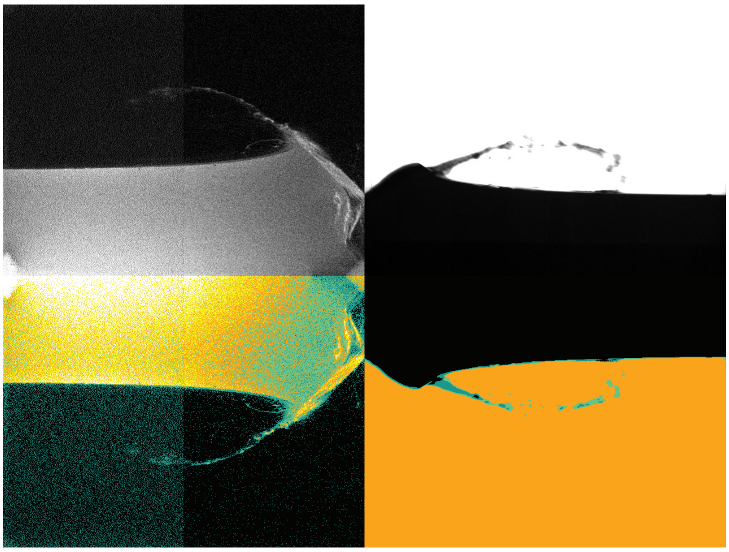
Capturing 2D Liquid Phenomena



Serkan Bozyigit

Master Thesis
August 2009

Supervisors:
Thomas Oskam
Dr. Nils Thuerey
Prof. Dr. Markus Gross

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Computer Graphics Laboratory ETH Zurich

Abstract

This thesis presents approaches and actions taken to enable capturing of two-dimensional water phenomena. In order to cope with the three-dimensional nature of water the construction of a specifically designed tank is shown. Additionally to this tank, a modifiable wedge-shaped device was built to create interesting splashes upon impact with the liquid. Reducing the three-dimensionality is not only based on the apparatus, but also on subsequent algorithmic steps. After eliminating illumination and recording difficulties that come along with a high-speed camera, the k-means segmentation algorithm, known from computer vision, is utilized to finally reduce the three-dimensional water appearance to two dimensions. In a final step, a graph data structure is introduced which is paired with a user-controllable tool in order to allow higher semantics to be added to the data.

Contents

List of Figures

List of Tables

1

Introduction

There has always been a fascination with water and its nature. It is remarkable that although, water is one of the most basic and essential things in life, it is also a very intricate matter. Thus, wanting to understand and trying to imitate and reproduce its behavior with the aid of computers seems reasonable since one's curiosity is being satisfied. Besides a purely curiosity-driven interest, there is also a demand from the movie and computer games industry for convincing animations of water. Finding new ways to capture its beauty suggests itself. But it turns out that the beauty as well as the devil is in the details.

There exist mathematical formulas that describe the behavior of water and are used for fluid simulations. The more details one demands to see from the simulation the more convincing it looks to us, but also the more numerically correct the according equations need to be solved. Solving them more precisely is coupled with computation, hence either one accepts the long waiting time or other approaches have to be found. One of these approaches are data-driven methods or, in our case, data-driven fluids simulations (DDFS).

In data-driven approaches the simulations are fed with real data to support and complement the underlying mathematical bases or animations rely entirely on them to generate meaningful output.

Specifically in this thesis, water phenomena are examined which are produced by an impacting wedge. Not only is the observation an important part of the thesis but the construction of a device that creates these phenomena. Furthermore, the apparatus shall also support modifiable parameters such that a wide range of impacts can be captured. This range can then be used to compile a database of different splashes where the distinct parameters span a multi-dimensional parameter space. Such parameters could consist of drop height, immersion angle, wedge length and wedge mass.

Due to the associated intricacy of capturing three-dimensional data the task is reduced to two dimensions. Nevertheless, the reduction does not make the situation trivial, on the contrary, water is inherently three-dimensional and therefore performing the reduction is quite challenging. The reduction itself is not achieved by a single step but rather divided into few smaller steps with each contributing to the final result.

