Saarland University
Faculty of Natural Sciences and Technology I
Department of Computer Science
Master's Program in Visual Computing

Master's Thesis

Optical Analysis of High Pressure Diesel Sprays using Image Segmentation Methods

submitted by

Timo Florian Adam

January 26, 2016

UNIVERSITÄT DES SAARLANDES

Supervisor:
Prof. Dr. Joachim Weickert,
Mathematical Image Analysis Group

First Reviewer:
Prof. Dr. Joachim Weickert

Second Reviewer:
Dr. Peter Ochs

Statement in Lieu of an Oath

I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis.

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Saarbrücken, January 26, 2016

Statement in Lieu of an Oath

I hereby confirm the congruence of the contents of the printed data and the electronic version of the thesis.

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides Statt, dass die vorliegende Arbeit mit der elektronischen Version übereinstimmt.

Saarbrücken, January 26, 2016

Declaration of Consent

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Einverständniserklärung

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlicht wird.

Saarbrücken, January 26, 2016

Abstract

Internal combustion engines operate on the principle of injecting a fuel and an oxidizer into a combustion chamber and igniting the gas mixture in the chamber. The resulting chemical reaction releases heat which causes the gas to expand. The increase in pressure inside the chamber is converted into mechanical energy, which can now be utilized elsewhere.

The effectiveness of such engines relies heavily on the characteristics of the fuel injection procedure. Acquiring visual data of this process and performing an optical analysis of the recorded data gives important insights into the design of modern injectors. To being able to evaluate the data in a meaningful and mathematically well-founded way, image processing methods are an indispensable tool.

In this work, we will investigate several image processing methods for analyzing such data. The main research focus will be on region-based active contour segmentation methods, using motion information as an additional input feature. To obtain such motion data, we will apply state-of-the-art variational optical flow methods.

This work is a joint teaching and research effort by the Mathematical Image Analysis Group at Saarland University and Delphi Automotive Systems, Technical Center Luxembourg.

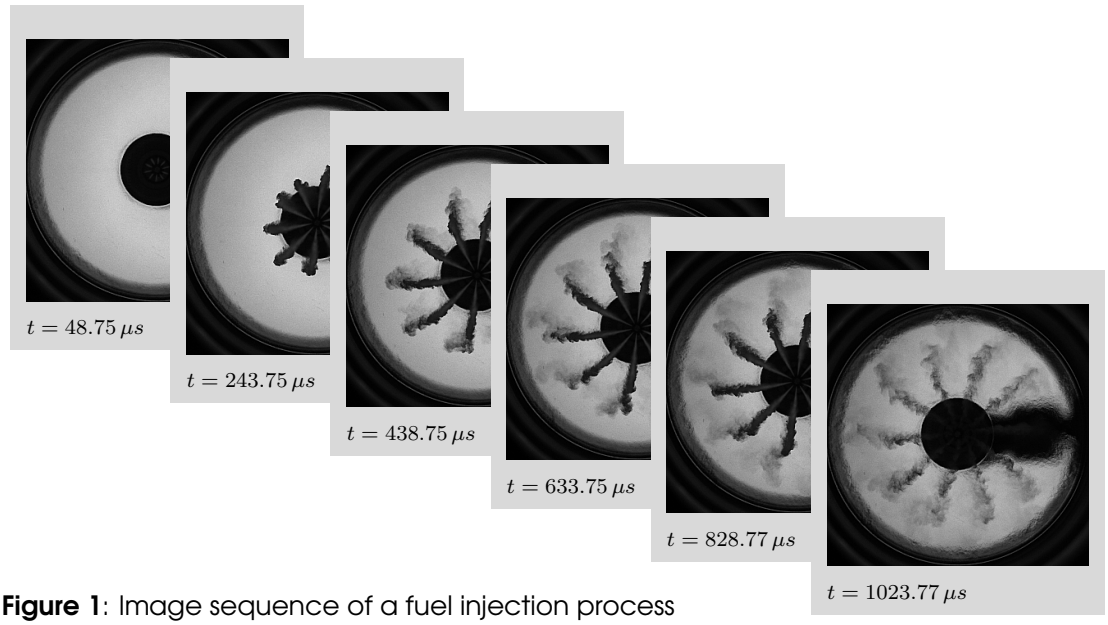


Figure 1: Image sequence of a fuel injection process

Acknowledgements

I want to thank Prof. Dr. Joachim Weickert for offering me this interesting topic and for providing me with ideas and guidance throughout my work. I would also like to thank all members of the Mathematical Image Analysis Group at Saarland University for their friendly assistance, including Dr. Peter Ochs for agreeing to review my thesis as a second reviewer.

I want to thank the Advanced Injection & Combustion Engineering Group at Delphi Technical Center Luxembourg, particularly the Diesel Combustion & Spray Optical Diagnostics Team headed by Dr. Kourosh Karimi, for offering me the opportunity to work at Delphi for an internship as part of my thesis. I would like to thank all my team members, Baptiste Lemaitre, Philippe Goossens and Ben Wagener. Ben spent a lot of his precious time with me introducing me to the exciting world of engineering and internal combustion engines. It was a valuable experience and a pleasure working at Delphi TCL.

Last but certainly not least, I want to thank my family for their unconditional support throughout my studies and during my internship in Luxembourg.

Contents

1 Introduction

1.1 Motivation

1

The purpose of an *internal combustion engine* is to convert chemical energy into mechanical energy. This is achieved by having two substances interact with each other in an exothermic redox-reaction, also known as *combustion*. An *exothermic reaction* is a type of chemical reaction in which energy is released. A *redox-reaction* is a chemical reaction between a reducing- and an oxidizing agent in which atoms have their oxidation state changed. A *reducing agent* loses electrons and is oxidized while an *oxidizing agent* gains electrons and is reduced. The oxidizing agent must be a chemical species or mixture that contains molecular oxygen, typically air. The reducing agent must be a material that stores potential energy, usually in the form of some hydrocarbon compounds. In an engineering context, the reducing agent is referred to as *fuel*. Fuels occur in solid, liquid and gaseous form and can either occur in nature or be created artificially. For a classification of fuels, see table 1.1.

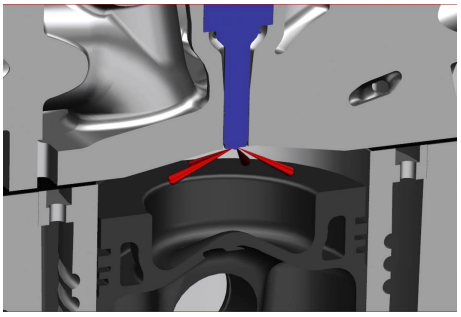


Figure 1.1: The cylinder of a gasoline engine. The injector (blue) injects the fuel into the combustion chamber where the spray is formed (red). After the combustion, the piston (black) is pushed outwards by the increase in pressure.

To being able to exploit the energy that is released during a combustion, the reaction has to take place in an enclosed space of the engine, the *combustion chamber*. The engine component enclosing the combustion chamber is referred to as *cylinder*. Using electronically controlled high pressure valves, called *injectors*, the fuel and air mixture is now introduced into the chamber. The heat that is released during the combustion leads to an expansion of the gas which increases the pressure in the combustion chamber. The *piston*, a moving engine component that is con-

Table 1.1: Types of chemical fuels

	Naturally occurring fuels	Artificially created fuels
Solid fuels	wood, coal,...	coke, charcoal
Liquid fuels	petroleum	diesel, gasoline, kerosine, ethanol,...
Gaseous fuels	natural gas	hydrogen, propane,...

tained in the cylinder, is pushed outside by the pressure. An engine usually consists of multiple cylinders, working in conjunction. A crank mechanism forces the pistons to move reciprocally in and out of the cylinder. The crankshaft converts this reciprocal motion into rotational motion. This mechanical energy in the form of rotational motion is now available to be used elsewhere.

In this work, we will only be concerned with injection sequences of *Diesel engines* as opposed to those of *gasoline engines*. In gasoline engines, the fuel air mixture is ignited externally via a spark. In a Diesel engine, the air in the combustion chamber is compressed and heats up. When the fuel is injected, the gas mixture ignites itself. The injection of fuel into the chamber is a crucial step in a Diesel engine. For the combustion to be effective, the liquid fuel has to be vaporized by injecting it into the combustion chamber under very high pressure. The injection process has a huge influence on the effectiveness, durability and emissions of an engine. Therefore, the design of injectors is an important research topic in engineering.

The optical analysis of an injection process is a very important tool in this context. The visual recording and the analysis of the data can give helpful insights. To being able to perform a thorough and convincing analysis, the image data has to be processed in a mathematically accurate manner, using advanced image processing methods.

1.2 Goals of Injection and Combustion Engineering

The research goals of injection and combustion engineering are to increase the efficiency, the power output and the reliability of engines while reducing fuel consumption, emissions and costs.

- The chemical reaction that occurs during a combustion usually leaves residues in the form of soot. This increases the friction of the piston inside the cylinder [GL08]. A more effective combustion process reduces the amount of soot residues. This enhances the overall *durability and reliability* of the engine and helps to *reduce maintenance costs*. Less friction inside the cylinder also allows for an increase in torque which leads to a *performance increase* and a higher power output of an engine.
- The ignition of the air fuel mixture in the combustion chamber leads to a rapid pressure rise that results in vibrations and audible noise [NKM12]. Having a more homogeneous distribution of fuel allows for a reduced ignition time which, in turn, helps in *reducing noise emissions*.
- The combustion and the friction cause an intense heat. An injector has to withstand temperatures of -30 up to 100 °C. If one assumes a life expectancy of 10 years or 300,000 km, an injector will perform over 2 billion injections. An important design goal is to *prevent deterioration* and increase the overall *driveability* and smooth operation of the engine.

- The combustion reaction turns fuel in the form of carbohydrate compounds ($C_xH_yO_z$), and air (N_2, O_2) into nitrogen (N_2), carbon dioxide (CO_2) and water (H_2O). This process usually produces several byproducts and leaves residues. The incomplete oxidation of carbon in Diesel fuel (caused by low temperatures or poor oxygen areas) emits carbon monoxide (CO). Particulates matters (soot), are set free by the incomplete fuel vaporization in the combustion chamber, referred to as liquid fuel coking. Unburned hydrocarbon (HC) deposits are created when fuel gets in contact with the colder combustion chamber walls. The oxidation of nitrogen in the air only occurs at high temperatures and a great excess of air. This causes emissions of oxidized nitrogen (NO_x), mostly occurring as species of nitric oxide (NO) and nitrogen dioxide (NO_2). Su et al. [SFN95] have shown that it is possible to *reduce exhaust emissions* and increase the range of environmental operation in Diesel engines if the injector nozzle produces smaller and more dispersed droplets.

- By modifying the injection flow, the timing and the pressure, it is possible to improve combustion control. This allows to *reduce fuel consumption* and gives rise to new possibilities for *engine tuning*. Since the whole injection process is controlled electronically, advances in injector design can also increase the *diagnostic capability* of an engine.

- Several different kinds of engines are required to *accommodate alternative types of fuel*. Investigating the injection characteristics of different types of fuel is therefore also an important research topic.

- The introduction of new injector designs also allows to *reduce the manufacturing- and assembly costs* of engine components.

1.3 Diesel Injector Characteristics and Design

The purpose of an injector is to admit fuel into the combustion chamber in a controlled manner. In contrast to carburetors that rely on the principle of suction, an

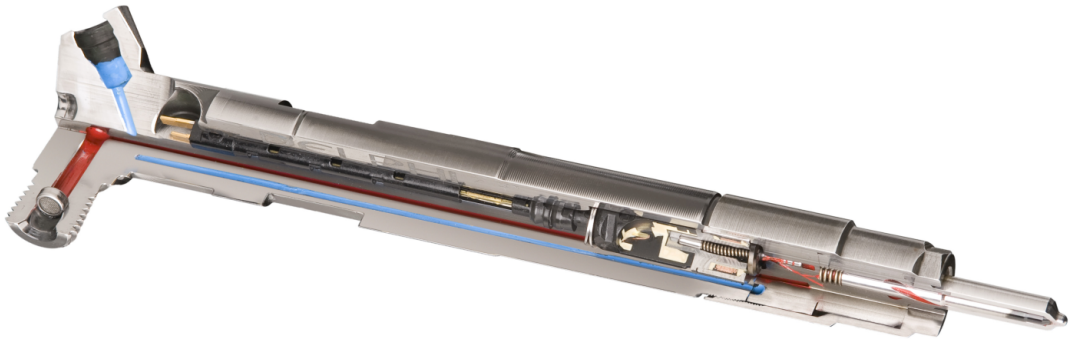


Figure 1.2: Cross section of a Delphi Multec Solenoid Diesel Injector (DFI1.5). The fuel is injected through the red tube. The blue tube channels the back-leak.

injector atomizes the fuel by injecting it under high pressure into the combustion chamber. The injector propels the fuel through tiny holes, usually with an exit diameter between 100 and 150 μm, into the chamber and turns the liquid fuel into an aerosol. In each injection, around 40,000 droplets of 15 μm diameter each enter the combustion chamber. High vaporization rates are achieved by applying pressures of up to 3000 bar. The separation between injections is usually around 100 μs. A single injection often comprises multiple pre- and post injections. The cross-section of an injector is depicted in figure 1.2.

The central component of an injector is the *needle*. When the fuel is inserted into the injector, it flows through conduits and notches, guided alongside the needle. At the pinpoint of the needle (*needle tip*), the fuel is collected in a small chamber, the *sac*. When the needle is in a lowered position, the area where the needle touches the injector body is called the *needle seat*. Every time the needle is lifted via a mechanism of springs, the fuel is admitted into the combustion chamber through tiny holes in the nozzle tip, the *nozzle holes*. The dispersed fuel in the combustion chamber is referred to as *spray*. An injector nozzle usually has between 3 and 12 holes. The spray released from a single nozzle hole is referred to as a *jet*. A cross section of this mechanism is shown in figure 1.3.

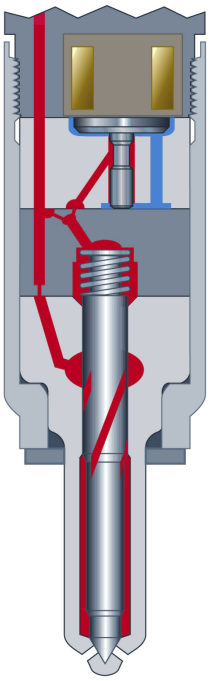


Figure 1.3: Illustration of an injector: After entering the injector under high pressure, the fuel flows along the needle and is collected in a chamber near the nozzle. When the needle lifts, the fuel is released. The nozzle breaks the fuel into droplets which form a spray pattern.

The injection process can be classified into four phases. When fuel enters the chamber, it is still in liquid form (*liquid phase*). After the droplets break apart, the fuel is now in the *vapor phase*. As soon as the gas mixture ignites, the *combustion phase* starts. The *soot formation* takes place when the chemical reaction has ended. In this work, we will only be concerned with the analysis of the liquid and vapor phases of an injection process.

There are certain aspects in the design of an injector, that have a huge influence on the goals outlined in section 1.2. Variations in the injection pressure, the amount of fuel or the spacing and geometry of the holes have an impact on spray symmetry, spray momentum, spray dispersion and the swirl[1] of the spray.

[1] When the spray enters the combustion chamber, it undergoes a spiraling effect, also called *swirl*, which affects the homogeneity and distribution of the fuel in the combustion chamber.

Another important aspect is injection timing. High injection frequencies cause severe vibrations and noise. The superimposition of such high frequencies can even reinforce these effects and lead to cracks in the injector. The frequency of injections is always a compromise. A high injection frequency causes more noise and stress while a lower injection frequency decreases the power output of the engine. By modifying the timing of injections, one can also achieve a higher power density and prevent back-leak of fuel into the injector.

The injector needle is also a crucial component. It allows for many modifications, such as the seat angle, the surface treatment (different coatings), the diameter, the spring load or the shape of the needle tip (truncated or not). The nozzle hole geometry can be modified by changing the angle, the shape or the diameter of the holes. Nozzle holes can also be tapered, sealed or coated. Additionally, the volume and shape of the sac also affect the spray behavior.

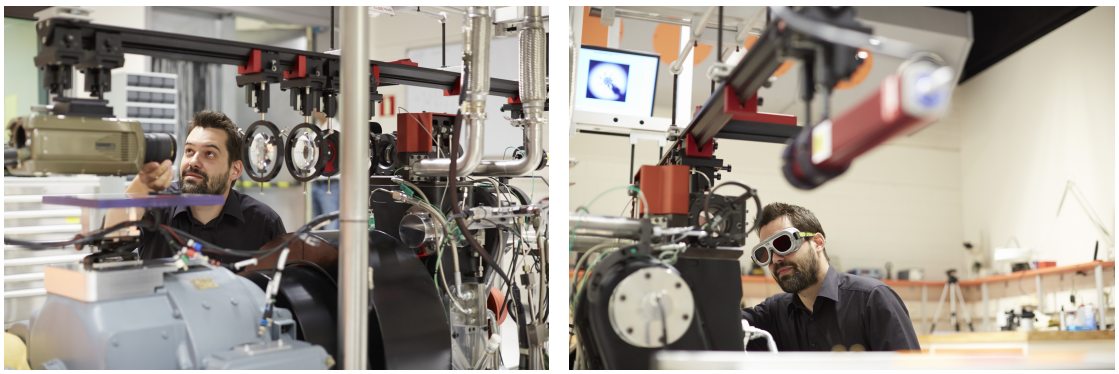


Figure 1.4: Optical single cylinder setup for Schlieren imaging at Delphi Technical Center Luxembourg

Despite the availability of methods from computational physics, such as flow simulations, it is still vital to see an injector perform under real world conditions. Once an injector prototype is available, it is installed in a laboratory setup, such as the one displayed in figure 1.4 and a visual analysis of the actual injection process is conducted.

2 Image Acquisition and Processing

This section will give a concise overview on the methods and tools that were used to acquire the data as well as the mathematical notations and image processing methods used in this work.

2.1 Interaction of Light with Media

Light propagates uniformly through homogeneous media. When we consider the geometrical model for optics, we observe four basic phenomena that occur when light interacts with matter in inhomogeneous media, namely absorption, reflection, refraction and scattering.

Absorption is the phenomenon when electromagnetic radiation is transformed into internal energy of the matter it interacts with. If the radiation is only absorbed partially, this is referred to as *attenuation*. *Reflection* is the change in direction of light at an interface between two different media such that the light returns into the medium from which it originated. *Refraction* is the phenomenon of light changing direction at the interface between two media, when entering the second medium. The angle of the change in direction is controlled by the speed of light in the two different media. The refractive index is defined as

Definition 2.1 Refractive Index

$$n = \frac{c_0}{c_1}$$

where c_0 and c_1 are the speed of light in the two respective media.

When light travels through a medium containing many small particles, the reflection and refraction phenomena that occur on a microscopic level (surface interactions on a particle) are referred to as *scattering* on the macroscopic level (when looking at a medium as a whole instead of the single particles).

In order to understand the spray characteristics, we have to apply image acquisition methods that provide us with information on the density of the spray. Scattering phenomena play a major role in the visual analysis of fuel sprays. One distinguishes between three different mathematical models to represent scattering phenomena. Scattering models are usually classified based on the average circumference of particles in the medium and the wavelength of the incident radiation. The interaction of light with particles that are small compared to the wavelength of the light is referred

to as *Rayleigh scattering*. If the particles are much larger than the wavelength of the light, one talks about *geometric scattering*. The *Mie scattering* model is preferred when the size of the particles roughly matches the wavelength of the light. For high pressure injections of Diesel fuel, the Mie scattering model is the most suitable one. It is only applicable if the droplets are more or less of spherical shape. However, because of the high injection pressure and the surface tension of fuel droplets, this assumption is reasonable.

2.2 Image Acquisition Techniques

Injected sprays are types of *compressible flows*[1]. To being able to capture the characteristics of an injection process and to evaluate it correctly, it makes sense to investigate different image acquisition techniques. These techniques enable us to capture data such as density changes in a medium that would remain hidden with conventional photography.

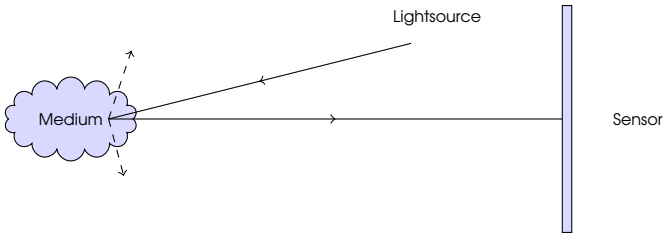


Figure 2.1: Illustration of scattering in visible light photography

2.2.1 Shadowgraphy

Shadowgraphy is an optical image acquisition method that allows to visualize non-uniformities in otherwise transparent media. Such non-uniformities can for example be caused by temperature or pressure differences in a medium. In regular visible light photography, these disturbances remain hidden. A simplified visible light photography setup is shown in figure 2.1.

In a shadowgraph setup, a parallel light source is placed on the opposite side of the sensor, with the medium in between the two. The medium refracts the light rays and casts shadows of the disturbances onto the sensor. Since the refractive index depends upon the density of a medium, the refraction of light waves in an otherwise transparent medium is visualized on the image plane. A simple illustration is displayed in figure 2.2.

[1] Compressible flows are flowing gases with variable density, caused by mixing dissimilar materials or by variations in speed, pressure or temperature

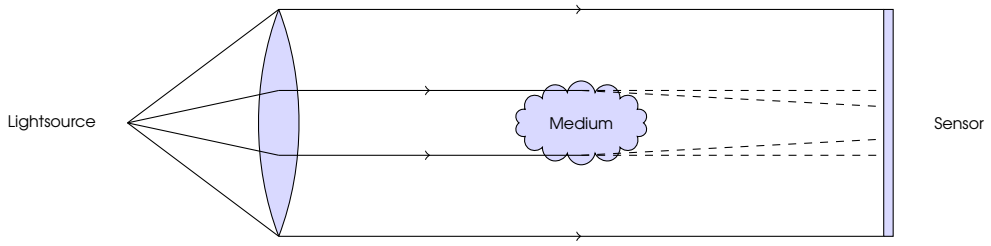


Figure 2.2: Illustration of optics in direct/parallel light shadowgraphy

2.2.2 Schlieren Photography

Schlieren imaging is based on the same principle as shadowgraphy, but uses a more sophisticated setup, which allows to reveal even more information. Human eyes have no way to discern phase differences in light beams. They only capture differences in amplitude and contrast. Schlieren imaging helps to translate phase differences into amplitude. The word Schlieren imaging is derived from the german word *Schliere* (smear). It describes gradient disturbances of inhomogeneous media. Optical inhomogeneities bend or refract light rays in proportion to their gradients of the refractive index of the respective materials. The illuminance level in a Schlieren image therefore responds to the first spatial derivative of the refractive index of the Schliere [Set01]. After the parallel light rays have traversed the medium, the light is refocused before hitting the image sensor. An obstacle, called a knife edge, is placed at this focal point. The rays that were refracted in the medium are partially blocked, which enables the observer to see the first derivative of the density of the medium in the direction of the knife-edge. If the knife is a straight edge, that is placed in orthogonal direction of the x/y-dimension of the image plane, the sensor will capture the gradient data in x/y-dimension. Instead of a rectangular knife edge, it is also possible to use an aperture shaped knife edge, which results in a mixture of the gradient information in both dimensions.

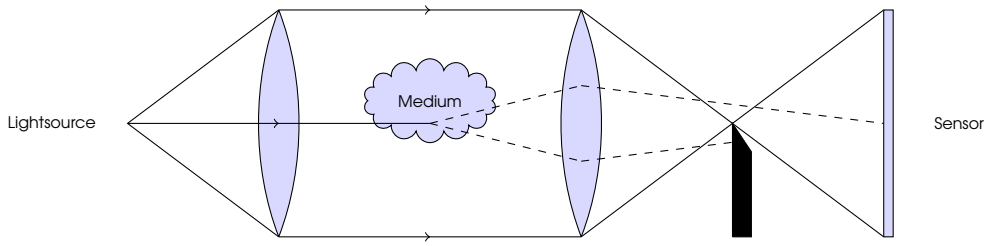


Figure 2.3: Illustration of optics in parallel light Schlieren photography

2.3 Image Acquisition

In order to record image data of an injection process, an elaborate and costly setup is necessary. The injector is mounted on a rig and injects the spray into an enclosed chamber. The chamber usually has multiple optical accesses for cameras and light sources. In order to prevent a combustion, the chamber is filled with nitrogen instead of air. The spray consists of conventional Diesel fuel. A Schlieren image sequence

recorded with the setup from figure 1.4 is shown in figure 2.11. All the Schlieren image sequences in this work were created using an aperture-shaped knife edge, thus mixing the first derivative density data in both spatial dimensions.

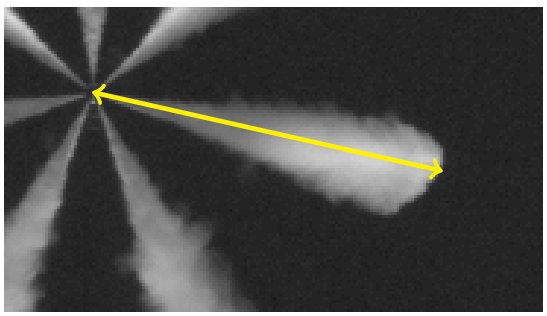
2.4 Objectives of Optical Analysis

The goal of optical spray analysis is to analyze the distribution and interactions of the spray, once it is released from the injector. The rate of a chemical reaction can be raised by increasing the surface area of a solid reactant [GW64], meaning that the reaction is sped up if the fuel is in a vaporized state. To analyze the spray behavior, it can therefore be helpful to distinguish between the formation of the liquid and the vapor phases. The goal is to have a uniform distribution of vaporized fuel in the chamber when the reaction starts. A high vaporization is also favorable, since large droplets will not be combusted completely and leave residues [Dry97]. It is also desirable for the combustion to occur before the fuel droplets get in contact with the walls of the combustion chamber, in order to prevent soot deposits.

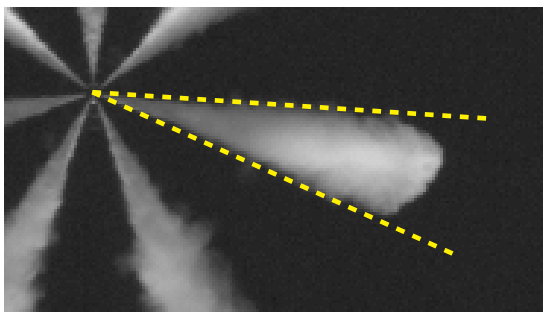
To asses the quality of an injection, we need a set of criteria and measurements. The most commonly used measurements are the *tip penetration*, the *dispersion angle*, the *spray volume* and the *spray homogeneity*.

The tip penetration measures the maximum distance between the exit point of the nozzle hole and the spray of a single jet, depicted in figure 2.4a. Since the recorded images are only a two-dimensional projection and the angle of the spray is not exactly parallel to the image plane, the angle has to be taken into account when computing this distance. The dispersion angle measures the angle up to which the spray disperses in lateral direction. The vertex of the dispersion angle is the center of the injector. An example can be seen in figure 2.4b.

A useful analysis of an injector design is only possible when the spray modeling is predictive. Ten repetitions (*cycles*) are performed for each injector under the same



(a) Tip penetration: The tip penetration measures the largest distance between the nozzle hole and the spray cloud.



