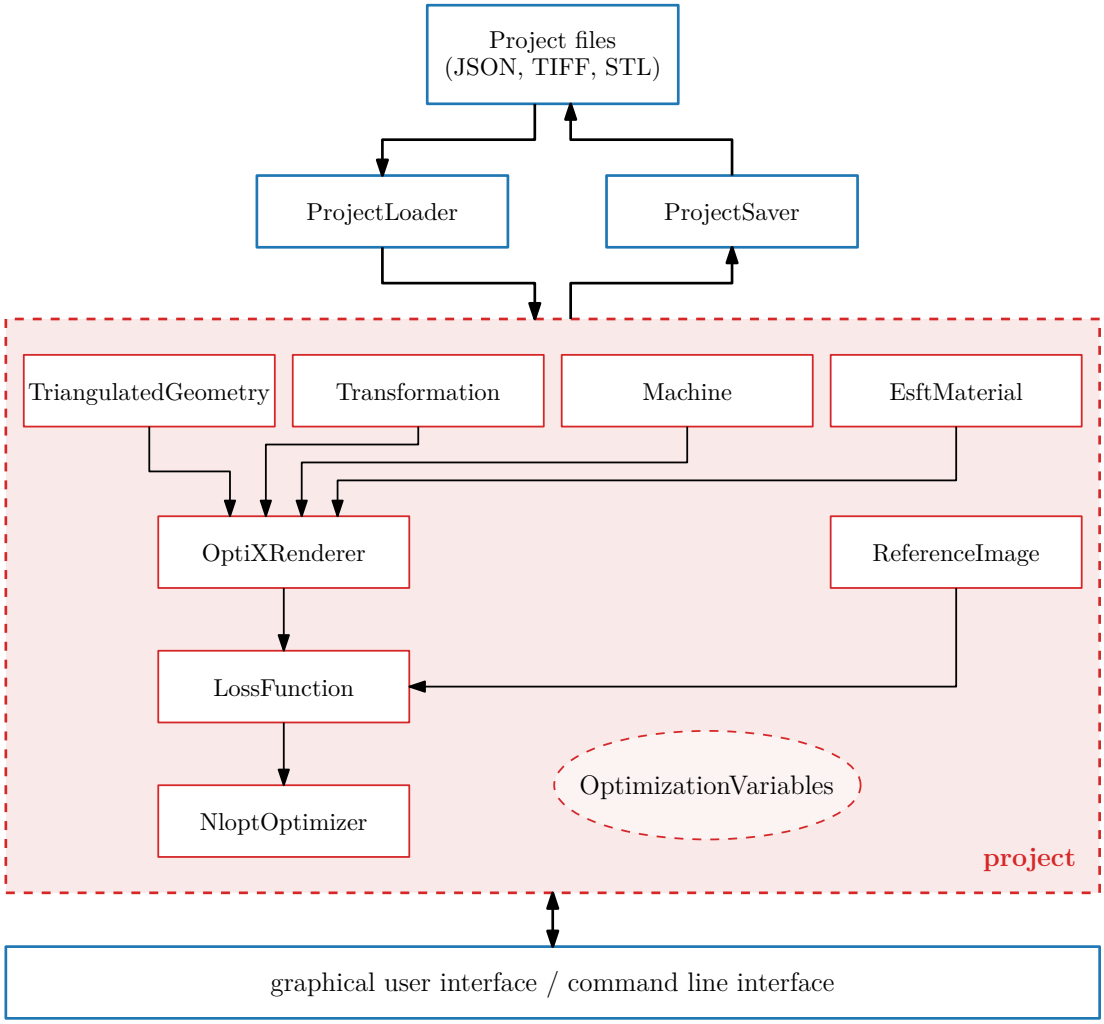


Figure 4.1: Implementation diagram. Compare to Figure 3.2 which represents the method diagram. The method corresponds to the red "project" part of the implementation diagram, the project loading and saving (above) and user interface (below) can be seen as utilities that enable interacting with the method.

example, a *loss function* configuration can contain a path to a *reference image* configuration, etc. So just by loading a *single* input file, the application *recursively* parses the whole project and constructs its dataflow graph. This was very useful during the development and evaluation of our method, as it enabled reusing only parts of the configurations and creating various test scenarios. For practical usage, of course, this would have to be replaced by a simpler user-friendly interface.

Output and interaction Certain nodes can be saved back to the file system. For example, the renderer can save the simulated radiographs to TIFF files just like regular radiographs, and the estimated dimensional measurements can be saved to a JSON file. Furthermore, the nodes can be interacted with in real time in the user interface (Figure 4.2), which is very valuable for debugging purposes.

Technologies The individual nodes are *classes* programmed in the C++ language, which is a typical language for high-performance and visual computing. To make the code simpler and clearer, we even use some of the modern features from C++11 to C++17 like `std::variant`. The GPU-accelerated parts, i.e., the loss function and the renderer, are written in CUDA and specifically the renderer is using Nvidia OptiX[2] ray-tracing framework [Parker et al., 2010].

The *user interface* uses OpenGL[3] and the OpenGL Shading Language (GLSL) for the rendering, glad[4] for loading the OpenGL, glfw[5] for window, context, and input management, and Dear ImGui[6] for the interactive user interface widgets.

The optimization algorithms Nelder-Mead, BOBYQA, and COBYLA (Section 3.8) are provided from NLopt[7] [Johnson, 2019].

For loading and saving projects and reference radiographs, we use libtiff[8] for handling TIFF images and nlohmann's JSON[9] for handling JSON configuration files. The triangulated STL files are loaded with our own code.

Additionally, we use three helper libraries. For vector and matrix mathematics, we use glm[10]. For string formatting and printing, we use fmt[11]. For unit testing of certain components, we use Catch2[12].

For source codes and executables, please see the electronic attachment (Attachment 2 – Electronic attachment contents). User reference is also available in Attachment 1 – User reference.

4.2 Component details

Let us now have a brief look at the dataflow graph implementation and the individual components from Figure 4.1. As we can see, the implementation is es-

[2]https://developer.nvidia.com/optix
[3]https://opengl.org/
[4]https://github.com/Dav1dde/glad
[5]https://github.com/glfw/glfw
[6]https://github.com/ocornut/imgui
[7]https://github.com/stevengj/nlopt
[8]http://www.libtiff.org/
[9]https://github.com/nlohmann/json
[10]https://github.com/g-truc/glm
[11]https://github.com/fmtlib/fmt
[12]https://github.com/catchorg/Catch2

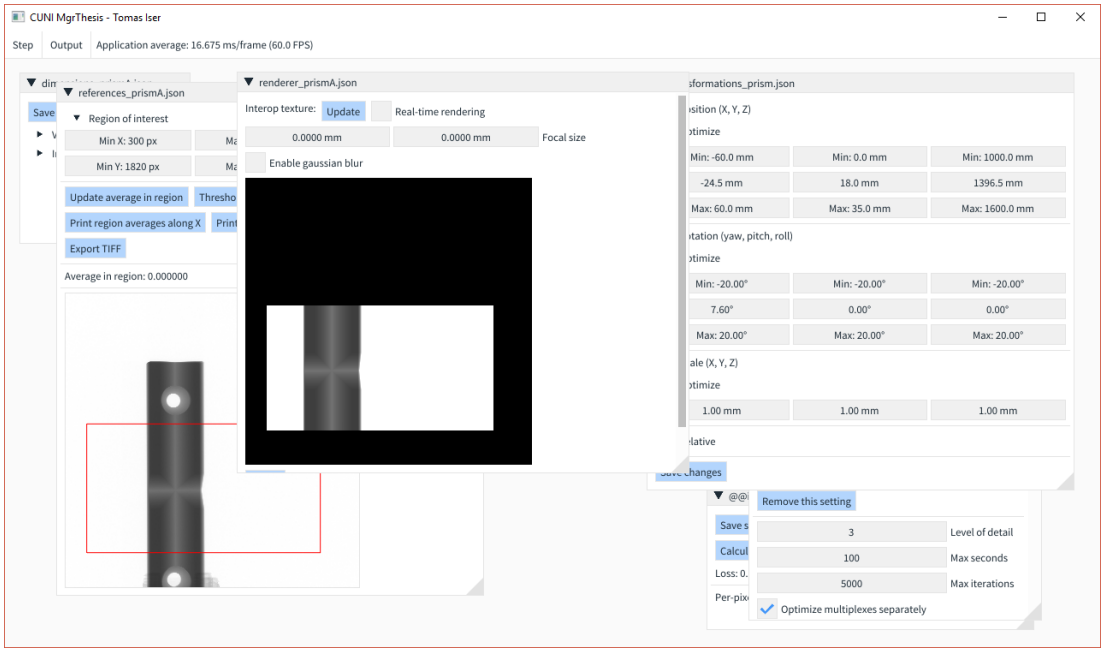


Figure 4.2: Our graphical user interface for debugging and experimenting with various configurations. See Attachment 1 – User reference.

sentially divided into three parts: the *project loading and saving*, i.e., constructing the graph and serializing it back to files, then the *dataflow graph* itself and its nodes, and finally a thin *user interface* layer.

Dataflow Each node is inherited from the `Node` class which declares a set of common methods such as `step()`, which executes a single step (calculation) of the given node in a single iteration. The nodes keep track of their own inputs which can be recursively traversed by `find_topological_ordering()`. Alongside the nodes, we have the `OptimizationVariables` class which keeps track of all the variables to be optimized, i.e., it corresponds to $\boldsymbol{\theta}$. We also keep track of the lower and upper bounds $\boldsymbol{\theta}_{\text{lb}} \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}_{\text{ub}}$ to make sure the parameters are not estimated with unreasonably low or high values. Additionally, `OptimizationVariables` also keeps track of the level of detail and *multiplex*, which is used when multiple reference images and their poses are present in the graph and we need to iterate over all of them in a single optimization iteration.

Optimizer The optimizer node is represented by the `Optimizer` abstract class, mainly implemented in the `NloptOptimizer` class, which uses the aforementioned NLopt library [Johnson, 2019] with the Nelder-Mead, BOBYQA, COBYLA, and other optimization algorithms (Section 3.8). When an optimization is started, we first acquire all optimization parameters from the predecessors of this node into `OptimizationVariables`. Then, in each iteration of the inner algorithm, `step()` is called on all the predecessors in the topological order, which ends with the `LossFunction` node. Then, the value of the `LossFunction` is read and passed to the optimization algorithm itself that decides on the new $\boldsymbol{\theta}$. We propagate the new $\boldsymbol{\theta}$ to the `OptimizationVariables` and notify all predecessors of the changes. Then, the graph is ready for a new iteration.

Loss function The `LossFunction` abstract class is implemented mainly in the `CompareTwoBuffers` class. It takes two image buffers (a reference image and a rendered simulation) and compares them pixel-by-pixel summing up the absolute values of the differences (Equation 3.13 and Section 3.7 in general). The core is implemented in CUDA, the per-pixel calculations are heavily parallelized on a GPU by striding over the 2D buffers. The final value is summed-up on a CPU after synchronization. The total difference value is calculated in `double` precision. We experimentally found out that `float` is losing precision for 4096×4096 pixels with up to 0.8 % difference from `double`.

To generate colorful difference images where each pixel difference is encoded in a color, there is another node `PerPixelLoss` that can be used for debugging purposes. We also implemented a `LossToCsv` class that can be connected to a loss function and used for tabulating loss function values to a `.csv` files. We used this component for generating data for the visualizations that we embedded in this thesis such as in Figure 3.15.

Renderer The abstract `Renderer` is implemented mainly in the `OptiXRenderer` class. It uses the low-level GPU-accelerated ray-tracing engine Nvidia OptiX [Parker et al., 2010], which can be used for generating rays and handling their intersections with geometry, which is exactly what we described in Section 3.6. OptiX supports various acceleration structures for detecting intersections and provides great performance with fully customizable programs.

For generating rays and computing the final pixel intensities, we wrote a *ray generation program*, where at least one new ray is created for every pixel. In case of supersampling or simulating the focal spot, we stochastically generate multiple rays between varying positions at the X-ray source and varying positions within the individual pixels. The 3D coordinates of the pixels are calculated from the projection parameters described in Section 3.6.

The intersections with geometry are calculated in *intersection programs* and handled in *any hit programs*. The *intersection program* calculates whether a ray hit a triangle from the triangulated geometry defined via a *vertex buffer* (vertex coordinates) and an *index buffer* (pointers to the vertices of the triangles). When *any* intersection is detected, the *any hit program* is called in an arbitrary order, which we use to accumulate the distances via Equation 3.11.

To handle the OptiX image buffers in our `CompareTwoBuffers` loss function, which uses CUDA directly, and to show the buffers in our user interface in OpenGL, we implemented the `InteropTextureBuffer` class that provides interoperability between OptiX, CUDA, and OpenGL.

Reference images A reference image is represented by the `ReferenceImage` class, loading and saving radiographs in TIFF files is handled in the `Image` class. To support level of detail (Section 3.8), MIP maps are stored in the `ImagePyramid` class which is also responsible for storing the image on the GPU for CUDA access from the `CompareTwoBuffers` (see above). The `ReferenceImage` class also supports normalizing the images (Section 3.5) and computing averages along regions, which we used for visualizations in this thesis.

Triangulated geometry The `TriangulatedGeometry` abstract class represents the triangulated geometry and its vertex and index buffers (see Renderer above and Section 3.3). Specifically, we have the `StlTriangulatedGeometry` class for creating the buffers from an `.stl` geometry file. Dimensional parameters and recalculating vertex positions (Section 3.3 and Chapter 5) is defined in the `Wedge` class for our wedge object and the `Prism` class for our prism jaw object.

Transformation Poses of objects, i.e., their positions and rotations in the 3D space (Section 3.5) are implemented in the `Transformation` class. It holds the parameters such as position, rotation, or relative translation and it can convert them into a *transformation matrix* used by the renderer. When multiple reference images are used, we have pose parameters for each of them, which we handle by the aforementioned *multiplexing*, i.e., we have an array of the poses and we keep track of an index to the currently active pose. Note that in our implementation, object translation is always in global world coordinates *independent* on local object rotation and scale, which we achieve by applying inverse object transformations to the translation axes.

Machine The projection parameters for the renderer (Section 3.6) are stored in the `Machine` class, which is a wrapper around the `MachineParameters` struct. We call it *machine* as the parameters are inherent to the whole X-ray "machine", i.e., how the source and detector are positioned with regards to each other.

Material Material calibration parameters are implemented in two independent classes. The exponential fitting explained in Section 3.4 are implemented in the `EsftMaterial` class, where ESFT stands for Exponential-Sum Fitting of Transmissions from the method by Wiscombe and Evans [1977]. The class stores k_1, k_2, k_3 and a_1, a_2 where $a_3 = 1 - a_1 - a_2$. During the evaluation, we found that $M = 3$ is enough to get an accurate fit.

We also had a previous implementation, in the `Material` class, which used tabulated function values instead of the analytical exponential-sums. The table (array) was constructed directly from a radiograph of a wedge object. But as the exponential fitting is very accurate, we stopped using this approach.

Project loading and saving The dataflow graphs are constructed from user input (Section 4.1) in the `ProjectLoader` class. Its constructor takes an array (vector) of inputs that can be either file paths or JSON strings directly. The inputs are parsed one-by-one and nodes are created and stored in an `unordered_map`, which keeps track of the canonical filesystem paths of the nodes to prevent loading a single node multiple times. The `ReferenceImage` nodes are loaded from `.tiff` files, the `StlTriangulatedGeometry` from `.stl` files, and the rest of the nodes are deserialized from `.json` configuration files. The parsing is *recursive* because each node must define its inputs, which have to be loaded as well. For example, if there a single last node, usually the optimizer, we can only load the optimizer as input and the rest of the graph nodes will be loaded recursively.

Some of the nodes of the graph can be saved (serialized), which is handled in the `ProjectSaver` class. This process is not recursive anymore as we already have the graph structure available in an array (vector), so each node is saved

individually. Usually, a node is serialized into a `.json` file in the same format that can be deserialized by the `ProjectLoader` again. The only nodes not saved to JSON are the `OptiXRenderer` and `PerPixelLoss`, which are directly converted to `.tiff` images that can be displayed in common image viewers.

Application and user interface The entry point is implemented in `main.cpp`. From there, we either construct a `CliApplication` instance for command line interface or a `UiApplication` instance for graphical user interface.

The *command line interface* simply loads all inputs from command line arguments via the `ProjectLoader`. Then, a topological ordering is constructed and `step()` is called on all of the nodes in order. If an `Optimizer` is present in the constructed graph, then an optimization is started and runs until the stopping conditions are met. After that, the graph final state is saved and serialized to the file system via the `ProjectSaver`. The application terminates.

The *graphical user interface* is more complicated. Its initialization is the same as in the previous paragraph, but then, we do not implicitly start the nodes. Instead, we wait for user input from the graphical interface. Each node has a little window showing possible configurations and previews. The nodes can be edited and viewed in real time, e.g., a live preview of the `OptiXRenderer` render is shown. Hence, the dataflow graph is modified and executed via the interactions with the buttons and other graphical widgets. Technically, the interface for all the nodes is implemented in the `NodeUserInterface` class. It uses a double dispatch technique to be called on each of the nodes with the appropriate method specialization for the corresponding dynamic types.