The first step consists of building the above mentioned device preferably showing as little of the third dimension as possible. One part of this device, the tank, contains the fluid to be filmed. The other part, the wedge, produces interesting splash phenomena. In order to take pictures of a scene a high-speed camera needs to be used along with a suited illumination. The transparency of water and the associated issues to perceive its behavior ask for a solution, and was found by adding a colorant.

In a next step, the recorded gray-scale images are made completely two-dimensional by applying segmentation algorithms from computer vision. The last step presents an approach to give the pixel data a higher meaning by grouping them into nodes of a traversable graph. In order to create a graph that contains the extracted information of an impact event, different algorithms, such as *optical flow*, an automated and a manual tracking algorithm, are presented and discussed.

2

Related Work

2.1. Water Entry Analysis

The hardware construction part of our work is mainly motivated by the publication of Alam, Kai and Suzuki in [AKS07]. They examine two-dimensional numerical simulations of water splash phenomena focusing on the existence of surface tension. Based on two dimensionless numbers they formulate the numerical MPS method. To compare their numerical results with experimental data they present a water tank with a wedge that creates the desired splash phenomena. Partly, their setup motivated the current form of *our* setup.
Surprisingly, experiments with free-falling objects slamming on water surfaces have a long research history. Already in 1932, Wagner studied the slamming effects on liquid surfaces in [Wag32] and several others followed him and extended his theory. The major part of these publications are motivated by maritime research. Water entry analysis is important for the marine in order to build ships and vessels optimized for re-entry into the water. During turbulent weather vessels can be exposed to strong waves and their exerted forces. Rescue boats which are usually dropped from a high location over the water surface need to stand the water impact forces to be effective. Idealized, rescue boats can be represented through simple geometric shapes such as triangles or cylinders.

In [Gre87] Greenhow uses numerical methods to quantify the pressures and forces measured on the body of an impacting triangular wedge. For that he compares his numerical method with experimental data from Greenhow and Lin in [GL83]. In this setup cylinders and wedges with different masses and deadrise angles hit the surface of the water which is contained by a tank made out of Plexiglas. They use a tricky mechanism to illuminate the scene for only one image per experiment in an otherwise dark room. The images are captured with an (old-fashioned) ordinary 35mm camera.

The work of Zhao and Faltinsen [ZF06] extends Greenhow's theoretical work by generalizing to objects with arbitrary cross-sections. Also Judge, Troesch and Perlin use numerical methods to analyze wedges at vertical and oblique angles in [JTP04]. Additionally, they perform experiments using various elements, such as the same dyeing substance as in part of our work (fluoresceine) in combination with a laser, that illuminates the wedge and the dye-laden liquid. Hollow glass spheres are used to visualize the flow of the liquid that surrounds the wedge during the impact. Colicchio et al.'s approach [CGML09] is similar to Judge et al.'s, by using fluoresceine, a laser and tiny ceramic spherical particles to examine the effects of cylindrical objects entering and exiting the water. A similar setup is chosen by Kang et al. in [KOK$^+$08]. However, their focus lies on the study of impacts on shallow water whereat the impacting object is flat, i.e. a deadrise angle of *0 degrees*.
As opposed to the rather handy-sized setups until now, Yettou et al. [YDC06] present a massive setup where they use a V-shaped wedge with a base area of *1.2 m* × *1.2 m* and variable weights of approximately *20 kg*, *40 kg* and *60 kg*. Instead of using a water tank they conduct the experiments in a water channel with the following dimensions: *2 m* × *2 m* × *30 m*. In addition, their wedge is equipped with pressure sensors distributed over the whole body surface in order to measure the forces acting on the object on impact.
Kleefsman et al. also experiment with impacting cylinders and wedges to verify their numerical volume-of-fluid (VOF) method in [KFVI04]. Their VOF method predicts the free surface development on object impact.
In contrast to the previously mentioned publications, Mei, Liu, and Yue [MLY99] also consider impact problems of general two-dimensional objects, but present an analytical solution that generalizes Wagner's method, instead of a numerical solution, like Greenhow et al. Due to Mei et al.'s publication's theoretical nature it is of rather low practical relevance for our work, however it was mentioned for completeness.

2.2. Segmentation and Tracking

The simple, yet quite effective region growing algorithm is first presented in [AB94] by Adams and Bischof. They propose two ways to employ the algorithm. Namely, by using a selected or a random pixel in the image to grow regions based on similarity measures. The work of Li et al. [LSLD03] extend Adam and Bischof's original algorithm and combine it with an online learning algorithm. Their aim is to aid to classify colored images in content-based retrieval systems.