(b) Dispersion angle: The dispersion angle measures the lateral propagation of a jet.

Figure 2.4: Characterization of spray patterns

conditions in a single run. Statistical measure such as the standard deviation and variance are used to make the data more reliable by reducing outliers.

2.5 Mathematical Representation of Image Data

Images can be mathematically represented as a function

Definition 2.2 Image

$$f(\boldsymbol{x}) : \Omega \to \mathbb{R}$$

where $\boldsymbol{x} := (x, y)^\top$ is the image plane and $\Omega \in \mathbb{R}^2$ is the image domain. For three-dimensional image data, we have $\boldsymbol{x} := (x, y, t)^\top$ and $\Omega \in \mathbb{R}^3$, where x and y are the spatial dimensions and t denotes the time direction. The co-domain of $f(\boldsymbol{x})$ describes the *intensity value* (*gray value*) at locations $\boldsymbol{x} := (x, y)^\top$ ($\boldsymbol{x} := (x, y, t)^\top$, respectively).

The intensity value is proportional to the amount of photons hitting the sensor during the acquisition of the image (see section 2.2). To being able to store and process these data, a *discretization* of both the image plane as well as the image domain are inevitable. A discrete image is defined as

Definition 2.3 Discrete Image

$$f(\boldsymbol{x}) := \Gamma \to \mathbb{N}$$

where $\Gamma := \{1, ..., n_x\} \times \{1, ..., n_y\}$ (or $\Gamma := \{1, ..., n_x\} \times \{1, ..., n_y\} \times \{1, ..., n_t\}$ for three-dimensional data) The elements of the set Γ are referred to as pixels for two-dimensional data (picture elements) and voxels for three-dimensional data (volume elements).

The discretization of the spatial domain is referred to as *spatial sampling*. The discretization of the image domain in temporal direction is referred to as *temporal sampling*.

As a result of the spatial sampling, one obtains data arranged in a pixel grid $\Gamma := \{1, ..., n_x\} \times \{1, ..., n_y\}$. As a result of the spatiotemporal sampling one obtains *frames* arranged in an *image sequence* on a three-dimensional grid $\Gamma := \{1, ..., n_x\} \times \{1, ..., n_y\} \times \{1, ..., n_t\}$. The variables n_x,n_y and n_t are the *image dimensions* in x, y and t direction, respectively. The *grid size* h_x, h_y and h_t represents the grid spacing between the data points in all the three dimensions.

For a two-dimensional image, we obtain a discretized image $f(x_i, y_j)$ with data points and spacing

$x_i := \left(i - \frac{1}{2}\right) h_x$ with $i \in \{1, ..., n_x\}$

$y_j := \left(j - \frac{1}{2}\right) h_y$ with $j \in \{1, ..., n_y\}$.

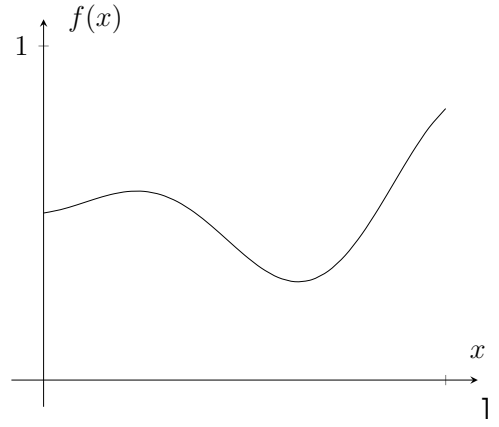
For a three-dimensional image, we obtain a discretized image $f(x_i, y_j, t_k)$ with data points and spacing

$x_i := \left(i - \frac{1}{2}\right) h_x$ with $i \in \{1, ..., n_x\}$

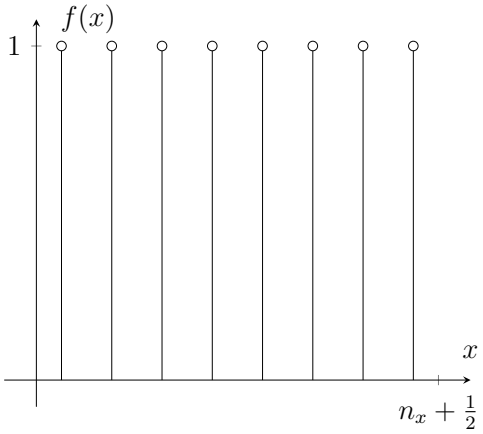
$y_j := \left(j - \frac{1}{2}\right) h_y$ with $j \in \{1, ..., n_y\}$

$t_k := \left(k - \frac{1}{2}\right) h_t$ with $k \in \{1, ..., n_t\}$

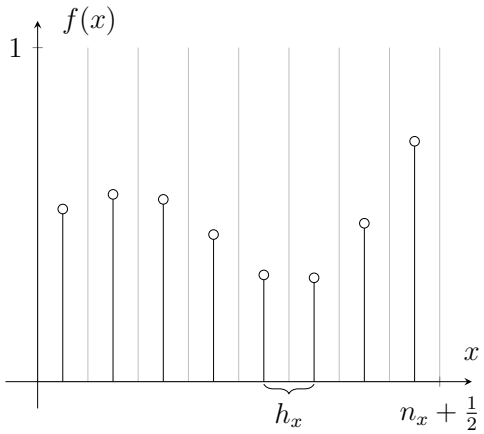
The discretization of the co-domain is referred to as *quantization*. Sampling and quantization of a continuous one-dimensional function is illustrated in figure 2.5.



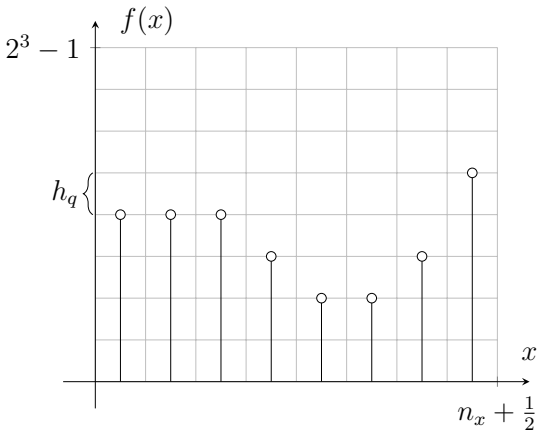
(a) A continuous signal



(b) Dirac Comb function, used for spatial sampling



(c) Spatial sampling of the function from a) by multiplication with the sampling function from b)



(d) Quantization of the sampled signal from c) with a bit depth of 3, resulting a range of values of $2^3 = 8$

Figure 2.5: Sampling and quantization of a one-dimensional signal

2.6 Frequency Filtering

Noise is always existent when recording images from electromagnetic radiation. This is especially true for data acquired with high speed cameras. To achieve extremely short exposure times, the camera sensor has to have a high sensitivity, which makes it even more susceptible to noise. Since noise mostly occurs in the high frequency spectrum, frequency filtering methods are a useful tool for removing noise in images. However, one has to be careful since any type of filtering usually also removes valuable image data in this respective part of the spectrum. A lowpass filter removes high frequencies from the signal. In the spatial domain, lowpass filtering can be achieved by convolving the signal with a Gaussian kernel [Wei15].

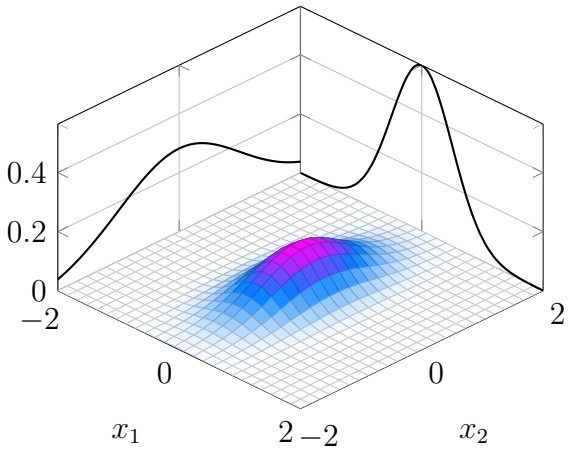


Figure 2.6: A two dimensional Gaussian kernel composed of two one-dimensional Gaussian kernels

It is useful to mention that one n-dimensional Gaussian Kernel is separable into n one-dimensional Gaussian kernels, which allows for a much more efficient computation. Additionally, Gaussian kernels are rotationally invariant, which means that they behave the same way, no matter how the image or the kernel are rotated in the image plane. A Gaussian kernel in two dimensions is depicted in figure 2.6. An image can be lowpass filtered by a simple convolution with a Gaussian kernel.

Definition 2.4 Gaussian Convolution

$$f(x,y,t) = (K_\sigma * f_0)(x,y,t) := \int_\Omega K_\sigma(\tilde{x},\tilde{y},\tilde{t}) f_0(x-\tilde{x}, y-\tilde{y}, t-\tilde{t})\, d\tilde{x}\, d\tilde{y}\, d\tilde{t}$$

where $K_\sigma = K_{\sigma_x} \cdot K_{\sigma_y} \cdot K_{\sigma_t}$ is the product of three one-dimensional Gaussian kernels

$$K_{\sigma_i}(i) := \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left(-\frac{i^2}{2\sigma_i^2}\right)$$

with standard deviation σ_i in all three dimensions.

The effects of filtering a gray value image with a Gaussian kernel are depicted in figure 2.7.

$\sigma_{xy} = 0.0$ $\sigma_{xy} = 1.0$ $\sigma_{xy} = 2.0$

Figure 2.7: Lowpass filtering of image data: Effect of modifying the standard deviation of the Gaussian Kernel

2.7 Partial Differential Equations

Partial differential equations (PDEs) are a versatile tool in image processing. They enable us to use optimization approaches from the calculus of variations to solve image analysis problems. They also allow us to formulate our model in the continuous setting by postponing the discretization to a later stage. The partial derivative of a function f in dimension x is given as

Definition 2.5 Partial Derivative

$$\frac{\partial}{\partial x} f = \partial_x f = f_x$$

The first derivative of a vector valued function $f(\boldsymbol{x})$, also called the *gradient*, is defined as

Definition 2.6 Gradient

$$\nabla_n f = \begin{pmatrix} f_{x_1} \\ \vdots \\ f_{x_n} \end{pmatrix}$$

The gradient of an image points in direction of the greatest change in value. Therefore, the gradient is a useful tool to detect edges in image data. The *magnitude of the gradient* is defined as:

Definition 2.7 Gradient Magnitude

$$|\nabla_n f| = \sqrt{f_{x_1}^2 + \cdots + f_{x_n}^2}$$

In contrast to the gradient, the gradient magnitude is rotationally invariant, since it discards directional information. If it is clear from the context, we just write ∇ instead of ∇_n.

The *divergence* is a vector operator that describes the magnitude of sources or sinks in vector-valued data, defined as

Definition 2.8 Divergence

$$div\,\boldsymbol{j} = \nabla_n^\top \boldsymbol{j} = (\partial_{x_1}, \cdots, \partial_{x_n}) \begin{pmatrix} j_1 \\ \vdots \\ j_n \end{pmatrix}$$

where $\boldsymbol{j} = (j_1(x_1, \cdots, x_n), \cdots, j_n(x_1, \cdots, x_n))^\top$

In some cases, we will have to apply finite differences in order to approximate derivatives. The *forward-*, *backward-* and *central differences* can be used to approximate the first derivative at location i. They will be denoted as follows

Definition 2.9 Forward Difference

$$f_x^+(i) = \frac{f_{i+1} - f_i}{h_x}$$

Definition 2.10 Backward Difference

$$f_x^-(i) = \frac{f_i - f_{i-1}}{h_x}$$

Definition 2.11 Central Difference

$$f_x(i) = \frac{f_{i+1} - f_{i-1}}{h_x^2}$$

2.8 Variational Methods

Variational methods are an image transformation that satisfies an optimality criterion based on an *energy functional* in the general form of

Definition 2.12 Energy Functional

$$E(u) = \int_\Omega F(x_1, \cdots, x_n, u, u_{x_1}, \cdots, u_{x_n})\, d\boldsymbol{x}$$

The functional rates the quality of a function w.r.t. certain model assumptions. The goal is to find those parameters that allow the energy to obtain its minimum/maximum value. The function that attains the minimum/maximum value and therefore fits best to the model assumptions is called the *minimizer*/*maximizer*.

The first variation of the energy, better known as *Euler Lagrange equation* is given as

Definition 2.13 Euler Lagrange Equation

$$\nabla E = F_u - \partial_{x_1} F_{u_{x_1}} - \cdots - \partial_{x_n} F_{u_{x_n}}$$

with natural boundary conditions $\boldsymbol{n}^\top \begin{pmatrix} F_{u_{x_1}} \\ \vdots \\ F_{u_{x_n}} \end{pmatrix} = 0$ at the image boundary with normal vector $\boldsymbol{n}$

∇E points in direction of the greatest positive change. Extrema occur at locations where $\nabla E = 0$. If it is strictly convex, it has a unique minimizer.

2.9 Optical Flow Computation

Detecting motion in an image sequence can be an important source of information. The goal of *optical flow* computation is to estimate apparent motion patterns in an image sequence and or motion patterns of the camera itself. The motion information will be represented as a two-dimensional vector field, the *optical flow field* (also *displacement field*, *image velocity field*). Using only one camera in each setup, one obtains a two-dimensional flow field which is computed from the projection of the actual motion onto the camera plane.

To find correspondences between objects at different points in time of an image sequence, one needs to extract features from the image sequence and then search for correspondences between these features. The two most obvious features in image data also turn out to be the most useful ones for our purpose. The gray value of the image sequence and its gradient. For the gray value, one obtains the so called *brightness constancy assumption*:

Definition 2.14 Brightness Constancy Assumption

$$\begin{aligned} f(x, y, t) &= f(x + u, y + v, t + 1) \qquad \Leftrightarrow \\ 0 &= f(x + u, y + v, t + 1) - f(x, y, t) \end{aligned}$$

This formula postulates the assumption that the gray value $f(x, y)$ remains constant between the frame t at image coordinates (x, y) and the frame $t+1$ at the displaced image coordinates $(x + u, y + v)$.

Since the image gradient is vector-valued, we obtain two constancy assumptions:

Definition 2.15 Gradient Constancy Assumption

$$f_x(x, y, t) - f_x(x + u, y + v, t + 1) = 0$$

$$f_y(x, y, t) - f_y(x + u, y + v, t + 1) = 0$$

The *gradient constancy assumption* makes the method more robust under illumination changes. However, while both assumptions are invariant under global additive illumination changes, the gradient constancy assumption may prove to be disadvantageous for situations with rotational motion, since it is inherently built on directional information. We are able to include both of these assumptions in our model by assigning weights to the two terms. The weights have to be adjusted depending on the type of image data.

As detailed in figure 2.8, instead of the ordinary quadratic penalization, we use a a *subquadratic penalizer* Ψ_D for the data term, introduced by [BA91]. This reduces the penalty for outliers and helps to make the method more robust. A more detailed introduction to subquadratic penalization in the context of image processing is given in [BWS05]. Such a function Ψ_D should be strictly convex, differentiable, positive, increasing and symmetric along the y-axis.

At many locations of the image sequence, it will not be possible to compute a flow vector, since there is not enough information available in the data. In order to still obtain a dense solution, the model is augmented with a *smoothness term*. This term allows us to model the assumption that the flow field only varies smoothly in spatial direction. The parameter that controls the weighting of the smoothness term is referred to as *regularization parameter*. We apply the same subquadratic penalization as in the data term and obtain an *isotropic flowdriven smoothness assumption* in the spatiotemporal domain. The concept of using a smoothness assumption on the optical flow field was introduced by [SH89].

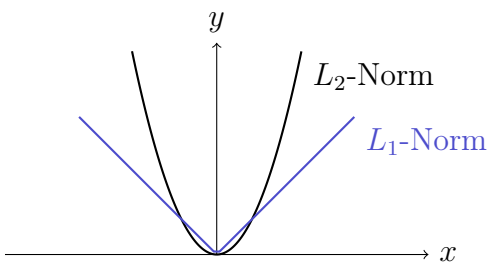


Figure 2.8: Error measure functions: $|x|$ denotes the deviation from the assumption, while the value of y denotes the applied penalty. As the deviation becomes larger, the quadratic L_2-Norm penalizes deviations much more severely compared to other regularizers such as the linear L_1-Norm

Since we are working with image derivatives, we have to make sure that the function is differentiable. As explained in section 2.6, the strong regularization properties of a Gaussian filtering ensure that the signal becomes infinitely many times differentiable. Finally, we are able to formulate the optical flow method of Brox et al. [Bro+04]

Definition 2.16 Optical Flow Method of Brox et al.

$$E(\boldsymbol{w}) = \int_{\Omega \times T} \Psi_D \left((f(\boldsymbol{x}) - f(\boldsymbol{x}+\boldsymbol{w}))^2 + \gamma \left| \nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{x}+\boldsymbol{w}) \right|^2 \right) d\boldsymbol{x} + \alpha \int_{\Omega \times T} \Psi_S \left(|\nabla_3 \boldsymbol{u}|^2 + |\nabla_3 \boldsymbol{v}|^2 \right) d\boldsymbol{x}$$

To find an optimal solution for the optical flow field $\boldsymbol{w}$, we use the minimization strategy explained in section 2.8. One obtains the following Euler Lagrange equations:

Definition 2.17 Euler Lagrange Equations of $E(\boldsymbol{w})$

$$\begin{aligned} 0 = {} & \Psi'_D \left((f(\boldsymbol{x}) - f(\boldsymbol{x}+\boldsymbol{w}))^2 + \gamma \left| \nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{x}+\boldsymbol{w}) \right|^2 \right) \\ & \cdot \Big(f_x(\boldsymbol{x}+\boldsymbol{w}) (f(\boldsymbol{x}) - f(\boldsymbol{x}+\boldsymbol{w})) \\ & + \gamma f_{xx}(\boldsymbol{x}+\boldsymbol{w}) (f_x(\boldsymbol{x}) - f_x(\boldsymbol{x}+\boldsymbol{w})) \\ & + \gamma f_{yx}(\boldsymbol{x}+\boldsymbol{w}) (f_y(\boldsymbol{x}) - f_y(\boldsymbol{x}+\boldsymbol{w})) \Big) \\ & + \alpha \operatorname{div} \left(\Psi'_S \left(|\nabla_3 \boldsymbol{u}|^2 + |\nabla_3 \boldsymbol{v}|^2 \right) \nabla_3 \boldsymbol{u} \right) \end{aligned}$$

$$\begin{aligned} 0 = {} & \Psi'_D \left((f(\boldsymbol{x}) - f(\boldsymbol{x}+\boldsymbol{w}))^2 + \gamma \left| \nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{x}+\boldsymbol{w}) \right|^2 \right) \\ & \cdot \Big(f_y(\boldsymbol{x}+\boldsymbol{w}) (f(\boldsymbol{x}) - f(\boldsymbol{x}+\boldsymbol{w})) \\ & + \gamma f_{xy}(\boldsymbol{x}+\boldsymbol{w}) (f_x(\boldsymbol{x}) - f_x(\boldsymbol{x}+\boldsymbol{w})) \\ & + \gamma f_{yy}(\boldsymbol{x}+\boldsymbol{w}) (f_y(\boldsymbol{x}) - f_y(\boldsymbol{x}+\boldsymbol{w})) \Big) \\ & + \alpha \operatorname{div} \left(\Psi'_S \left(|\nabla_3 \boldsymbol{u}|^2 + |\nabla_3 \boldsymbol{v}|^2 \right) \nabla_3 \boldsymbol{v} \right) \end{aligned}$$

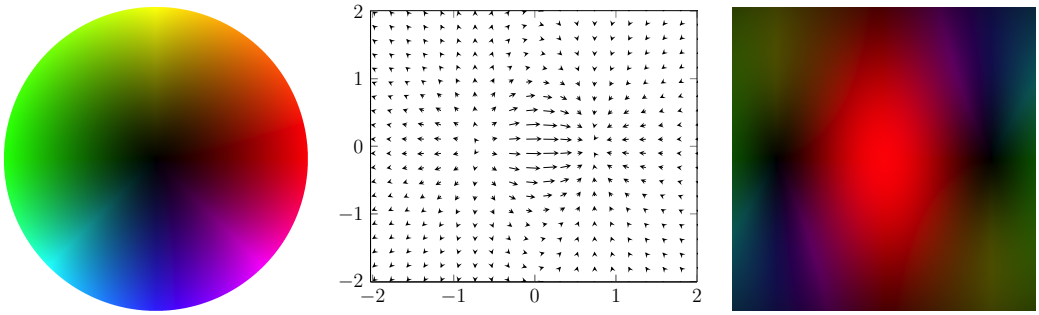
To find a numerical solution, we now apply a coarse-to-fine strategy as explained in detail in [Pap+06].

To represent a displacement field graphically, instead of the *arrow plot* that is commonly used in physics (also called *vector plot* or *quiver plot*), shown in Figure 2.9b, we are going to use a pseudocolor representation scheme from Bruhn [Bru06], where the hue represents the direction of the vector $(u(x,y,t), v(x,y,t))^\top$ at location $(x,y)^\top$ and time t, while the saturation represents its magnitude. A distinct advantage of the pseudocolor representation is that it allows to display motion information in the vector field much more precisely (especially at motion discontinuities) compared to the coarse representation of the arrow plot.

The optical flow method of Brox et al. was implemented in MATLAB as well as C, using the approach presented in [Bro+04]. An example of the optical flow field

between two frames of a Schlieren sequence is shown in figure 2.10.

The optical flow field of the Schlieren sequence from figure 2.11 computed with the method from definition 2.16 is shown below. Figure 2.12 shows the optical flow of a sequence in color-coded representation, figure 2.13 shows the same sequence with an overlay of the arrow plot of the flow field. The following parameters were used: $\alpha = 10.0$, $\gamma = 0.0$, $\sigma_x y = 0.4$, $\sigma_t = 0.2$

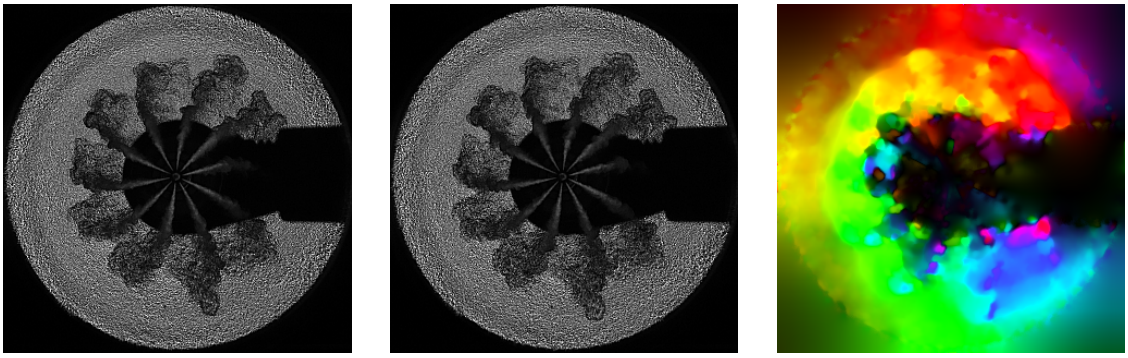


(a) Pseudocolor scheme to represent displacement vectors

(b) Gradient vector field of the function $x \exp(-x^2-y^2)$ using an arrow representation

(c) Gradient vector field of the same function as in b) using the color-coded representation from a)

Figure 2.9: Visualization of vector fields using a color-coded representation



Frame 22 Frame 23 Optical flow

Figure 2.10: Optical flow result of Brox et al. method between two frames of a sequence. Parameters used: $\alpha = 8.0$, $\gamma = 0.2$, $\sigma_x y = 0.6$, $\sigma_t = 0.3$

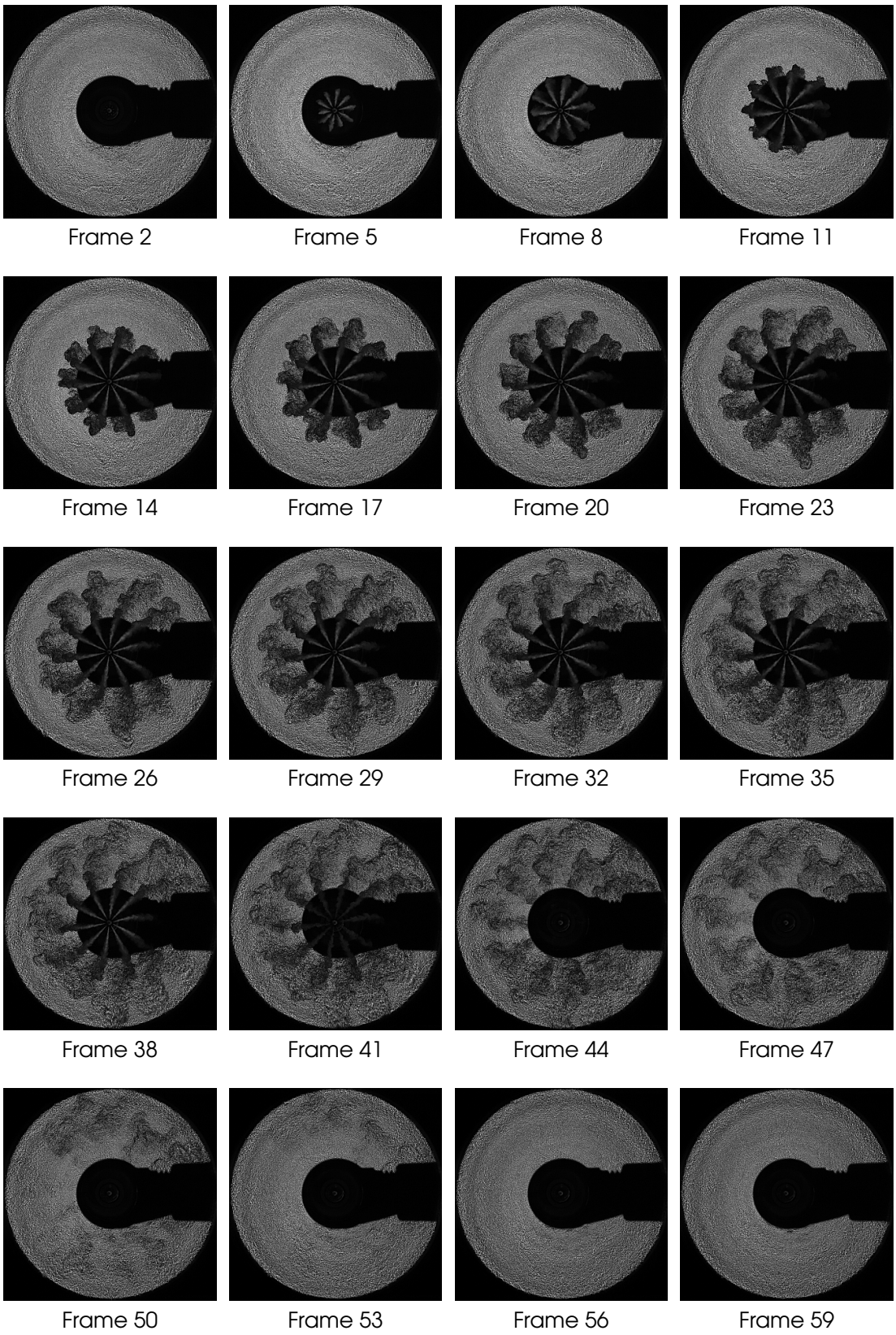


Figure 2.11: Schlieren sequence of an injection process (Sequence S3-1-6)