5. Results and evaluation

Finally, in this chapter, we evaluate the accuracy and overall performance of our method and we show that it is indeed capable of accurate dimensional measurements from a very limited number of radiographs. We begin with describing our evaluation setup (Section 5.1), then we show our results achieved using three test objects (two prism jaws, Section 5.2, and a wedge, Section 5.3), and then we discuss and evaluate the limitations caused by the radiography setup itself (Section 5.4) and caused by our method (Section 5.5).

5.1 Our setup

Our evaluation was performed using the setup as shown previously in Figure 1.1 and now in Figure 5.1. This is a standard industrial X-ray setup similar to these shown in other works such as by Kruth et al. [2011], Mery [2015], Villarraga-Gómez et al. [2018], and Du Plessis et al. [2018]. In order to evaluate our method reliably, we used high-quality components that we now describe technically.

Source Our X-ray source was developed by *YXLON*, type *FXE-225.48*. Its maximum tube voltage is $U = 225\,\mathrm{kV}$. For our dimensional measurements of aluminum objects, we always used voltage $U = 220\,\mathrm{kV}$. Hence, the maximum photon energy present in the source spectrum should be $\varepsilon_{\max} = 220\,\mathrm{keV}$ (Section 1.3). When measuring the wedge, we used the power of $P = 60\,\mathrm{W}$, so the tube current was approximately $I \approx 0.27\,\mathrm{mA}$. The spectrum itself should be expected to be roughly as in Figure 1.8, but its exact properties remain unknown — we did not measure the spectrum in any way as it is not required for our method.

The focal size varies with the power and according to the source manufacturer, it should be between $2\,\mu\mathrm{m}$ and $6\,\mu\mathrm{m}$. Exact source optical properties are not provided and we did not attempt to measure them in any way. According to the manufacturer, the focus is expected to be elliptical with a Gaussian distribution.

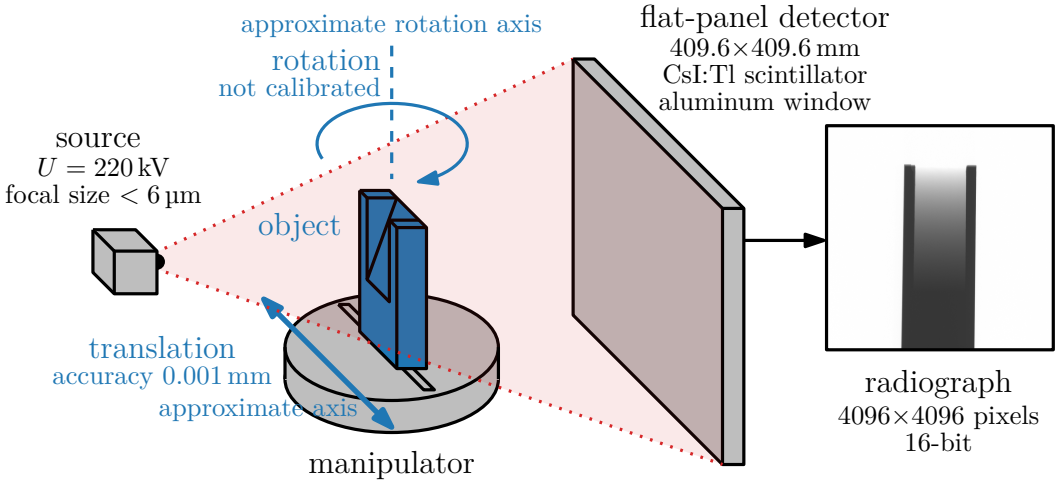


Figure 5.1: Our evaluation setup based on Figure 1.1 with specific details.

Manipulator The measured object was always placed in our manipulator. We used it for two types of motion:

- First, the manipulator enables *translational motion* in the horizontal axis approximately parallel to the detector (consider again Figure 5.1). The axis itself is not calibrated, i.e., we cannot rely on it being exactly parallel to the detector, but the translation *magnitude* can be measured with a 0.001 mm accuracy, which is important for our pose estimation (Section 3.5).
- Then, the manipulator enables *rotation* around the vertical axis (Figure 5.1). The axis itself nor the magnitude are calibrated, so we cannot rely on the rotations being exact and they must be estimated via the optimization.

Detector Our detector was developed by *Varex Imaging*, type *XRD 1611 AP*. It is an indirect flat-panel detector (Section 1.4). It has an aluminum entrance window, which means that the received spectrum is partially filtered and weakened by a thin block of aluminum roughly up to around 40 keV.

It uses a single substrate amorphous silicon active TFT-diode array with a direct deposition CsI:Tl scintillator. The pixels are 100 µm squares, there are 4096×4096 pixels, so the active area is 409.6×409.6 mm large. The detector has 16-bit electronics and the final radiographs are 16-bit 4096×4096 digital images. Later in Section 5.4, we show the image quality and limitations caused by the detector and its sctintillation layer that we already mentioned in Section 1.5.

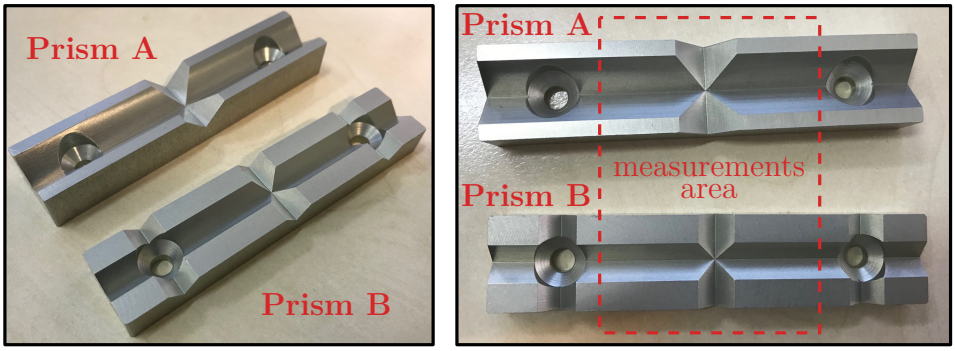
Computation hardware For the optimization process itself, that is, for running the application that we implemented (Chapter 4), we used a modern computer with a high-performance GPU. The configuration was the following:

- **CPU:** AMD Ryzen 7 2700X, 8 cores, 3.70 GHz,
- **GPU:** NVIDIA GeForce RTX 2080, 8.0 GB memory,
- **RAM:** 64.0 GB,
- **Operating system:** Windows 10, 64-bit.

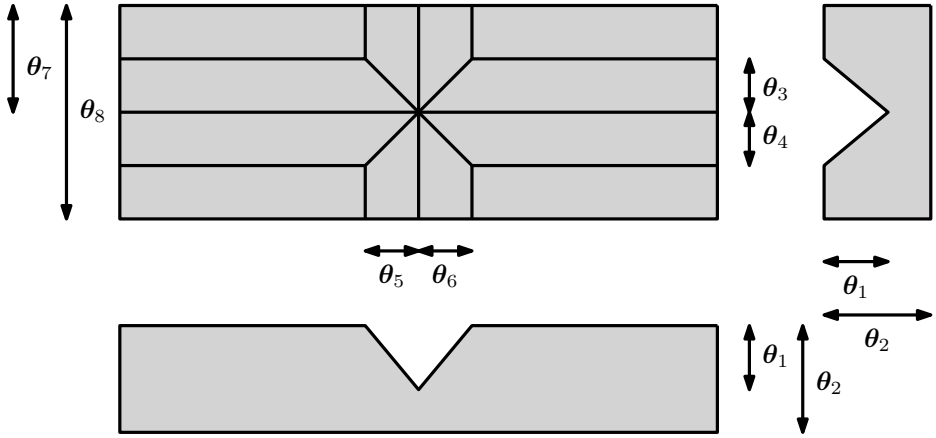
5.2 Dimensional measurements of prism jaws

Let us now evaluate our method in the context of dimensional measurements, which is the main topic of this thesis. In this section, we perform measurements of *prism jaws* that we have already seen in previous chapters. Jaws are used in vice, which is a mechanical tool for clamping objects in a steady position. The jaws typically have a prism shape with holes for attaching them to the vice and one or more cut-outs to better lock the objects in position.

We have two of these prism jaws available that we call *Prism A* and *Prism B* (Figure 5.2). The overall shape of these prisms is the same, so we can parametrize them only once as a single object. This perfectly simulates the conditions in a real manufacturing process: we have multiple objects of the same general shape but different dimensions.



(a) Photos of Prism A and Prism B. We are interested in dimensional measurements of features inside the red area without the holes. Notice that Prism A has bigger cut-outs and is about 3 mm thicker than Prism B.



(b) The dimensional parametrization $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_8$ originally introduced in Figure 3.4b can be used for both Prism A and Prism B. This is exactly what we want to simulate a real manufacturing process where the objects of the same shape have varying dimensions due to the manufacturing process.

Figure 5.2: Prism A and Prism B.

Process overview We propose to perform the dimensional measurements in the following way. We use the same parametrization as we already explained in Section 3.3 and Figures 3.4 and 5.2b. Then, we have to calibrate the source-material-detector pipeline according to Section 3.4. For this purpose, we assume that both prisms are made of identical aluminum.

As we do not have a reference wedge made of the same aluminum with the same thickness ranges, we propose to use Prism B as a reference object for material calibration via optimization (Section 3.4b), which we describe below in more details. Once we have the calibration ready, we use it for dimensional measurements of Prism A. Then, we evaluate the accuracy and repeatability of these measurements in comparison to computed tomography (CT, Section 2.1).

Reference radiographs dataset We acquired and manually normalized a dataset of 14 reference radiographs for each Prism A and Prism B (Figure 5.3). Some of these radiographs represent 10.000 mm translations along the same fixed axis, the rest of them are rotations around the vertical y-axis. Out of this dataset we can always choose a subset of radiographs for evaluating our approach.

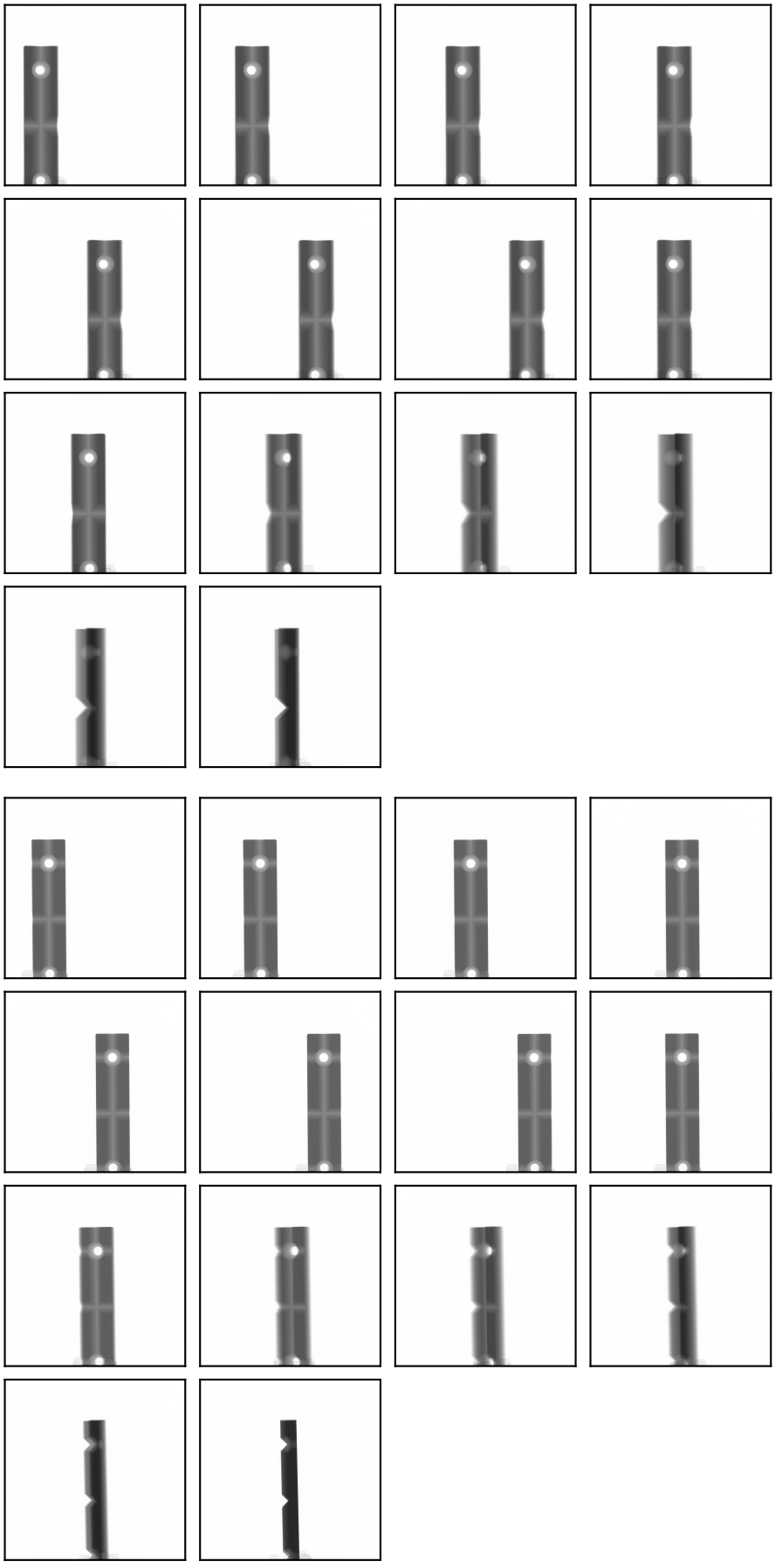


Figure 5.3: Prism A (top) and Prism B (bottom) reference radiographs. The first 7 radiographs in each dataset represent translations by 10.000 mm precisely. The remaining 7 radiographs represent rotations around the vertical y-axis.

1) Material calibration from Prism B We use Prism B to calibrate the source-material-detector pipeline according to Section 3.4b. For this purpose, we need to first accurately measure Prism B via a different method. We used precise computed tomography measurements with an approximately 0.02 mm accuracy (Figure 5.4). They allowed us to find the dimensional parameters $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_4$ and $\boldsymbol{\theta}_7, \boldsymbol{\theta}_8$. The only parameters that remain unknown are $\boldsymbol{\theta}_5$ and $\boldsymbol{\theta}_6$ as they cannot be easily measured from the slices. However, from contact measurements, we know that they are roughly 4 mm each.

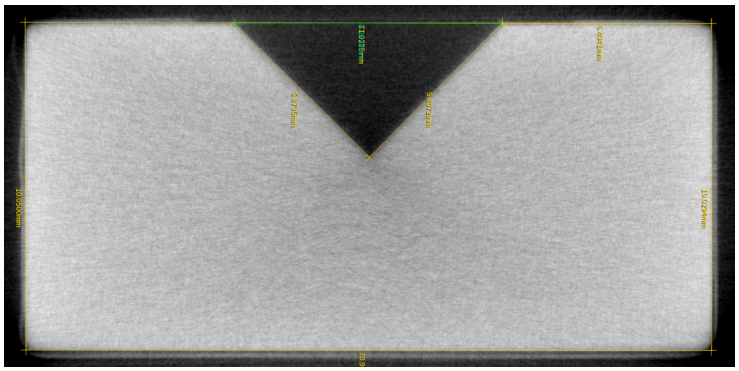


Figure 5.4: Prism reference measurements were acquired via computed tomography. In the figure, we can see a reconstructed slice through the object.

The calibration itself was performed together with pose estimation and estimating the dimensions $\boldsymbol{\theta}_5$ and $\boldsymbol{\theta}_6$. As varying material thicknesses are best seen on rotated prisms, we decided to perform the optimization on two reference radiographs, the 9th and 12th, that represent rotations roughly 15° and 75°.

An important question is how to select the best radiographs for material calibrations. In our case, we decided on the 9th and 12th, but we also tried calibrating from other rotations with similar results. For example, calibration using the 7th, 9th, and 11th radiographs (roughly 0°, 30°, and 60°) also converged to a very similar intensity function. It is obvious that the references should cover wide thickness ranges over which the material parameters are optimized. On the other hand, the radiographs should not be redundant as that would overly prioritize the thicknesses that are dominant. Hence, selecting the 9th and 12th radiographs that represent very far rotations is certainly a correct choice.

The whole optimization process was the following:

1. **Initial estimates** We manually found rough initial pose estimates for both reference radiographs. We fixed all dimensions $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_4$ and $\boldsymbol{\theta}_7, \boldsymbol{\theta}_8$ to the reference measurements. We set $\boldsymbol{\theta}_5 = \boldsymbol{\theta}_6 = 4\,\text{mm}$. We set the material coefficients to $a_1 = 1$, $\mu_1 = 0.116$, and the rest $a_2 = \mu_2 = \mu_3 = 0$ exactly as described in Section 3.4b. The coefficient μ_1 was calculated from an average intensity where the thickness was roughly 10 mm.

2. **First optimization** Then, we performed a first optimization on the pose parameters, $\boldsymbol{\theta}_5$, $\boldsymbol{\theta}_6$, and material $(a_1, a_2, \mu_1, \ldots, \mu_3)$ but only on radiograph number 9 (15°). We used Nelder-Mead simplex with 10000 iterations with the original image resolution.