MacQueen [M$^+$66] introduces an algorithm that groups iteratively n-dimensional data into k groups. The basic algorithm is widely know as k-means clustering algorithm. The randomness built into this algorithm may not find the global optimum of the solution space. This fact prompted Likas et al. to investigate on this matter and to present the *global k-means* and the *fast global k-means* algorithm in [LVJV03]. Both algorithms basically search through the solution space with alternating configurations, whereas the *fast* variant uses heuristics to reduce the computing time.

The work of Boykov, Veksler and Zabih [BVZ01] uses graph cuts to approximately minimize energy functions in order to restore images and aid in stereo and motion images. They use customizable data and smoothing functions with the restriction only to use pixel pairs resulting in a fast segmentation algorithm. Kolmogorov et al. [KZ04] and Boykov et al. [BK04] base their work on the previous publication and elaborate on the matter.

In our work some elements of the image are contours of an evolving water surface. Some approaches based on contour tracking such as Isard and Blake's [IB98] the probabilistic CONDENSATION or gradient vector flows in [XP97] might work well. The latter extends Kass et al.'s *active contours or snakes* presented in [KWT88]. The algorithm locates object contours by also using a sum of internal and external energies which are to minimize similar to the graph cut approach.

Optical flow algorithms such as the one of Lucas and Kanade [LK81] or Horn and Schunck in [HS81] calculate the relative movement in a sequence of images. However, this type of algorithms are not well suited for our work due to assumption violation as will be explained in section 7.1.1.

2.3. Other Related Work

Our work focuses on the capturing of two-dimensional liquid phenomena. There is other work that is concerned with the three-dimensional nature of water. Ihrke, Goldluecke and Magnor's approach [IGM05] models three-dimensional water bodies. They dye the water with a fluorescent substance and measure the emissions when the substance is saturated by a UV light source. The emission along specified rays are used to calculate the three-dimensional representation of the water body.

Similarly, the work of Wang et al. [WLZ$^+$09] also uses a colorant in combination with a LCD projector to model real water scenes. Based on image reconstruction and physically-based models they are able to produce convincing water animations.

3

Hardware Construction

The necessity of building a custom tank evolved out of the reason that common fish tanks usually do not have the adequate sizing to properly capture two-dimensional liquid phenomena. The deeper the tank is (in z-direction) the more three-dimensional liquid features there will be. The three-dimensional nature of water can, of course, not simply be reduced to two dimensions. There will always be "artifacts" in the captured images, but by narrowing the tank these artifacts can be reduced and hence minimize the post-processing on the images.
Nonetheless, some experiments were conducted with a small sized fish tank which is shown in Figure 3.1.

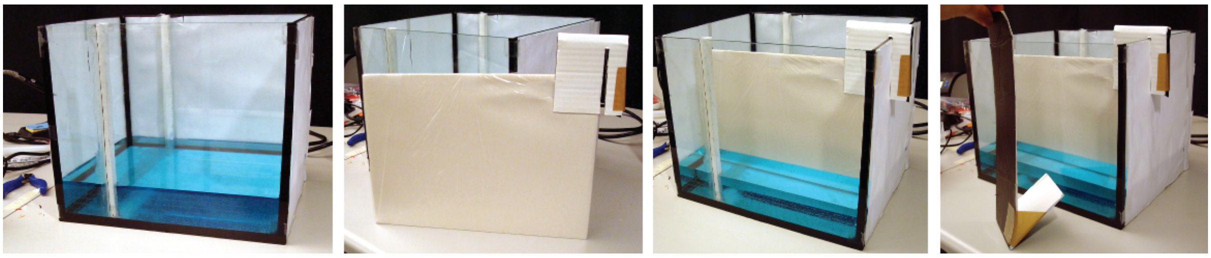


Figure 3.1.: *These images show the provisional tank with which the first experiments were conducted. The first image from the left shows the tank alone. In the second and third picture one can see the board that was inserted to create a narrow tank. The last image shows the wedge made out of card board used for the first splashes.*

Since commercially available fish tanks do not serve the purpose there is the option of letting a craftsman build a custom-sized tank. As in every branch of craftsmanship special constructions are expensive. Neglecting the high pricing, the finished product would still need some adaptations and extensions from the user's side, since the tank itself is just the container of the fluid and is still missing the apparatus which produces the impacts on the water. Therefore, there is no way around constructing some self-made parts. With this in mind, one of the goals of this thesis is to build such a tank with a working machinery that produces interesting liquid phenomena. In the following subsections, all self-constructed parts of the setup are described and hints will be given in order to reproduce the tank.