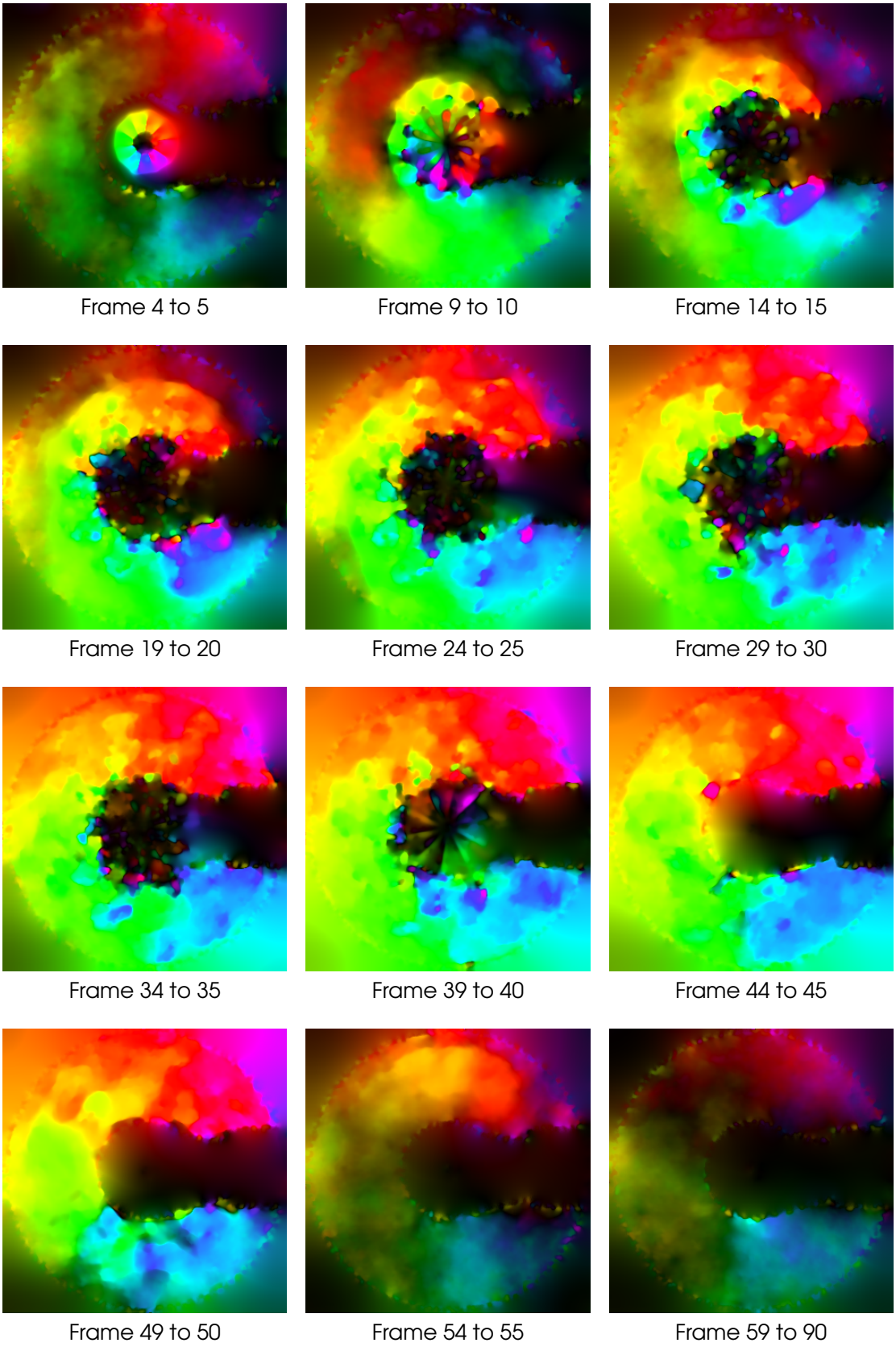


Figure 2.12: Color coded representation of the optical flow field of sequence S3-1-6

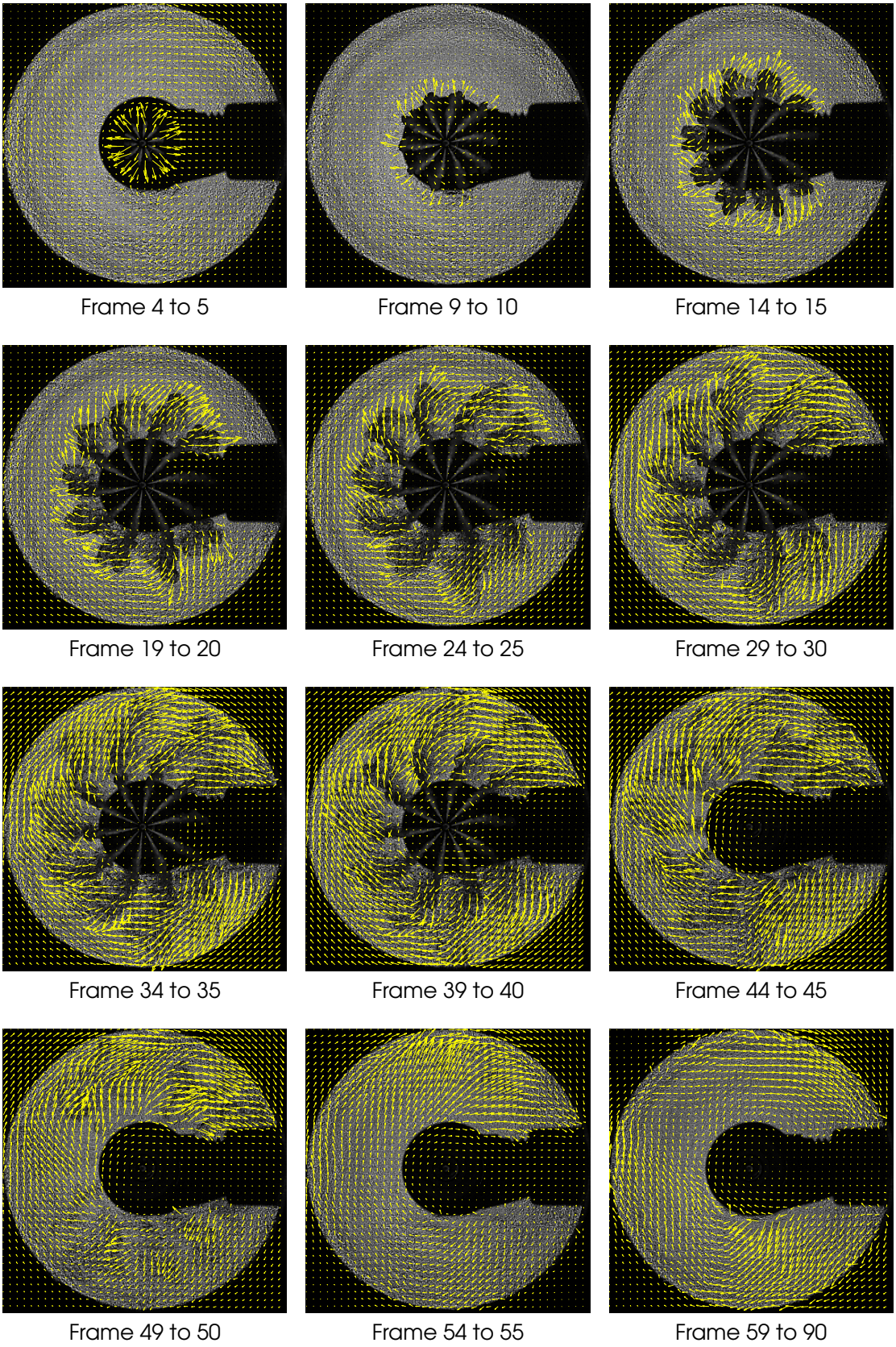


Figure 2.13: Arrow representation of the optical flow field of sequence S3-1-6

3 Optical Analysis Part I: Feature Detection

The goal of optical spray analysis is to evaluate the characteristics of the spray when injected into the combustion chamber. Features of interest are for example the dispersion angle of the jets or the penetration curve of each jet. The first step in processing the data is to extract suitable features from the sequence.

3.1 Histogram Equalization

Depending on the type of input data, it can make sense to perform a *histogram equalization* on the input image data. Histogram equalization is a method to increase the global contrast of the image, using the image's histogram [Wei15]. To perform a histogram equalization on a sequence, we use MATLAB's `histeq` method. An example of an image with its histogram before histogram equalization (figure 3.1) and after histogram equalization (figure 3.2) is shown below.

In a histogram equalized sequence, it becomes easier to distinguish the spray from the background, since the difference in gray value between the different phases is increased. The segmentation also becomes more robust under parameter adjustments which makes the segmentation result more precise and reliable. Histogram equalization is an idempotent image processing operation, meaning that applying it to the same input data multiple times has the same effect as applying it once.

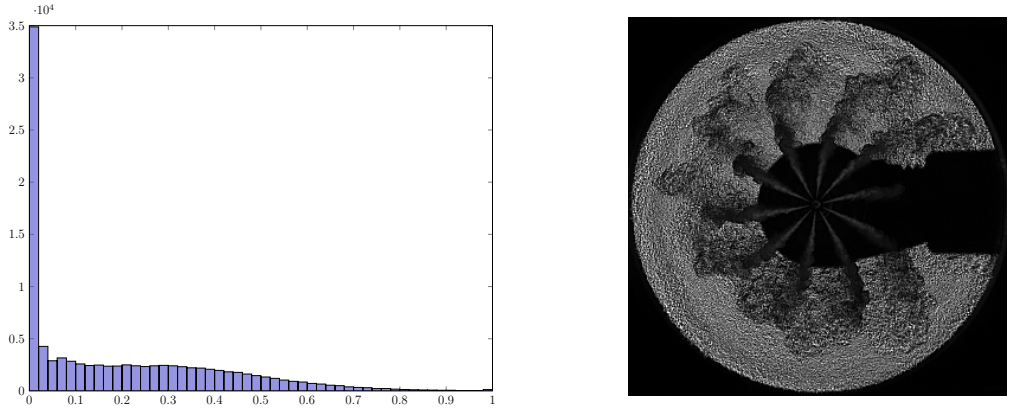


Figure 3.1: A Schlieren sequence and its histogram before performing a histogram equalization

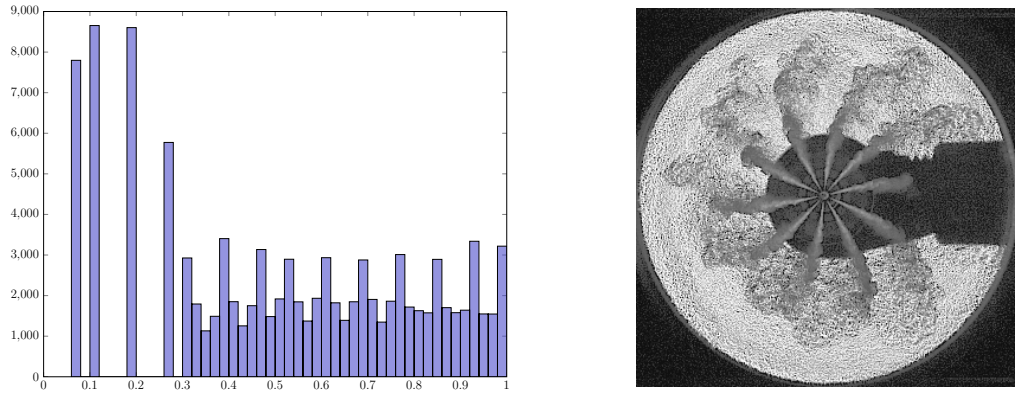


Figure 3.2: A Schlieren sequence and its histogram after performing a histogram equalization

3.2 Detection of First Sight of Light

Because of the need for high temporal resolution and steady illumination, the background laser light source and the electronic controls of the injector are coupled and synchronized with the camera. The camera usually starts recording before the laser light reaches the image sensor. Since these dark frames of the sequence are of no interest and might interfere with other image processing operations, they should be discarded.

The *first sight of light* (FSOL) t_{FSOL} is determined by comparing the average gray value of subsequent frames within a certain time window. If we know that the FSOL occurs within the first n frames of the sequence, we only consider the time interval $[t_0, t_n]$. Since we do not have any prior information on the brightness of the light source or the distribution of light over the image plane, we use a simple thresholding to determine t_{FSOL} The sequence is processed in succession from t_0 to t_n. If the difference exceeds a threshold thr_{FSOL} between frames t_i and t_{i+1}, the light reaches the sensor in frame t_{i+1} and we assign $t_{FSOL} = t_{i+1}$.

$t = 1$ $t = 2$ $t = 3$

Figure 3.3: The first three frames of a shadowgraph sequence. The first sight of light is detected as $t_{FSOL} = 2$.

The FSOL detection algorithm was implemented in MATLAB. The user interface also allows to set t_{FSOL} manually, in case the automatic detection should be unreliable.

3.3 Spatial Localization of the Injector Nozzle

Because of the circular nature of the images, it makes sense to perform certain image processing operations in polar coordinates. To this end, it is necessary to detect the location of the nozzle tip of the injector in the image sequence.

There are some aspects to consider, when trying to detect the center of the nozzle:

- The nozzle itself is not visible. Since the injector is only illuminated from the opposite side of the camera, the nozzle lies in the shadow of the injector.
- The nozzle is not guaranteed to be located at the center of the image.
- Depending on how the injector is installed in the mounting, the outer circle (the shadow of the optical access to the chamber) is not necessarily centered around the inner circle (the shadow of the injector).
- The outer circle is not always guaranteed to be located completely inside the boundaries of the images.

By performing a circular *Hough transformation*, we are able to find circular structures in images. Since an injector, by design, always has a circular shape when seen from below, we can apply a circular Hough transformation to find this circle. If we assume that the injector is aligned in normal direction to the camera plane and the nozzle is located at the center of the injector shadow, we are able to estimate the coordinates of the nozzle. Depending on the shape of the obstacles, the algorithm might detect multiple circles in the image. In such a case, the circle with its center closest to the image center $(\frac{n_x}{2}, \frac{n_y}{2})$ in terms of Euclidean distance is chosen.

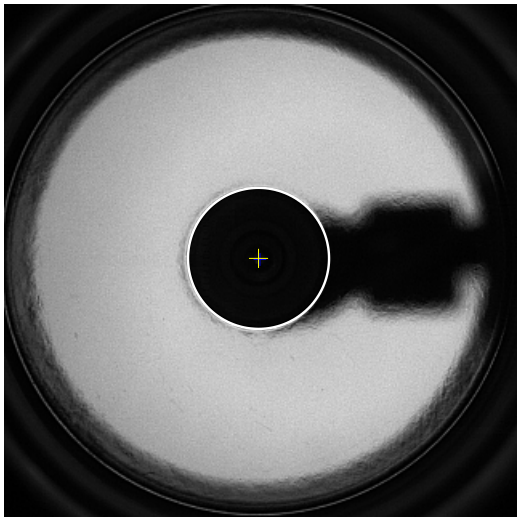
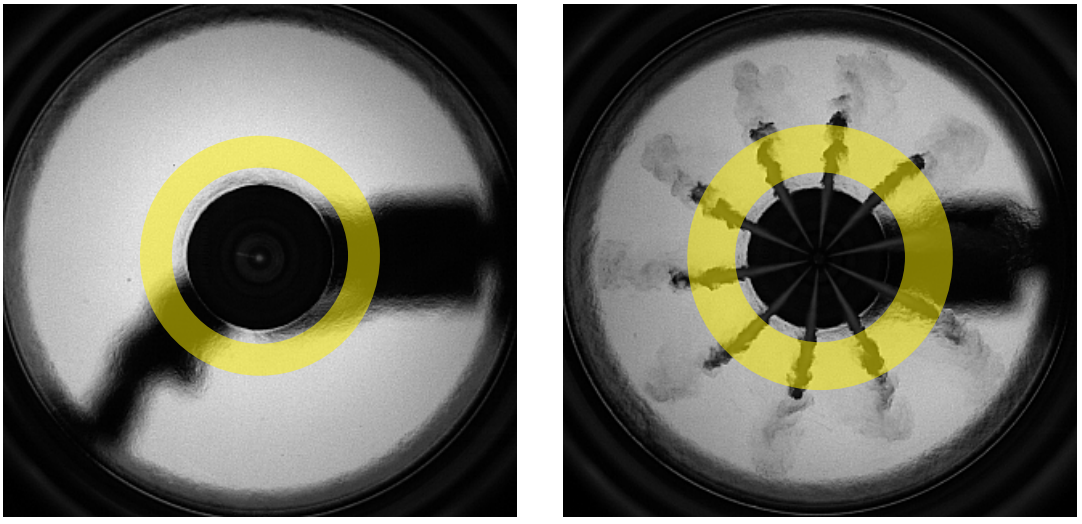


Figure 3.4: Result of a Hough transformation

Our implementation of the Hough transformation is partially based on the MATLAB function `imcircles`.

3.4 Localization of Obstacles

Since the injector has to be mounted to the rig and has to be connected to a fuel supply, the light travelling alongside the shadow of the injector is partially obstructed. Only very limited information of the jets in front of these obstacles can be obtained. That is why it makes sense to identify which jets are located in front of obstacles and to neglect those jets for further processing. Depending on the setup, there can be multiple obstacles at arbitrary angles and locations. We use only a single frame, t_{FSOL}, to automatically identify the angular range in which the obstacles are located. Since we know that for all frames $t > t_{FSOL}$, the light source is visible and that the first sight of fuel will occur after the first sight of light $t_{FSOF} > t_{FSOL}$, the frame t_{FSOL} gives an unperturbed view of the obstacles.



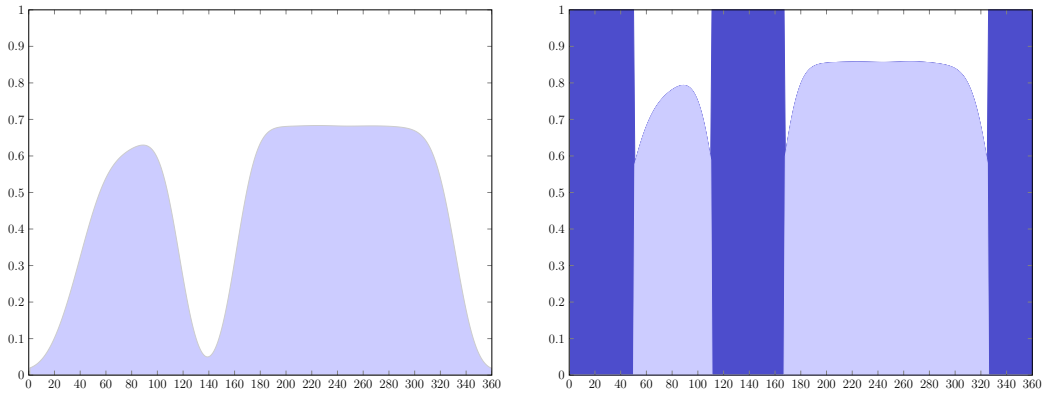
(a) The frame t_{FSOL} is used to detect obstacles. The area that is transformed to polar coordinates is displayed with a color overlay.

(b) The area marked with the color overlay is converted to the polar coordinate system. Using the polar data, it is possible to partition the jets.

Figure 3.5: Conversion to polar coordinates

Since we are interested in the angular range of the obstacles, we convert the colored area of the image shown in figure 3.5a to the polar domain, centered around the nozzle. The resulting image in polar coordinates is shown in figure 3.8. The vertical axis represents the angular axis of the polar image while the horizontal axis represents the radial axis of the polar domain. To account for noise and variations in shape of the obstacles, we perform a dimensionality reduction, by computing the mean of the values in radial dimension. One obtains a one-dimensional function in

Figure 3.6: The marked area from figure 3.5a transformed to polar coordinates.



(a) The polar image from figure 3.5a after dimensionality reduction.

(b) The obstructed angular areas that are below the threshold $thr_{obstacle}$ are marked with dark blue.

Figure 3.7: Obstacle curve in polar coordinates

the domain $[0, 360]$, shown in figure 3.7a. The curve reflects the average gray value along the donut shaped area from figure 3.5a. The values of this function which are below a threshold $thr_{obstacle}$ are considered to be obstructed, since the light from the background light source does not reach them. Consequently, we will ignore the jets located inside these angular areas (shown in figure 3.7b).

In addition to this automatic detection algorithm that was implemented, our MATLAB graphical user interface also allows to define these angular areas manually.

3.5 Angular Partitioning of the Jets

In order to partition the jets into jet regions, it is necessary to identify the number and angles of the jets. As mentioned before, we can assume that the number of jets N_{Jets} is within the interval $[3, 12]$ and that the nozzle holes are distributed at equidistant angle around the nozzle tip. Since the injector is backlit, the nozzle holes cannot be detected visually. However, using a similar polar coordinate conversion as before, we are able to partition the jets themselves. Instead of the frame t_{FSOL}, we now use a frame where the jets are already visible. The frame $t_{FSOF} + 15$, 15 frames after the first sight of fuel is shown in figure 3.5b.

To detect the angles of the jets, we again convert the region shown in figure 3.5b to polar coordinates and obtain the gray value curve f_{ang} seen in figure 3.9.

Figure 3.8: Representation in polar coordinates of the marked area from figure 3.5b.

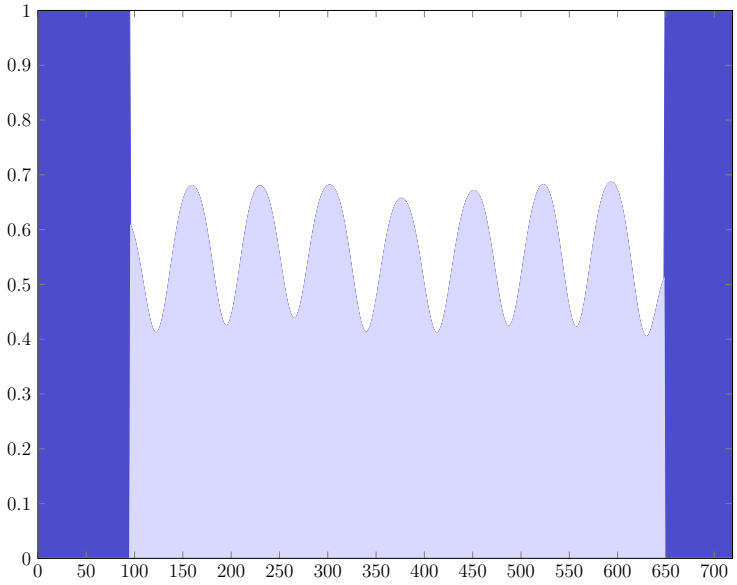


Figure 3.9: Dimensionality reduction of the polar image from figure 3.8 by averaging over the radial dimension of the polar plot. We obtain a one dimensional function of the average gray value in polar dimension. The local maxima represent the bright areas of the polar image (the background area between jets), while the local minima correspond to the dark areas of the polar image (the jets).

The locations of the jets now correspond to the local minima of the function. We compute the angle by fitting the equidistant points to the minima of the function. This can be formulated as a minimization problem:

Definition 3.1 Angular Jet Partitioning

$$\underset{\Phi}{\operatorname{argmin}} \sum_{i=1}^{numjets} f_{ang}(i) \tag{3.1}$$

with $i \in [0, 360]$ where $j * \Phi$ is the desired angle for jet j. A solution for this optimization problem and the resulting partitioning are shown in figure 3.10.

Our MATLAB implementation also allows to set the number of jets and the spacing of the jets manually. If a jet region is only partially obstructed, it is possible to keep

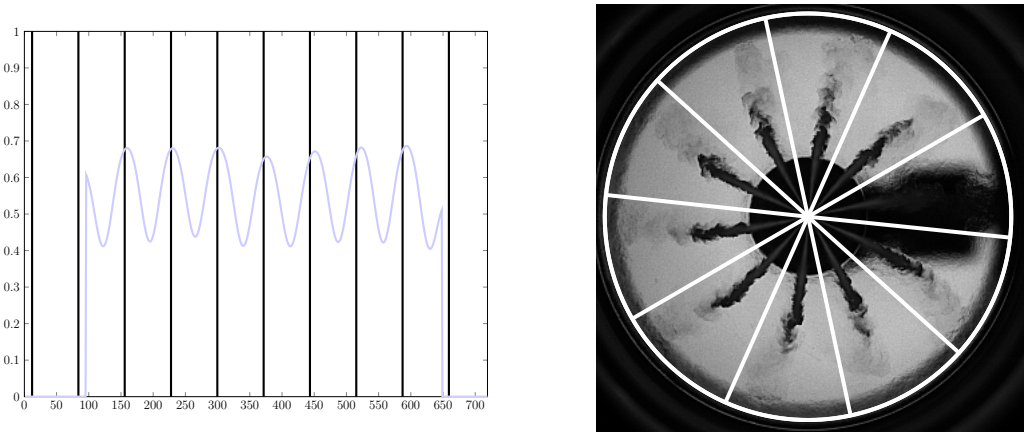


Figure 3.10: Partitioning result of sequence 3-1-7 frame 18 using the method from 3.1

the region, but restrict it by a certain degree on each side.

3.6 Detection of First Sight of Fuel

To being able to correctly measure the propagation of the jets, it is necessary to determine the exact point in time when the *first sight of fuel* (FSOF) occurs, denoted as t_{FSOF}. The detection of t_{FSOF} proves to be a difficult task, since the nozzle is illuminated differently, depending on the setup. In some setups, the nozzle tip might be coated to prevent reflections. To determine t_{FSOF}, we will only consider a circular region around the nozzle and discard the rest of the image, since it doesn't contribute any useable information.

By performing the Hough transformation from section 3.3, we obtain the radius r_{inj} and the center point $\left(c_{inj_x}, c_{inj_y}\right)$ of the circle that is cast by the shadow of the injector. The masked region consists of a donut shaped area with a radius between $0.1 * r_{inj}$ and $0.5 * r_{inj}$ We can use prior information, since we already know that t_{FSOF} will occur in the time window $[t_{FSOL} + 1, t_{FSOL} + n]$, where n is usually 15. Therefore we only consider this subsequence of 15 frames.

To detect the first sight of fuel, we will use a *background subtraction*. Since the background is static, we are able to use a *frame differencing method*, where t_{FSOL} is chosen as the *reference frame* (or *background frame*). We compute for each frame t the average difference, shown in figure 3.13.

$$\sum_{i=1}^{n_x} \sum_{j=1}^{n_y} |f_{i,j,t_{FSOL}} - f_{i,j,t}| \tag{3.2}$$

When this difference rises above a threshold thr_{FSOF} at time t, we know that the first fuel is visible and we assign $t_{FSOF} = t$.

3.7 Detection of the End of the Injection

Using the same method as for the detection of the first sight of fuel, it is also possible to determine the point in time when the injection stops. We know that the *last sight of fuel* (LSOF) occurs shortly after the end of the injection. We can now automatically crop the sequence which facilitates the computation.

Figure 3.11: The area marked with the color overlay used to compute the background subtraction for the detection of the first sight of fuel.