3. **Second optimization** Using the estimates from the previous optimization, we ran the same optimization process once again, this time on radiograph number 12 (75°).

4. **Final optimization** Finally, we ran the optimization again for both radiographs together so that the material is optimized on all the varying thicknesses at the same time.

Via this process, we got the following intensity function:

$$I(x) \doteq 0.563 \exp(-0.068x) + 0.024 \exp(-x) + 0.440 \exp(-0.279x). \quad (5.1)$$

In Figure 5.5, we can see the per-pixel difference between the optimized simulation and the reference radiograph depicting roughly the 15° rotation. As we can see, the highest errors are located at the object edges. This is highly likely caused by scattering of the visible light on the detector, which we discuss later in Section 5.4. Inside the object itself, the errors are very low. Notice also the orange line roughly at y-coordinates 1800. We suspect that this line may be caused by ghosting from the previous image, which we also discuss in Section 5.4.

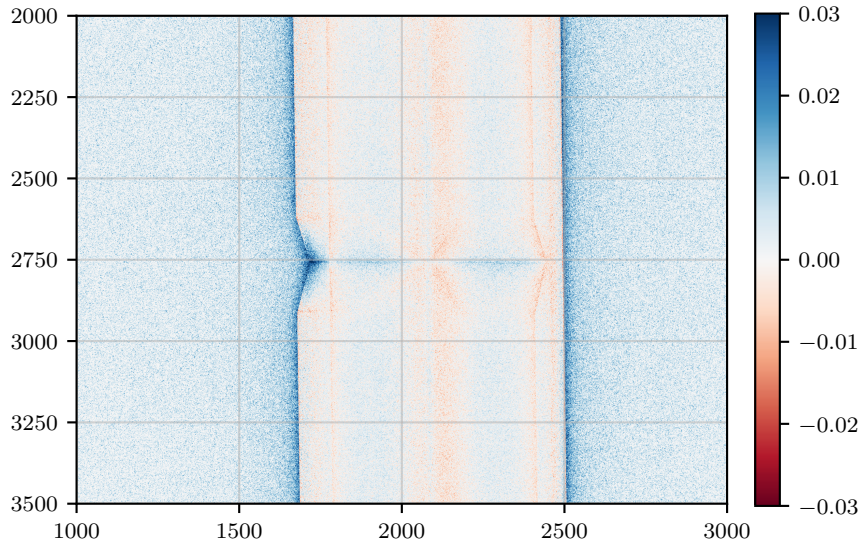


Figure 5.5: Per-pixel differences between the final calibrated simulated radiograph and the first reference radiograph. Image axes are pixel coordinates. The blue shadows around the object are most likely caused by scattering on the detector, which is discussed later in Section 5.4.

2) Prism A dimensional measurements Now, using the calibrated material from Equation 5.1, we estimate the Prism A dimensions. We are mostly interested in *accuracy* and *repeatability* of the measurements. By accuracy, we mean how far our estimates are from the reference measurements acquired from computed tomography. By repeatability, we mean if we can converge to the same results from different radiographs.

For this purpose, we will report two different properties that were also used for example by Butzhammer and Hausotte [2019]. The *mean absolute error* is the average over all absolute values of the differences between our estimates and

reference dimensions. The *maximum absolute error* is the worst result that we have achieved, i.e., the maximum over all the absolute errors. The reason we use absolute values is that the errors may be both positive and negative, but we do not want them to average out to zero, because in real industrial situations, we would only perform a *single* measurement.

Measurements from two radiographs First, we performed a series of measurements using only two reference radiographs. In total, we used $\binom{7}{2} = 21$ pairs of translations ranging from 10 mm to 60 mm. In these experiments, we did *not* use the rotation radiographs at all. But keep in mind that the Prism B rotations were used for calibrating the material. The optimization was performed hierarchically exactly as described in Section 3.8.

In Table 5.1, we can see the mean and maximum absolute errors for various parameters. It can be clearly seen that $\boldsymbol{\theta}_7$ and $\boldsymbol{\theta}_8$ were estimated with the lowest errors. The mean absolute errors are within the reference measurements accuracy of 0.020 mm, which means they completely match the CT accuracy. That is most likely because these two dimensions represent the object widths (total width and width from the middle) and they are clearly visible[1] in the reference radiographs.

The highest errors were achieved in $\boldsymbol{\theta}_3$ and $\boldsymbol{\theta}_4$. These parameters correspond to the inner cut-out, whose boundaries are much subtler than the prism width. Finally, the parameters $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ represent the prism thickness (depth). Since we did not use any rotation radiographs, the thickness could not be estimated from any other information than the intensity values. It shows that the material calibration correctly fitted the intensity-thickness relationship.

These experiments show that our method can be successfully used for dimensional measurements with a very high accuracy, for some parameters even comparable to industrial CT, even though we only used two radiographs.

Parameter:	$\boldsymbol{\theta}_1$	$\boldsymbol{\theta}_2$	$\boldsymbol{\theta}_3$	$\boldsymbol{\theta}_4$	$\boldsymbol{\theta}_7$	$\boldsymbol{\theta}_8$	**Total**
Reference:	6.923	13.020	6.848	6.963	10.438	20.977	
Mean |error|	0.041	0.025	0.055	0.125	0.019	0.013	0.046
Max |error|	0.097	0.080	0.165	0.207	0.049	0.024	0.207

Table 5.1: Mean and maximum absolute errors collected from $\binom{7}{2} = 21$ evaluations of Prism A from two radiographs. References obtained via computed tomography. Columns represent the individual dimensions (Figure 5.2). The right-most column represents all dimensions together. All values in millimeters.

Measurements from three radiographs We saw that measurements from only two radiographs achieve very high accuracies. In our next experiments, we tried to add an additional reference radiograph depicting the prism in a rotated pose. We wanted to evaluate whether adding new information could improve the accuracy. We performed $6 \cdot \binom{7}{2} = 126$ measurements in total, $\binom{7}{2} = 21$ pairs of translations per each rotation.

[1] In this context, when we say that a dimensional parameter is "*visible*", we intuitively mean that it influences the loss function. For example, dimensions that are "not visible" do not contribute to the loss function, so the optimization algorithm cannot estimate them correctly.

In Table 5.2, we can see that the mean and maximum absolute errors usually did not decrease. In fact, some of them are even slightly worse. The best results were achieved with rotations 50° and more, which is probably as they brought the most additional information to the optimization, because they represent the side view of the prism. That could also explain why the highest improvement can be seen for $\boldsymbol{\theta}_1$: this parameter corresponds to the cut-out depth, which is much better visible from the side views.

It is worth noting that optimizing with three radiographs is slower as three radiographs have to be simulated per each iteration. In our case, optimizing a single pair took approximately 136 seconds, while optimizing from three radiographs took approximately 206 seconds, which is roughly 3/2 of the former.

We also evaluated the influence of spatial resolution (level of detail). Surprisingly, when using two radiographs, the accuracies are almost the same even from downsampled images, which significantly reduces the optimization time. We discuss the spatial resolution further in Section 5.4.

Parameter:		$\boldsymbol{\theta}_1$	$\boldsymbol{\theta}_2$	$\boldsymbol{\theta}_3$	$\boldsymbol{\theta}_4$	$\boldsymbol{\theta}_7$	$\boldsymbol{\theta}_8$	**Total**
Reference:		6.923	13.020	6.848	6.963	10.438	20.977	
5°	Mean |error|	0.048	0.023	0.072	0.176	0.025	0.017	0.060
	Max |error|	0.080	0.059	0.143	0.258	0.045	0.041	0.258
20°	Mean |error|	0.060	0.053	0.146	0.191	0.039	0.018	0.085
	Max |error|	0.084	0.088	0.206	0.252	0.064	0.033	0.252
35°	Mean |error|	0.027	0.040	0.133	0.198	0.048	0.014	0.077
	Max |error|	0.062	0.084	0.200	0.267	0.098	0.031	0.267
50°	Mean |error|	0.020	0.036	0.035	0.159	0.034	0.011	0.049
	Max |error|	0.078	0.131	0.122	0.250	0.079	0.031	0.250
65°	Mean |error|	0.027	0.044	0.058	0.151	0.032	0.009	0.053
	Max |error|	0.072	0.111	0.128	0.242	0.087	0.022	0.242
80°	Mean |error|	0.023	0.039	0.053	0.160	0.034	0.010	0.053
	Max |error|	0.085	0.128	0.172	0.214	0.061	0.022	0.214
Total mean |error|		0.034	0.039	0.083	0.172	0.035	0.013	0.063
Total max |error|		0.085	0.131	0.206	0.267	0.098	0.041	0.267

Table 5.2: Mean and maximum absolute errors collected from $6 \cdot \binom{7}{2} = 126$ evaluations of Prism A from three radiographs (two translations and one rotation). The values are grouped by the rotation that was used (left column), the total is in the last two rows. References were obtained via computed tomography. The columns represent the individual dimensions from Figure 5.2. The right-most column represents all the dimensions together. All values in millimeters.

5.3 Dimensional measurements of a wedge

In the previous Section 5.2, we evaluated dimensional measurements of prism jaws. Now, we verify that our method is suitable for other shapes as well. As another example, we now evaluate our method on a *wedge* object manufactured from an aluminum block (Figure 5.6).

Unlike the prism jaws, this object is manufactured with a much lower quality with uneven edges and certain defects where a piece of aluminum broke off. On this object, we show how we can calibrate the material from a wedge radiograph directly (as in Section 3.4a), how our method behaves for worse quality products, and how we can use it also for defectoscopy.

Process overview Unlike in Section 5.2, we calibrate the source-material-detector pipeline from a wedge (Section 3.4a). Then, we use this calibration for dimensional measurements of the parametrized object (Figure 5.6b). The reference measurements in this case were obtained using digital calipers with a 0.01 mm resolution, but the measurements were complicated as the wedge is uneven and the accuracy is certainly worse than the caliper resolution.

Wedge radiograph dataset We acquired and manually normalized 17 reference radiographs (Figure 5.7a) depicting the wedge being translated 7.000 mm along the same uncalibrated axis in each radiograph subsequently. Unlike for prism jaws, we did not acquire any rotations as they were not important for our dimensions. Together with these, we also acquired another radiograph (Figure 5.7b) where the wedge is closer to the X-ray source to cover a larger part of the wedge itself. We used this radiograph for our material calibration.

1) Material calibration The calibration was performed on a single reference radiograph (Figure 5.7b) exactly as explained in Section 3.4a and in Figure 3.7. During the normalization, we assumed $x_{\text{min}} = 0\,\text{mm}$ and $x_{\text{max}} = 20\,\text{mm}$, where the latter is the thickness of the whole aluminum block that the wedge was cut-out from. We sampled the intensity table between image horizontal x-coordinates 1575 and 2585. By manually examining the intensity values, we determined the edge beginning to be at y-coordinate 477 (from top) and the end at 3197. Please refer again to Figure 3.7 from Section 3.4a.

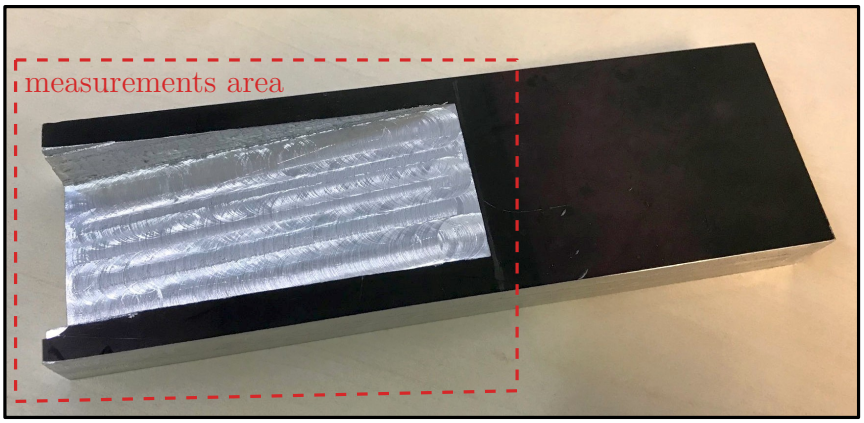
The fitting itself was calculated in Wolfram Mathematica using the built-in function `NonlinearModelFit` for a sum of three exponentials. The result was the following equation (Figure 5.8):

$$I(x) \doteq 0.206 \exp(-0.238x) + 0.715 \exp(-0.074x) + 0.067 \exp(-0.012x)\,. \quad (5.2)$$

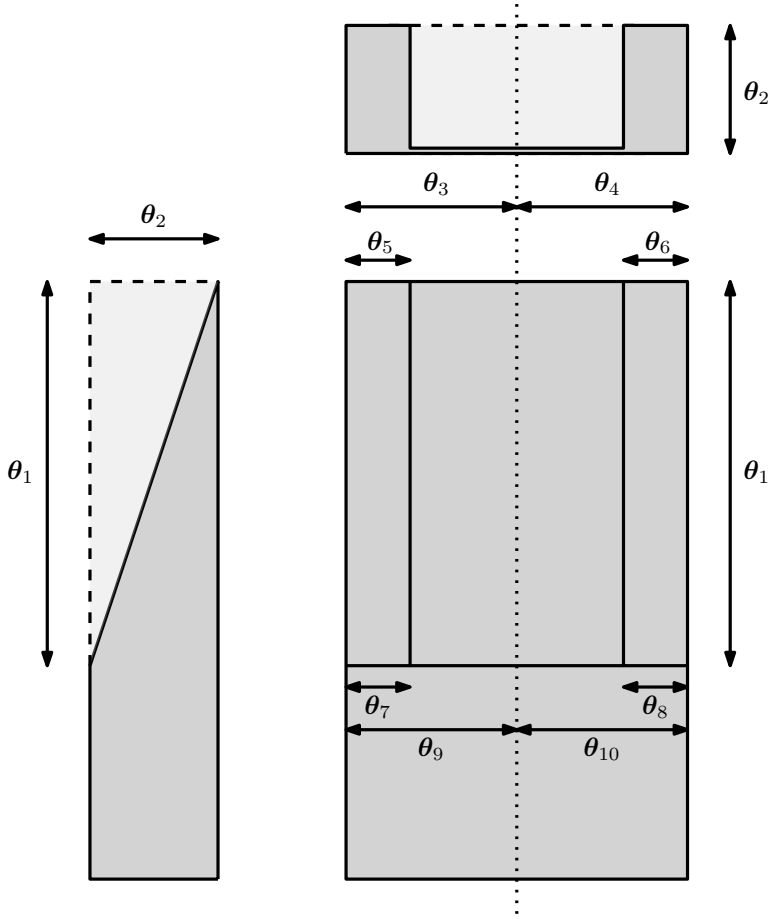
Then we found the x_0 for which $I(x_0) = 1$, which was $x_0 \doteq -0.107$. We then shifted and scaled Equation 5.2 according to Equation 3.7 and we got the following final result (Figure 5.8):

$$I^*(x) \doteq 0.212 \exp(-0.239x) + 0.721 \exp(-0.075x) + 0.067 \exp(-0.012x)\,. \quad (5.3)$$

Notice that it holds $I^*(0) = 1$ and $I^*(20) = I(20)$, which is exactly what we wanted to achieve by shifting and scaling the intensity function.

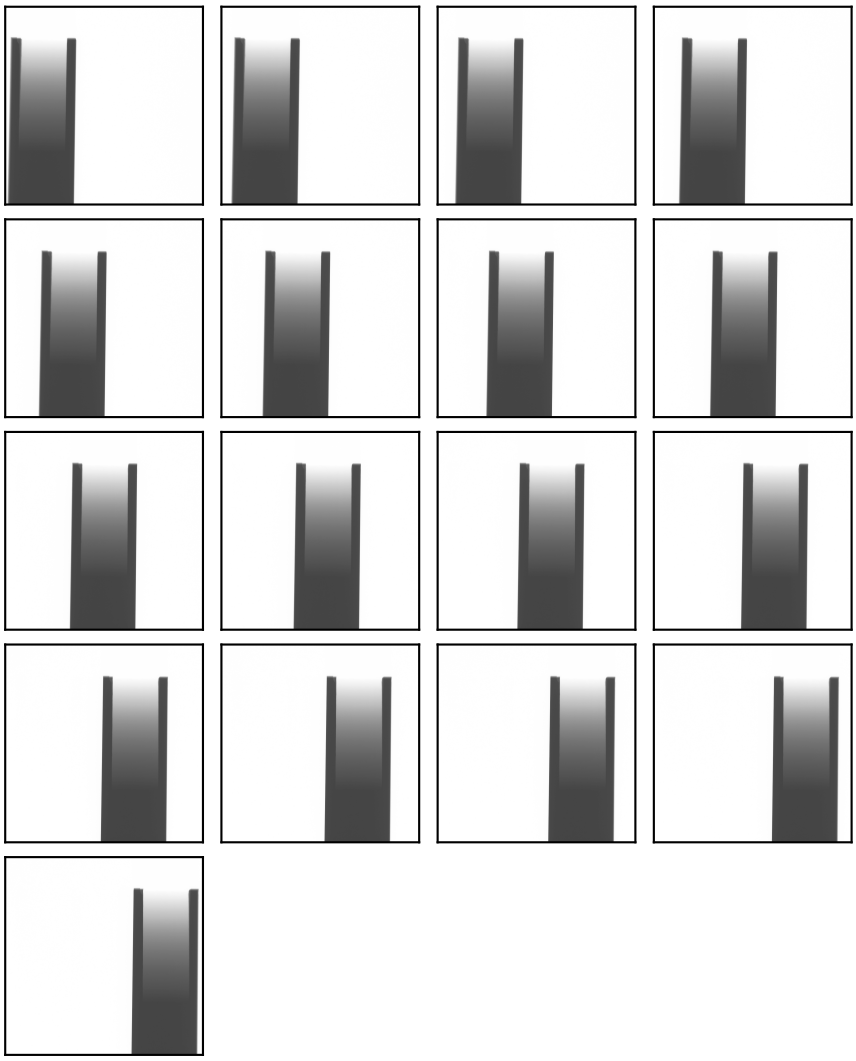


(a) Photo of our wedge object. We are interested in dimensional measurements of features inside the red area, i.e., the wedge itself, not the rest of the aluminum block.