3.1. Liquid Tank

3.1.1. Description

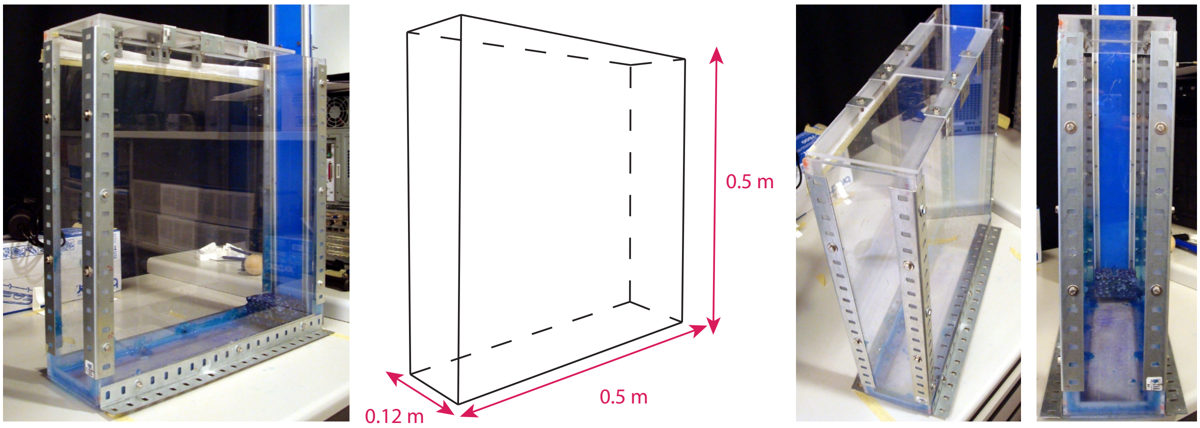
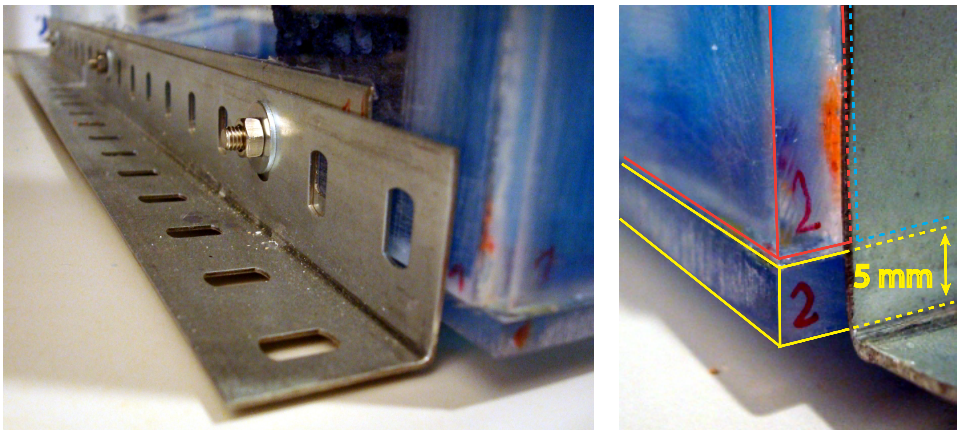


Figure 3.2.: *These pictures show the tank from different perspectives and its dimensions.*



a) b)

Figure 3.3.: *The left picture shows the L-shaped steel bars that were added outwardly to stabilize the tank. The right picture shows how the vertical acrylic boards rest* ***on*** *the horizontal board to reduce the stress on the glued seams. Distinct colors visualize the positions of those boards.*

The constructed tank is, compared to a common fish tank, rather narrow in depth with respect to its height and width[1]. The walls and the bottom consist of *5 mm* thick transparent acrylic