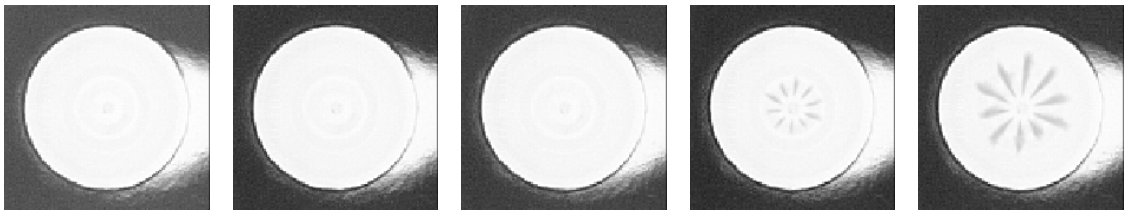


Figure 3.12: The first five frames of the injection sequence 3-1-2 (inverted for better visibility). The first sight of fuel occurs at $t = 4$.

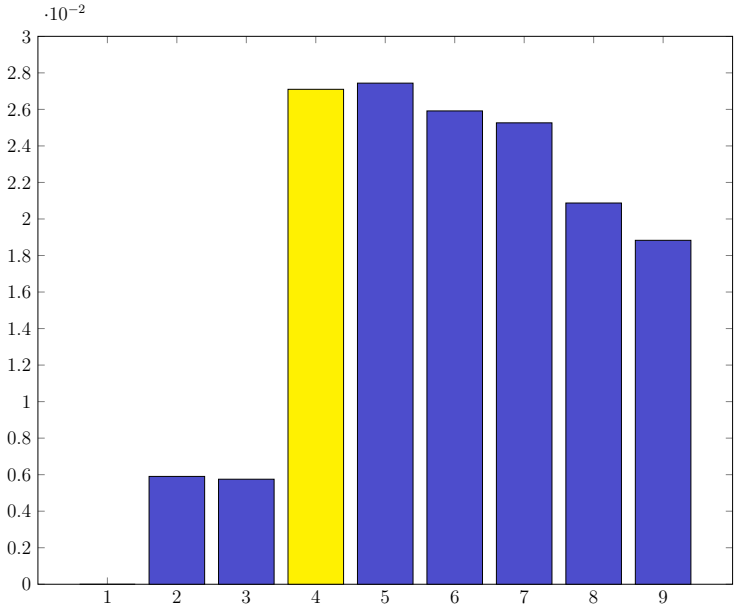


Figure 3.13: Bar plot of the background subtraction from sequence 3-1-2. The x-axis shows the single frames while the value on the y-axis reflects the difference in gray value between frames t and frame t_{FSOL}. The bar marked in yellow corresponds to the frame that is selected as t_{FSOF}. The according images are shown in figure 3.12.

4 Optical Analysis Part II: Image Segmentation

In order to correctly analyze the data, it is crucial to distinguish the liquid and vapor phases from each other as well as from the background. To this end, we will perform a *segmentation* of the image sequence. Segmentation is the process of partitioning an image into a set of distinct regions, which differ in some qualitative or quantitative way. The segmentation of visual data is an essential step in nearly all higher level object recognition tasks.

4.1 Active Contour Models

To perform a segmentation on the injection sequences, we will implement an *active contour model* (also called *snakes*, *energy minimizing curves*). The essential idea of active contour models is to evolve a curve in direction of an object boundary subject to certain constraints on the curve itself (*internal energy/force*) as well as on the image data (*external energy/force*, *image force*). Active contours were introduced by Kass, Witkin and Terzopoulus [KWT88].

The requirements on such an algorithm are that it should be robust under illumination changes, that the model parameters should be easy and intuitive to understand and that the method should not rely on any manual interaction besides the choice of the parameters. Also, the computational complexity and the memory requirements need to be reasonable, in order to being able to handle frequent and large amounts of data.

4.2 Level Set Representation

The segmentation methods that we are going to investigate make use of an implicit level set formulation in an Eulerian framework to represent a segmentation mathematically. This allows numerical computations of contours such as curves and (hyper-)surfaces on a Cartesian coordinate system without the need to parameterize these objects. This representation offers many advantages compared to the parametric formulation in a Lagrangian framework, as we will see later. Level set methods are popular in fields such as medical image analysis and were introduced by Osher and Sethian [OS88].

A segmentation algorithm can be expressed as a *curve evolution*. Let us consider the general case of a curve evolution

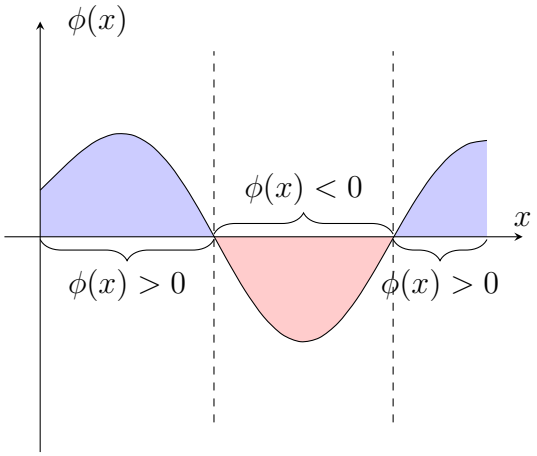


Figure 4.1: A level set function: The x-axis line represents the level set. The dashed lines represent the locations of the two contour points. The level set function partitions the domain into three segments. The two segments where $\phi > 0$ belong to the foreground phase (blue). The segment in the middle where $\phi < 0$ belongs to the background phase (red).

Definition 4.1 Curve Evolution

$$\boldsymbol{c}_t = \beta(\kappa)\,\boldsymbol{n}$$

where κ is the curvature, $\boldsymbol{n}$ is the normal vector and β is some function that controls the local speed of the evolution.

As an example for a curve evolution, let us consider the *motion by mean curvature*, which is the underlying principle of all segmentation techniques considered in this work.

Definition 4.2 Mean Curvature Motion

$$\overrightarrow{V} = -b\kappa\overrightarrow{N}$$

where κ is the curvature, b is a constant and $\overrightarrow{N}$ is a vector field representing the local unit normal.

This definition tells us that the interface moves in normal direction with a velocity proportional to its curvature. If $b > 0$, the motion will point inwards of the curve and the interface will shrink. When $b < 0$, the curve will expand. This case may be ill-posed and we do not need to consider it further in our context.

Instead of representing the segmentation process as a curve evolution, we can embed the curve evolution into a higher dimensional evolution of a *level set function*. The level set equation is given as

Definition 4.3 Level Set Equation

$$\phi_t + \overrightarrow{V}\nabla\phi = 0$$

where ϕ is the so called level set function and $\overrightarrow{V}$ is an arbitrary velocity field.

To elevate a curve to the higher-dimensional level set representation, we can apply the *Euclidean signed distance function* to the curve, defined as:

Definition 4.4 Euclidean Signed Distance Function

$$\psi(\boldsymbol{x}) = \begin{cases} |\boldsymbol{x} - \phi_t(\boldsymbol{x})| & \boldsymbol{x} \text{ inside } \phi \\ -|\boldsymbol{x} - \phi_t(\boldsymbol{x})| & \text{otherwise} \end{cases}$$

In case of curve evolution by mean curvature, the level set equation becomes

$\phi_t - b\kappa \boldsymbol{n} \nabla(\phi) = 0$

It follows from $\boldsymbol{n}\nabla\phi = |\nabla\phi|$ that

$\phi_t - b\kappa|\nabla(\phi)| = 0$

It can be shown that any curve evolution can be embedded into the evolution of a level set function [Set99]. The curve is now implicitly represented as a *level set*.

Definition 4.5 C Level Set

$$L_c(\phi) = \{(x_1, \cdots, x_n) | f(x_1, \cdots, x_n) = c\}$$

In two dimensions, the level set $L_c(\phi) = \{(x, y) | f(x, y) = c\}$ is referred to as *level curve*, *isoline*, *interface* or *contour line*. In three dimensions, the level set is defined as $L_c(\phi) = \{(x, y, z) | f(x, y, z) = c\}$ and is referred to as *isosurface*, *contour surface* or *level surface*. If the level set function is n-dimensional, the interface is always $n-1$-dimensional. Since c is just an arbitrary constant value, without loss of generality, we set $c = 0$ in the following. $L_0(\phi)$ is also called the *zero level set*.

The normal vector for points on this interface is exactly perpendicular to the contour of our segmentation. The mean curvature on the interface is exactly the divergence of the normal vector.

4.3 Chan Vese Active Contour Model

The active contour segmentation model by Chan and Vese [CV01] is a reduction of the piecewise constant segmentation model introduced by Mumford and Shah [MS89]. The idea is to start with some initial level set function ϕ and to evolve the contour under certain constraints to obtain a useful segmentation. Because of the level set formulation, the algorithm creates a *partitioning* of the data, which means the resulting data will not have vacuums or overlaps. The level set representation also allows us to automatically detect interior contours. A major restriction of the original model is that it only allows two phases, namely the *foreground phase* and the *background phase*. We will see later how to deal with this limitation. The model consists of three terms; a *length term*, an *area term* and a *data term* . It is given as

Definition 4.6 Chan Vese Active Contour Segmentation Model

$$\begin{aligned} E_{CV}(u_{in}, u_{out}, C) = &\alpha \cdot \text{Length}(C) + \mu \cdot \text{Area}(inside(C)) \\ &+ \lambda_1 \int\limits_{inside(C)} (f(\boldsymbol{x}) - u_{in})^2 \, d\boldsymbol{x} \\ &+ \lambda_2 \int\limits_{outside(C)} (f(\boldsymbol{x}) - u_{out})^2 \, d\boldsymbol{x} \end{aligned}$$

where $f(\boldsymbol{x})$ is the image, C is the level set curve, u_{in} and u_{out} are the arithmetic means of the foreground and the background phases respectively. λ_1 and λ_2 are the weights of the data term, penalizing the deviation of a gray value to the average gray value within each phase. The length term penalizes the length of the contour using a weighting parameter α. The area term with weighting parameter $\mu > 0$ penalizes the area of the foreground phase. If $\mu < 0$, the area of the background phase will be penalized.

Instead of using the level set curve C, we now introduce the aforementioned level set function $\phi(\boldsymbol{x}) : \Omega \mapsto R$. ϕ must be a Lipschitz continuous function. This is to ensure that it is differentiable everywhere and has a bounded first derivative.

$$\begin{cases} C = \partial\omega = \{\boldsymbol{x} \in \Omega : \phi(\boldsymbol{x}) = 0\} \\ \text{inside}(C) = \omega = \{\boldsymbol{x} \in \Omega : \phi(\boldsymbol{x}) > 0\} \\ \text{outside}(C) = \Omega \backslash \overline{\omega} = \{\boldsymbol{x} \in \Omega : \phi(\boldsymbol{x}) < 0\} \end{cases} \tag{4.1}$$

where $\partial\omega$ is the border of ω.

Since it is mathematically very inconvenient to work with different integration domains, using the *Heaviside function*

Definition 4.7 Heaviside Function

$$H(z) = \begin{cases} 1 & z \geq 0 \\ 0 & z < 0 \end{cases}$$

we can reformulate the single terms of the model from definition 4.6 in the previously introduced level set formalism explained in section 4.2 as follows

$$\begin{aligned}
\text{Length}\,(C) = \text{Length}\,\{\phi = 0\} &= \int_{\Omega} |\nabla H\,(\phi\,(\boldsymbol{x}))|\; d\boldsymbol{x} \\
\text{Area}\,(C) = \text{Area}\,\{\phi \geq 0\} &= \int_{\Omega} H(\phi\,(\boldsymbol{x}))\, d\boldsymbol{x} \\
\int_{\phi>0} (f\,(\boldsymbol{x}) - u_{in})^2\; d\boldsymbol{x} &= \int_{\Omega} (f\,(\boldsymbol{x}) - u_{in})^2\, H(\phi\,(\boldsymbol{x}))\, d\boldsymbol{x} \\
\int_{\phi<0} (f\,(\boldsymbol{x}) - u_{out})^2\; d\boldsymbol{x} &= \int_{\Omega} (f\,(\boldsymbol{x}) - u_{out})^2\, (1 - H(\phi\,(\boldsymbol{x})))\, d\boldsymbol{x}
\end{aligned} \tag{4.2}$$

and obtain the rewritten model in level set representation as

$$\begin{aligned}
E_{CV}(u_{in}, u_{out}, \phi) = \quad & \alpha \int_{\Omega} |\nabla H\,(\phi\,(\boldsymbol{x}))|\; d\boldsymbol{x} + \mu \int_{\Omega} H(\phi\,(\boldsymbol{x}))\, d\boldsymbol{x} \\
& + \lambda_1 \int_{\Omega} (f\,(\boldsymbol{x}) - u_{in})^2\, H(\phi\,(\boldsymbol{x}))\, d\boldsymbol{x} \\
& + \lambda_2 \int_{\Omega} (f\,(\boldsymbol{x}) - u_{out})^2\, (1 - H(\phi\,(\boldsymbol{x})))\, d\boldsymbol{x}
\end{aligned} \tag{4.3}$$

We can now solve this minimization problem by alternately updating u_{in}, u_{out} and ϕ.

For the first update step, the arithmetic means of the two phases are computed (assuming a nonempty interior and exterior in Ω, to ensure that the values are well defined):

$$u_{in}\,(\phi) = \frac{\int_{\Omega} f\,(\boldsymbol{x}) H\,(\phi\,(\boldsymbol{x}))\, d\boldsymbol{x}}{\int_{\Omega} H\,(\phi\,(\boldsymbol{x}))\, d\boldsymbol{x}} \tag{4.4}$$

$$u_{out}\,(\phi) = \frac{\int_{\Omega} f\,(\boldsymbol{x})\,(1 - H\,(\phi\,(\boldsymbol{x})))\, d\boldsymbol{x}}{\int_{\Omega} (1 - H\,(\phi\,(\boldsymbol{x})))\, d\boldsymbol{x}} \tag{4.5}$$

In the second update step, we compute the level set function ϕ using the fixed mean values obtained from equations 4.4 and 4.5

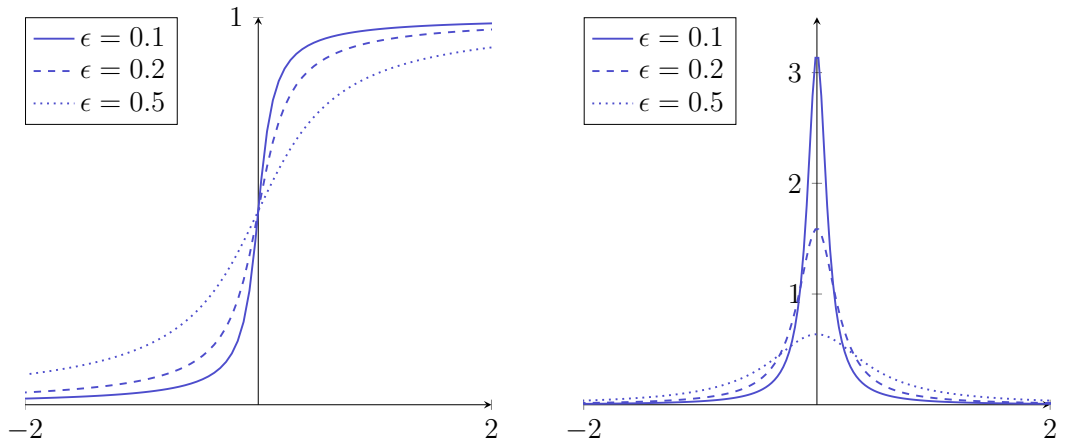
$$\begin{aligned} E_{CV}(\phi) = \quad & \alpha \int_{\Omega} |\nabla H(\phi(\boldsymbol{x}))| \, d\boldsymbol{x} + \mu \int_{\Omega} H(\phi(\boldsymbol{x})) \, d\boldsymbol{x} \\ & + \lambda_1 \int_{\Omega} (f(\boldsymbol{x}) - u_{in})^2 H(\phi(\boldsymbol{x})) \, d\boldsymbol{x} \\ & + \lambda_2 \int_{\Omega} (f(\boldsymbol{x}) - u_{out})^2 (1 - H(\phi(\boldsymbol{x}))) \, d\boldsymbol{x} \end{aligned} \tag{4.6}$$

This process is repeated iteratively until convergence.

The computation of the mean values in each iteration is just a simple arithmetic operation. The update of the level set function on the other hand turns out to be a more challenging task. The energy functional from equation 4.6 is non-convex. We solve it using a *gradient descent approach*. This involves the computation of derivatives. Since the Heaviside function is not differentiable at $x = 0$, Chan and Vese use a regularized version of the Heaviside function.

Definition 4.8 Regularized Heaviside Function

$$H_\epsilon(z) = \frac{1}{2}\left(1 + \frac{2}{\pi}\arctan\left(\frac{z}{\epsilon}\right)\right)$$



(a) Regularized Heaviside function with different values for ϵ

(b) Regularized Dirac function

Figure 4.2: Regularization effect of Heaviside and Dirac functions

The distributional derivative of this function is now given as the *regularized Dirac function*

Definition 4.9 Regularized Dirac Function

$$\delta_\epsilon(z) = \frac{\epsilon}{\pi(\epsilon^2 + z^2)}$$

which is the first order approximation of original Dirac delta function. ϵ is a numerical parameter that determines the size of the bandwidth of numerical smearing. A small value approximates the original model better, but leads to numerical problems. The closer ϵ approaches 0, the smaller the function values become. A high value for ϵ makes the method numerically more stable, but increases the deviation from the original model assumptions. According to Osher and Fedkiw [OF03], the error in calculation for the regularized Heaviside function is in $O(h_x)$.

According to 2.13, we now deduce the Euler Lagrange equations from equation 4.6 which gives:

Definition 4.10 Euler Lagrange Equations of E_{CV}

$$\delta_\epsilon(\phi(\boldsymbol{x}))\left(\lambda_2(f(\boldsymbol{x}) - u_{out})^2 - \lambda_1(f(\boldsymbol{x}) - u_{in})^2 - \mu + \alpha\,\mathrm{div}\left(\frac{\nabla\phi(\boldsymbol{x})}{|\nabla\phi(\boldsymbol{x})|}\right)\right) = 0$$

with the boundary condition

$$\frac{\partial_\epsilon(\phi(\boldsymbol{x}))}{|\nabla\phi(\boldsymbol{x})|}\frac{\partial\phi(\boldsymbol{x})}{\partial\boldsymbol{n}(\boldsymbol{x})} = 0$$

and $\boldsymbol{x} \in \partial\Omega$

With the Euler Lagrange Equations at hand, we can now embed definition 4.10 into a gradient descent scheme as follows:

Definition 4.11 Gradient Descent of E_{CV}

$$\partial_t\phi(\boldsymbol{x}) = \delta_\epsilon(\phi(\boldsymbol{x}))\left(\lambda_2(f(\boldsymbol{x}) - u_{out})^2 - \lambda_1(f(\boldsymbol{x}) - u_{in})^2 - \mu + \alpha\,\mathrm{div}\left(\frac{\nabla\phi(\boldsymbol{x})}{|\nabla\phi(\boldsymbol{x})|}\right)\right)$$

with the boundary condition

$$\frac{\partial_\epsilon(\phi(\boldsymbol{x}))}{|\nabla\phi(\boldsymbol{x})|}\frac{\partial\phi(\boldsymbol{x})}{\partial\boldsymbol{n}(\boldsymbol{x})} = 0$$

$$\phi(\boldsymbol{x}, 0) = \phi_0(\boldsymbol{x})$$

with $\boldsymbol{x} \in \partial\Omega$

4.3.1 Discretization of the Chan Vese Model

In order to apply this approach to an actual discrete data set, we have to discretize it first. The divergence term appearing in equations 4.6 and 4.10, which corresponds to the length term from our original model can also be written as

$$div\left(\frac{\nabla\phi\left(\boldsymbol{x}\right)}{\left|\nabla\phi\left(\boldsymbol{x}\right)\right|}\right)=\partial_x\left(\frac{\partial_x\phi}{\sqrt{\left(\partial_x\phi\right)^2+\left(\partial_y\phi\right)^2}}\right)+\partial_y\left(\frac{\partial_y\phi}{\sqrt{\left(\partial_y\phi\right)^2+\left(\partial_x\phi\right)^2}}\right) \tag{4.7}$$

To discretize this term, we use a mixture of forward- and backward differences. The reasoning for using a combination of forward- and backward differences instead of central differences is that the result will be better localized[1]. We obtain:

$$\begin{aligned}\frac{\phi_{,i,j}^{n+1}-\phi_{i,j}^{n}}{\tau}=&\delta_\epsilon\left(\phi_{i,j}^{n}\right)\Bigg(\lambda_2\left(f_{i,j}-u_{out}\right)^2-\lambda_1\left(f_{i,j}-u_{in}\right)^2-\mu\\&+\alpha\left(\partial_x^-\left(\frac{\partial_x^+\phi_{i,j}^{n+1}}{\sqrt{\left(\partial_x^+\phi_{i,j}^{n}\right)^2+\left(\partial_y\phi_{i,j}^{n}\right)^2}}\right)\right.\\&\left.\left.+\partial_y^-\left(\frac{\partial_y^+\phi_{i,j}^{n+1}}{\sqrt{\left(\partial_y^+\phi_{i,j}^{n}\right)^2+\left(\partial_x\phi_{i,j}^{n}\right)^2}}\right)\right)\Bigg)\end{aligned} \tag{4.8}$$

By using the following curvature terms

$$\begin{aligned}C_{1,i,j}&=\frac{\alpha}{\sqrt{\left(\partial_x^+\phi_{i,j}^{n}\right)^2+\left(\partial_y\phi_{i,j}^{n}\right)^2}}\\C_{2,i,j}&=\frac{\alpha}{\sqrt{\left(\partial_x^+\phi_{i-1,j}^{n}\right)^2+\left(\partial_y\phi_{i-1,j}^{n}\right)^2}}\\C_{3,i,j}&=\frac{\alpha}{\sqrt{\left(\partial_y^+\phi_{i,j}^{n}\right)^2+\left(\partial_x\phi_{i,j}^{n}\right)^2}}\\C_{4,i,j}&=\frac{\alpha}{\sqrt{\left(\partial_y^+\phi_{i,j-1}^{n}\right)^2+\left(\partial_x\phi_{i,j-1}^{n}\right)^2}}\end{aligned} \tag{4.9}$$

and by writing the data and area terms as

$$D_{i,j}=\lambda_2\left(f_{i,j}-u_{out}\right)^2-\lambda_1\left(f_{i,j}-u_{in}\right)^2-\mu \tag{4.10}$$

[1] When applying the central difference consecutively, the central pixel would not be even be included in the computation once. When applying forward and backward differences, one also obtains a centralized result, but the central pixel actually occurs twice in the computation. (see definition of finite differences 2.9-2.11

we can now rewrite the whole problem as

$$\begin{aligned}\frac{\phi_{i,j}^{n+1}-\phi_{i,j}^{n}}{\tau} =&\delta_\epsilon\left(\phi_{i,j}^{n}\right)\left(\lambda_2\left(f_{i,j}-u_{out}\right)^2-\lambda_1\left(f_{i,j}-u_{in}\right)^2-\mu\right.\\ &+C_{1,i,j}\left(\phi_{i+1,j}^{n}-\phi_{i,j}^{n+1}\right)-C_{2,i,j}\left(\phi_{i,j}^{n+1}-\phi_{i-1,j}^{n}\right)\\ &\left.+C_{3,i,j}\left(\phi_{i,j+1}^{n}-\phi_{i,j}^{n+1}\right)-C_{4,i,j}\left(\phi_{i,j}^{n+1}-\phi_{i,j-1}^{n}\right)\right)\end{aligned} \tag{4.11}$$

Finally, we obtain a system of equations that we solve using Jacobi iterations:

$$\phi_{,i,j}^{n+1}=\frac{\phi_{i,j}^{n}+\tau\delta_\epsilon\left(\phi_{i,j}^{n}\right)\left(D_{i,j}+C_{1,i,j}\phi_{i+1,j}^{n}+C_{2,i,j}\phi_{i-1,j}^{n+1}+C_{3,i,j}\phi_{i,j+1}^{n}+C_{4,i,j}\phi_{i,j-1}^{n+1}\right)}{1+\tau\delta_\epsilon\left(C_{1,i,j}+C_{2,i,j}+C_{3,i,j}+C_{4,i,j}\right)} \tag{4.12}$$

The number of equations to solve corresponds to the number of pixels. Since there are no dependencies between the equations, all the pixels in each iteration step can be solved in parallel which makes the method extremely efficent. For a 320 x 320 image, we therefore have to solve 102,400 equations in each iteration step. The integration domain must not necessarily be rectangular. It would also be possible to use a masked region of the image domain.

The Chan Vese algorithm presented here was implemented both in MATLAB and in C. A contour evolution of the Chan Vese method on a Schlieren sequence is shown in figure 4.6. A two-dimensional heat map of the level set function of the same evolution is shown in figure 4.7. A three-dimensional depiction of the level set evolution is shown in figure 4.8.

4.3.2 Initialization of the Level Set Function

The iterative procedure described above needs to start with an initial value for the level set function. In our implementation, we enable the user to choose between several basic shapes (rectangle, circle, donut, gray value thresholding, cirular- or rectangular patterns). Using some simple parameters such as the inner and outer radius/diameter, the shapes can be easily adapted to the data at hand. Some possibilities are shown in figure 5.12. The choice of the initial level set also affects the computation. The curve tends to evolve slowly if it has a low curvature and faster if it has a higher curvature.

We initialize all level set functions, using the Euclidean signed distance function, as proposed by [CV01]. With a gradient of 1, the Euclidean signed distance function helps to avoid rapidly changing features, which makes the result numerically more stable. Mulder, Osher and Sethian [MOS92] showed that initializing ϕ to a Euclidean signed distance function results in more accurate numerical solutions than an initialization with the Heaviside function.

Our implementation allows the user to choose between the Heaviside function and the Euclidean signed distance function. The Euclidean signed distance function was implemented using the `bwdist` method from MATLAB.

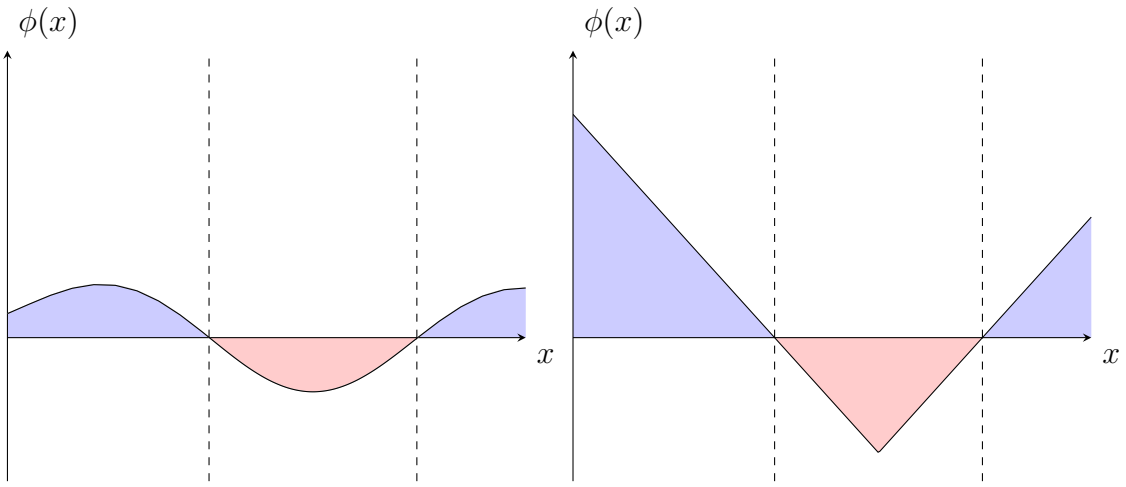


Figure 4.3: Reinitialization of the level set function: Left: Original level set function. Right: Level set function after reinitialization with the Euclidean signed distance function. The dashed vertical lines denote the location of the contour.