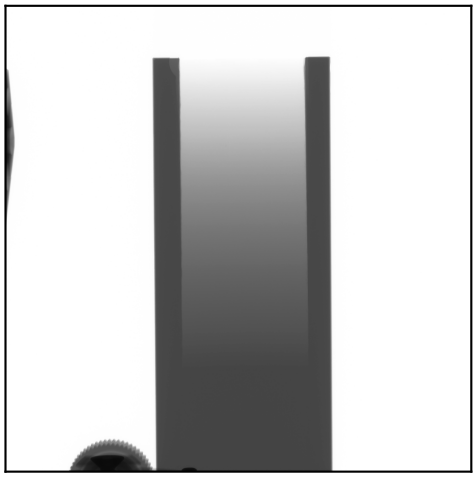


(b) The dimensional parametrization $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{10}$. Note that the width parameters at the top $(\boldsymbol{\theta}_3, \ldots, \boldsymbol{\theta}_6)$ are duplicated in the middle $(\boldsymbol{\theta}_7, \ldots, \boldsymbol{\theta}_{10})$, which is because the edges are not perfectly parallel, so the object is not a perfect rectangular block, which has to be taken into account in the parametrization (compare to Figure 3.3).

Figure 5.6: Wedge.



(a) Reference radiograph depicting subsequent 7.000 mm translations.



(b) Reference radiograph for material calibration. The wedge is moved closer to the X-ray source so that a larger part is covered. The circle at the bottom is a part of the manipulator irrelevant to the measurements.

Figure 5.7: Wedge reference radiographs dataset.

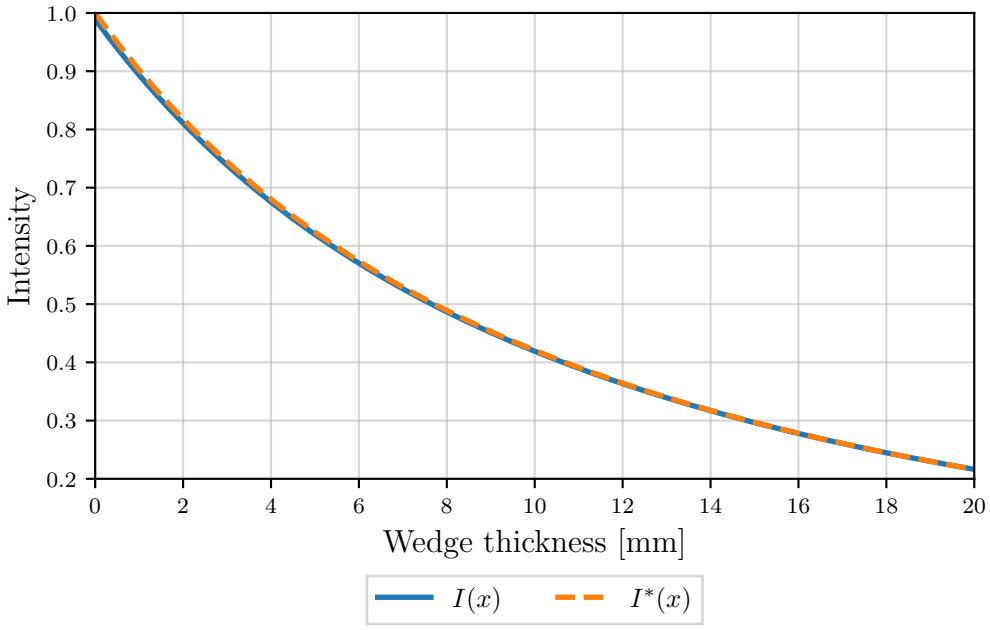


Figure 5.8: Calibrated material intensity functions $I(x)$ and $I^*(x)$.

2) Dimensional measurements Similarly to the prism jaws (Section 5.2), we now use the calibrated material from Equation 5.3 for dimensional measurements using the hierarchical optimization from Section 3.8. However, there are a few differences from the prism jaws. The first difference is that we now measure the same object which we used for the material calibration. And the second is that the wedge was manufactured with much worse quality, which affects our accuracy.

In Table 5.3, we can see our mean and maximum absolute errors obtained from $\binom{11}{2} = 55$ radiograph pairs in total. The reference measurements were not obtained by computed tomography, rather using a simple digital caliper, and they are only approximate because of the low manufacturing quality.

As we can see, the best accuracy was obtained in the middle width parameters represented by the sum $\boldsymbol{\theta}_9 + \boldsymbol{\theta}_{10}$. That is, again, most likely because the width is clearly visible in the radiographs. However, the top width $\boldsymbol{\theta}_3 + \boldsymbol{\theta}_4$ has much lower accuracy, which is likely caused by the manufacturing defects present in the top area (Figure 5.9). For example, a piece of aluminum is chipped off in the top-left corner, which gives the optimization algorithm incorrect intensity data. Furthermore, the right edge is not smooth, which leads the optimization to incorrect pose estimations (it estimates the wedge slightly rotated around the vertical axis, because that gives a lower image difference), which in turn results in poor dimensional estimates of the edge widths as well (parameters $\boldsymbol{\theta}_5, \ldots, \boldsymbol{\theta}_8$).

3) Defectoscopy Since we have fitted simulations together with reference radiographs, it is possible to use the difference images to detect defects unrelated to dimensional measurements. This is what Dael et al. [2017] (Section 2.3) used for classifying horticultural products into bad (with internal defects such as cavities or mold) and good (without defects). In our case, we can, for example, spot the chipped-off part of the wedge in the top-left corner (Figure 5.9).

Parameter:	$\boldsymbol{\theta}_3 + \boldsymbol{\theta}_4$	$\boldsymbol{\theta}_5$	$\boldsymbol{\theta}_6$	$\boldsymbol{\theta}_7$	$\boldsymbol{\theta}_8$	$\boldsymbol{\theta}_9 + \boldsymbol{\theta}_{10}$	**Total**
Reference:	59.24	8.80	8.40	9.20	8.10	59.35	
Mean |error|:	0.34	0.16	0.28	0.23	0.29	0.14	0.24
Max |error|:	1.36	0.71	1.10	1.16	0.90	0.99	1.36

Table 5.3: Mean and maximum absolute errors collected from $\binom{11}{2} = 55$ evaluations of the wedge from two radiographs (4th to 14th). References are only approximate and were obtained via a digital caliper. The columns represent the individual dimensions from Figure 5.6, some of them are summed together as we could not measure them individually by the caliper. The right-most column represents all the dimensions together. All values in millimeters.

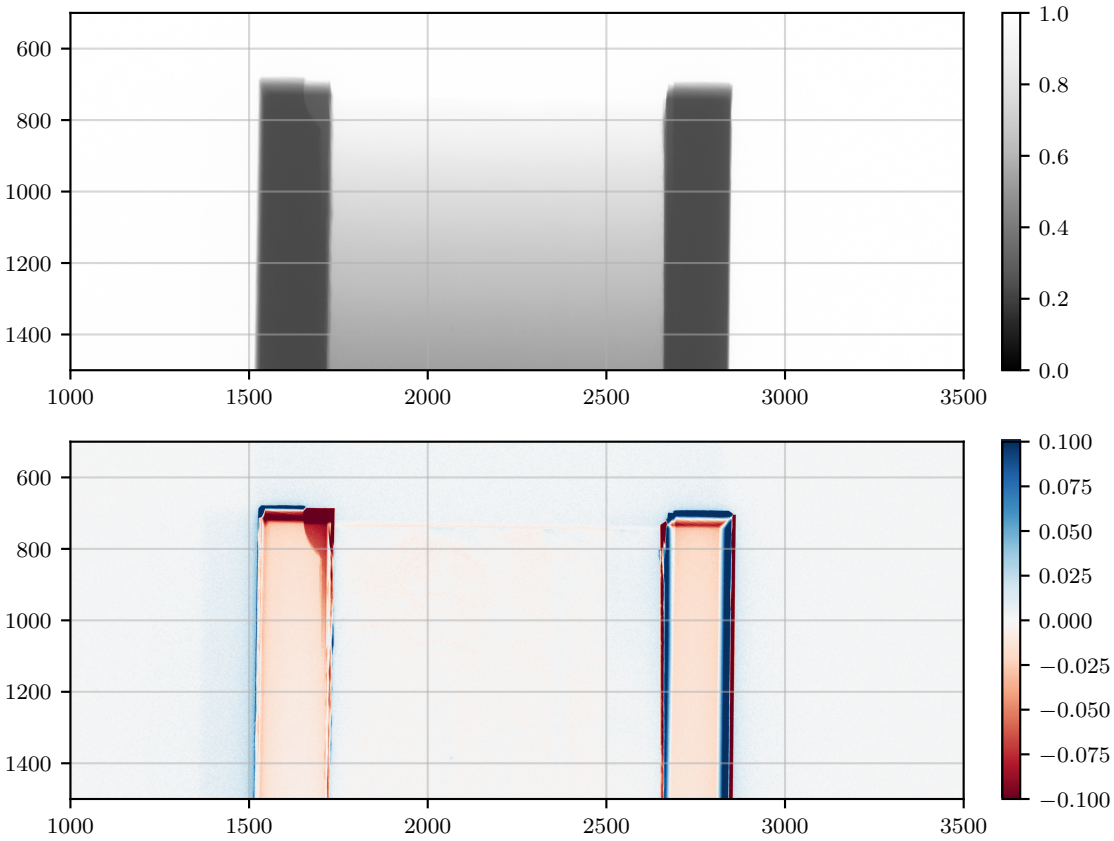


Figure 5.9: The zoomed-in wedge reference radiograph (top) and the per-pixel difference to our fitted simulation (bottom). Notice that a part of the wedge has chipped off in the top left corner, which can be clearly seen in red color in the bottom image. Other defects can be seen at right, where the edge is not smooth, which caused our optimization to misalign the simulated radiograph. Notice that the wedge inner parts are fitted correctly from the calibrated material. Image axes are pixel coordinates.

5.4 Limitations caused by the setup

In the previous sections, we saw how accurate our method is. But note that all dimensional measurements that our method performs are based solely on the *inputs* that we give to the method, mainly the reference radiographs that we acquired. We already know that X-ray setups are never perfect and that the radiograph quality is limited by physics and the available technologies (Chapter 1 and mainly Section 1.5). Let us now mention the most important limitations of our method caused by the X-ray technologies themselves.

This section is motivated mainly by Figure 5.10. There, we can see one-dimensional cross-sections through the prism jaws and we can compare the reference radiographs to our simulations. These visualizations are extremely valuable as they provide a visual guide to our following discussion.

Noise One of the first things one can notice is the noise present in the reference data. Note that the data in Figure 5.10 are averaged over 11 pixels in the vertical direction, so the noise is already much reduced. To see the original amount of noise, refer to Figure 5.11, which depicts noise in a flat white area. Notice that our simulations are noiseless as we use deterministic ray tracing with a single sample per pixel and we do not take any scattering processes into account that would require stochastic simulations.

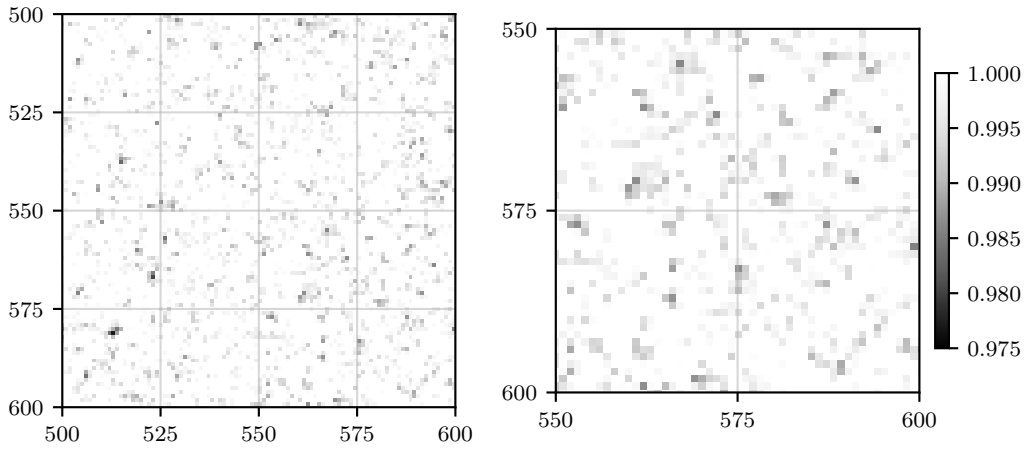
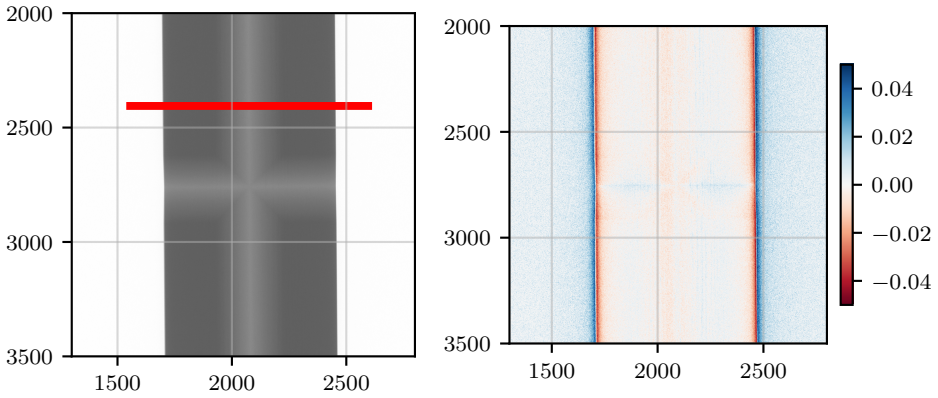


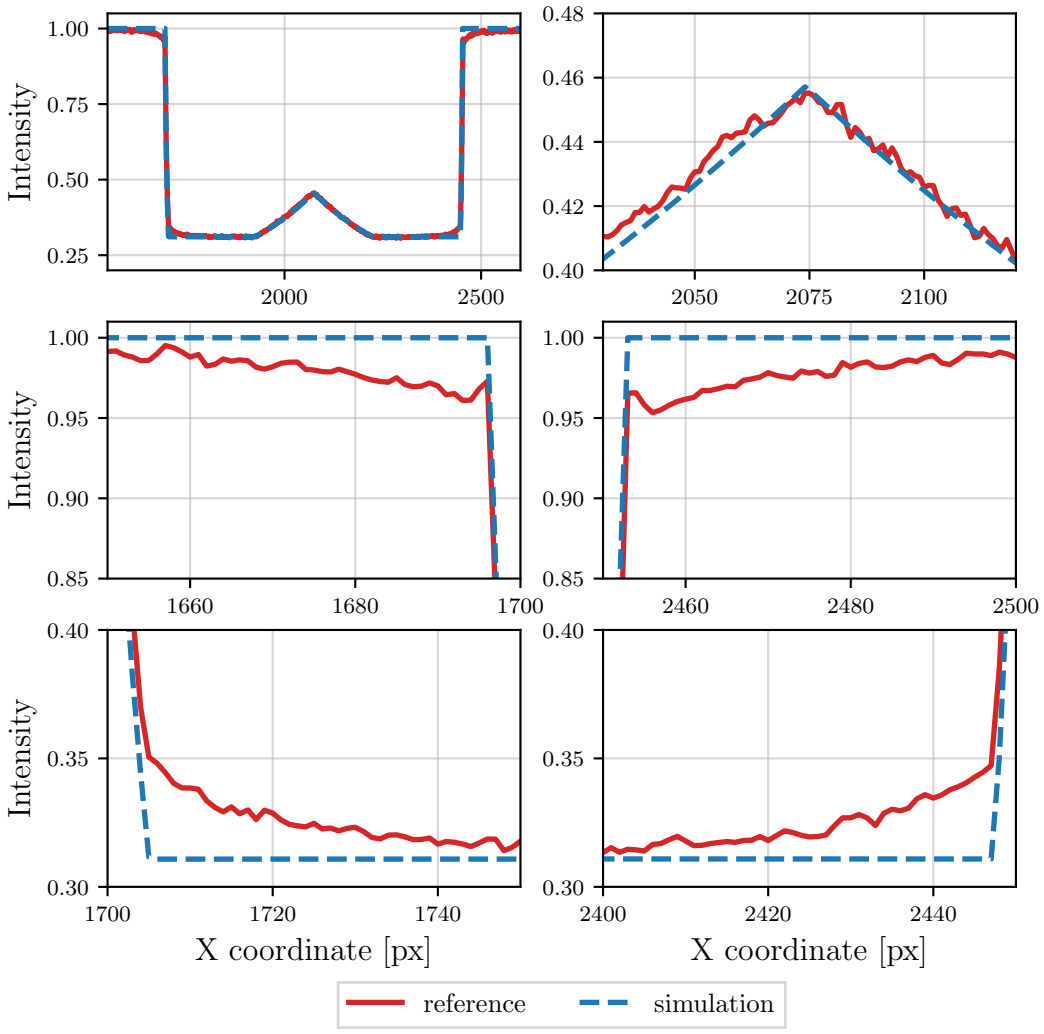
Figure 5.11: Noise present in a normalized reference radiograph in an area that should be completely white. Image axes are pixel coordinates. The colors are clamped between 0.975 and 1.000, the darkest pixel has intensity 0.977.