[1] *50 cm* × *50 cm* × *12 cm*, in our setup

glass (Plexiglas). To counteract the emerging splashes which would leave the tank during an experiment and possibly harm electrical devices around it, there is also a removable acrylic glass lid.
To fortify the tank structure against the water pressure, some edges are reinforced with L-shaped steel bars. These bars contain holes through which the acrylic boards are connected with stainless M5 screws and screw nuts.
The inner edges and corners are sealed with a "cold welding" glue on top of which bath silicone is applied. The screws, as well, are treated with both sealing products on the inside of the tank to avoid leaking through the screw holes.
There is a high chance of tipping the tank over because of the rather slim and high form. Due to this risk the bottom's long edges were outwardly reinforced with two L-shaped steel bars, which also might come in handy in case the tank needs to be mounted to the ground. The Figures 3.2, 3.3 and 3.5 show several aspects of the setup.

3.1.2. Compilation

The goal of this subsection is to allow the reader to rebuild the above described tank. Therefore, it will be rather instructive.

Building this device is based on many things: intuition, craftsmanship, luck and also on Internet forums. The latter one is especially interesting. Although it gives answers to many practical questions it is also afflicted with a high insecurity about statements, for example which adhesives to use. Most of the people who are participating and posting in such forums are adding their experience they gained with the subject. However, their claims are usually not verified or even guaranteed to work. Luck comes into play, when the right hint needs to be chosen out of many available, whereof the chosen one needs to work.

Before one can start with the compilation of the parts, they need to be treated in a particular way in order to achieve the best possible result. Usually, the acrylic boards and the L-shaped steel bars are not manufactured in the desired sizes and therefore need to be cut into the appropriate dimension, whereas the measures should be determined in such a way that the walls come to lie on the bottom part. This makes sure that the weight of the structure will lie on acrylic material instead of on the glue which would be holding the acrylic glasses (cf. Figure 3.3b).
One should also make sure that the cut edges are treated with a file afterwards since unfiled edges might lead to cut wounds. If possible, let the supplier of the acrylic glass cut the material since it requires a special saw. Cutting it on your own is also possible with an ordinary metal saw, but the cut itself will not be as smooth as it is with a special saw[2]. A very annoying property of acrylic glass is that it gets scratched very easily. Therefore, in order to not be forced to re-buy the materials, being careful with hard and sharp tools is a valuable piece of advice.

Material Preparation

1. **Marking:** Mark the spots along the edges of the acrylic boards for drilling the screw holes. A few are enough, since too many weaken the structure (see subsection 3.1.3).

[2] A saw especially made for acrylic glass.

About *18 cm* of distance from screw to screw worked well for this tank. Be aware that short edges need not be reinforced with steel bars since the edge is stronger without the drilling on this short length. Screws at corners should not be too close to each other or if possible do not place any of them close to a corner, since the assembly at these points are cumbersome due to narrow conditions. An example is depicted in Figure 3.6c.

2. **Drilling:** Drill at the spots with the according drill head. Acrylic glass is a sort of plastics which means that fast drilling will melt the acrylic glass due to friction. The melting can be avoided by drilling just a few millimeters at a time and resting for half a second. By doing so one gives the drilling spot time to cool down.
3. **Sanding:** Each acrylic board needs to be sanded along the edges that will be glued. Sanding is necessary because the glue and the silicone adhere much better on coarse surfaces than on flat or even polished ones, as it is the case with acrylic glass.
4. **Cleaning:** Clean the sanded areas as good as possible, but do not use any glass cleaner or acetone, which are or contain chemicals that can affect the acrylic glass' surface. This hint is also important when trying to remove limescale or color spots after experiments. Because these chemicals are superb cleaners for common glass, it might mislead one to also use them on acrylic glasses. Instead, use clean water or water with dish washing detergent.