4.3.3 Reinitialization of the Level Set Function

During a level set evolution, the level set function usually develops irregularities which can cause numerical errors and harm the stability of the method. From a numerical perspective, it is desirable to keep the evolving level-set function close to the Euclidean signed distance function ([OF03], [Pen+99]). Level set functions can develop shocks [2] during the evolution which makes further computations highly inaccurate. If ϕ is not smooth or if ϕ is much steeper on one side of interface than on the other, the contour can be moved incorrectly ([Set99], [Pen+99], [OF03]). In oder to maintain a stable level set evolution, we apply a method called *reinitialization*. Reinitialization is performed by periodically disrupting the level set evolution and reshaping the degraded level set function as a Euclidean signed distance function (Definition 4.4).

However, there are some practical and theoretical problems regarding the practice of reinitialization. In a conventional level set formulation, the motion of the contour should only be guided by some speed function that is purely based on the model assumptions. It has been proven Barles et al. [BSS93] that solutions to the Hamilton Jacobi Equation of such a form are not Euclidean signed distance functions. Therefore one has to deal with a difference between theory and implementation. An investigation on this problem is given by Gomes and Faugeras [GF00].

In the user interface, we allow the user to choose an arbitrary period of steps during iterations in which to perform the reinitialization. The implementation of the Euclidean signed distance function in MATLAB is based on Maurer, Rensheng and Vijay [Mau+03]. In order to maintain a stable curve evolution, we apply a reinitialization after each single iteration step for the examples shown in this work. By reinitializing more often, the method converges faster. Hence, there is barely any

[2] Shocks are very sharp and/or flat shaped structures in the graph of a function.

performance penalty when applying a frequent initialization.

4.3.4 Termination Criterion

As a termination criterion, we use the residual. For an equation system $Au = b$, we obtain a residual $r_k = Au^k - b$. As soon as $||r_k|| \leq \epsilon * ||r_0||$ with $\epsilon = 0.001$, we stop. The implementation also allow to stop the process manually, either after a user-predefined number of iterations or during live execution of the segmentation.

4.4 Multiphase Chan Vese Segmentation Model

Using one level set, it is possible to distinguish between two phases. Embedding multiple level set functions in one model enables us to identify more than only two phases. For example, by considering the zero level sets of two different level set functions ϕ_1 and ϕ_2, we obtain the four phases:

$\phi_1 < 0 \wedge \phi_2 < 0$

$\phi_1 \geq 0 \wedge \phi_2 < 0$

$\phi_1 < 0 \wedge \phi_2 \geq 0$

$\phi_1 \geq 0 \wedge \phi_2 \geq 0$

The generalized *multiphase Chan Vese model* with m level set functions and $n = 2^m$ phases is defined as

Definition 4.12 Multiphase Chan Vese Active Contour Segmentation Model

$$E^n_{CVM}(\boldsymbol{u}, \boldsymbol{C}) = \sum_{j=1}^{m} \alpha_j \cdot \text{Length}(C_j) + \sum_{j=1}^{m} \mu_j \cdot \text{Area}(inside\,(\chi_j\,(C_j))) + \sum_{K=1}^{n} \lambda_K \cdot \int_{\Omega} (f(\boldsymbol{x}) - u_K)^2 \, d\boldsymbol{x}$$

where χ is the characteristic function, used to distinguish the n phases

In the previously introduced level set formalism, the multiphase model is given as

$$\begin{aligned} E^n_{CVM}(\boldsymbol{u}, \boldsymbol{\phi}) = & \sum_{j=1}^{m} \alpha_j \int_{\Omega} |\nabla H\,(\phi_j\,(\boldsymbol{x}))| \, d\boldsymbol{x} \\ & + \sum_{j=1}^{m} \mu_j \int_{\Omega} H\,(\chi_j(\phi\,(\boldsymbol{x}))) \, d\boldsymbol{x} \\ & + \sum_{K=1}^{n} \lambda_K \int_{\Omega} (f\,(\boldsymbol{x}) - u_K)^2 \, d\boldsymbol{x} \end{aligned} \tag{4.13}$$

Since we are interested in differentiating between four phases (liquid, vapor, chamber, background), we restrict ourselves to the special case of two level set functions

Definition 4.13 Four Phase Chan Vese Model

$$\begin{aligned} E^4_{CVM}(\boldsymbol{u}, \boldsymbol{\phi}) = & \sum_{j=1}^{2} \alpha_j \int_\Omega \left|\nabla H\left(\phi_j\left(\boldsymbol{x}\right)\right)\right| \, d\boldsymbol{x} \\ & + \sum_{j=1}^{2} \mu_j \int_\Omega H\left(\chi_j(\phi\left(\boldsymbol{x}\right))\right) \, d\boldsymbol{x} \\ & + \sum_{K=1}^{4} \lambda_K \int_\Omega \left(f\left(\boldsymbol{x}\right) - u_K\right)^2 \, d\boldsymbol{x} \end{aligned}$$

where $\boldsymbol{u} = (u_1, u_2, u_3, u_4)^\top$ and $\boldsymbol{\phi} = (\phi_1, \phi_2)^\top$

we can now rewrite the functional, just as in equation 4.3, as

$$\begin{aligned} E^4_{CVM}\left(\boldsymbol{u}, \boldsymbol{\phi}\right) = & \lambda_1 \int_\Omega \left(f\left(\boldsymbol{x}\right) - u_1\right)^2 H\left(\phi_1\right) H\left(\phi_2\right) d\boldsymbol{x} \\ & + \lambda_2 \int_\Omega \left(f\left(\boldsymbol{x}\right) - u_2\right)^2 H\left(\phi_1\right) \left(1 - H\left(\phi_2\right)\right) d\boldsymbol{x} \\ & + \lambda_3 \int_\Omega \left(f\left(\boldsymbol{x}\right) - u_3\right)^2 \left(1 - H\left(\phi_1\right)\right) H\left(\phi_2\right) d\boldsymbol{x} \\ & + \lambda_4 \int_\Omega \left(f\left(\boldsymbol{x}\right) - u_4\right)^2 \left(1 - H\left(\phi_1\right)\right) \left(1 - H\left(\phi_2\right)\right) d\boldsymbol{x} \\ & + \mu_1 \int_\Omega H\left(\chi_1(\phi\left(\boldsymbol{x}\right))\right) \, d\boldsymbol{x} \\ & + \mu_2 \int_\Omega H\left(\chi_2(\phi\left(\boldsymbol{x}\right))\right) \, d\boldsymbol{x} \\ & + \alpha_1 \int_\Omega \left|\nabla H\left(\phi_1\left(\boldsymbol{x}\right)\right)\right| d\boldsymbol{x} \\ & + \alpha_2 \int_\Omega \left|\nabla H\left(\phi_2\left(\boldsymbol{x}\right)\right)\right| d\boldsymbol{x} \end{aligned} \tag{4.14}$$

To obtain the arithmetic means in the first update step of the iterative procedure, we compute

$$\begin{aligned}
u_1(\boldsymbol{\phi}) &= \frac{\int_\Omega f(\boldsymbol{x}) H(\phi_1(\boldsymbol{x})) H(\phi_2(\boldsymbol{x})) \, d\boldsymbol{x}}{\int_\Omega H(\phi_1(\boldsymbol{x})) H(\phi_2(\boldsymbol{x})) \, d\boldsymbol{x}} \\
u_2(\boldsymbol{\phi}) &= \frac{\int_\Omega f(\boldsymbol{x}) H(\phi_1(\boldsymbol{x})) (1 - H(\phi_2(\boldsymbol{x}))) \, d\boldsymbol{x}}{\int_\Omega H(\phi_1(\boldsymbol{x})) (1 - H(\phi_2(\boldsymbol{x}))) \, d\boldsymbol{x}} \\
u_3(\boldsymbol{\phi}) &= \frac{\int_\Omega f(\boldsymbol{x}) (1 - H(\phi_1(\boldsymbol{x}))) H(\phi_2(\boldsymbol{x})) \, d\boldsymbol{x}}{\int_\Omega (1 - H(\phi_1(\boldsymbol{x}))) H(\phi_2(\boldsymbol{x})) \, d\boldsymbol{x}} \\
u_4(\boldsymbol{\phi}) &= \frac{\int_\Omega f(\boldsymbol{x}) (1 - H(\phi_1(\boldsymbol{x}))) (1 - H(\phi_2(\boldsymbol{x}))) \, d\boldsymbol{x}}{\int_\Omega (1 - H(\phi_1(\boldsymbol{x}))) (1 - H(\phi_2(\boldsymbol{x}))) \, d\boldsymbol{x}}
\end{aligned} \tag{4.15}$$

In the second update step, we now have to find a solution for the model

$$\begin{aligned}
E^4_{CVM}(\boldsymbol{\phi}) = \quad & \lambda_1 \int_\Omega (f(\boldsymbol{x}) - u_1)^2 H(\phi_1) H(\phi_2) \, d\boldsymbol{x} \\
& + \lambda_2 \int_\Omega (f(\boldsymbol{x}) - u_2)^2 H(\phi_1) (1 - H(\phi_2)) \, d\boldsymbol{x} \\
& + \lambda_3 \int_\Omega (f(\boldsymbol{x}) - u_3)^2 (1 - H(\phi_1)) H(\phi_2) \, d\boldsymbol{x} \\
& + \lambda_4 \int_\Omega (f(\boldsymbol{x}) - u_4)^2 (1 - H(\phi_1)) (1 - H(\phi_2)) \, d\boldsymbol{x} \\
& + \mu_1 \int_\Omega H(\chi_1(\phi(\boldsymbol{x}))) \, d\boldsymbol{x} \\
& + \mu_2 \int_\Omega H(\chi_2(\phi(\boldsymbol{x}))) \, d\boldsymbol{x} \\
& + \alpha_1 \int_\Omega |\nabla H(\phi_1(\boldsymbol{x}))| \, d\boldsymbol{x} \\
& + \alpha_2 \int_\Omega |\nabla H(\phi_2(\boldsymbol{x}))| \, d\boldsymbol{x}
\end{aligned} \tag{4.16}$$

In the two phase scenario, one obtains the following Euler Lagrange equations

Definition 4.14 Euler Lagrange Equations of E^4_{CVM}

$$\begin{aligned}
0 =& \delta_\epsilon\left(\phi_1\left(\boldsymbol{x}\right)\right)\left(\alpha \operatorname{div}\left(\frac{\nabla\phi_1\left(\boldsymbol{x}\right)}{|\nabla\phi_1\left(\boldsymbol{x}\right)|}\right) - \mu_1\right.\\
&- \left(\lambda_1\left(f\left(\boldsymbol{x}\right)-u_1\right)^2 - \lambda_3\left(f\left(\boldsymbol{x}\right)-u_3\right)^2\right) H\left(\phi_1\right)\\
&\left.- \left(\lambda_2\left(f\left(\boldsymbol{x}\right)-u_2\right)^2 - \lambda_4\left(f\left(\boldsymbol{x}\right)-u_4\right)^2\right)\left(1-H\left(\phi_1\right)\right)\right)\\
0 =& \delta_\epsilon\left(\phi_2\left(\boldsymbol{x}\right)\right)\left(\alpha \operatorname{div}\left(\frac{\nabla\phi_2\left(\boldsymbol{x}\right)}{|\nabla\phi_2\left(\boldsymbol{x}\right)|}\right) - \mu_2\right.\\
&- \left(\lambda_1\left(f\left(\boldsymbol{x}\right)-u_1\right)^2 - \lambda_3\left(f\left(\boldsymbol{x}\right)-u_3\right)^2\right) H\left(\phi_2\right)\\
&\left.- \left(\lambda_2\left(f\left(\boldsymbol{x}\right)-u_2\right)^2 - \lambda_4\left(f\left(\boldsymbol{x}\right)-u_4\right)^2\right)\left(1-H\left(\phi_2\right)\right)\right)
\end{aligned}$$

with the boundary conditions

$$\frac{\partial_\epsilon\left(\phi_1\left(\boldsymbol{x}\right)\right)}{|\nabla\phi_1\left(\boldsymbol{x}\right)|}\frac{\partial\phi_1\left(\boldsymbol{x}\right)}{\partial\boldsymbol{n}\left(\boldsymbol{x}\right)} = 0$$

$$\frac{\partial_\epsilon\left(\phi_2\left(\boldsymbol{x}\right)\right)}{|\nabla\phi_2\left(\boldsymbol{x}\right)|}\frac{\partial\phi_2\left(\boldsymbol{x}\right)}{\partial\boldsymbol{n}\left(\boldsymbol{x}\right)} = 0$$

with $\boldsymbol{x} \in \partial\Omega$

Using the same gradient descent approach as in equation 4.11, we now formulate a gradient descent approach for the multiphase Chan Vese segmentation method. The gradient descent scheme reads:

Definition 4.15 Gradient Descent of E^4_{CVM}

$$\begin{aligned}
\partial_t\phi\left(\boldsymbol{x}\right) =& \delta_\epsilon\left(\phi_1\left(\boldsymbol{x}\right)\right)\left(\alpha \operatorname{div}\left(\frac{\nabla\phi_1\left(\boldsymbol{x}\right)}{|\nabla\phi_1\left(\boldsymbol{x}\right)|}\right) - \mu_1\right.\\
&- \left(\lambda_1\left(f\left(\boldsymbol{x}\right)-u_1\right)^2 - \lambda_3\left(f\left(\boldsymbol{x}\right)-u_3\right)^2\right) H\left(\phi_1\right)\\
&\left.- \left(\lambda_2\left(f\left(\boldsymbol{x}\right)-u_2\right)^2 - \lambda_4\left(f\left(\boldsymbol{x}\right)-u_4\right)^2\right)\left(1-H\left(\phi_1\right)\right)\right)\\
\partial_t\phi\left(\boldsymbol{x}\right) =& \delta_\epsilon\left(\phi_2\left(\boldsymbol{x}\right)\right)\left(\alpha \operatorname{div}\left(\frac{\nabla\phi_2\left(\boldsymbol{x}\right)}{|\nabla\phi_2\left(\boldsymbol{x}\right)|}\right) - \mu_2\right.\\
&- \left(\lambda_1\left(f\left(\boldsymbol{x}\right)-u_1\right)^2 - \lambda_3\left(f\left(\boldsymbol{x}\right)-u_3\right)^2\right) H\left(\phi_2\right)\\
&\left.- \left(\lambda_2\left(f\left(\boldsymbol{x}\right)-u_2\right)^2 - \lambda_4\left(f\left(\boldsymbol{x}\right)-u_4\right)^2\right)\left(1-H\left(\phi_2\right)\right)\right)
\end{aligned}$$

with the boundary condition

$$\frac{\partial_\epsilon\left(\phi_1\left(\boldsymbol{x}\right)\right)}{|\nabla\phi_1\left(\boldsymbol{x}\right)|}\frac{\partial\phi_1\left(\boldsymbol{x}\right)}{\partial\boldsymbol{n}\left(\boldsymbol{x}\right)} = 0$$

$$\phi_1\left(\boldsymbol{x},0\right) = \phi_{1,0}\left(\boldsymbol{x}\right)$$

$$\frac{\partial_\epsilon\left(\phi_2\left(\boldsymbol{x}\right)\right)}{\left|\nabla\phi_2\left(\boldsymbol{x}\right)\right|}\frac{\partial\phi_2\left(\boldsymbol{x}\right)}{\partial\boldsymbol{n}\left(\boldsymbol{x}\right)}=0$$
$$\phi_2\left(\boldsymbol{x},0\right)=\phi_{2,0}\left(\boldsymbol{x}\right)$$

with $\boldsymbol{x}\in\partial\Omega$

4.4.1 Discretization of the Multiphase Model

The discretization approach in the two phase case is similar to the one phase scenario.

$$\begin{aligned}\frac{\phi_{1,i,j}^{n+1}-\phi_{1,i,j}^{n}}{\tau}=&\delta_\epsilon\left(\phi_{1,i,j}^{n}\right)\left(\left(\lambda_1\left(f_{i,j}-u_1\right)^2-\lambda_3\left(f_{i,j}-u_3\right)^2\right)H\left(\phi_{2,i,j}\right)\right.\\&-\left(\lambda_2\left(f_{i,j}-u_2\right)^2-\lambda_4\left(f_{i,j}-u_4\right)^2\right)\left(1-H\left(\phi_{2,i,j}\right)\right)-\mu_1\\&+\alpha\left(\partial_x^-\left(\frac{\partial_x^+\phi_{1,i,j}^{n+1}}{\sqrt{\left(\partial_x^+\phi_{1,i,j}^{n}\right)^2+\left(\partial_y\phi_{1,i,j}^{n}\right)^2}}\right)\right.\\&\left.\left.+\partial_y^-\left(\frac{\partial_y^+\phi_{1,i,j}^{n+1}}{\sqrt{\left(\partial_y^+\phi_{1,i,j}^{n}\right)^2+\left(\partial_x\phi_{1,i,j}^{n}\right)^2}}\right)\right)\right)\end{aligned}\tag{4.17}$$

$$\begin{aligned}\frac{\phi_{2,i,j}^{n+1}-\phi_{2,i,j}^{n}}{\tau}=&\delta_\epsilon\left(\phi_{2,i,j}^{n}\right)\left(\left(\lambda_1\left(f_{i,j}-u_1\right)^2-\lambda_3\left(f_{i,j}-u_3\right)^2\right)H\left(\phi_{1,i,j}\right)\right.\\&-\left(\lambda_2\left(f_{i,j}-u_2\right)^2-\lambda_4\left(f_{i,j}-u_4\right)^2\right)\left(1-H\left(\phi_{1,i,j}\right)\right)-\mu_2\\&+\alpha\left(\partial_x^-\left(\frac{\partial_x^+\phi_{2,i,j}^{n+1}}{\sqrt{\left(\partial_x^+\phi_{2,i,j}^{n}\right)^2+\left(\partial_y\phi_{2,i,j}^{n}\right)^2}}\right)\right.\\&\left.\left.+\partial_y^-\left(\frac{\partial_y^+\phi_{2,i,j}^{n+1}}{\sqrt{\left(\partial_y^+\phi_{2,i,j}^{n}\right)^2+\left(\partial_x\phi_{2,i,j}^{n}\right)^2}}\right)\right)\right)\end{aligned}\tag{4.18}$$

We rewrite the curvature terms the same way as shown in equation 4.9

$$\begin{aligned}C_{1,m,i,j}&=\frac{\alpha}{\sqrt{\left(\partial_x^+\phi_{m,i,j}^{n}\right)^2+\left(\partial_y\phi_{m,i,j}^{n}\right)^2}}\\C_{2,m,i,j}&=\frac{\alpha}{\sqrt{\left(\partial_x^+\phi_{m,i-1,j}^{n}\right)^2+\left(\partial_y\phi_{m,i-1,j}^{n}\right)^2}}\\C_{3,m,i,j}&=\frac{\alpha}{\sqrt{\left(\partial_y^+\phi_{m,i,j}^{n}\right)^2+\left(\partial_x\phi_{m,i,j}^{n}\right)^2}}\\C_{4,m,i,j}&=\frac{\alpha}{\sqrt{\left(\partial_y^+\phi_{m,i,j-1}^{n}\right)^2+\left(\partial_x\phi_{m,i,j-1}^{n}\right)^2}}\end{aligned}\tag{4.19}$$

With the data term

$$\begin{aligned} D_{m,i,j} = &- \left(\left(\lambda_1 \left(f_{i,j} - u_1 \right)^2 - \lambda_3 \left(f_{i,j} - u_3 \right)^2 \right) H \left(\phi_{m,i,j} \right) \right. \\ &\left. + \left(\lambda_2 \left(f_{i,j} - u_2 \right)^2 - \lambda_4 \left(f_{i,j} - u_4 \right)^2 \right) \left(1 - H \left(\phi_{m,i,j} \right) \right) \right) \end{aligned} \tag{4.20}$$

we now obtain the following discretization for our four phase model

$$\begin{aligned} \frac{\phi_{1,i,j}^{n+1} - \phi_{1,i,j}^{n}}{\tau} =& \delta_\epsilon \left(\phi_{1,i,j}^{n} \right) \left(D_{2,i,j} - \mu_1 \right. \\ &+ C_{1,1,i,j} \left(\phi_{1,i+1,j}^{n} - \phi_{1,i,j}^{n+1} \right) - C_{2,1,i,j} \left(\phi_{1,i,j}^{n+1} - \phi_{1,i-1,j}^{n} \right) \\ &\left. + C_{3,1,i,j} \left(\phi_{1,i,j+1}^{n} - \phi_{1,i,j}^{n+1} \right) - C_{4,1,i,j} \left(\phi_{1,i,j}^{n+1} - \phi_{1,i,j-1}^{n} \right) \right) \end{aligned} \tag{4.21}$$

$$\begin{aligned} \frac{\phi_{2,i,j}^{n+1} - \phi_{2,i,j}^{n}}{\tau} =& \delta_\epsilon \left(\phi_{2,i,j}^{n} \right) \left(D_{1,i,j} - \mu_2 \right. \\ &+ C_{1,2,i,j} \left(\phi_{2,i+1,j}^{n} - \phi_{2,i,j}^{n+1} \right) - C_{2,2,i,j} \left(\phi_{2,i,j}^{n+1} - \phi_{2,i-1,j}^{n} \right) \\ &\left. + C_{3,2,i,j} \left(\phi_{2,i,j+1}^{n} - \phi_{2,i,j}^{n+1} \right) - C_{4,2,i,j} \left(\phi_{2,i,j}^{n+1} - \phi_{2,i,j-1}^{n} \right) \right) \end{aligned} \tag{4.22}$$

Again, we solve the resulting system of equations with the Jacobi method

$$\begin{aligned} \phi_{1,i,j}^{n+1} = & \frac{\phi_{1,i,j}^{n} + \tau \delta_\epsilon \left(\phi_{1,i,j}^{n} \right) \left(D_{2,i,j} \right) - \mu_1}{1 + \tau \delta_\epsilon \left(C_{1,1,i,j} + C_{2,1,i,j} + C_{3,1,i,j} + C_{4,1,i,j} \right)} \\ &+ \frac{\tau \delta_\epsilon \left(\phi_{1,i,j}^{n} \right) \left(C_{1,1,i,j} \phi_{1,i+1,j}^{n} + C_{2,1,i,j} \phi_{1,i-1,j}^{n+1} + C_{3,1,i,j} \phi_{1,i,j+1}^{n} + C_{4,1,i,j} \phi_{1,i,j-1}^{n+1} \right)}{1 + \tau \delta_\epsilon \left(C_{1,1,i,j} + C_{2,1,i,j} + C_{3,1,i,j} + C_{4,1,i,j} \right)} \end{aligned} \tag{4.23}$$

$$\begin{aligned} \phi_{2,i,j}^{n+1} = & \frac{\phi_{2,i,j}^{n} + \tau \delta_\epsilon \left(\phi_{2,i,j}^{n} \right) \left(D_{1,i,j} \right) - \mu_2}{1 + \tau \delta_\epsilon \left(C_{1,2,i,j} + C_{2,2,i,j} + C_{3,2,i,j} + C_{4,2,i,j} \right)} \\ &+ \frac{\tau \delta_\epsilon \left(\phi_{2,i,j}^{n} \right) \left(C_{1,2,i,j} \phi_{2,i+1,j}^{n} + C_{2,2,i,j} \phi_{2,i-1,j}^{n+1} + C_{3,2,i,j} \phi_{2,i,j+1}^{n} + C_{4,2,i,j} \phi_{2,i,j-1}^{n+1} \right)}{1 + \tau \delta_\epsilon \left(C_{1,2,i,j} + C_{2,2,i,j} + C_{3,2,i,j} + C_{4,2,i,j} \right)} \end{aligned} \tag{4.24}$$

The multiphase Chan Vese model was implemented both in MATLAB and in C.

4.5 Extension to the Spatiotemporal Domain

Since we are working with image sequences, it is sensible to include information from the temporal domain in order to improve the segmentation result. Instead of integrating over only the spatial domain and treating each image independently, we can treat the image sequence as a three-dimensional image and perform the segmentation of the whole sequence in one step. The disadvantage of this approach is of course, that the sequence always has to be processed as a whole, using one large system of equations. It is not possible to compute a segmentation of single frames of the sequence independently from the other frames. However, the computation remains as efficient as in the spatial domain, since we are still able to process every voxel independently of the others, allowing for a high degree of parallelization.

The multiphase Chan Vese model in the spatiotemporal setting can be written as

Definition 4.16 Spatiotemporal Multiphase Chan Vese Model

$$E^n_{CVM3}(\boldsymbol{u}, \boldsymbol{C}) = \sum_{j=1}^{m} \alpha_j \cdot \text{Surface}(C_j) + \sum_{j=1}^{m} \mu_j \cdot \text{Volume}(\mathit{inside}\,(\chi_j\,(C_j))) + \sum_{K=1}^{n} \lambda_K \int_{\Omega_3} \left(f(\boldsymbol{x}) - u_K\right)^2 d\boldsymbol{x}$$

where m is the number of level set functions and $n = 2^m$ is the number of phases.

The interface has now become a three-dimensional surface in four-dimensional space. Instead of penalizing the length of a line, we now penalize the area of a surface. The area term from the original model is now a volume in three-dimensional space.