Practically, the amount of noise can be reduced by longer exposures and averaging the data, but that of course prolongs the image acquisition time. Generally, noise can also be reduced in post-processing, but we wanted to avoid any image post-processing except normalization as that could introduce bias and inaccuracies in the measurements that we would have to evaluate as well.

Paradoxically, the presence of noise may contain additional useful information about the signal. Let us show an example for a hypothetical 2-bit detector whose only possible pixel intensity values are 0 and 1. Remember that the pixel intensity (gray level) is just the charge from the incoming X-ray photons accumulated and



(a) Prism B reference radiograph (left) and the per-pixel difference to our simulation (right). Image axes are pixel coordinates. The red area in the left image corresponds to the cross sections in (b) below.



(b) One-dimensional cross sections of the red area from (a) showing the reference signal and our simulation (both averaged over 11 pixels vertically). Top-left image shows the overall view, the remaining images are zoomed-in to specific parts across the x-coordinates. Notice that our simulation almost perfectly matches the reference signal except for the noise and "smoothing", see text for discussion.

Figure 5.10: Prism B reference signal in comparison to our simulation.

quantized over the physical pixel area. Think of a 2×2 pixels square and incoming X-ray photons with the total intensity 0.25. If there was no noise present, the detector would simply detect 0. But as quantum noise occurs, it is possible that one of the pixels would go above the threshold and be detected with the intensity 1. Hence, the 2×2 pixels area would contain intensities 0, 0, 0, and 1 that average to 0.25. Of course, this is not a proof or statement that noise is useful, we just wanted to mention that when noise is accumulated over larger regions, it may not harm our method. Notice that our optimization algorithm optimizes the total loss summed-up through the whole radiograph and the dimensional features that we measure are typically present in areas bigger than one pixel.

Focal size Another physical property that we discuss is the X-ray source focal spot diameter that may result in geometric blur. For a general discussion about this issue, please refer back to Section 1.5. According to our X-ray source manufacturer, the reported focal size of our source is lower than 6 µm, which should be practically unnoticeable.

To verify that the geometric blur caused by the focal size is negligible, we implemented stochastic supersampling into our ray-tracer. For each pixel sample, we randomly sample the position on the X-ray source using Gaussian distribution and we also randomly sample the pixel position within the pixel area. In Figure 5.12, we can see that for 2048 random samples per pixel, the geometric blur is, indeed, insignificant for focal sizes in the range of a few micrometers.

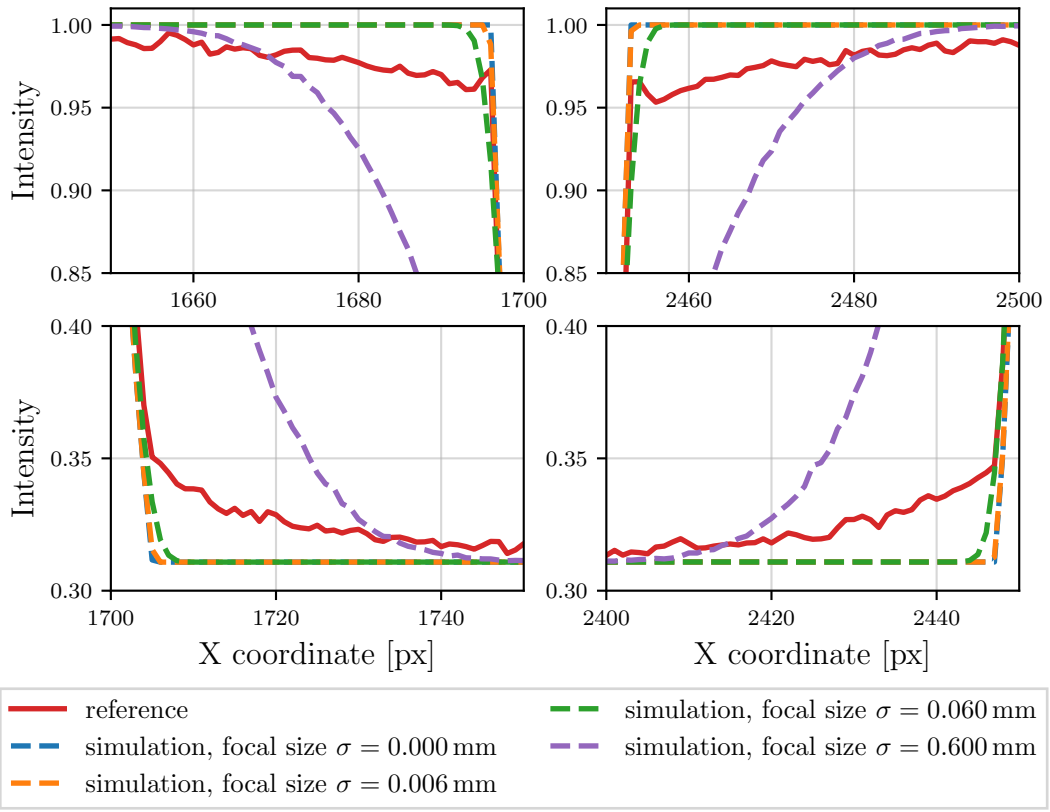


Figure 5.12: Effects of various focal spot diameters simulated by randomly sampling the source by a Gaussian distribution with different deviations σ. Notice that in the range of several micrometers, the geometric blur is insignificant, which is the case of our X-ray source.

Detector scattering Much more significant than the focal spot diameter is probably the scattering happening inside the detector itself. We have an indirect flat-panel that contains a scintillator layer where scattering of the transformed visible light is occuring as we discussed in more details in Sections 1.4 and 1.5.

In Figure 5.10, it is highly noticeable that the reference signal is partially "smoothed", i.e., the signal intensity decreases or increases in large neighborhoods around sharp edges. Edges smoothing can, of course, be a result of the focal size, but we already saw that in our case, this effect is negligible, and also it cannot influence such large areas. Furthermore, notice that the edges are still present in the reference signal, the edges themselves are not smoothed, only their neighborhoods. The effect is highly likely caused by the detector scattering.

To verify our statement, we implemented a post-processing simulation of this effect. We assume the scattering to be Gaussian with an unknown deviation. But just blurring the whole image is not correct as that would also blur the edges themselves. Rather, we must assume that the Gaussian blurring is additive. Hence, we propose to linearly *weight* it together with the original signal. In our simulation, we calibrated (optimized) the parameters using Prism B and the results of our simulation can be seen in Figure 5.13. We assume the Gaussian to have a deviation $\sigma = 500\,\text{px}$ and weight $w = 0.139$, so the blurred image is:

$$\mathsf{I}^* = 0.139 \cdot \mathsf{G}(\mathsf{I}, \sigma = 500) + (1 - 0.139) \cdot \mathsf{I}, \tag{5.4}$$

where I is the original ray-tracing simulation and G represents Gaussian filtering.

By simulating the detector scattering, we can significantly lower the total loss (difference) between the reference and simulated image. Unfortunately, computing the blurring with such large standard deviations is very slow even with a separable convolution filter. Rendering the image with the scattering is about 400 times slower than standard rendering, so it becomes unsuitable for our method and we did not use it for dimensional measurements. Later, in our Conclusion and Future work, we discuss the possibility of pre-processing the reference radiographs to remove the detector scattering instead of simulating it in post-processing.

Image resolution Our detector image output size is $4096 \times 4096\,\text{px}$. To discuss if the resolution is sufficient for the dimensional measurements, we offer the following example. In our reference radiographs, Prism A is roughly 750 px wide. In physical dimensions, the prism is roughly 21 cm wide. That means that one pixel corresponds to roughly 0.028 mm of the prism front side.

However, in Section 5.2, we saw that we can estimate the width $\boldsymbol{\theta}_8$ with the accuracy of 0.013 mm, which is much better than the pixel resolution. Furthermore, we verified how the accuracy decreases when we downsample the reference radiographs to lower resolutions of $2048 \times 2048\,\text{px}$, $1024 \times 1024\,\text{px}$, $512 \times 512\,\text{px}$, or even $256 \times 256\,\text{px}$. The $\boldsymbol{\theta}_8$ mean absolute error got only slightly higher to 0.016 mm, 0.017 mm, 0.024 mm, and 0.043 mm, respectively. How is that possible?

We have to take the redundancy into account. Notice that the prism covers a very large area and our method is fitting the whole prism with regards to the whole radiograph, not just a single line or pixel. Hence, changing the width by even a very small value still influences the total loss in the whole image. As a result, our method can estimate dimensions with high accuracies even on

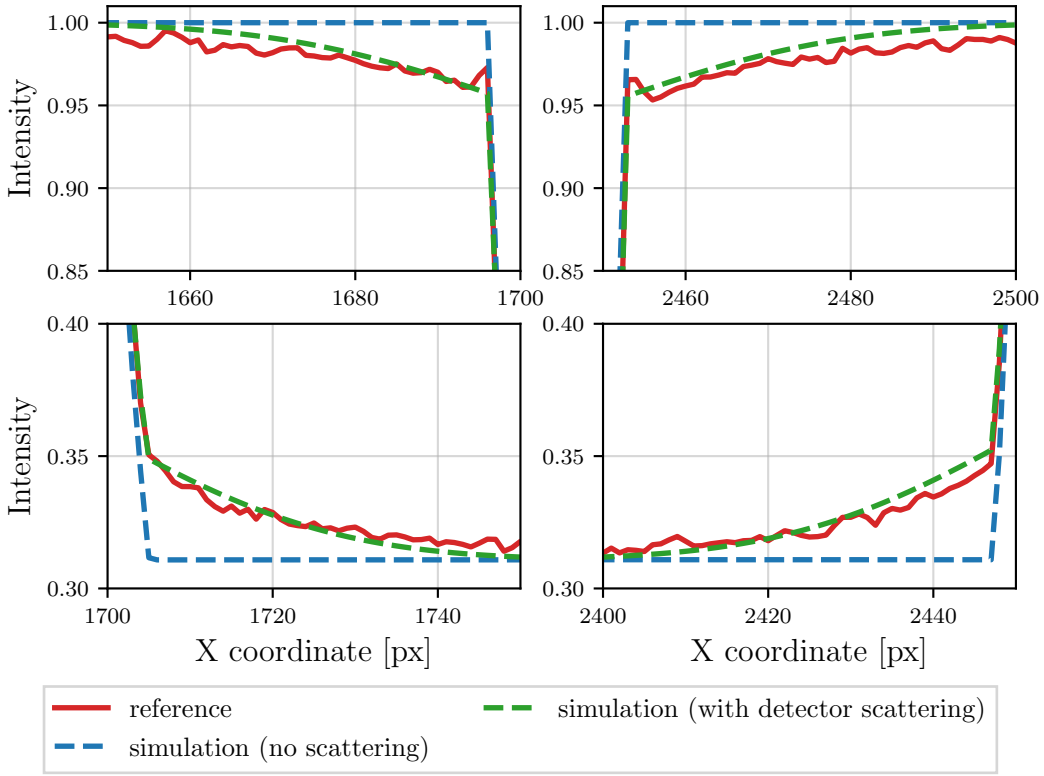


Figure 5.13: Detector scattering simulated by Equation 5.4. Notice that the simulation almost perfectly matches the reference signal, which proves that the "shadows" around objects are, indeed, most likely caused by the scattering. The parameters were partially optimized from Prism B. In fact, the σ could probably be even higher, but that would slow the fitting even more, so it was not tested.

seemingly insufficient image resolutions, which we also verified on the remaining dimensional parameters, not only $\boldsymbol{\theta}_8$.

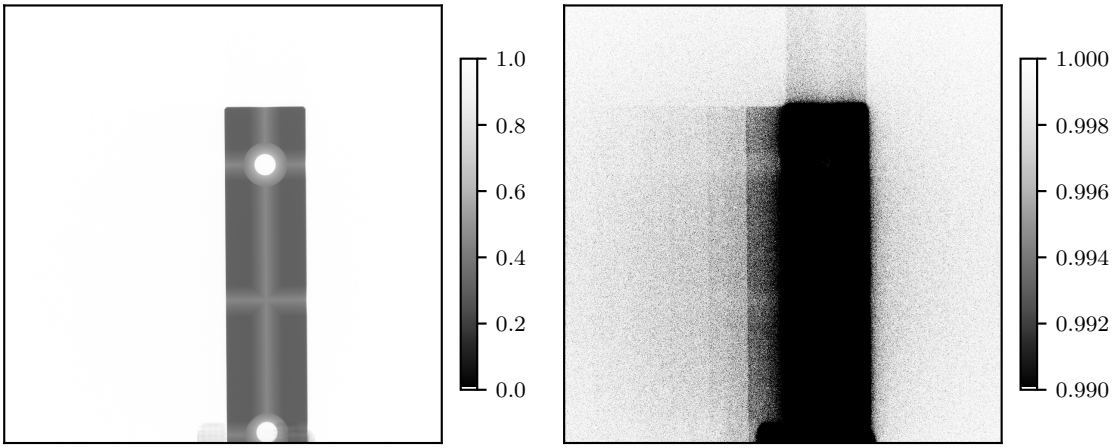
Gray-level resolution Our detector is 16-bit, so the output signal has up to $2^{16} = 65536$ distinct intensity values. In reality, we cannot benefit from the whole range, because the signal black level is not exactly zero, but roughly from around 3000 to 5000 for our detector. Furthermore, the signal is normalized by the white average, which again slightly impacts the resolution.

For our measurements, the gray-level resolution seems to be perfectly sufficient. For our aluminum materials, the normalized intensity was roughly 0.2 for thicknesses around 20 mm. That means that if there were 60000 distinct values between 0.0 and 1.0, that would be 48000 values between 0.2 and 1.0, so without taking the exponentiality into account, we could recognize thicknesses with resolutions even better than 0.001 mm. Of course, since the intensities are exponential, the resolution gets lower when passing very thick materials, which must be taken into account when measuring thick objects. Notice that in our measurements, the thicknesses were at maximum around 20 mm.

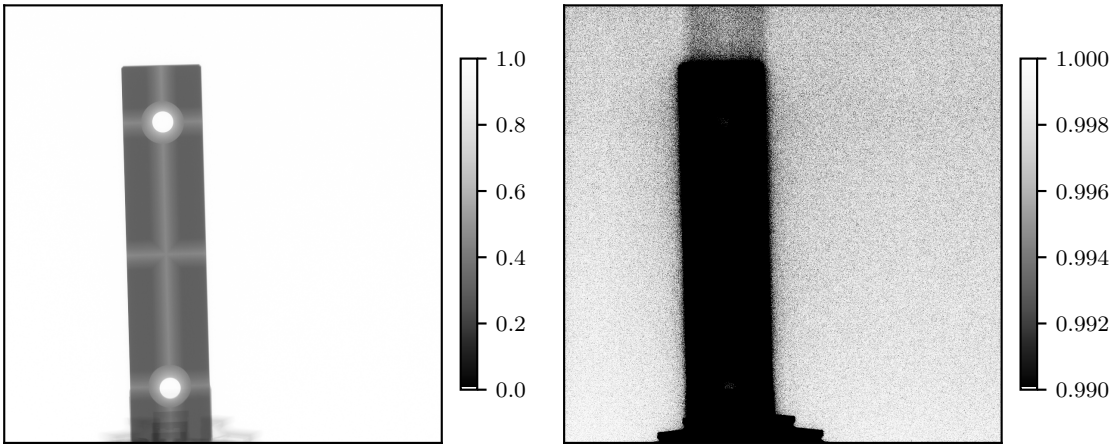
Ghosting and other artefacts Finally, we briefly discuss the ghosting (image lag) and other artefacts previously mentioned in Section 1.5. Examples are shown in Figure 5.14, where we can mainly see that ghosting really does occur in real

detectors. In Figure 5.14b, we specifically tried to remove the ghosting artefacts by turning the setup off between the projections, which of course significantly prolongs the acquisition time. Practically, we think that the ghosting does not influence the dimensional measurements in any significant way and we performed all our measurements on regular radiographs *with* the ghosting. First of all, it can be clearly seen that the ghosting intensity is rather low and much lower than, for example, the detector scattering (compare to Figure 5.10). And then, notice that in our prism radiographs depicting the translations, the ghosting only appears on the left side of the prism, which could only affect fitting the prism width, where we achieve accuracies comparable to computed tomography.