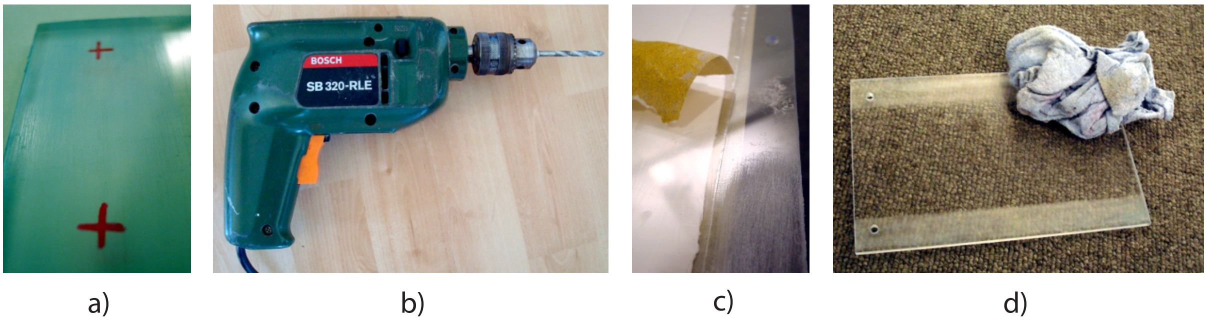


Figure 3.4.: *Material preparation: a) Mark the glasses with an appropriate marker. b) Drill them carefully. c) Sand the surfaces along the edges that will be glued together. d) Remove the sanding dust.*

Assembly

1. **Screw together:** After making sure that the parts fit to each other, i.e. the dimensions are correct, screw the parts[3] together and tighten them.
2. **1st glueing:** Two sorts of glue and a bath silicone will be used in the next steps. The glues are supposed to be *cold-welding*. One of them has a viscosity comparable to water and the other one has a rather thick viscosity comparable to honey. In every step it is very important to work precisely and to pay attention not to leave any unfilled holes to avoid fluid leaking.

[3] All parts need to be screwed. The glue will be applied later, also on holes and screws.

First, take the low viscous glue[4] and apply it to all joining edges. If the acrylic boards were cut precisely the glue should literally be sucked in by the gaps between the edges due to the capillary force. Although, usually the glue is not perfectly suited for connecting metals and acrylic glasses it still has an adhesive effect on this combination. Therefore, the glue should also be applied to the screws inside the tank. By applying the glue to the screws one can also observe the capillary force sucking in the adhesive material. It is very important to detect any unfilled holes and fill them, which can be done in an additional control pass.

3. **2nd glueing:** After letting the first coating dry, which should not take longer than 10 minutes, apply the more viscous glue[5] on top of the first one in a similar fashion as in the previous step.

4. **1st drying:** Letting those two coatings dry (in a well aerated room) for at least a day is important since the glues contain solvents which need to volatilize. Otherwise, they might affect the silicone coating which will be applied in the next step.

5. **Applying silicone:** This step uses a commercially available bath silicone. There are different colored ones. Using a transparent one is a bit cumbersome in the application step because it is difficult to see where it has already been applied and where not, but the transparency property will come in handy during usage of the tank. It is important to note that a thicker layer of silicone does not improve the sealing. There should only be as much applied as needed. The thicker a layer is, the longer it takes to dry, which is usually up to five to seven days. As long as the layer is moist it will not properly fulfill its purpose of sealing the tank.

6. **2nd drying:** After letting the tank dry, do not tighten or untighten the screws anymore because this might lead to holes in the seal and cause leaking.

3.1.3. Additional Hints for a Successful Compilation

When purchasing a wardrobe from a Swedish furniture store, where its parts to assemble are already chosen, precut, neatly packed and come with a handy assembly instruction, the challenge there is the assembly itself. The first difficulty in this thesis was even more basic which one does not usually have to worry about, namely to determine the size, relations and needed materials.
Publications[6] with similar setups were investigated in order to get an idea about this topic. Most of these publications with similar tank constructions were not always very descriptive about the tank's dimensions and even less about the used materials. The majority of these publications were funded by governmental institutions to contribute to naval research. The governmental help is also the reason why their setups were usually bigger in size[7], more automated and more sophisticated. Sophisticated in the sense that the impacting devices included electrical pres-

[4] for example: UHU plast special

[5] for example: UHU allplast

[6] for example: Alam, Kai and Suzuki's work [AKS07] *"Two-dimensional numerical simulation of water splash phenomena with and without surface tension"*. For additional related work see section 2.1.

[7] In Yettou et al. [YDC06]'s case even huge!