The four-dimensional level set function is consequently defined as

$$\begin{cases} C_i = \partial\omega_i = \{\boldsymbol{x} \in \Omega_3 : \phi_i\,(\boldsymbol{x}) = 0\} \\ \text{inside}\,(C_i) = \omega_i = \{\boldsymbol{x} \in \Omega_3 : \phi_i\,(\boldsymbol{x}) > 0\} \\ \text{outside}\,(C_i) = \Omega_3 \backslash \overline{\omega_i} = \{\boldsymbol{x} \in \Omega_3 : \phi_i\,(\boldsymbol{x}) < 0\} \end{cases} \tag{4.25}$$

For the special case of two level set functions, the multiphase spatiotemporal Chan Vese model is given as

Definition 4.17 Four Phase Spatiotemporal Chan Vese Model

$$\begin{aligned} E^4_{CVM3}(\boldsymbol{u},\boldsymbol{\phi}) = & \sum_{j=1}^{2} \alpha_j \int\limits_{\Omega_3} \left|\nabla_3 H\left(\phi_j\left(\boldsymbol{x}\right)\right)\right| \, d\boldsymbol{x} \\ & + \sum_{j=1}^{2} \mu_j \int\limits_{\Omega} H\left(\chi_j(\phi\left(\boldsymbol{x}\right))\right) \, d\boldsymbol{x} \\ & + \sum_{K=1}^{4} \lambda_K \int\limits_{\Omega_3} \left(f\left(\boldsymbol{x}\right) - u_K\right)^2 \, d\boldsymbol{x} \end{aligned}$$

where $\boldsymbol{u} = (u_1, u_2, u_3, u_4)^\top$ and $\boldsymbol{\phi} = (\phi_1, \phi_2)^\top$

In our first update step of the alternating iterative procedure, again we have to compute the means of all four phases in the spatiotemporal domain as follows:

$$\begin{aligned} u_1\left(\boldsymbol{\phi}\right) &= \frac{\int\limits_{\Omega_3} f\left(\boldsymbol{x}\right) H\left(\phi_1\left(\boldsymbol{x}\right)\right) H\left(\phi_2\left(\boldsymbol{x}\right)\right) d\boldsymbol{x}}{\int\limits_{\Omega_3} H\left(\phi_1\left(\boldsymbol{x}\right)\right) H\left(\phi_2\left(\boldsymbol{x}\right)\right) d\boldsymbol{x}} \\ u_2\left(\boldsymbol{\phi}\right) &= \frac{\int\limits_{\Omega_3} f\left(\boldsymbol{x}\right) H\left(\phi_1\left(\boldsymbol{x}\right)\right) \left(1 - H\left(\phi_2\left(\boldsymbol{x}\right)\right)\right) d\boldsymbol{x}}{\int\limits_{\Omega_3} H\left(\phi_1\left(\boldsymbol{x}\right)\right) \left(1 - H\left(\phi_2\left(\boldsymbol{x}\right)\right)\right) d\boldsymbol{x}} \\ u_3\left(\boldsymbol{\phi}\right) &= \frac{\int\limits_{\Omega_3} f\left(\boldsymbol{x}\right) \left(1 - H\left(\phi_1\left(\boldsymbol{x}\right)\right)\right) H\left(\phi_2\left(\boldsymbol{x}\right)\right) d\boldsymbol{x}}{\int\limits_{\Omega_3} \left(1 - H\left(\phi_1\left(\boldsymbol{x}\right)\right)\right) H\left(\phi_2\left(\boldsymbol{x}\right)\right) d\boldsymbol{x}} \\ u_4\left(\boldsymbol{\phi}\right) &= \frac{\int\limits_{\Omega_3} f\left(\boldsymbol{x}\right) \left(1 - H\left(\phi_1\left(\boldsymbol{x}\right)\right)\right) \left(1 - H\left(\phi_2\left(\boldsymbol{x}\right)\right)\right) d\boldsymbol{x}}{\int\limits_{\Omega_3} \left(1 - H\left(\phi_1\left(\boldsymbol{x}\right)\right)\right) \left(1 - H\left(\phi_2\left(\boldsymbol{x}\right)\right)\right) d\boldsymbol{x}} \end{aligned} \tag{4.26}$$

The second update step of the four phase spatiotemporal model reads:

$$\begin{aligned} E^4_{CVM3}(\boldsymbol{\phi}) = \quad & \lambda_1 \int_{\Omega_3} (f(\boldsymbol{x}) - u_1)^2 H(\phi_1) H(\phi_2)\, d\boldsymbol{x} \\ & + \lambda_2 \int_{\Omega_3} (f(\boldsymbol{x}) - u_2)^2 H(\phi_1) (1 - H(\phi_2))\, d\boldsymbol{x} \\ & + \lambda_3 \int_{\Omega_3} (f(\boldsymbol{x}) - u_3)^2 (1 - H(\phi_1)) H(\phi_2)\, d\boldsymbol{x} \\ & + \lambda_4 \int_{\Omega_3} (f(\boldsymbol{x}) - u_4)^2 (1 - H(\phi_1)) (1 - H(\phi_2))\, d\boldsymbol{x} \\ & + \mu_1 \int_{\Omega_3} H(\chi_1(\phi(\boldsymbol{x})))\, d\boldsymbol{x} \\ & + \mu_2 \int_{\Omega_3} H(\chi_2(\phi(\boldsymbol{x})))\, d\boldsymbol{x} \\ & + \alpha_1 \int_{\Omega_3} |\nabla_3 H(\phi_1(\boldsymbol{x}))|\, d\boldsymbol{x} \\ & + \alpha_2 \int_{\Omega_3} |\nabla_3 H(\phi_2(\boldsymbol{x}))|\, d\boldsymbol{x} \end{aligned} \tag{4.27}$$

Next, we compute the two Euler Lagrange equations of the four phase spatiotemporal model:

Definition 4.18 Euler Lagrange Equations of E^4_{CVM3}

$$\begin{aligned} 0 =& \delta_\epsilon(\phi_1(\boldsymbol{x})) \left(\alpha \operatorname{div} \left(\frac{\nabla_3 \phi_1(\boldsymbol{x})}{|\nabla_3 \phi_1(\boldsymbol{x})|} \right) - \mu_1 \right. \\ & - \left(\lambda_1 (f(\boldsymbol{x}) - u_1)^2 - \lambda_3 (f(\boldsymbol{x}) - u_3)^2\right) H(\phi_1) \\ & \left. - \left(\lambda_2 (f(\boldsymbol{x}) - u_2)^2 - \lambda_4 (f(\boldsymbol{x}) - u_4)^2\right) (1 - H(\phi_1)) \right) \\ 0 =& \delta_\epsilon(\phi_2(\boldsymbol{x})) \left(\alpha \operatorname{div} \left(\frac{\nabla_3 \phi_2(\boldsymbol{x})}{|\nabla_3 \phi_2(\boldsymbol{x})|} \right) - \mu_2 \right. \\ & - \left(\lambda_1 (f(\boldsymbol{x}) - u_1)^2 - \lambda_3 (f(\boldsymbol{x}) - u_3)^2\right) H(\phi_2) \\ & \left. - \left(\lambda_2 (f(\boldsymbol{x}) - u_2)^2 - \lambda_4 (f(\boldsymbol{x}) - u_4)^2\right) (1 - H(\phi_2)) \right) \end{aligned}$$

with the boundary conditions

$$\frac{\partial_\epsilon(\phi_1(\boldsymbol{x}))}{|\nabla_3 \phi_1(\boldsymbol{x})|} \frac{\partial \phi_1(\boldsymbol{x})}{\partial \boldsymbol{n}(\boldsymbol{x})} = 0$$

$$\frac{\partial_\epsilon\left(\phi_2\left(\boldsymbol{x}\right)\right)}{\left|\nabla_3\phi_2\left(\boldsymbol{x}\right)\right|}\frac{\partial\phi_2\left(\boldsymbol{x}\right)}{\partial\boldsymbol{n}\left(\boldsymbol{x}\right)}=0$$

with $\boldsymbol{x}\in\partial\Omega$

Using the same gradient descent approach as in 4.15, we end up with

Definition 4.19 Gradient Descent of E^4_{CVM3}

$$\begin{aligned}\partial_t\phi\left(\boldsymbol{x}\right)=&\delta_\epsilon\left(\phi_1\left(\boldsymbol{x}\right)\right)\left(\alpha\,\mathrm{div}\left(\frac{\nabla_3\phi_1\left(\boldsymbol{x}\right)}{\left|\nabla_3\phi_1\left(\boldsymbol{x}\right)\right|}\right)-\mu_1\right.\\&-\left(\lambda_1\left(f\left(\boldsymbol{x}\right)-u_1\right)^2-\lambda_3\left(f\left(\boldsymbol{x}\right)-u_3\right)^2\right)H\left(\phi_1\right)\\&\left.-\left(\lambda_2\left(f\left(\boldsymbol{x}\right)-u_2\right)^2-\lambda_4\left(f\left(\boldsymbol{x}\right)-u_4\right)^2\right)\left(1-H\left(\phi_1\right)\right)\right)\\\partial_t\phi\left(\boldsymbol{x}\right)=&\delta_\epsilon\left(\phi_2\left(\boldsymbol{x}\right)\right)\left(\alpha\,\mathrm{div}\left(\frac{\nabla_3\phi_2\left(\boldsymbol{x}\right)}{\left|\nabla_3\phi_2\left(\boldsymbol{x}\right)\right|}\right)-\mu_2\right.\\&-\left(\lambda_1\left(f\left(\boldsymbol{x}\right)-u_1\right)^2-\lambda_3\left(f\left(\boldsymbol{x}\right)-u_3\right)^2\right)H\left(\phi_2\right)\\&\left.-\left(\lambda_2\left(f\left(\boldsymbol{x}\right)-u_2\right)^2-\lambda_4\left(f\left(\boldsymbol{x}\right)-u_4\right)^2\right)\left(1-H\left(\phi_2\right)\right)\right)\end{aligned}$$

with the boundary conditions

$$\frac{\partial_\epsilon\left(\phi_1\left(\boldsymbol{x}\right)\right)}{\left|\nabla_3\phi_1\left(\boldsymbol{x}\right)\right|}\frac{\partial\phi_1\left(\boldsymbol{x}\right)}{\partial\boldsymbol{n}\left(\boldsymbol{x}\right)}=0$$

$$\phi_1\left(\boldsymbol{x},0\right)=\phi_{1,0}\left(\boldsymbol{x}\right)$$

$$\frac{\partial_\epsilon\left(\phi_2\left(\boldsymbol{x}\right)\right)}{\left|\nabla_3\phi_2\left(\boldsymbol{x}\right)\right|}\frac{\partial\phi_2\left(\boldsymbol{x}\right)}{\partial\boldsymbol{n}\left(\boldsymbol{x}\right)}=0$$

$$\phi_2\left(\boldsymbol{x},0\right)=\phi_{2,0}\left(\boldsymbol{x}\right)$$

with $\boldsymbol{x}\in\partial\Omega_3$

4.5.1 Discretization of Spatiotemporal Multiphase Approach

The discretization of the length term in the multiphase spatiotemporal model looks similar to the previous discretization, but now contains an additional term with derivatives in time direction:

$$\begin{aligned}\frac{\phi_{1,i,j}^{n+1}-\phi_{1,i,j}^{n}}{\tau} =&\delta_\epsilon\left(\phi_{1,i,j}^{n}\right)\left(\left(\lambda_1\left(f_{i,j}-u_1\right)^2-\lambda_3\left(f_{i,j}-u_3\right)^2\right)H\left(\phi_{2,i,j}\right)\right.\\ &-\left(\lambda_2\left(f_{i,j}-u_2\right)^2-\lambda_4\left(f_{i,j}-u_4\right)^2\right)\left(1-H\left(\phi_{2,i,j}\right)\right)-\mu_1\\ &+\alpha\left(\partial_x^-\left(\frac{\partial_x^+\phi_{1,i,j,t}^{n+1}}{\sqrt{\left(\partial_x^+\phi_{1,i,j,t}^{n}\right)^2+\left(\partial_y\phi_{1,i,j,t}^{n}\right)^2+\left(\partial_t\phi_{1,i,j,t}^{n}\right)^2}}\right)\right.\\ &+\partial_y^-\left(\frac{\partial_y^+\phi_{1,i,j,t}^{n+1}}{\sqrt{\left(\partial_y^+\phi_{1,i,j,t}^{n}\right)^2+\left(\partial_x\phi_{1,i,j,t}^{n}\right)^2+\left(\partial_t\phi_{1,i,j,t}^{n}\right)^2}}\right)\\ &\left.\left.+\partial_t^-\left(\frac{\partial_t^+\phi_{1,i,j,t}^{n+1}}{\sqrt{\left(\partial_t^+\phi_{1,i,j,t}^{n}\right)^2+\left(\partial_x\phi_{1,i,j,t}^{n}\right)^2+\left(\partial_y\phi_{1,i,j,t}^{n}\right)^2}}\right)\right)\right)\end{aligned} \tag{4.28}$$

$$\begin{aligned}\frac{\phi_{2,i,j}^{n+1}-\phi_{2,i,j}^{n}}{\tau} =&\delta_\epsilon\left(\phi_{2,i,j}^{n}\right)\left(\left(\lambda_1\left(f_{i,j}-u_1\right)^2-\lambda_3\left(f_{i,j}-u_3\right)^2\right)H\left(\phi_{1,i,j}\right)\right.\\ &-\left(\lambda_2\left(f_{i,j}-u_2\right)^2-\lambda_4\left(f_{i,j}-u_4\right)^2\right)\left(1-H\left(\phi_{1,i,j}\right)\right)-\mu_2\\ &+\alpha\left(\partial_x^-\left(\frac{\partial_x^+\phi_{2,i,j,t}^{n+1}}{\sqrt{\left(\partial_x^+\phi_{2,i,j,t}^{n}\right)^2+\left(\partial_y\phi_{2,i,j,t}^{n}\right)^2+\left(\partial_t\phi_{2,i,j,t}^{n}\right)^2}}\right)\right.\\ &+\partial_y^-\left(\frac{\partial_y^+\phi_{2,i,j,t}^{n+1}}{\sqrt{\left(\partial_y^+\phi_{2,i,j,t}^{n}\right)^2+\left(\partial_x\phi_{2,i,j,t}^{n}\right)^2+\left(\partial_t\phi_{2,i,j,t}^{n}\right)^2}}\right)\\ &\left.\left.+\partial_t^-\left(\frac{\partial_t^+\phi_{2,i,j,t}^{n+1}}{\sqrt{\left(\partial_t^+\phi_{2,i,j,t}^{n}\right)^2+\left(\partial_x\phi_{2,i,j,t}^{n}\right)^2+\left(\partial_y\phi_{2,i,j,t}^{n}\right)^2}}\right)\right)\right)\end{aligned} \tag{4.29}$$

We now obtain six curvature terms

$$\begin{aligned}
C_{1,m,i,j,t} &= \frac{\alpha}{\sqrt{\left(\partial_x^+ \phi_{m,i,j,t}^n\right)^2 + \left(\partial_y \phi_{m,i,j,t}^n\right)^2 + \left(\partial_t \phi_{m,i,j,t}^n\right)^2}} \\
C_{2,m,i,j,t} &= \frac{\alpha}{\sqrt{\left(\partial_x^+ \phi_{m,i-1,j,t}^n\right)^2 + \left(\partial_y \phi_{m,i-1,j,t}^n\right)^2 + \left(\partial_t \phi_{m,i-1,j,t}^n\right)^2}} \\
C_{3,m,i,j,t} &= \frac{\alpha}{\sqrt{\left(\partial_y^+ \phi_{m,i,j,t}^n\right)^2 + \left(\partial_x \phi_{m,i,j,t}^n\right)^2 + \left(\partial_t \phi_{m,i,j,t}^n\right)^2}} \\
C_{4,m,i,j,t} &= \frac{\alpha}{\sqrt{\left(\partial_y^+ \phi_{m,i,j-1,t}^n\right)^2 + \left(\partial_x \phi_{m,i,j-1,t}^n\right)^2 + \left(\partial_t \phi_{m,i,j-1,t}^n\right)^2}} \\
C_{5,m,i,j,t} &= \frac{\alpha}{\sqrt{\left(\partial_t^+ \phi_{m,i,j,t}^n\right)^2 + \left(\partial_x \phi_{m,i,j,t}^n\right)^2 + \left(\partial_y \phi_{m,i,j,t}^n\right)^2}} \\
C_{6,m,i,j,t} &= \frac{\alpha}{\sqrt{\left(\partial_t^+ \phi_{m,i,j,t-1}^n\right)^2 + \left(\partial_x \phi_{m,i,j,t-1}^n\right)^2 + \left(\partial_y \phi_{m,i,j,t-1}^n\right)^2}}
\end{aligned} \tag{4.30}$$

and the new three-dimensional data term

$$\begin{aligned}
D_{m,i,j,t} = &- \left(\left(\left(f_{i,j,t} - u_1\right)^2 - \left(f_{i,j,t} - u_3\right)^2\right) H\left(\phi_{m,i,j,t}\right)\right. \\
&\left. + \left(\left(f_{i,j,t} - u_2\right)^2 - \left(f_{i,j,t} - u_4\right)^2\right)\left(1 - H\left(\phi_{m,i,j,t}\right)\right)\right)
\end{aligned} \tag{4.31}$$

This leads to the following discrete scheme:

$$\begin{aligned}
\frac{\phi_{1,i,j,t}^{n+1} - \phi_{1,i,j,t}^n}{\tau} =& \delta_\epsilon\left(\phi_{1,i,j,t}^n\right)\left(D_{2,i,j,t} - \mu_1\right. \\
&+ C_{1,1,i,j,t}\left(\phi_{1,i+1,j,t}^n - \phi_{1,i,j,t}^{n+1}\right) - C_{2,1,i,j,t}\left(\phi_{1,i,j,t}^{n+1} - \phi_{1,i-1,j,t}^n\right) \\
&+ C_{3,1,i,j,t}\left(\phi_{1,i,j+1,t}^n - \phi_{1,i,j,t}^{n+1}\right) - C_{4,1,i,j,t}\left(\phi_{1,i,j,t}^{n+1} - \phi_{1,i,j-1,t}^n\right) \\
&\left. + C_{5,1,i,j,t}\left(\phi_{1,i,j,t+1}^n - \phi_{1,i,j,t}^{n+1}\right) - C_{6,1,i,j,t}\left(\phi_{1,i,j,t}^{n+1} - \phi_{1,i,j,t-1}^n\right)\right)
\end{aligned} \tag{4.32}$$

$$\begin{aligned}
\frac{\phi_{2,i,j,t}^{n+1} - \phi_{2,i,j,t}^n}{\tau} =& \delta_\epsilon\left(\phi_{2,i,j,t}^n\right)\left(D_{1,i,j,t} - \mu_2\right. \\
&+ C_{1,2,i,j,t}\left(\phi_{2,i+1,j,t}^n - \phi_{2,i,j,t}^{n+1}\right) - C_{2,2,i,j,t}\left(\phi_{2,i,j,t}^{n+1} - \phi_{2,i-1,j,t}^n\right) \\
&+ C_{3,2,i,j,t}\left(\phi_{2,i,j+1,t}^n - \phi_{2,i,j,t}^{n+1}\right) - C_{4,2,i,j,t}\left(\phi_{2,i,j,t}^{n+1} - \phi_{2,i,j-1,t}^n\right) \\
&\left. + C_{5,2,i,j,t}\left(\phi_{2,i,j,t+1}^n - \phi_{2,i,j,t}^{n+1}\right) - C_{6,2,i,j,t}\left(\phi_{2,i,j,t}^{n+1} - \phi_{2,i,j,t-1}^n\right)\right)
\end{aligned} \tag{4.33}$$

We can solve this scheme using the Jacobi method. The system of equations now

consists of $n_x \times n_y \times n_t \times 2$ equations which have to be solved

$$\begin{aligned}\phi^{n+1}_{1,i,j,t} = & \frac{\phi^n_{1,i,j,t} + \tau\delta_\epsilon\left(\phi^n_{1,i,j,t}\right)\left(D_{2,i,j,t}\right) - \mu_1}{1 + \tau\delta_\epsilon\left(C_{1,1,i,j,t} + C_{2,1,i,j,t} + C_{3,1,i,j,t} + C_{4,1,i,j,t} + C_{5,1,i,j,t} + C_{6,1,i,j,t}\right)} \\ & + \frac{\tau\delta_\epsilon\left(\phi^n_{1,i,j,t}\right)\left(C_{1,1,i,j,t}\phi^n_{1,i+1,j,t} + C_{2,1,i,j,t}\phi^{n+1}_{1,i-1,j,t} + C_{3,1,i,j,t}\phi^n_{1,i,j+1,t}\right)}{1 + \tau\delta_\epsilon\left(C_{1,1,i,j,t} + C_{2,1,i,j,t} + C_{3,1,i,j,t} + C_{4,1,i,j,t} + C_{5,1,i,j,t} + C_{6,1,i,j,t}\right)} \\ & + \frac{\tau\delta_\epsilon\left(\phi^n_{1,i,j,t}\right)\left(C_{4,1,i,j,t}\phi^{n+1}_{1,i,j-1,t} + C_{5,1,i,j,t}\phi^n_{1,i,j,t+1} + C_{6,1,i,j,t}\phi^{n+1}_{1,i,j,t-1}\right)}{1 + \tau\delta_\epsilon\left(C_{1,1,i,j,t} + C_{2,1,i,j,t} + C_{3,1,i,j,t} + C_{4,1,i,j,t} + C_{5,1,i,j,t} + C_{6,1,i,j,t}\right)}\end{aligned} \tag{4.34}$$

$$\begin{aligned}\phi^{n+1}_{2,i,j,t} = & \frac{\phi^n_{2,i,j,t} + \tau\delta_\epsilon\left(\phi^n_{2,i,j,t}\right)\left(D_{1,i,j,t}\right) - \mu_2}{1 + \tau\delta_\epsilon\left(C_{1,2,i,j,t} + C_{2,2,i,j,t} + C_{3,2,i,j,t} + C_{4,2,i,j,t} + C_{5,2,i,j,t} + C_{6,2,i,j,t}\right)} \\ & + \frac{\tau\delta_\epsilon\left(\phi^n_{2,i,j,t}\right)\left(C_{1,2,i,j,t}\phi^n_{2,i+1,j,t} + C_{2,2,i,j,t}\phi^{n+1}_{2,i-1,j,t} + C_{3,2,i,j,t}\phi^n_{2,i,j+1,t}\right)}{1 + \tau\delta_\epsilon\left(C_{1,2,i,j,t} + C_{2,2,i,j,t} + C_{3,2,i,j,t} + C_{4,2,i,j,t} + C_{5,2,i,j,t} + C_{6,2,i,j,t}\right)} \\ & + \frac{\tau\delta_\epsilon\left(\phi^n_{2,i,j,t}\right)\left(C_{4,2,i,j,t}\phi^{n+1}_{2,i,j-1,t} + C_{5,2,i,j,t}\phi^n_{2,i,j,t+1} + C_{6,2,i,j,t}\phi^{n+1}_{2,i,j,t-1}\right)}{1 + \tau\delta_\epsilon\left(C_{1,2,i,j,t} + C_{2,2,i,j,t} + C_{3,2,i,j,t} + C_{4,2,i,j,t} + C_{5,2,i,j,t} + C_{6,2,i,j,t}\right)}\end{aligned} \tag{4.35}$$

All the segmentation methods presented in this chapter were implemented both as spatial and as spatiotemporal methods in MATLAB as well as in C.

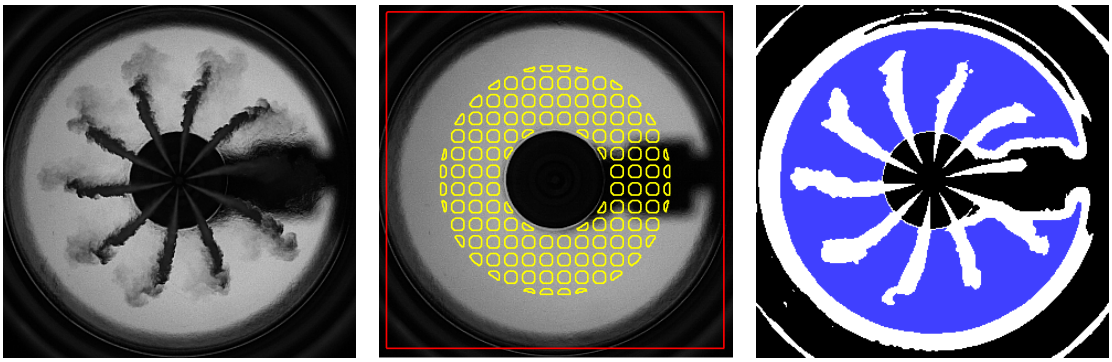
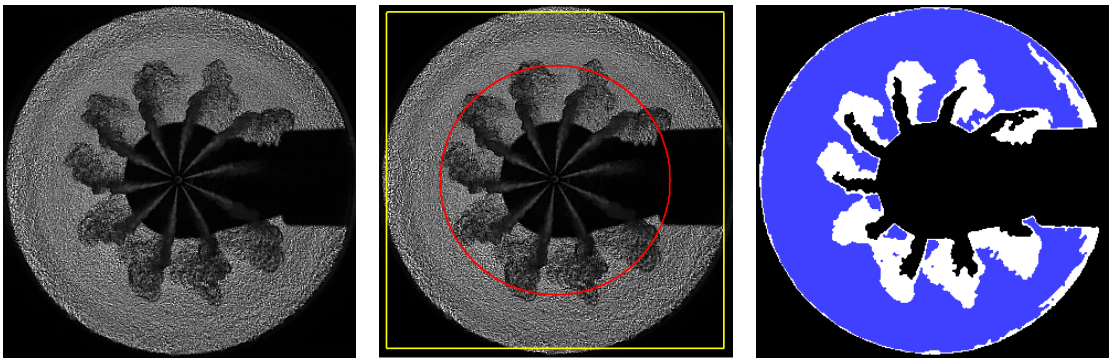


Figure 4.4: Spatiotemporal fourphase Chan Vese segmentation. The red square denotes the contour of level set function 1, while the yellow lines denote the contour of level set function 2. Segmentation parameters: $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1.0$, $\alpha_1 = \alpha_2 = 0.3$, $\mu_1 = 2.0$, $\mu_2 = 0.0$, $\sigma_{xy} = 0.8$, $\sigma_t = 0.5$



Original image | Initial contours | Segmentation result

Figure 4.5: Spatiotemporal fourphase Chan Vese segmentation of liquid and vapor phases. The yellow square denotes the contour of level set function 1, while the red circle denotes the contour of level set function 2. Segmentation parameters: $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1.0$, $\alpha_1 = \alpha_2 = 0.3$, $\mu_1 = 0.0$, $\mu_2 = 5.3$, $\sigma_{xy} = 0.9$, $\sigma_t = 0.6$

4.6 Optical Flow as a Feature for Segmentation

Instead of only considering the gray value of the image sequence as input feature, we can control the segmentation by adding more relevant features to the data term. The assumption in the data term

$$\begin{aligned} &\lambda_1 \int\limits_{inside(C)} (f(\boldsymbol{x}) - u_{in})^2 \, d\boldsymbol{x} \\ +&\lambda_2 \int\limits_{outside(C)} (f(\boldsymbol{x}) - u_{out})^2 \, d\boldsymbol{x} \end{aligned} \tag{4.36}$$

can be easily extended to a vector-valued data term as introduced by [CSV00]

$$\begin{aligned} &\frac{1}{V} \int\limits_{inside(C)} \sum_{v=1}^{V} \lambda_1^v \, (f^v(\boldsymbol{x}) - u_{in}^v)^2 \, d\boldsymbol{x} \\ +&\frac{1}{V} \int\limits_{outside(C)} \sum_{v=1}^{V} \lambda_2^v \, (f^v(\boldsymbol{x}) - u_{out}^v)^2 \, d\boldsymbol{x} \end{aligned} \tag{4.37}$$

Since the liquid and vapor phase are governed by the laws of compressible flows, it makes sense to incorporate an assumption from physics into the data term. Before performing the segmentation, we compute the optical flow field $\boldsymbol{u}$ of the image sequence, as explained in section 2.9. The magnitude of an optical flow vector provides information about the speed of movement at a particular location in the image. In our case, this is a useful input feature, since the liquid and vapor phases propagate with slightly different speed. The further the particles are dispersed, the slower their movement becomes.

The vapor phase can not easily be distinguished from the background, since they have very similar brightness. However, using motion information, it becomes easier to distinguish the vapor phase from the background, since they follow different motion patterns.

To include motion information in the Chan Vese model, we simply add an additional data term $(f(\boldsymbol{x}) - |\boldsymbol{u}|)^2$ to the vector valued data term from equation 4.37 and assign a new weight ρ.