Furthermore, in Figure 5.14, we can also see slightly darker areas above the prism. We believe they may be caused by correlations in the detector columns, which may happen for example because of crosstalks in the read-out electronics.



(a) Prism B radiograph taken in a sequence of translations. When we rescale the colors (right), we can clearly see the ghosting (image lag) from the previous translations on the left side of the prism. Furthermore, we can see also see a slightly darker area above the prism, which may be caused by correlations or crosstalks in the read-out electronics.



(b) Here, the radiograph was taken after turning off and "cooling down" the setup for a couple of seconds. As we can see, the ghosting has disappeared, but the dark area above the prism remained, which supports our theory.

Figure 5.14: Detector ghosting and other artefacts.

5.5 Limitations caused by our method

In the previous section, we discussed the limitations caused by the reference radiographs quality, which is influenced by the whole X-ray setup. Now, we discuss the limitations that our method would have even if the input radiographs had an absolutely perfect quality.

Calibration Our method requires calibrating the material in the source-detector pipeline (Section 3.4). The calibration is tied to a specific material in a specific polychromatic spectrum and also takes the detector characteristics into account. That means that if we had multiple X-ray machines or even the same machine but with different configurations, such as different source voltages, we would have to calibrate each of these situations individually with regards to the same material. The good thing is that the calibration is performed semi-automatically and only requires one reference object. In our case, we always used the same machine and voltage to take the radiographs, so we only required one calibration.

We showed two ways in which the material may be calibrated, but both have their own limitations. The first method requires us to manufacture a reference wedge from exactly the same material as the measured objects. The problem is that manufacturing an ideally smooth wedge whose thickness linearly decreases towards zero is not a trivial operation. It is possible to buy calibration wedges, but these are usually step-wedges, i.e., they are not smooth, but instead they have steps of constant thicknesses, so we would have to interpolate the intensities between the steps. Also, these wedges are very expensive and they may not be manufactured from exactly the same material that we want to measure.

The second method uses an arbitrary parametrized object. The limitation is that we first need to measure the reference object dimensions with very high accuracy. That essentially means that we must first use computed tomography on this object before we can use our own method for measuring other objects of the same material. Again, the reference object should be manufactured with as high quality as possible. However, in this thesis, we proved than even reasonably accurate prism jaws can be used for a successful calibration.

Inhomogeneous materials Our simulations assume that the areas inside object boundaries are filled with a homogeneous material, i.e., we assume that the attenuation properties are uniform within the whole volume. While this assumption was reasonably valid for our aluminum test objects, it may not always be true for all materials. Important examples are wood [Jordan et al., 1998] and composite materials that are neither homogeneous nor isotropic [Amir et al., 2019]. Composite materials such as carbon-reinforced fiber plastics or glass fiber-reinforced aluminum are made by combining multiple materials with different properties with boundaries still existing between them.

The problem with inhomogeneousity is that we have no way of knowing how exactly the material properties are distributed in the volume, which causes a problem for our simulations. Computed tomography, on the other hand, estimates the attenuation properties in each voxel separately, so it can be successfully used even for scanning composite materials [Garcea et al., 2018].

Multi-material objects Dimensional measurements of objects consisting of multiple materials is also something which our current method does not support. Same as with inhomogeneous materials, the problem is with the attenuation properties not being consistent throughout the whole volume. But unlike inhomogeneousity, this problem may be easier to solve.

We would have to decide on an object representation that would contain information about the material boundaries, e.g., we could model the individual materials separately, then render them individually via the ray tracing, and finally merge the attenuations along the paths. Of course, evaluations would have to be performed whether such simulations would be plausible for dimensional measurements. That is beyond the scope of this thesis.

Parametrizing and triangulating complex shapes Another limitation of our method is that it requires parametrizing and triangulating the object shapes. That may be problematic for spherical or circular features where accuracy may be lost during the process.

Practically, the triangulation step could be replaced for example by constructive solid geometry (CSG), which is an alternative representation for modeling complex shapes by combining simple primitives via various operations [Stewart, 2008]. In that case, we could easily represent spheres and circles or even multiple materials without any triangulation.

On the other hand, for simple models such as the prism jaws or the wedge that we evaluated in our thesis, triangulations are easy to achieve and provide good performance. That is because we can avoid sorting the intersections, which would be problematic with various CSG operations. Also, new versions of the OptiX ray-tracing framework can find ray-triangle intersections very fast due to hardware acceleration in modern GPUs.

Stochastic X-ray photons scattering In Sections 2.4 and 3.6, we have already discussed the various rendering approaches and why we decided to use attenuation-only ray tracing. Of course, by using such a deterministic simulation, we do not simulate the fact that X-ray photons flying towards a certain pixel A may be scattered on their path towards a different pixel B. Practically, as we could see, we can still achieve accurate dimensional measurements even without taking this effect into account. Vidal and Villard [2016] compared their attenuation-only renderer with a stochastic Monte-Carlo particle-physics simulation and they found out that the normalized cross-correlation between the images is higher than 0.997. In our opinion, it is caused by the fact that X-ray photons scatter either via Rayleigh or Compton scattering (Section 1.2).

Rayleigh scattering changes the photon direction and the energy remains the same. But notice that in aluminum, the Rayleigh scattering coefficient significantly decreases for photon energies above 10 keV (Figure 1.4, remember that both axes are logarithmic) and our detector filters energies up to around 40 keV.

Compton scattering, on the other hand, changes the photon direction but it also *decreases* the photon energy during the process. But since the photon now has lower energy with each Compton scattering event, the probability that such photon will get absorbed by the photoelectric effect increases (again, consider Figure 1.4) or, again, it may get filtered by the detector. Hence, notice that

when the photons really do scatter, they have very low probabilities of actually reaching the detector at any pixel.

Rendering artefacts Our ray tracer is based on ray-triangle intersections and Equation 3.11. Unfortunately, in specific situations, such ray tracing may cause rendering artefacts that manifest themselves as black or white pixels inside an object (Figure 5.15). The issue was previously described for example by Vidal and Villard [2016], even though they used a rasterizer instead of ray tracing, so the issue had a slightly different cause.

In our case, the problem is caused by non-watertight ray-triangle intersections [Woop et al., 2013]. That means that occasionally, an intersection may be missed and there is a microscopic "crack" inside the object surface. When that happens in our X-ray tracing, we incorrectly miss the fact that an object was entered or exited. That means that the accumulated distance x through the object is miscalculated and is either too high (black pixel, we forgot to exit the object) or too low (white pixel, we forgot to enter the object).

Practically, we can avoid these artefacts either by implementing watertight intersections as described for example by Woop et al. [2013], but that is not trivial. Or, we can detect these pixels and replace them by the averages from neighboring pixels. In our case, we did not work around the issue in any way because the bad pixels usually only appear in very specific object poses and they disappear with even a slight pose change (Figure 5.15). Hence, these artefacts manifest themselves as loss function noise that should not affect the robust local optimization algorithms that we use.

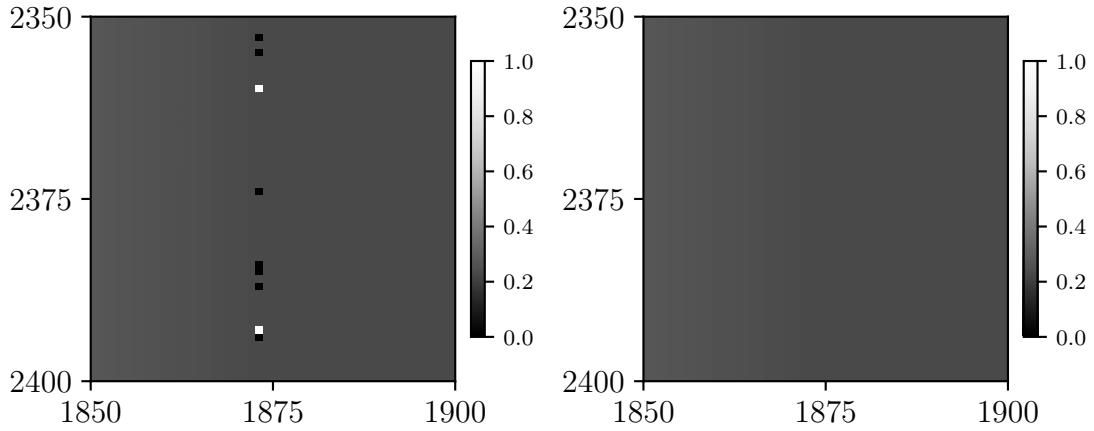


Figure 5.15: Zoomed-in simulated radiographs, image axes are pixel coordinates. The left simulation was specifically chosen to contain bad pixels (see text for explanation). When the object was slightly rotated by 0.01°, the bad pixels disappeared (right image), which shows that they are caused by numerical errors in the intersection calculations (the line along which the bad pixels appear is a boundary between two triangles).

Optimization parameters Finally, we mention that our method requires manually setting a few parameters for the optimization process. First of all, it is necessary to define the lower and upper bounds of all optimization parameters, which is necessary for the gradient-free methods. For example, if we knew that

our object is not visible in the radiograph anymore if its x-position is lower than $-60\,\mathrm{mm}$, we can safely use this value as the lower bound for the x-position, and so on. Using tighter bounds is helpful for the optimization process to make sure that it does not diverge from the correct solution (Section 3.8). Fortunately, it is usually plausible to only set the lower and upper bounds once for multiple radiographs. We do not have to set these values individually all the time.

For some parameters, we also have to set their initial estimates, which we already discussed in Section 3.8. In case of pose estimation, we solved this problem by designing our hierarchical optimization that finds the initial pose by sampling the parameter space. Hence, pose estimation happens fully automatically within the lower and upper bounds. For the material optimization, we explained how the initial estimate can be computed in Section 3.4b, and that only has to be done once. For dimensional parameters, we typically roughly know the expected dimensions, so we can use these as initial estimates. During our experiments, we verified that slightly changing the initial dimensions does not influence the final accuracy in any substantial way.

And finally, we have to set the stopping conditions, in our case, the maximum number of iterations or a time limit, which we also discussed in Section 3.8. For our dimensional measurements of the prism jaws and the wedge, we always used 5000 iterations as a stopping condition and we found this value to be perfect for these objects. The only exception was material calibration, where we stopped the process after 10000 iterations. For different objects, it might be necessary to increase the limit in case it improves the accuracy, or it may be beneficial to decrease the limit for faster measurements.

Conclusion

In this thesis, we tackled an important quality control problem. Our main goal was to propose, fully implement, and thoroughly evaluate a new method that would allow dimensional measurements from only a few X-ray projections. Inventing such solution would solve one of the main computed tomography limitations: the requirement to acquire hundreds or thousands of projections. Let us now summarize our contributions and verify that we have fullfiled all our goals.

At first, in Introduction, we briefly discussed the manufacturing processes and measurement approaches. We followed by explaining the necessary physical and technical background in X-rays and radiography in general (Chapter 1). Then, we examined the existing and related solutions and we saw that even in the recent state-of-the-art research in computed tomography, tens and hundreds of projections are still necessary (Chapter 2).

Our main contribution is proposing our novel approach that does not depend on computed tomography at all. Rather, we formulate the dimensional measurements as a minimization problem using prior knowledge. We estimate such dimensional parameters that minimize the difference between our parametric model simulations and reference radiographs (Chapter 3). We carefully designed and explained the individual components required for the method to work.

Namely, we proposed how the measured objects can be efficiently parametrized, how to calibrate their materials without any complicated measurements, how to easily take and parametrize reference X-ray projections, how to simulate radiographs in a scalable and GPU-accelerated way, and finally, how to perform the minimization itself in a hierarchical way to ensure it converges to accurate results.

We have successfully implemented the whole method in a prototype demo application (Chapter 4). We performed an exhaustive evaluation with real radiographs and real physical objects, we verified the method accuracy and repeatability, and thoroughly discussed all limitations (Chapter 5).

Our results successfully show that our proposed method could provide significant benefits for industrial quality control. When measuring prism jaws, one of our test objects, from only two and three radiographs, we achieved the mean absolute error in the range from 0.013 mm to 0.172 mm on different features, and the maximum absolute error from 0.024 mm to 0.267 mm in comparison to computed tomography. These results were also consistent when verified on more than one hundred combinations of input radiographs. We also evaluated our method on a different object shape, but unfortunately with worse results as that object was manufactured with much lower quality than the prism jaws.

The major advantage is that unlike hundreds or thousands of projections for computed tomography, acquiring two or three radiographs with different translations is possible in a couple of seconds. That means that the X-ray machine time is significantly reduced. Hence, the same machine can be used for quality control in more production lines or for other purposes, which can lower the total quality control costs. Furthermore, it reduces the time in which any defects or manufacturing errors are detected, so errors can be corrected much faster leading to fewer rejected products. That, again, lowers the total production costs.

Future work

The dimensional measurements problem is not easy to solve. Especially relying on only two or three radiographs is heavily underdetermined and requires using prior knowledge such as carefully parametrizing the object and acquiring accurate radiographs. While our method achieves high accuracies for our test objects, it is of course not perfect and we have thoroughly discussed some of the limitations in Sections 5.4 and 5.5. There are still problems that remain to be solved during the possible future work as they were already beyond the scope of this thesis. Let us now briefly mention a few of them.

Differentiable renderer While this is not really a limitation, it would be very interesting to propose and implement a differentiable X-ray renderer. In our method, we work with a ray tracer with unknown gradients with regards to the optimized parameters. We got around this problem by using gradient-free optimization methods (Section 3.8). However, if we had a differentiable renderer, we could probably achieve much faster optimizations as the algorithms would use pre-computed gradient information. So far, we do not know of any differentiable X-ray renderer, but such renderers were already researched for standard rendering [Loper and Black, 2014, Palazzi et al., 2018] and they have its usage in deep learning [Palazzi et al., 2018].

Multi-material objects We already discussed the problem of measuring multi-material objects in Section 5.5. Generally, our method only supports objects with a single homogeneous material. However, proposing a multi-material representation and simulation could be important for quality control of whole products where different materials are combined. It would be interesting to evaluate if these objects could be rendered by simply ray tracing the materials individually and accumulating the attenuations.

Complex geometries and constructive solid geometry A problem partially linked to the one above is parametrizing complex geometries, which was also discussed in Section 5.5. We believe that proposing and implementing a constructive solid geometry (CSG) ray tracer for X-rays could bring the benefit of supporting complex spherical or circular geometries, or internal features such as circular holes that are complicated to triangulate (many vertices would have to be recalculated every time the parameters change). More details about CSG primitives and operations are available for example in Stewart [2008], Chapter 2. Furthermore, CSG could also help parametrize multi-material objects as the boundaries between materials would be calculated from the CSG operations.

Of course, implementing complex geometries is only a part of the problem. Thorough evaluations would be necessary to verify the dimensional measurements accuracies for such geometries. In our method, we only used relatively simple shapes and it is not clear how well the method would scale with geometry complexity and an increasing number of optimization parameters.

Removing detector scattering The detector scattering that we discussed in Sections 1.5 and 5.4 is significant in our indirect flat-panel detector. It would

be interesting to evaluate how such scattering influences the accuracy. Unfortunately, post-processing the rendered radiographs by Gaussian blurring is very slow, so in our method, we perform all optimizations on naïve simulations without taking the scattering into account. Instead of post-processing the renders, it would be an interesting future research to propose a way to remove the scattering directly from the reference radiographs.

Evaluation in a real manufacturing process When evaluating the dimensional measurements of prism jaws (Section 5.2), we used two prisms, one reference for the material calibration, and one with different and unknown dimensions. In a real manufacturing process, however, there may be not two, but thousands or even more objects of the same shape produced every day. Evaluating our method on thousands of objects was, of course, impossible in our research conditions and limited timeframe. However, it would be a very important future work to implement an automated demo application for a real industrial process and evaluate the accuracies achieved in real factory conditions.

It would be especially interesting to measure the real time required for a human operator to acquire the radiographs, i.e., place the measured object in a manipulator, take a radiograph with a fixed configuration, move the object with the manipulator, and take another radiograph. Of course, an automated machine could be constructed for these measurements just like the industrial machines that already exist for computed tomography measurements.

Defectoscopy Furthermore, it would be interesting to evaluate if our method could also be used for defectoscopy by evaluating the difference images. We briefly discussed this idea at the end of Section 5.3 and it is similar to what Dael et al. [2017] presented for detecting defects in horticultural products. Unfortunately, in case of the wedge, we saw that the optimization algorithm may converge to wrong pose estimations because of the defects, which may negatively influence the accuracy.

Performance optimizations Our implementation is only a prototype intended mainly for experimenting and evaluating the approach with different parameters and in various situations. In no way it is optimized for the best performance, even though we implemented the ray tracing in a GPU-accelerated framework and we also used CUDA for computing the loss function, which significantly reduces the time per iteration. It would be interesting to compare the accuracy and performance of ray tracing in comparison to GPU rasterization as proposed by Vidal et al. [2009] (Section 2.4).

Removing rendering artefacts Our ray tracer currently suffers from occasional artefacts that we discussed in Section 5.5. These artefacts are tied to the triangulation and they do not influence the loss function significantly as they disappear even with a subtle change of the parameters, hence they behave more like loss function noise. In our opinion, removing the artefacts would not bring any benefits to our simple objects, but it could be more important for complex triangulated shapes with thousands of triangles or even more. As the artefacts

are caused by ray-triangle intersections, increasing the number of triangles also increases the number of bad pixels.