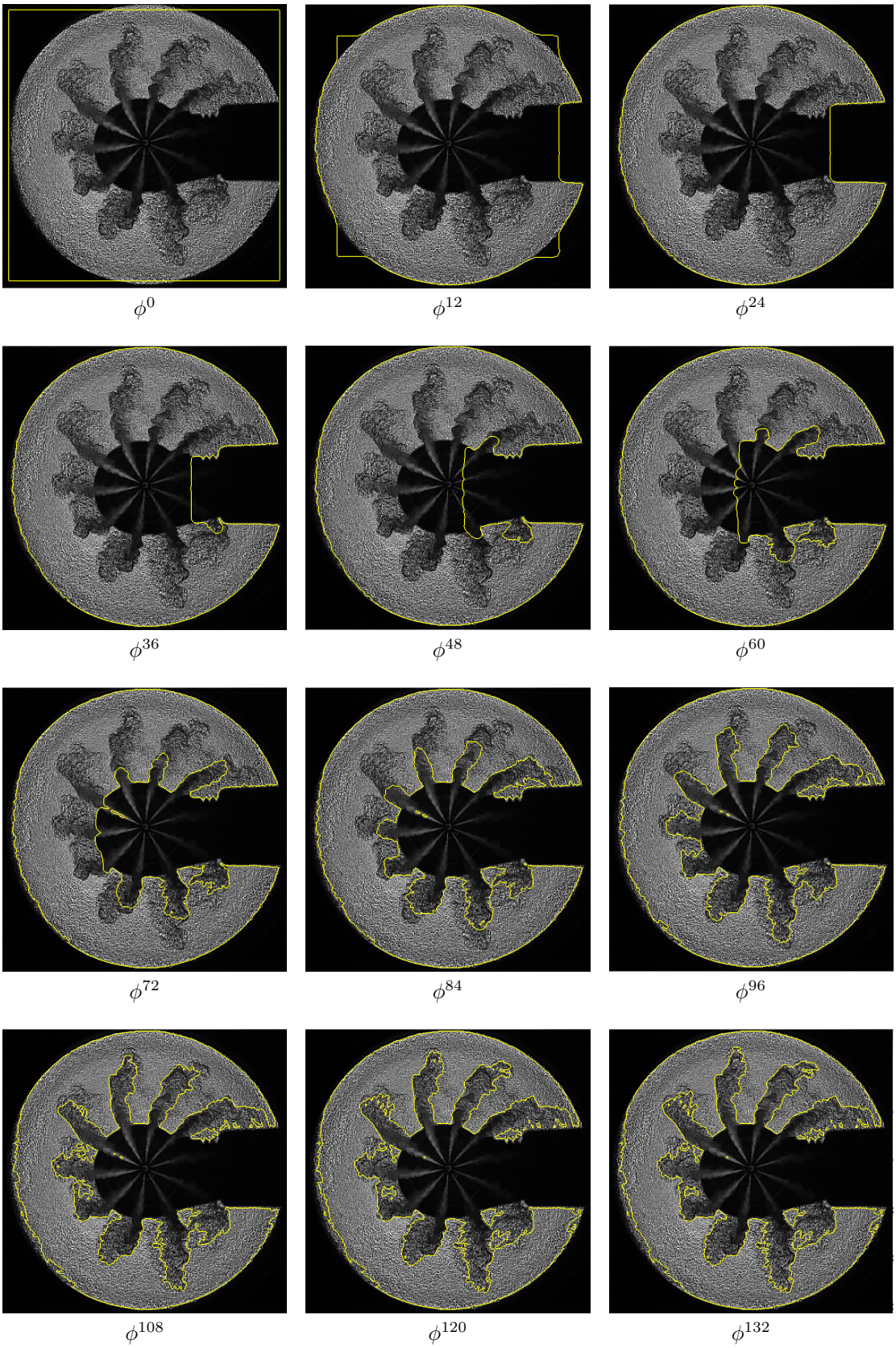


Figure 4.6: Contour evolution of Chan Vese segmentation with 132 iterations. Parameters: $\lambda_1 = 1.0$, $\lambda_2 = 1.0$, $\alpha = 0.3$, $\mu = 4.8$, $\sigma_x = \sigma_y = \sigma_t = 0.8$

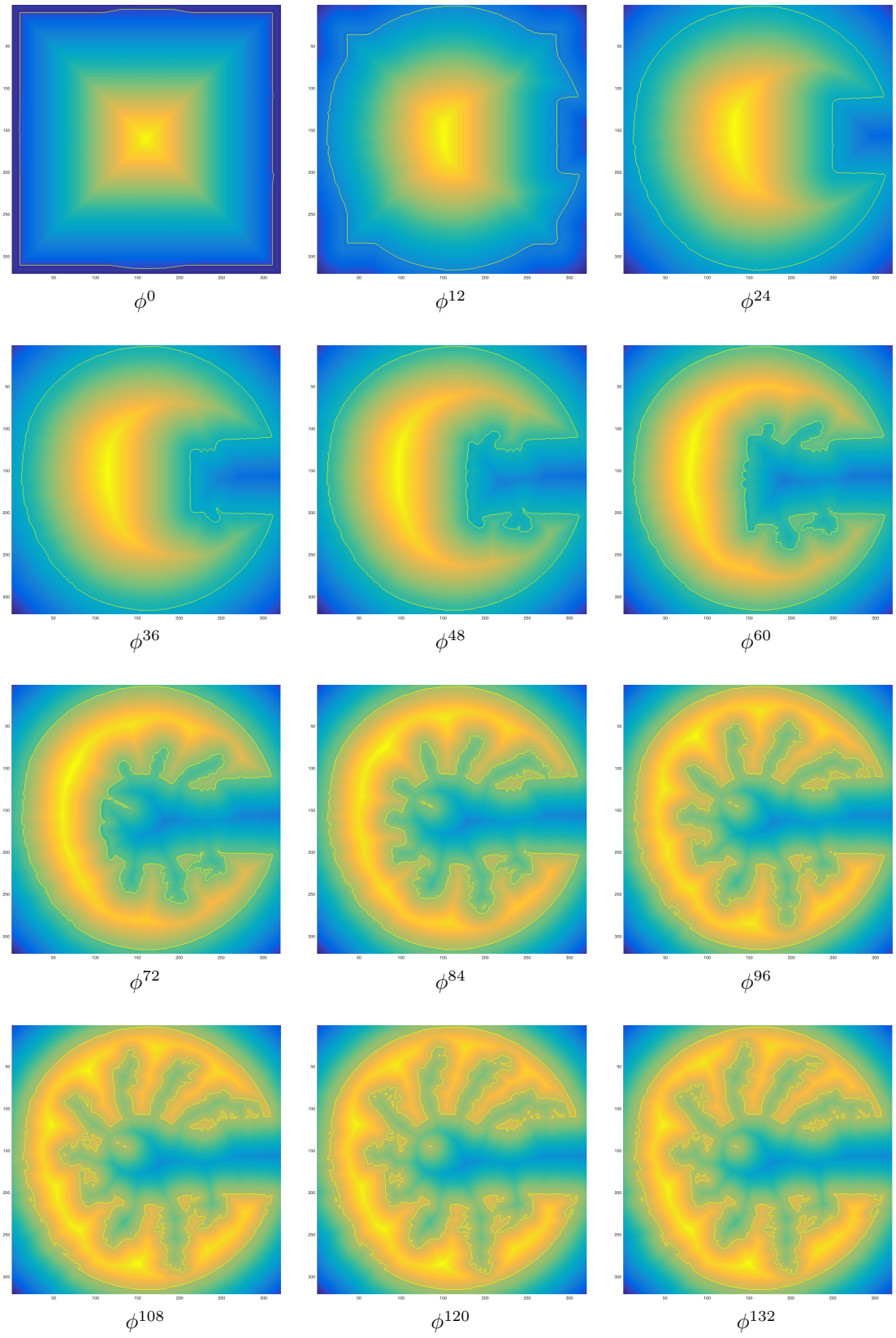


Figure 4.7: 2D level set evolution of Chan Vese segmentation with 132 iterations. Parameters: $\lambda_1 = 1.0$, $\lambda_2 = 1.0$, $\alpha = 0.3$, $\mu = 4.8$, $\sigma_x = \sigma_y = \sigma_t = 0.8$

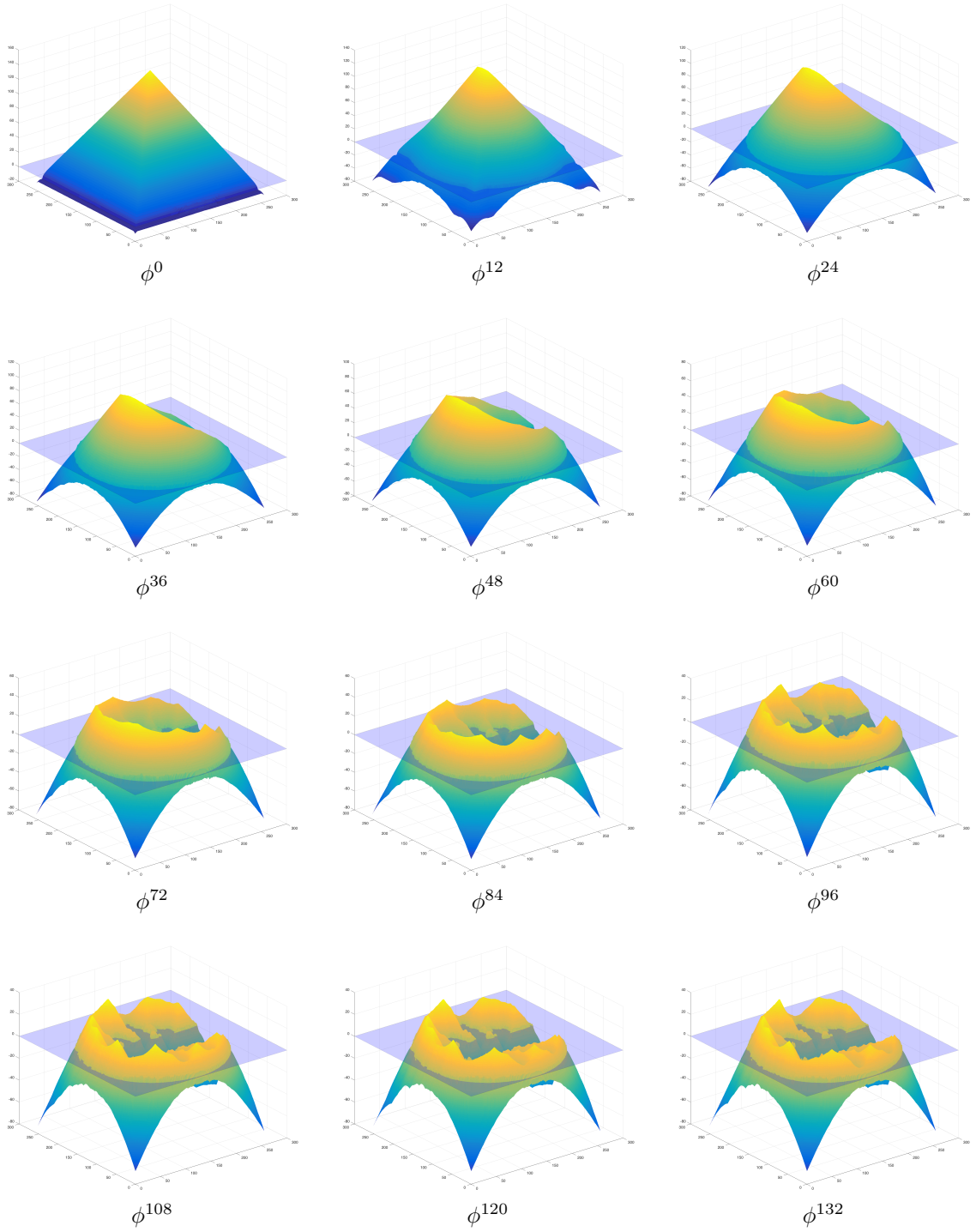


Figure 4.8: 3D level set evolution of Chan Vese segmentation with 132 iterations. Parameters: $\lambda_1 = 1.0$, $\lambda_2 = 1.0$, $\alpha = 0.3$, $\mu = 4.8$, $\sigma_x = \sigma_y = \sigma_t = 0.8$

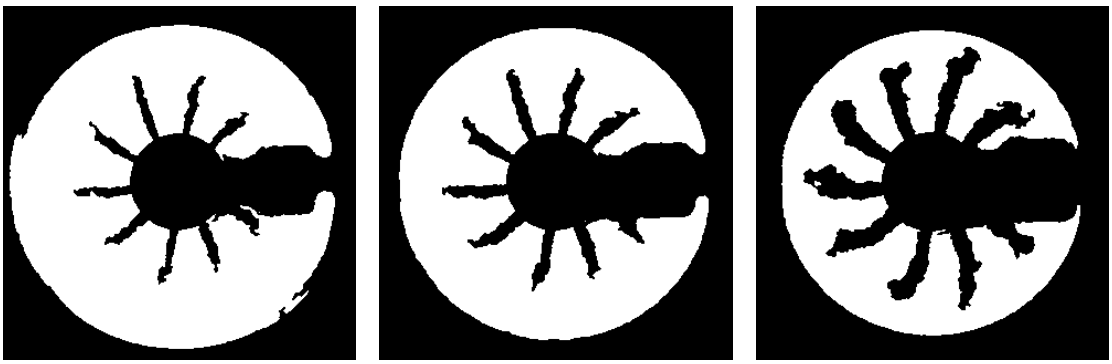
5 Results and Conclusions

5.1 Parameter Selection

In order to being able to interpret the results correctly, one has to understand how the parameter values affect the resulting segmentation and how the parameters influence each other.

5.1.1 Weight of the Data Term

The data term ensures that the segmentation stays close to the original data by penalizing the discrepancy between the input image and the arithmetic mean of each phase. The influence of this term is controlled using a weighting parameters λ_1 (for the foreground phase) and λ_2 (for the background phase). If $\lambda_1 < \lambda_2$, the energy fits the background phase more accurately to the original image. If $\lambda_1 > \lambda_2$, the energy will fit the foreground with higher priority. In our experiments, we keep both weight fixed at $\lambda_1 = \lambda_2 = 1.0$.



$\lambda_2 > \lambda_1$ $\lambda_1 = \lambda_2$ $\lambda_1 > \lambda_2$

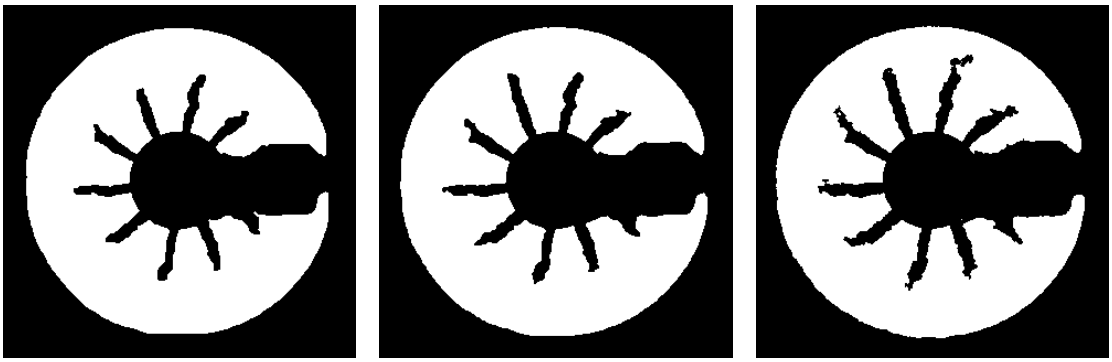
Figure 5.1: Effect of modifying the weights of data term. The foreground phase is shown in white while the background phase is shown in black. Left: The segmentation tries to fit the background phase with higher fidelity. Therefore, in the critical area of the vapor phase, the minimum is attained by assigning more pixels to the foreground phase. Right: The foreground phase is required to be more faithful to the input data. In the questionable region of the vapor phase, the algorithm assigns less pixels to the foreground phase.

Beware: The gray value range of the input image has a direct effect on the weight of the data term. For example, if the range of an image is normalized to $[0, 1]$, the maximum deviation between two values is only 1. However, if the input image has a range of $[0, 255]$, the deviation is as high as 255. Depending on the quantization

of the input image (see section 2.5), the weight of the data term has to be adjusted accordingly. Our implementation also allows to automatically normalize the input data to the range $[0, 1]$, if desired.

5.1.2 Weight of the Length Term

The weight α penalizes the length of the contour. When the weight is chosen low, the contour is allowed to be longer, which results in a more detailed boundary between the two phases. If one imagines the contour as an elastic band, than the length parameter controls the tension on the band. A larger weight will increase the tension and produce a smoother boundary between the two phases. The lower the length penalty, the more detailed is the segmentation and the more objects will be detected. To fit the vapor phase of the Schlieren sequences more accurately, we usually choose a low value such as $\alpha = 0.3$. A low value of the length parameter can be susceptible to noise, but poses no problem if the data is lowpass filtered (see section 2.6).



$\alpha = 1.7$ $\alpha = 1.0$ $\alpha = 0.0$

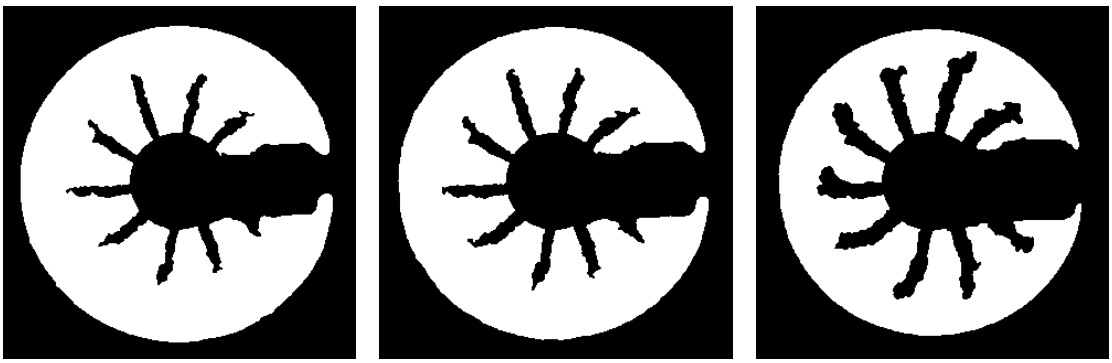
Figure 5.2: Effects of modifying the weights of the length term. The foreground phase is shown in white, the background phase in black. Left: A large value produces a smooth boundary between the liquid phase and the background. Right: By neglecting the length term, we obtain a much more detailed segmentation of the liquid phase.

5.1.3 Weight of the Area Term

The weight μ penalizes the area of the foreground phase (or the background phase if $\mu < 0$).

This term is only meaningful when there exists a prescribed inside and outside of the segmentation boundary and one has a rough idea of the volume of each phase. If the value is chosen large, the background phase will expand and move the boundary in direction of the foreground phase. If the value is lower, the area of the foreground phase will be expanded. Since we have a rough idea of the volume of each phase, the area term μ provides us with a convenient tool to adjust the boundary between the phases in an injection image sequence. When separating the vapor phase from

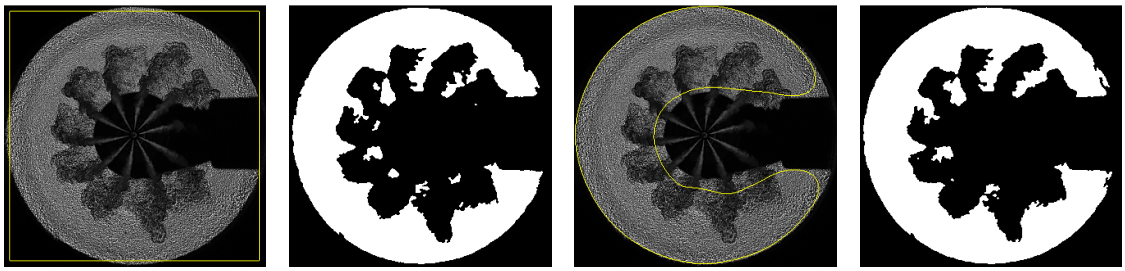
the background, a larger value ($\mu > 4.0$) is used. When separating the liquid from the vapor phase, a smaller value is recommended ($0 < \mu < 4.0$).



$\mu = -2.0$ $\mu = 0.0$ $\mu = 4.0$

Figure 5.3: Effects of modifying the weights of the area term: The foreground phase is shown in white. The background phase is displayed in black. Left: A negative penalty for the volume of the foreground phase will increase the size of the foreground phase. Right: A larger weight to penalize the foreground phase results in a shrinking of the foreground phase.

5.1.4 Choice of Initialization



(a) (b) (c) (d)

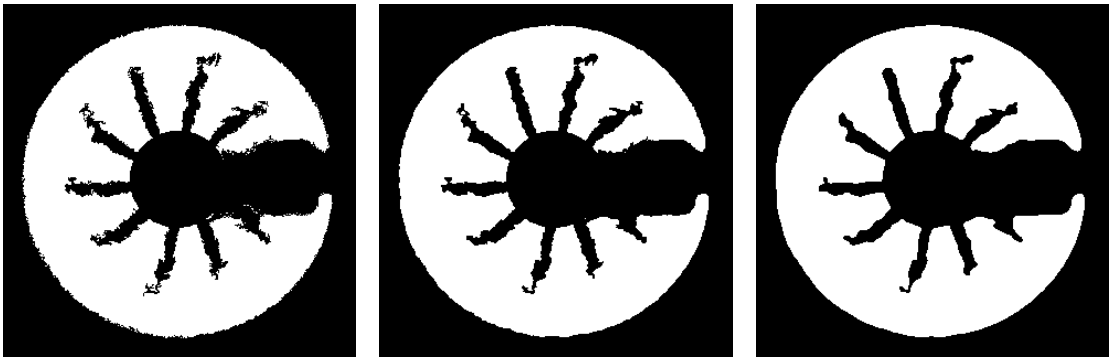
Figure 5.4: Effect of initial level set on evolution time. The rectangular initialization on the left leads to a similar result as the thresholding initialization on the right. However, the rectangular initialization needs 159 iterations while the thresholding initialization only needs 54 iterations with the same time step size.

Since the energy is non-convex, it has multiple local minimizers. By choosing the appropriate type of initialization, we can influence the outcome of the segmentation. Five different shapes are offered to the user to initialize the segmentation process. A circular/donut-shaped contour, a rectangular contour, a thresholding boundary contour and circular and rectangular pattern contours. By simply inverting a level set function, it is possible to exchange the foreground- and background phase. This can be important when working with more than one level set. All level set functions created from these initial contours are designed as Euclidean signed distance functions, as already mentioned in section 4.3.3. Some examples of initial contours and their respective level sets are shown in figure 5.12.

Rectangular and circular initializations are useful if one wants to enclose a certain area of the image. A thresholding contour uses the average gray value of the data as a threshold between the two phases. The image is lowpass filtered and each pixel is assigned to a phase according to its gray value. A thresholding initialization can be useful if the areas that should be segmented in the image differ in gray value. Pattern contours are especially suitable as a general-purpose tool, for example if one has no prior knowledge on the type of data to be segmented.

The initial contour not only influences the segmentation result, but also the evolution time. Contours with higher curvature, such as the pattern contours, evolve distinctly faster than contours with lower curvature, such as rectangles. An example is given in figure 5.4.

5.1.5 Gaussian Kernel for Presmoothing



$\sigma_{xy} = 0.0$ $\sigma_{xy} = 0.8$ $\sigma_{xy} = 1.6$

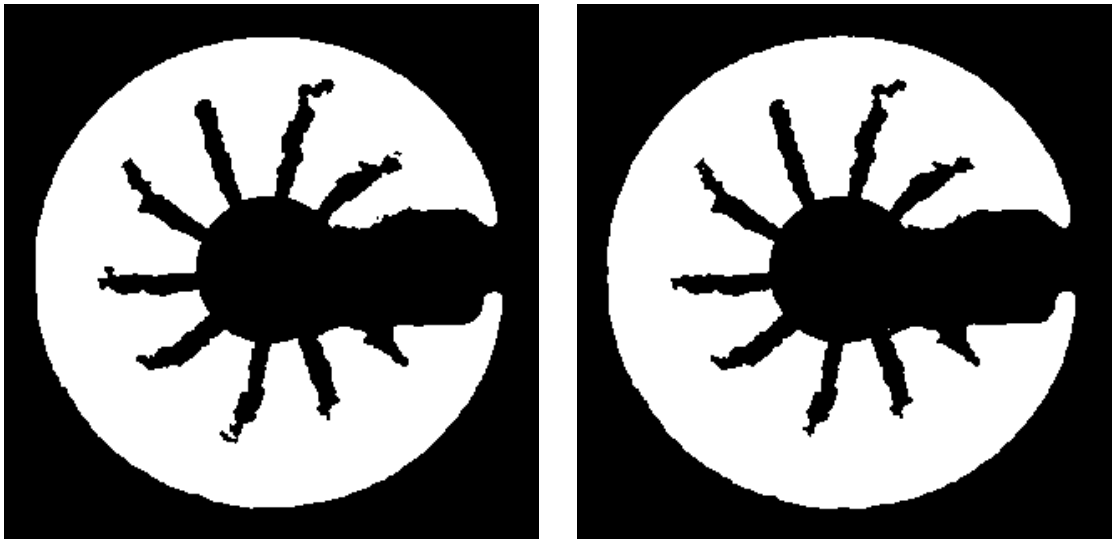
Figure 5.5: Effect of modifying the standard deviation of the Gaussian kernel. Without presmoothing, the contour of the liquid phase and along the convection boundaries of the chamber is very crude. A high standard deviation creates a very smooth contour but also overlooks subtleties in the contour of the liquid phase.

The parameters σ_x, σ_y and σ_t control the amount of presmoothing of the original image data before performing the actual segmentation. Noise is always present as a result of the image acquisition process. Schlieren photography poses additional problems for image processing. The free convection boundary layer between the hot chamber and the room air leads to a severe amount of background Schlieren. Presmoothing the input data can be a helpful tool in discriminating the background Schlieren from the Schlieren of the fuel. If the parameters are chosen very low, the segmentation will be more faithful to the original data, including the noise. By increasing the value, noise can be removed from the signal. (Remember that Gaussian convolution acts as a lowpass filter 2.6.) On the downside, together with the noise, also important high frequency image structures might get lost. In our experiments with the sequence of figure 2.11, the best compromise was a spatial presmoothing of $\sigma_x = \sigma_y = 0.8$ and a temporal presmoothing of $\sigma_t = 0.5$, depending of course on the tyoe of input data. The effect of modifying the standard deviation

of the Gaussian kernel on the segmentation result is shown in figure 5.5.

5.1.6 Reinitialization of Level Set Function

As mentioned in section 4.3.3, the level set function can develop shocks during evolution. A reinitialization helps to keep the level set function in shape, but also prevents the growth of inner contours. How this affects the Schlieren data is explained in figure 5.6.