Removing the artefacts could be done in three ways. Either by post-processing the pixels as proposed by Vidal and Villard [2016], or by implementing watertight ray-triangle intersections [Woop et al., 2013] as in the renderer by Marinovszki et al. [2018], or by not using triangulation and instead relying on CSG or other representations, but they could, of course, have their own artefacts.

Bibliography

Jens Als-Nielsen and Des McMorrow. *Elements of Modern X-ray Physics: Als-Nielsen/Elements*. John Wiley & Sons, Inc., Hoboken, NJ, USA, March 2011. ISBN 978-1-119-99836-5 978-0-470-97395-0. doi: 10.1002/9781119998365. URL http://doi.wiley.com/10.1002/9781119998365.

Siti Madiha Muhammad Amir, M.T.H. Sultan, Mohammad Jawaid, Ahmad Hamdan Ariffin, Shukri Mohd, Khairul Anuar Mohd Salleh, Mohamad Ridzwan Ishak, and Ain Umaira Md Shah. Nondestructive testing method for Kevlar and natural fiber and their hybrid composites. In *Durability and Life Prediction in Biocomposites, Fibre-Reinforced Composites and Hybrid Composites*, pages 367–388. Elsevier, 2019. ISBN 978-0-08-102290-0. doi: 10.1016/B978-0-08-102290-0.00016-7. URL https://linkinghub.elsevier.com/retrieve/pii/B9780081022900000167.

Carsten Bellon, Andreas Deresch, Christian Gollwitzer, and Gerd-Rüdiger Jaenisch. Radiographic simulator aRTist: version 2. *18th WCNDT - World conference on nondestructive testing (Proceedings)*, 2012. URL https://opus4.kobv.de/opus4-bam/frontdoor/index/index/docId/27398.

Anand K Bewoor and Vinay A Kulkarni. *Metrology & measurement*. 2009. ISBN 978-0-07-014000-4. OCLC: 879567943.

BIPM. *The International System of Units*. Bureau international des poids et mesures, Sèvres, 9th edition, 2019. ISBN 978-92-822-2272-0.

Franc¸ois Blais. Review of 20 years of range sensor development. *Journal of Electronic Imaging*, 13(1):231, January 2004. ISSN 1017-9909. doi: 10.1117/1.1631921. URL http://electronicimaging.spiedigitallibrary.org/article.aspx?doi=10.1117/1.1631921.

R. H. Bossi, J. L. Cline, and G. E. Georgeson. X-Ray Computed Tomographic Inspection of Castings. In Donald O. Thompson and Dale E. Chimenti, editors, *Review of Progress in Quantitative Nondestructive Evaluation: Volume 10B*, pages 1783–1790. Springer US, Boston, MA, 1991. ISBN 978-1-4615-3742-7. doi: 10.1007/978-1-4615-3742-7_84. URL https://doi.org/10.1007/978-1-4615-3742-7_84.

Brierley, Bellon, and Lazaro Toralles. Optimized multi-shot imaging inspection design. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2216):20170319, August 2018. doi: 10.1098/rspa.2017.0319. URL https://royalsocietypublishing.org/doi/abs/10.1098/rspa.2017.0319.

Budianto and Daniel P. K. Lun. Robust Fringe Projection Profilometry via Sparse Representation. *IEEE Transactions on Image Processing*, 25(4):1726–1739, April 2016. ISSN 1057-7149, 1941-0042. doi: 10.1109/TIP.2016.2530313. URL http://ieeexplore.ieee.org/document/7407374/.

Mai Bui, Shadi Albarqouni, Michael Schrapp, Nassir Navab, and Slobodan Ilic. X-Ray PoseNet: 6 DoF Pose Estimation for Mobile X-Ray Devices. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1036–1044, Santa Rosa, CA, USA, March 2017. IEEE. ISBN 978-1-5090-4822-9. doi: 10.1109/WACV.2017.120. URL http://ieeexplore.ieee.org/document/7926703/.

Matteo Busi, Ulrik Olsen, Erik Knudsen, Jeppe Frisvad, Jan Kehres, Erik Dreier, Mohamad Khalil, and Kristoffer Haldrup. Simulation tools for scattering corrections in spectrally resolved x-ray computed tomography using McXtrace. *Optical Engineering*, 57:1, March 2018. doi: 10.1117/1.OE.57.3.037105.

Lorenz Butzhammer and Tino Hausotte. Effect of iterative sparse-view CT reconstruction with task-specific projection angles on dimensional measurements. February 2019. URL http://www.ndt.net/?id=23654.

Szu-yuan Chen, Anatoly Maksimchuk, and Donald Umstadter. Experimental observation of relativistic nonlinear Thomson scattering. *Nature*, 396(6712): 653–655, December 1998. ISSN 0028-0836, 1476-4687. doi: 10.1038/25303. URL http://www.nature.com/articles/25303.

Gregory R. Choppin, Jan-Olov Liljenzin, Jan Rydberg, and Christian Ekberg. *Radiochemistry and nuclear chemistry*. Elsevier/AP, Academic Press is an imprint of Elsevier, Amsterdam ; Boston, 4th edition edition, 2013. ISBN 978-0-12-405897-2. OCLC: ocn852806072.

A. Criminisi, I. Reid, and A. Zisserman. Single View Metrology. *International Journal of Computer Vision*, 40(2):123–148, November 2000. ISSN 1573-1405. doi: 10.1023/A:1026598000963. URL https://doi.org/10.1023/A:1026598000963.

M. van Dael, P. Verboven, J. Dhaene, L. Van Hoorebeke, J. Sijbers, and B. Nicolai. Multisensor X-ray inspection of internal defects in horticultural products. *Postharvest Biology and Technology*, 128:33–43, June 2017. ISSN 0925-5214. doi: 10.1016/j.postharvbio.2017.02.002. URL http://www.sciencedirect.com/science/article/pii/S0925521416304690.

Anton Du Plessis, Igor Yadroitsev, Ina Yadroitsava, and Stephan Le Roux. X-Ray Microcomputed Tomography in Additive Manufacturing: A Review of the Current Technology and Applications. *3D Printing and Additive Manufacturing*, 5, July 2018. doi: 10.1089/3dp.2018.0060.

O. Elek. *Efficient Methods for Physically-Based Rendering of Participating Media*. PhD Thesis, Max Planck Institute for Computer Science, 2016.

L. Feldkamp and G. Jesion. 3-D X-Ray Computed Tomography. *Review of Progress in Quantitative Nondestructive Evaluation*, January 1986. URL https://lib.dr.iastate.edu/qnde/1986/allcontent/57.

Lee Feldkamp, L. C. Davis, and James Kress. Practical Cone-Beam Algorithm. *J. Opt. Soc. Am*, 1:612–619, January 1984.

Shijie Feng, Liang Zhang, Chao Zuo, Tianyang Tao, Qian Chen, and Guohua Gu. High dynamic range 3d measurements with fringe projection profilometry: a review. *Measurement Science and Technology*, 29(12):122001, December 2018. ISSN 0957-0233, 1361-6501. doi: 10.1088/1361-6501/aae4fb. URL http://stacks.iop.org/0957-0233/29/i=12/a=122001?key=crossref.949689f3403b182385158d87ab5505b2.

L. Finkelstein. Fundamental concepts of measurement. *Acta IMEKO*, 3:10–15, May 2014.

Andreas Fischer, Tobias Lasser, Michael Schrapp, Jürgen Stephan, and Peter B. Noël. Object Specific Trajectory Optimization for Industrial X-ray Computed Tomography. *Scientific Reports*, 6:19135, January 2016. ISSN 2045-2322. doi: 10.1038/srep19135. URL https://www.nature.com/articles/srep19135.

S.C. Garcea, Y. Wang, and P.J. Withers. X-ray computed tomography of polymer composites. *Composites Science and Technology*, 156:305–319, March 2018. ISSN 02663538. doi: 10.1016/j.compscitech.2017.10.023. URL https://linkinghub.elsevier.com/retrieve/pii/S0266353817312460.

E. Bruce Goldstein and James R. Brockmole. *Sensation and perception*. Cengage Learning, Boston, 10th edition, 2017. ISBN 978-1-305-58029-9. OCLC: 952665360.

G. N. Hounsfield. Computerized transverse axial scanning (tomography): Part 1. Description of system. *The British Journal of Radiology*, 46 (552):1016–1022, December 1973. ISSN 0007-1285. doi: 10.1259/0007-1285-46-552-1016. URL https://www.birpublications.org/doi/abs/10.1259/0007-1285-46-552-1016.

Tomáš Iser. *Real-Time Light Transport in Analytically Integrable Participating Media*. Bachelor Thesis, Charles University, Prague, 2017.

Steven G. Johnson. The NLopt nonlinear-optimization package, 2019. URL https://github.com/stevengj/nlopt. Accessed: 2019-06-13.

D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, October 1993. ISSN 0022-3239, 1573-2878. doi: 10.1007/BF00941892. URL http://link.springer.com/10.1007/BF00941892.

G. A. Jones and P. Huthwaite. Limited view X-ray tomography for dimensional measurements. *NDT & E International*, 93:98–109, January 2018. ISSN 0963-8695. doi: 10.1016/j.ndteint.2017.09.002. URL http://www.sciencedirect.com/science/article/pii/S0963869517302074.

R. Jordan, F. Feeney, N. Nesbitt, and J.A. Evertsen. Classification of wood species by neural network analysis of ultrasonic signals. *Ultrasonics*, 36(1-5):219–222, February 1998. ISSN 0041624X. doi: 10.1016/S0041-624X(97)00148-0. URL https://linkinghub.elsevier.com/retrieve/pii/S0041624X97001480.

M. Zahangir Kabir and Safa Kasap. Photoconductors for X-Ray Image Detectors. In Safa Kasap and Peter Capper, editors, *Springer Handbook of Electronic and Photonic Materials*, Springer Handbooks. Springer International Publishing, Cham, 2017. ISBN 978-3-319-48933-9. URL https://doi.org/10.1007/978-3-319-48933-9_45.

Safa Kasap, Joel B. Frey, George Belev, Olivier Tousignant, Habib Mani, Jonathan Greenspan, Luc Laperriere, Oleksandr Bubon, Alla Reznik, Giovanni DeCrescenzo, Karim S. Karim, and John A. Rowlands. Amorphous and Polycrystalline Photoconductors for Direct Conversion Flat Panel X-Ray Image Sensors. *Sensors*, 11(5):5112–5157, May 2011. ISSN 1424-8220. doi: 10.3390/s110505112. URL http://www.mdpi.com/1424-8220/11/5/5112.

Angus I. Kingon, Theodore F. Morse, Peter M. Weber, Adil Akif, Rajiv Gupta, Timothy P. Murphy, Nerine J. Cherepy, Bernhard W. Adams, Thomas G. Bifano, Brian Stankus, and Nicholas Mostovych. Demonstration of a high resolution x-ray detector for medical imaging. In Gary P. Grim, H. Bradford Barber, Lars R. Furenlid, and Jeffrey A. Koch, editors, *Radiation Detectors in Medicine, Industry, and National Security XIX*, page 11, San Diego, United States, September 2018. SPIE. ISBN 978-1-5106-2097-1 978-1-5106-2098-8. doi: 10.1117/12.2320723. URL https://www.spiedigitallibrary.org/conference-proceedings-of-spie/10763/2320723/Demonstration-of-a-high-resolution-x-ray-detector-for-medical/10.1117/12.2320723.full.

Erik Knudsen, Andrea Prodi, Jana Baltser, Maria Thomsen, Peter Willendrup, Manuel Sánchez del Río, Claudio Ferrero, Emmanuel Farhi, Kristoffer Haldrup, Anette Vickery, Robert Feidenhans'l, Kell Mortensen, Martin Nielsen, H.F. Poulsen, Søren Schmidt, and Kim Lefmann. McXtrace: A Monte Carlo software package for simulating X-ray optics, beamlines and experiments. *Journal of Applied Crystallography*, 46:679–696, June 2013. doi: 10.1107/S0021889813007991.

Anastasios C. Konstantinidis, Magdalena B. Szafraniec, Robert D. Speller, and Alessandro Olivo. The Dexela 2923 CMOS X-ray detector: A flat panel detector based on CMOS active pixel sensors for medical imaging applications. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 689:12–21, October 2012. ISSN 01689002. doi: 10.1016/j.nima.2012.06.024. URL https://linkinghub.elsevier.com/retrieve/pii/S0168900212006730.

J. P. Kruth, M. Bartscher, S. Carmignato, R. Schmitt, L. De Chiffre, and A. Weckenmann. Computed tomography for dimensional metrology. *CIRP Annals*, 60(2):821–842, January 2011. ISSN 0007-8506. doi: 10.1016/j.cirp.2011.05.006. URL http://www.sciencedirect.com/science/article/pii/S0007850611002083.

Markus Körner, Christof H. Weber, Stefan Wirth, Klaus-Jürgen Pfeifer, Maximilian F. Reiser, and Marcus Treitl. Advances in Digital Radiography: Physical Principles and System Overview. *RadioGraphics*, 27(3):675–686,

May 2007. ISSN 0271-5333, 1527-1323. doi: 10.1148/rg.273065075. URL http://pubs.rsna.org/doi/10.1148/rg.273065075.

Chris Lefteri. *Making it: manufacturing techniques for product design*. Laurence King, London, 2. ed edition, 2012. ISBN 978-1-85669-749-1. OCLC: 760291946.

Matthew M. Loper and Michael J. Black. OpenDR: An Approximate Differentiable Renderer. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, volume 8695, pages 154–169. Springer International Publishing, Cham, 2014. ISBN 978-3-319-10583-3 978-3-319-10584-0. doi: 10.1007/978-3-319-10584-0_11. URL http://link.springer.com/10.1007/978-3-319-10584-0_11.

Árpád Marinovszki, Jan De Beenhouwer, and Jan Sijbers. An efficient CAD projector for X-ray projection based 3d inspection with the ASTRA Toolbox. In *8th Conference on Industrial Computed Tomography, Wels, Austria*, 2018.

C H McCollough. The AAPM/RSNA physics tutorial for residents. X-ray production. *RadioGraphics*, 17(4):967–984, July 1997. ISSN 0271-5333, 1527-1323. doi: 10.1148/radiographics.17.4.9225393. URL http://pubs.rsna.org/doi/10.1148/radiographics.17.4.9225393.

Domingo Mery. *Computer Vision for X-Ray Testing: Imaging, Systems, Image Databases, and Algorithms*. Springer International Publishing, 2015. ISBN 978-3-319-20746-9. URL https://www.springer.com/gp/book/9783319207469.

S. Miao, A. Tuysuzoglu, Z. J. Wang, and R. Liao. Real-time 6dof pose recovery from X-ray images using library-based DRR and hybrid optimization. *International Journal of Computer Assisted Radiology and Surgery*, 11(6):1211–1220, June 2016. ISSN 1861-6410, 1861-6429. doi: 10.1007/s11548-016-1387-2. URL http://link.springer.com/10.1007/s11548-016-1387-2.

Douglas C. Montgomery. *Introduction to statistical quality control*. Wiley, Hoboken, N.J, 6th ed edition, 2009. ISBN 978-0-470-16992-6.

Christopher I. Moore. *Observation of the Transition from Thomson to Compton Scattering in Optical Multiphoton Interactions with Electrons.* PhD Thesis, 1995. URL https://www.semanticscholar.org/paper/Observation-of-the-Transition-from-Thomson-to-in-Moore/c6e75d4318197f761dc4db3b466e584136cd6ebf.

John C Nash. *Nonlinear parameter optimization using R tools*. 2014. ISBN 978-1-118-88396-9 978-1-118-88475-1 978-1-118-88400-3 978-1-118-56928-3 978-1-306-63878-4. URL http://public.eblib.com/choice/publicfullrecord.aspx?p=1666497. OCLC: 867284265.

J. A. Nelder and R. Mead. A Simplex Method for Function Minimization. *The Computer Journal*, 7(4):308–313, January 1965. ISSN 0010-4620, 1460-2067. doi: 10.1093/comjnl/7.4.308. URL https://academic.oup.com/comjnl/article-lookup/doi/10.1093/comjnl/7.4.308.

Van Nguyen, Jan De Beenhouwer, Joaquim Sanctorum, S Van Wassenbergh, P Aerts, C Van Ginneken, J J J Dirckx, and Jan Sijbers. A low-cost and easy-to-use phantom for cone-beam geometry calibration of a tomographic X-ray system. February 2019.

J. A. Noble, R. Gupta, J. Mundy, A. Schmitz, and R. I. Hartley. High precision X-ray stereo for automated 3d CAD-based inspection. *IEEE Transactions on Robotics and Automation*, 14(2):292–302, April 1998. ISSN 1042-296X. doi: 10.1109/70.681247.

Dominika Oborska-Kumaszyńska and Sylwia Wiśniewska-Kubka. Analog and digital systems of imaging in roentgenodiagnostics. *Polish Journal of Radiology*, 75(2):73–81, 2010. ISSN 1733-134X. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3389874/`.

Andrea Palazzi, Luca Bergamini, Simone Calderara, and Rita Cucchiara. End-to-end 6-DoF Object Pose Estimation through Differentiable Rasterization. In *The European Conference on Computer Vision (ECCV) Workshops*, September 2018.

Steven G. Parker, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllister, Morgan McGuire, Keith Morley, Austin Robison, and Martin Stich. OptiX: A General Purpose Ray Tracing Engine. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, pages 66:1–66:13, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0210-4. doi: 10.1145/1833349.1778803. URL `http://doi.acm.org/10.1145/1833349.1778803`. event-place: Los Angeles, California.

M. J. D. Powell. A Direct Search Optimization Method That Models the Objective and Constraint Functions by Linear Interpolation. In Susana Gomez and Jean-Pierre Hennart, editors, *Advances in Optimization and Numerical Analysis*, Mathematics and Its Applications, pages 51–67. Springer Netherlands, Dordrecht, 1994. ISBN 978-94-015-8330-5. doi: 10.1007/978-94-015-8330-5_4. URL `https://doi.org/10.1007/978-94-015-8330-5_4`.

M. J. D. Powell. The BOBYQA Algorithm for Bound Constrained Optimization without Derivatives. *Technical Report, Department of Applied Mathematics and Theoretical Physics*, January 2009.

Bertrand Russell. *History of Western Philosophy.* Routledge, London, 2004. ISBN 978-0-203-48797-6 978-0-415-32505-9. URL `http://public.eblib.com/choice/publicfullrecord.aspx?p=182683`. OCLC: 437055799.

Stuart J. Russell, Peter Norvig, and Ernest Davis. *Artificial intelligence: a modern approach*. Prentice Hall series in artificial intelligence. Prentice Hall, Upper Saddle River, 3rd ed edition, 2010. ISBN 978-0-13-604259-4.

Paul D. Scott and Evstatin Krastev. 2d X-Ray inspection With Materials and Thickness Identification. In *SMTA International Conference Proceedings*, September 2016.

Euclid Seeram. *Digital Radiography: Physical Principles and Quality Control*. Springer Singapore, Singapore, second edition, 2019. ISBN 9789811332432 9789811332449. doi: 10.1007/978-981-13-3244-9. URL http://link.springer.com/10.1007/978-981-13-3244-9.

Stephen Seltzer. XCOM-Photon Cross Sections Database, NIST Standard Reference Database 8, 1987. URL http://www.nist.gov/pml/data/xcom/index.cfm. type: dataset.

N. Stewart. *An image-space algorithm for hardware-based rendering of constructive solid geometry*. PhD Thesis, 2008. URL http://researchbank.rmit.edu.au/view/rmit:9552.

Frank E. Swindells. Some Physical Properties of Fluorescent Screens. *Radiology*, 11(5):424–428, November 1928. ISSN 0033-8419, 1527-1315. doi: 10.1148/11.5.424. URL http://pubs.rsna.org/doi/10.1148/11.5.424.

Joachim Tabary, P. Hugonnard, Francoise Mathy, and CEA-LETI MINATEC. SINDBAD : a realistic multi-purpose and scalable X-ray simulation tool for NDT applications. 2007.

Roger Bradley Ulrich. *Roman woodworking*. Yale University Press, New Haven [CT], 2007. ISBN 978-0-300-10341-0. OCLC: ocm64083838.

Franck Vidal, M Garnier, N Freud, Jm Létang, and Nigel John. Accelerated Deterministic Simulation of X-ray Attenuation Using Graphics Hardware. page Poster 5011, May 2010.

Franck P. Vidal and Pierre-Frédéric Villard. Development and validation of real-time simulation of X-ray imaging with respiratory motion. *Computerized Medical Imaging and Graphics*, 49:1–15, April 2016. ISSN 08956111. doi: 10.1016/j.compmedimag.2015.12.002. URL https://linkinghub.elsevier.com/retrieve/pii/S0895611115001895.

Franck P. Vidal, Manuel Garnier, Nicolas Freud, Jean Michel Létang, and Nigel W. John. Simulation of X-ray Attenuation on the GPU, 2009. URL http://diglib.eg.org/handle/10.2312/LocalChapterEvents.TPCG.TPCG09.025-032.

Herminso Villarraga-Gómez. X-ray Computed Tomography for Dimensional Measurements. July 2016.

Herminso Villarraga-Gómez, Edward Morse, Robert J. Hocken, and Stuart Smith. Dimensional metrology of internal features with X-ray computed tomography. November 2014.

Herminso Villarraga-Gómez, ChaBum Lee, and Stuart T. Smith. Dimensional metrology with X-ray CT: A comparison with CMM measurements on internal features and compliant structures. *Precision Engineering*, 51:291–307, January 2018. ISSN 0141-6359. doi: 10.1016/j.precisioneng.2017.08.021. URL http://www.sciencedirect.com/science/article/pii/S014163591630157X.

T. Wawrzinek, U. Zscherpel, and C. Bellon. Wall thickness determination in digital radiography. *The online Journal of Nondestructive Testing & Ultrasonics*, Vol.2(No.10), October 1997. URL https://www.ndt.net/article/wt1097/zscherp/zscherp.htm. Accessed: 2019-06-02.

W.J. Wiscombe and J.W. Evans. Exponential-sum fitting of radiative transmission functions. *Journal of Computational Physics*, 24(4):416–444, August 1977. ISSN 00219991. doi: 10.1016/0021-9991(77)90031-6. URL https://linkinghub.elsevier.com/retrieve/pii/0021999177900316.

Sven Woop, Carsten Benthin, and Ingo Wald. Watertight Ray/Triangle Intersection. 2013.

Dawei Xu, Fang Cheng, Yu Zhou, Thaddie Matalaray, Pei Xian Lim, and Liping Zhao. Process optimization: internal feature measurement for additive-manufacturing parts using x-ray computed tomography. page 40, March 2019. doi: 10.1117/12.2511429.

M J Yaffe and J A Rowlands. X-ray detectors for digital radiography. *Physics in Medicine and Biology*, 42(1):1–39, January 1997. ISSN 0031-9155, 1361-6560. doi: 10.1088/0031-9155/42/1/001. URL http://stacks.iop.org/0031-9155/42/i=1/a=001?key=crossref.6b789f4e194e9f93abd008c341d01fee.

Min Yao, Philippe Duvauchelle, Valerie Kaftandjian, Angela Peterzol, and Andreas Schumm. Simulating radiographic inspections with imaging plates. volume 1706, page 110005, February 2016. doi: 10.1063/1.4940576.

Leonid Ja. Zhmud. "All is number"? *Phronesis*, 34(1-3):270–292, 1989. ISSN 0031-8868, 1568-5284. doi: 10.1163/156852889X00189. URL https://brill.com/view/journals/phro/34/1-3/article-p270_18.xml.

U. Zscherpel, A. Alekseychuk, S. Bär, and P. Rost. Corrosion and Wall Thickness Measurement, April 2007. URL https://www.ndt.net/article/imagingNDE2007/Zscherpel2.pdf. Accessed: 2019-06-02.

Chao Zuo, Shijie Feng, Lei Huang, Tianyang Tao, Wei Yin, and Qian Chen. Phase shifting algorithms for fringe projection profilometry: A review. *Optics and Lasers in Engineering*, 109:23–59, October 2018. ISSN 01438166. doi: 10.1016/j.optlaseng.2018.04.019. URL https://linkinghub.elsevier.com/retrieve/pii/S0143816618302203.

List of Figures

Attachment 1 – User reference

As described in Chapter 4, we implemented our method in a configurable prototype demo application. In the electronic attachment, we can find the main application and other utilities, their executables and source codes. Here we explain how the demo can be used from the user point of view.

System requirements

We tested the prototype on several computers. Based on our observations, we recommend the following system configuration:

- Operating system: Windows 10, 64-bit version,
- CPU: x64, multi core, frequency 2.0 GHz or higher,
- Physical memory: 4 GB or more,
- GPU: NVIDIA required, Compute Capatibility 3.0 (Kepler architecture) or higher, at least 2 GB of memory, newest drivers (396.65 or higher); it might be necessary to have NVIDIA CUDA Toolkit 10.0 installed[1],
- C++ components: Microsoft Visual C++ Redistributable for Visual Studio 2017 needs to be installed[2].

Graphical user interface

The application is started by simply running the file `build/CUNI_MgrThesis.exe`. This batch file executes the graphical user interface (GUI) of an empty project corresponding to an empty dataflow graph (Chapter 4). The GUI consists of an almost empty window with a framerate counter on the top. Together with the main window, a console is opened and various kinds of messages are logged there.

Working with a project

The simplest way to *load a project* (a set of interconnected graph nodes, Chapter 4), is to drag and drop a node file or multiple files onto the main window. These files are typically JSON files (`.json`) with an exception of radiographs (`.tiff`) and 3D STL models (`.stl`).

When a project is loaded, its graph nodes (Figure 4.1) are constructed and small windows appear inside the main window (Figure 4.2). These subwindows correspond to the project nodes and show their settings and possible actions. Together with these, there is a subwindow called *Optimization Variables* that enables changing the current level of detail and multiplex (Section 4.2), i.e., switching between multiple reference images and their poses if present.

In the top menu, additional two buttons are displayed:

- **Step** executes a single step of each graph node in the topological order, i.e., it forces the graph to recalculate once. Beware that if an optimizer node exists in the graph, performing a single step means executing the whole

[1] `https://developer.nvidia.com/cuda-toolkit`

[2] `https://aka.ms/vs/15/release/VC_redist.x64.exe`

optimization process! The optimization process can be stopped by pressing *Ctrl+C* inside the console window once (terminates only the current optimization in the hierarchy) or twice (terminates the whole optimization). Results of the optimization can be seen in the console window and the graph state can be saved and serialized using the following button:

- **Output** saves the supported graph nodes to the output folder (see Command line interface). Note that only the following nodes can be saved: *EsftMaterial*, *Wedge*, *Prism*, *Transformation* (all of them saved as `.json` that can be loaded again), *OptiXRenderer* (saves the render as `.tiff`), *PerPixelLoss* (calculates and saves the loss function values per pixel as `.tiff`).

Note that a completely new project cannot be created via the application. It either has to be created manually as new `.json` files or you can save modifications of existing projects via the GUI and the "Output" button.

Mouse and keyboard interactions

Our GUI is designed using the Dear ImGui framework[3]. The main window corresponds to a workspace that contains project subwindows as already explained.

The subwindows have a *title bar*, which contains an arrow-button for collapsing and expanding each window, and a title which corresponds to the file name of the node, or for inline-defined nodes that do not have a dedicated file it says `@@inline_argument_XYZ`, where `XYZ` is the number of the inline node.

Each subwindow can be moved by clicking and dragging either the title bar or the window area itself if there is not any interactive element under the cursor. Note that the subwindows may be dragged *outside* the main window area, i.e., the subwindows can float wherever on the screen not limited by the main window. Their positions are remembered when the application terminates and are restored when the nodes of the same names are loaded.

Inside the subwindows, all clickable buttons are light blue and can be activated by clicking them with a mouse. Number inputs are represented by gray rectangles, their values can be changed either by double-clicking them and using a keyboard input, or by clicking and dragging the mouse over them in the left and right directions. Drop-down inputs are represented by gray rectangles with an arrow on the right side, they can be clicked and a new value selected by mouse clicking. Finally, there are checkboxes that can be (de)activated by clicking them. Keyboard navigation is also possible by pressing *Tab*, arrow keys, and *Spacebar*, and by pressing *Ctrl+Tab* to switch between subwindows. For more details, please consult the Dear ImGui repository.

Command line interface

For convenience and automated executions, the application can also be executed without the graphical user interface and controlled by command line arguments. The supported arguments are the following:

[3] `https://github.com/ocornut/imgui`

- `--cli` runs the application without the graphical user interface. A project will be loaded immediately, a topological ordering of the nodes will be found, and all nodes will be executed and their outputs saved to an output folder. That will happen exactly once and then the application will terminate.
- `--output XYZ` sets the output path (directory) to `XYZ` (relative or absolute). This path is used to save and serialize graph nodes. If not specified, the default output path is `./output` relative to the current working directory.
- `--assets XYZ` sets the assets path to `XYZ` (relative or absolute). This path contains assets such as fonts and compiled ray-tracing programs. If not specified, the application tries to find the first directory called `assets` in the current working directory or parent directories.
- All other arguments not beginning with `--` are interpreted as paths to project files that will be loaded when the application starts. If not specified, an empty project is loaded.

For example:

```
build/CUNI_MgrThesis.exe --cli --output test optimizer.json
```

loads and executes a project defined in a file `optimizer.json` and saves the results in the `test` directory.

Example projects

The electronic attachment contains a few example projects that can be used to verify and reproduce our results from Chapter 5 and possibly create new projects based on them. We mention the following:

- `projects/prism_measurements/prismB_material`
  This directory contains a project for fitting the prism material from Prism B (Section 5.2). The project can be loaded from `optimizer_prismB.json`. The reference images and their transformations for the optimization can be changed in `references_prismA.json` and `transformations_prism.json`. When the optimization is executed and finishes, the fitted material can be saved and used for the following project.
- `projects/prism_measurements/prismA_dimensions`
  This directory contains a project for measuring Prism A dimensions from the material fitted from Prism B in the project above. The project can be loaded from `optimizer_prismA.json`. References and their transformations can again be changed in the appropriate files as in the project above.
- `projects/prism_measurements/statistics_generator`
  In this directory, there is a simple C++ source code and executable for automatically generating project files and executing all optimizations for Prism A dimensional measurements from all possible combinations of two or three reference images (two translations and one rotation). The results are automatically parsed and printed to the standard output. Note that executing this example may take several hours based on the hardware. This project was used for generating the statistics from Section 5.2.
- `projects/wedge_measurements/dimensions`
  Similarly to the prism measurements above, this directory contains a project

for measuring the wedge dimensions. The project can be loaded from `optimizer_wedge.json`. The material in `material_wedge_plotfit.json` is calibrated from the wedge directly by fitting exponentials to the intensity table via Wolfram Mathematica (Section 3.4a). The Mathematica notebook with the fitting is available in `material_wedge_plotfit.nb` together with experimental fitting of a varying number of exponentials in the sum.

- `projects/wedge_measurements/statistics_generator`
  Similarly to the statistics generator for Prism A, this directory contains a source code and executable for automatically generating project files and executing all optimizations for wedge dimensional measurements from all possible combinations of two reference images (without rotations).

Building the demo

All executables for the 64-bit Windows platform are already available in the `build` directory. In order to build the project from scratch, one can use the `CMakeLists.txt` configuration for CMake[4] to generate the solution for Visual Studio 2017 that can be compiled afterwards. If unfamiliar with CMake, please consult their official documentation.

Note that it is necessary to further define the following CMake options when generating the project. We need to set the OptiX SDK path `OptiX_INSTALL_DIR` and then we also need to set the CUDA Toolkit include directory `CMAKE_CUDA_TOOLKIT_INCLUDE_DIRECTORIES_WORKAROUND`. Both options can be set from the command line using the CMake `-D` argument.

Beware that building the project was only tested with Visual Studio 2017 and Windows 10, so remember to use the `Visual Studio 15 2017` CMake generator. It also requires NVIDIA OptiX SDK 5.1.1 or 6.0.0[5] and NVIDIA CUDA Toolkit 9.0 or 10.0[6] installed on the machine. We recommend consulting the official NVIDIA documentations and forums especially for OptiX support[7].

Furthermore, building the project requires the `libtiff` library for loading and saving TIFF. The repository of this library has to be downloaded[8] to the `lib/libtiff` directory.

The remaining libraries are already present in the attachment.

Building the statistics generators Building the statistics generators from the previous section is handled separately as they are just a single C++ file. These can be compiled directly without any complicated settings, e.g., via the *Developer Command Prompt for VS 2017* such as:

```
cl /std:c++17 /EHsc /I"../../../lib/json/include" statistics_generator.cpp
```

[4] `https://cmake.org/`

[5] `https://developer.nvidia.com/optix`

[6] `https://developer.nvidia.com/cuda-toolkit`

[7] `https://devtalk.nvidia.com/default/board/254/`

[8] `https://libtiff.gitlab.io/libtiff/` or `http://www.simplesystems.org/libtiff/` or `https://gitlab.com/libtiff/libtiff` are all mirrors of the same library.

Attachment 2 – Electronic attachment contents

The contents of the accompanying electronic attachment are organized as follows:

- `assets/` — assets (fonts, geometry, and compiled OptiX programs) required to run the demo application,
- `build/CUNI_MgrThesis.exe` — the demo application,
- `cmake/` — utility files for CMake,
- `lib/` — C++ libraries required to compile the demo application,
- `projects/` — the demo project files and selected reference radiographs in full resolution,
- `projects/prism_measurements/` — the demo projects for prism measurements, see Attachment 1 – User reference for more details,
- `projects/wedge_measurements/` — the demo projects for wedge measurements, see Attachment 1 – User reference for more details,
- `projects/prism/` — normalized prism jaws reference radiographs in full resolution,
- `projects/wedge/` — normalized wedge reference radiographs in full resolution, only a subset of the original dataset is attached to save space,
- `src/` — our demo application C++ and CUDA source codes,
- `CMakeLists.txt` — main configuration file for building the demo application via CMake.