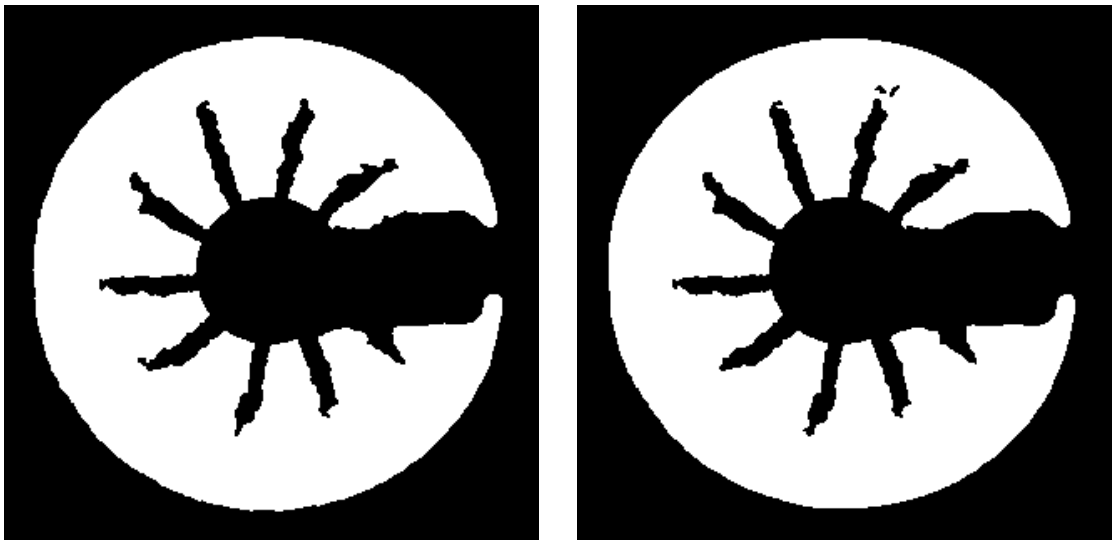


(a) Reinitialization after every 10 iterations (b) Reinitialization after every iteration

Figure 5.6: Effect of different reinitialization strategies: If the reinitialization is performed only sparsely, inner contours such as droplets of liquid are still preserved (Figure a). A reinitialization after every iteration discards the droplets. (Figure b)

5.1.7 Choice of Integration Domain

The segmentation of an image sequence can either be performed on all single images or on one three dimensional image object. When performing the segmentation in 2D, the algorithm can not exploit information in time direction. An example of a segmentation using the same parameters both in the spatial and the spatiotemporal domain is shown in figure 5.7

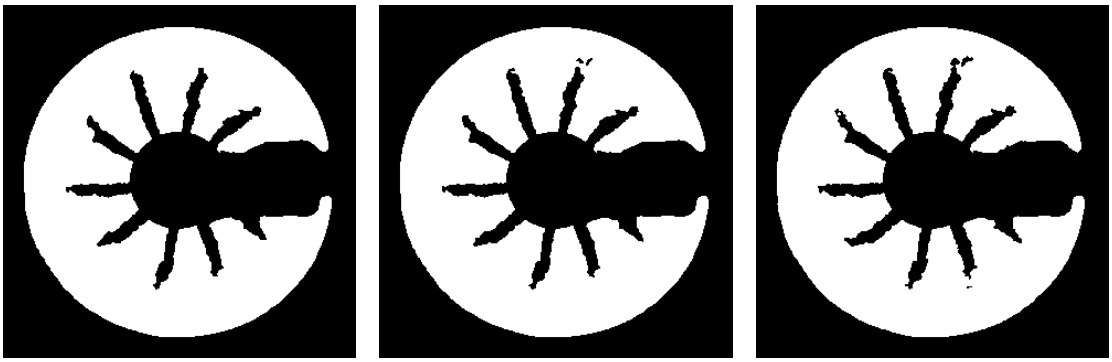


(a) Spatial domain (b) Spatiotemporal domain

Figure 5.7: Integration in the spatial and spatiotemporal domain. At the contour of the liquid phase, the fuel breaks apart into droplets. The spatiotemporal approach carries over information from neighboring frames. The droplets in b) are still detected as part of the liquid phase.

5.1.8 Modifying the Grid Size

The grid size affects how fast the segmentation progresses in a certain dimension. By modifying the spatial grid size in only one direction, it is possible to artificially speed up the segmentation along only this dimension. Since we are only working with data using a squared pixel grid, we keep both the spatial grid sizes at $h_x = h_y = 1.0$ and only modify the temporal grid size h_t accordingly. When choosing a small grid size in time direction, the segmentation process will evolve faster in time direction, and vice versa. The effects of modifying the temporal grid size is shown in figure 5.8.



$h_t = 5.0$ $h_t = 1.0$ $h_t = 0.1$

Figure 5.8: Effect of varying grid size in temporal direction. A small grid size adds more droplets from the previous frames into the segmentation result (right). If the temporal grid size is large, the segmentation will progress slower in time direction, therefore discarding the droplets from the previous frames (left)

Table 5.1: Evaluation of optical analysis preprocessing

	Correctly classified	Within margin	Falsely classified
Nozzle Detection	97.47%	-	2.53%
FSOL Detection	82.28%	16.45%	1.27%
FSOF Detection	65.82%	27.85%	6.33%

5.2 Analysis and Results

5.2.1 Preprocessing

The preprocessing methods presented in chapter 3 were evaluated on a test dataset of 79 injection sequences. The test dataset contained shadowgraph- and schlieren sequences, sequences with one or multiple obstacles, coated and uncoated nozzle tips and injection pressures ranging from 400 to 2000 bar. The sequences were first evaluated manually by an optical diagnostics engineer and then processed by the algorithm. The results were compared.

The algorithm for nozzle detection worked flawlessly on 97.47% of the scenes. In most of the scenes, it is not possible to install the injector right in the center and in normal direction to the image plane. If the injector is slightly skewed, the nozzle tip is not located exactly in the center of the shadow circle. This is why it is recommended to manually adjust the coordinates of the nozzle tip to within pixel precision, since the exact location of the nozzle tip affects all other measurements. In the few cases with multiple obstacles, the Hough transformation did not work as desired and had to be readjusted manually by the user.

The exact detection of the first sight of light and the first sight of fuel prove to be a difficult task even for a trained and skilled human observer. If it is not possible to pinpoint one exact frame as the correct one and there are two consecutive frames in question, we consider those two frames to be within the margin. If the detected frame by the algorithm lies outside of this margin, we consider it as falsely classified.

For the detection of the first sight of light, in 98.73% of the cases the algorithm was able to obtain a result at least as good as a human. In 1.27% of the cases, the result was falsely classified, especially in those sequences with severe fluctuations by the laser light source across multiple frames.

The detection of the first sight of fuel was correctly classified in 93.67% of the cases. In 27.85% of the cases where the FSOF could not be located exactly in one frame, the FSOF detection was still within the margin of the two frames. In 1.34% of the cases where the FSOF was falsely classified, the FSOF computed by the algorithm was at a maximum off by one frame from the margin.

5.2.2 Penetration Curve

Using the segmentation result, we are able to compute the penetration curve of each jet. An example is shown in figure 5.9. Since the jet plume drifts in clockwise direction, it is advised to also move the partitioning boundaries in clockwise direction by a few degrees to prevent the jet from crossing into the neighboring partition later in the sequence.

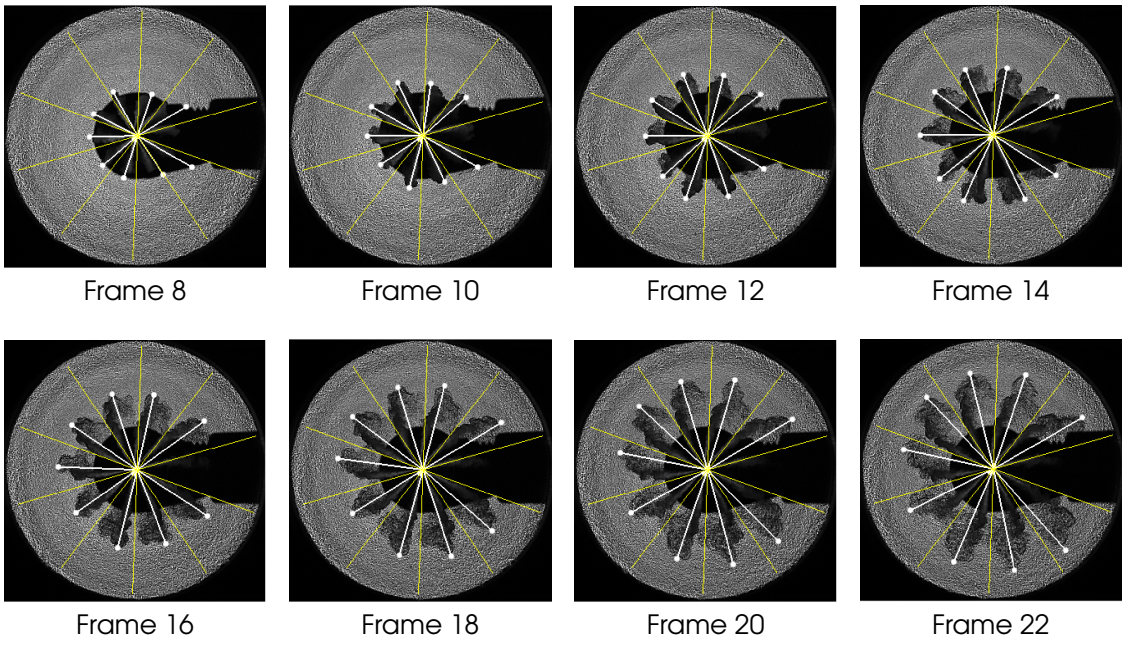


Figure 5.9: Jet penetration of the vapor phase

5.2.3 Chan Vese Segmentation with Optical Flow Data Term

Chan Vese segmentation is an active contour segmentation model, based on the theory of curve evolution. It allows us to detect objects whose boundaries are not necessarily defined by gradients. As we have seen for the case of injection sequences, this is a very desirable property, since the phases of compressible flows usually have a smeared and diffuse boundary and can not easily be distinguished using only edge information. When working with image sequences of fluids, it makes sense to include ideas from physics into the model, in order to better capture the characteristics of such flow patterns.

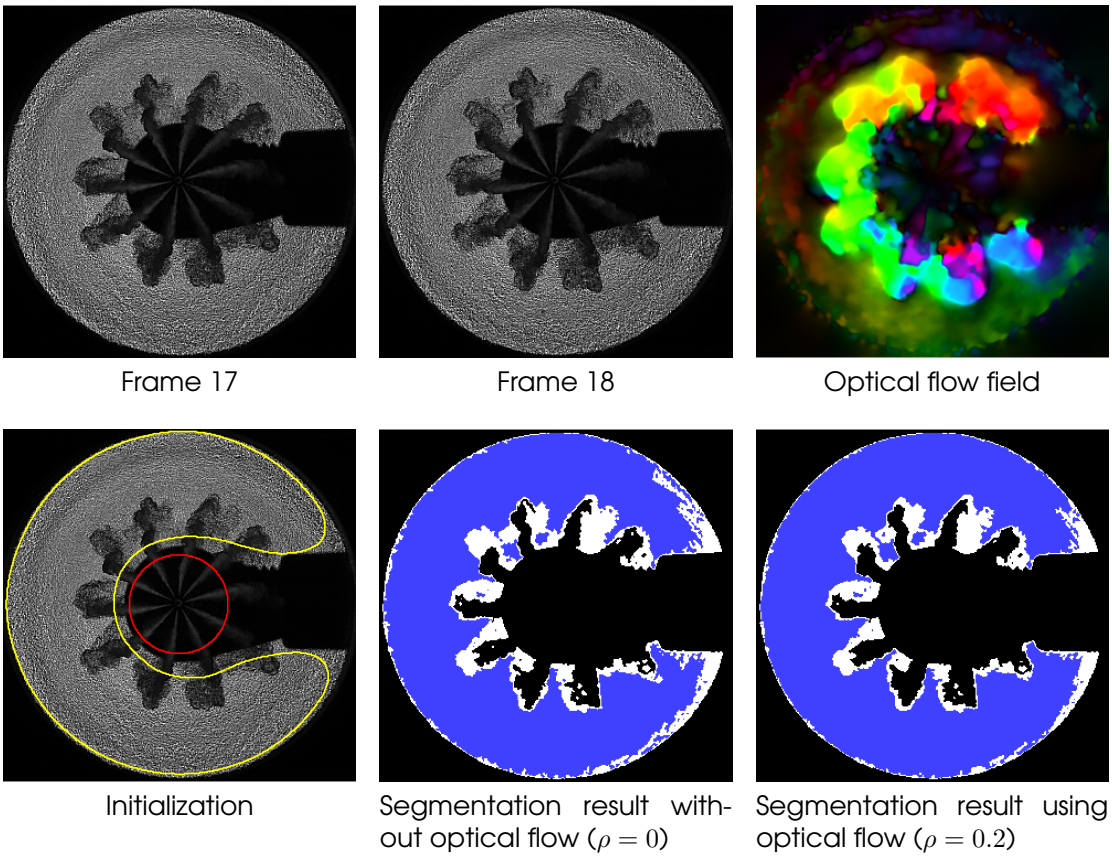


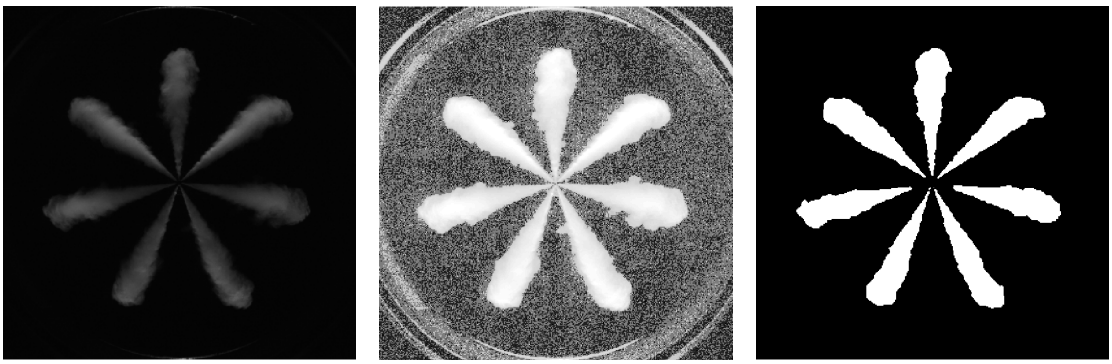
Figure 5.10: Chan Vese segmentation using the magnitude of the optical flow field of Brox et al. in the data term. Optical flow parameters: $\alpha = 6.0$, $\gamma = 0.2$, $\sigma_x yt = 0.8$Segmentation parameters: $\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1.0$, $\alpha_1 = \alpha_2 = 0.3$, $\mu_1 = 5.5$, $\mu_2 = 0.3$, $\sigma_x y = 0.8$, $\sigma_t = 0.6$, $\rho = 0.2$. The yellow line denotes the contour of the first level set function, the red one of the second.

As explained in section 4.6, the optical flow field can be included in the data term of the Chan Vese model. The optical flow field contains valuable information that can be used to improve the segmentation result. Using the optical flow in the data term allows to better distinguish the vapor phase from the background, especially in the area between the jets close to the image center. However, it can be tedious to find a set of suitable parameters for both approaches, that yields a notable improvement compared to the regular data term. We obtained the best results by

assigning a weight $\rho = 0.2$ to the optic flow term of the Chan Vese data term and by choosing a larger smoothness term weight with around $6.0 < \alpha < 10.0$, in order to avoid contradictory information from the optical flow to perturb the segmentation. The temporal presmoothing should be kept low at $\sigma_t < 0.3$, otherwise, the speed of the vapor phase can not be distinguished from the Schlieren movement of the background. An example of a segmentation using the optical flow is shown in figure 5.10.

By extending the Chan Vese model to a multiphase scenario with more than just two phases, we are able to distinguish the liquid and vapor phases from the background, using only one consistent mathematical approach. We have seen that performing a histogram equalization allows us to distinguish the four phases even better, especially in the multiphase case. It was shown that, by using an appropriate initialization, the four phases of the Chan Vese model can be exactly matched with the four physical phases seen in the data (liquid phase, vapor phase, illuminated background, chamber background).

Another advantage of the Chan Vese method is that it is very versatile and can be applied to any type of input data. Figure 5.11 shows the Chan Vese algorithm applied to a sequence acquired with the Mie scattering method, introduced in chapter 2.



Frame 80 | Frame 80 Histogram equalized | Segmentation result

Figure 5.11: Chan Vese segmentation of a Mie scattering image. Left: Original image. Center: Histogram equalized image from a). Right: ChanVese segmentation result using the parameters $\lambda_1 = \lambda_2 = 1.0$, $\alpha = 0.3$, $\mu = 0.0$, $\sigma_x y = 0.3$, $\sigma_t = 0.1$

5.3 Future Work

- Increasing the temporal and/or spatial resolution of the data could improve the optical flow results. This of course comes at the cost of hardware expenses and increased computation time.
- Using an advanced camera setup that can record color images would also allow to analyze the combustion and soot formation phases of a combustion and would allow to distinguish the spray even better from the background light.
- Some additional image corrections (such as background subtraction, non-local means or bilateral filtering) might be helpful in separating the actual Schlieren created by the spray from the Schlieren of the gas in the background.
- By extrapolating the data obtained from the penetration curve, it is possible to determine t_{FSOF} with subframe precision, which would help to make the measurements even more precise.
- When the jets have penetrated deeply into the combustion chamber, they start to drift. When using a partitioning scheme as described in section 3.5, a jet might partially cross over into another partition, distorting the results. Instead of the pie shaped partitioning scheme, it might be desirable to apply a partitioning that respects the shape of the jets, for example by using a clustering method such as a Voronoi partitioning on the segmented data.
- All the injection sequences are circular in nature. Performing a Fourier Transformation and analyzing the data in the frequency domain might reveal additional information and simplify some of the operations from chapter 3.

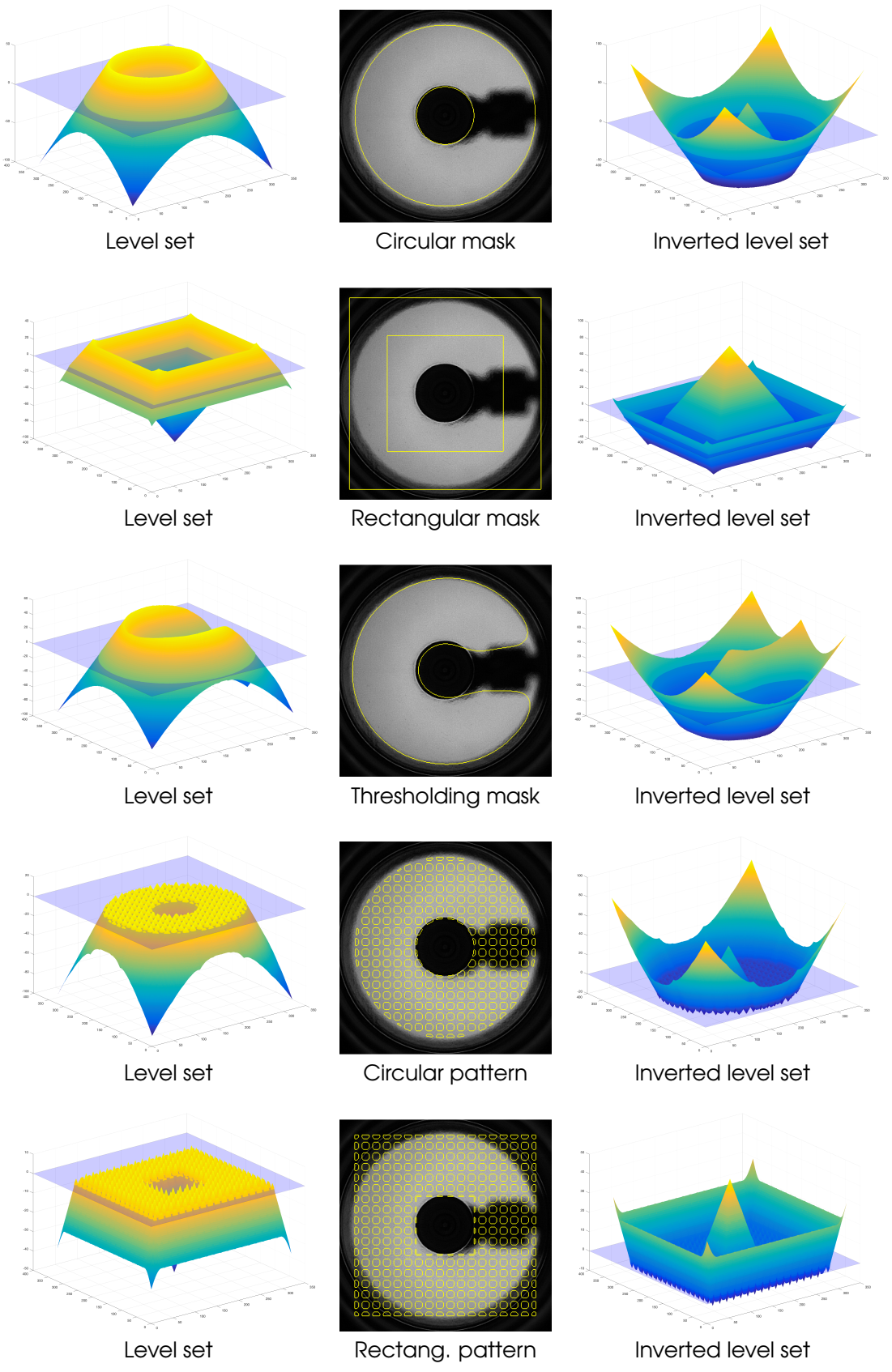


Figure 5.12: Possible initializations for the level set function

List of Figures

List of Tables

References

(BA91) Michael Julian Black and P. Anandan. "Robust dynamic motion estimation over time". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Maui, Hawaii, USA, 1991, pp. 296–302 (cit. on p. 17).

(Bro+04) Thomas Brox et al. "High accuracy optical flow estimation based on a theory for warping". In: *European Conference on Computer Vision*. Vol. 3024. Prague, Czech Republic, 2004, pp. 25–36 (cit. on pp. 17, 18).

(Bru06) Andrés Bruhn. "Variational Optic Flow Computation, Accurate Modelling and Efficient Numerics". PhD thesis. Saarland University, Saarbrücken, Germany, Aug. 2006 (cit. on p. 18).

(BSS93) Guy Barles, H Mete Soner, and Panagiotis E Souganidis. "Front Propagation and phase field theory". In: *SIAM Journal on Control and Optimization* 31.2 (1993), pp. 439–469 (cit. on p. 40).

(BWS05) Andrés Bruhn, Joachim Weickert, and Christoph Schnörr. "Lucas Kanade Meets Horn Schunck: Combining Local and Global Optic Flow Methods". In: *International Journal of Computer Vision*. Vol. 61. 3. Springer Science and Business Media, 2005, pp. 211–231 (cit. on p. 17).

(CSV00) Tony F. Chan, B. Yezrielev Sandberg, and Luminita A. Vese. "Active Contours without Edges for Vector-Valued Images". In: *Journal of Visual Communication and Image Representation* 11 (2000), pp. 130–141 (cit. on p. 54).

(CV01) Tony Chan and Luminita Vese. "Active Contours Without Edges". In: *IEEE Transactions on Image Processing* 10.2 (2001) (cit. on pp. 33, 39).

(Dry97) F Dryer. *Physical and Chemical Aspects of Combustion: A Tribute to Irvin Glassman*. Taylor and Francis, 1997 (cit. on p. 10).

(GF00) Jose Gomes and Olivier D. Faugeras. "Reconciling Distance Functions and Level Sets". In: *J. Visual Communication and Image Representation* 11.2 (2000), pp. 209–223 (cit. on p. 40).

(GL08) D. A. Green and R. Lewis. "The effects of soot-contaminated engine oil on wear and friction: A review". In: *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering* 222.9 (2008) (cit. on p. 2).

(GW64) M L Goldberger and K M Watson. *Collision Theory*. New York: Wiley, 1964 (cit. on p. 10).

(KWT88) Michael Kass, Andrew Witkin, and Demetri Terzopoulos. "Snakes: Active contour models". In: *International Journal of Computer Vision* 1.4 (1988), pp. 321–331 (cit. on p. 31).

(Mau+03) Calvin R. Maurer et al. "A linear time algorithm for computing exact Euclidean distance transforms of binary images in arbitrary dimensions". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2003), pp. 265–270 (cit. on p. 40).

(MOS92) W. Mulder, S. Osher, and James A. Sethian. "Computing interface motion in compressible gas dynamics". In: *J. Comput. Phys.* 100.2 (June 1992), pp. 209–228 (cit. on p. 39).

(MS89) David Mumford and Jayant Shah. "Optimal approximations by piecewise smooth functions and associated variational problems". In: *Communications on Pure and Applied Mathematics* 42.5 (1989), pp. 577–685. ISSN: 00103640 (cit. on p. 33).

(NKM12) Tuan Anh Nguyen, Yuichiro Kai, and Masato Mikami. "Study on Combustion Noise from a Running Diesel Engine Based on Transient Combustion Noise Generation Model". In: *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering* 3.4 (2012) (cit. on p. 2).

(OF03) Stanley Osher and Ronald Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces*. 1st ed. Vol. 153. Applied Mathematical Sciences. Springer-Verlag New York, 2003 (cit. on pp. 37, 40).

(OS88) Stanley Osher and James A. Sethian. "Fronts Propagating with Curvature-dependent Speed: Algorithms Based on Hamilton-Jacobi Formulations". In: *J. Comput. Phys.* 79.1 (Nov. 1988), pp. 12–49 (cit. on p. 31).

(Pap+06) Nils Papenberg et al. "Highly Accurate Optic Flow Computation with Theoretically Justified Warping". In: *International Journal of Computer Vision* 67.2 (2006), pp. 141–158 (cit. on p. 18).

(Pen+99) Danping Peng et al. "A PDE-Based Fast Local Level Set Method". In: *Journal of Computational Physics* 155 (1999) (cit. on p. 40).

(Set01) G. S. Settles. *Schlieren and Shadowgraph Techniques*. Springer Verlag, 2001 (cit. on p. 9).

(Set99) J.A. Sethian. *Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision, and Materials Science*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 1999. ISBN: 9780521645577 (cit. on pp. 33, 40).

(SFN95) T. Su, P. Farrell, and R. Nagarajan. "Nozzle Effect on High Pressure Diesel Injection". In: *SAE 1995 Transactions: Journal of Engines* 104.3 (1995) (cit. on p. 3).

(SH89) D. Shulman and J.-Y. Herve. "Regularization of discontinuous flow fields". In: *Proceedings of the Workshop on Visual Motion*. 1989, pp. 81–86. DOI: 10.1109/WVM.1989.47097 (cit. on p. 17).

(Wei15) Joachim Weickert. *Image Processing and Computer Vision Lecture*. 2015. URL: http://www.mia.uni-saarland.de/Teaching/ipcv15.shtml (cit. on pp. 13, 23).

Additional Sources and Further Reading

(Ada14) Timo Florian Adam. "Object Tracking using Variational Optic Flow Methods". MA thesis. 66123 Saarbrücken, Germany: Universität des Saarlandes, 2014.

(Boo92) Rein van den Boomgaard. "The morphological equivalent of the Gauss convolution". In: *Nieuw Archief voor Wiskunde* 10.3 (1992), pp. 219–236.

(Dem14) Oliver Demetz. *Correspondence Problems in Computer Vision Lecture*. 2014. URL: `http://www.mia.uni-saarland.de/Teaching/COPCV14/copcv14.shtml`.

(GW06) Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing (3rd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006. ISBN: 013168728X.

(Mül14) Sabine Müller. "Image Processing Methods for Analysing Glioblastoma Multiforme in MRI Data". MA thesis. 66123 Saarbrücken, Germany: Universität des Saarlandes, 2014.

(OP03) Stanley Osher and Nikos Paragios. *Geometric Level Set Methods in Imaging, Vision and Graphics*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2003. ISBN: 0387954880.

(SAS11) Delphi France SAS. *Principles of Operation Common Rail*. DDGX200B(EN). Delphi Aftermarket, 2011.

(Sch13) Christian Schmaltz. *Advanced Image Analysis Lecture*. 2013. URL: `http://www.mia.uni-saarland.de/Teaching/aia13.shtml`.

LaTeX-typesetting based on stylesheets created by Oliver Barth, 2013