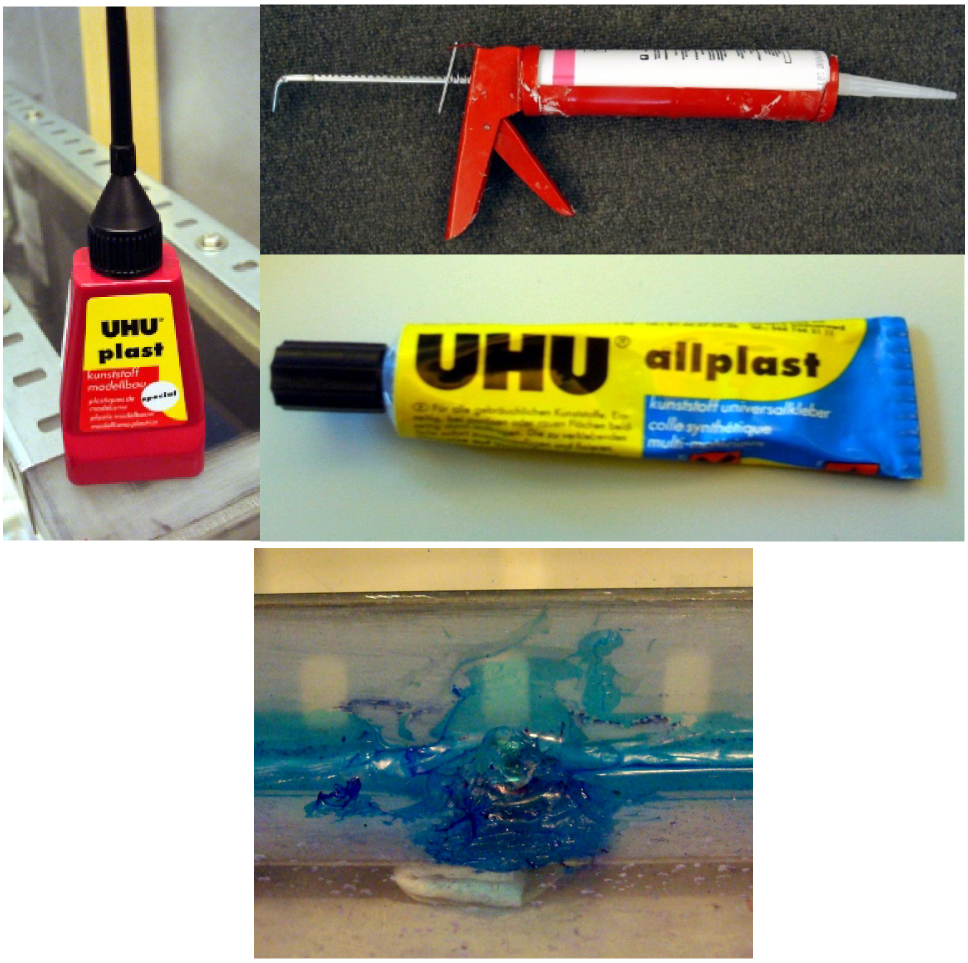


Figure 3.5.: *The upper images show the three sealing products which were used. The lower picture shows how a screw needs to be covered by silicone in order to seal it. Originally, the silicone was transparent but as a result of the utilized colorant to dye the water it took on its color.*

sure sensors on their surfaces to determine the forces of the impact acting on the object which originated in a completely different research objective as compared to this thesis.

Due to the sparse information found to answer the material question, it was assumed that commercially available materials in Do-It-Yourself stores should do the trick, i.e. giving sufficient stability. Walls were chosen to be made out of acrylic glass due to its excellent transparency, its relative stability compared to its weight, and its easiness of processing for drilling, sanding, or similar manual works. In order to reinforce the corners and edges of the boards a metallic supporting structure was searched which happened to be the L-shaped steel bars. To connect the bars with the boards screws and screw nuts were used. Since the tank will be filled with water (and dye) the screws needed to be stainless to prevent corrosion.

As the tanks in the related papers are ranging from medium table-sized up to building-sized constructions, a first intuitive guess of the volume considering available setup space and handiness was made. Choosing the tank's depth plays a major role with regard to the thesis' goal. The less deep the tank is chosen to be the less "three-dimensionality" will be observed during

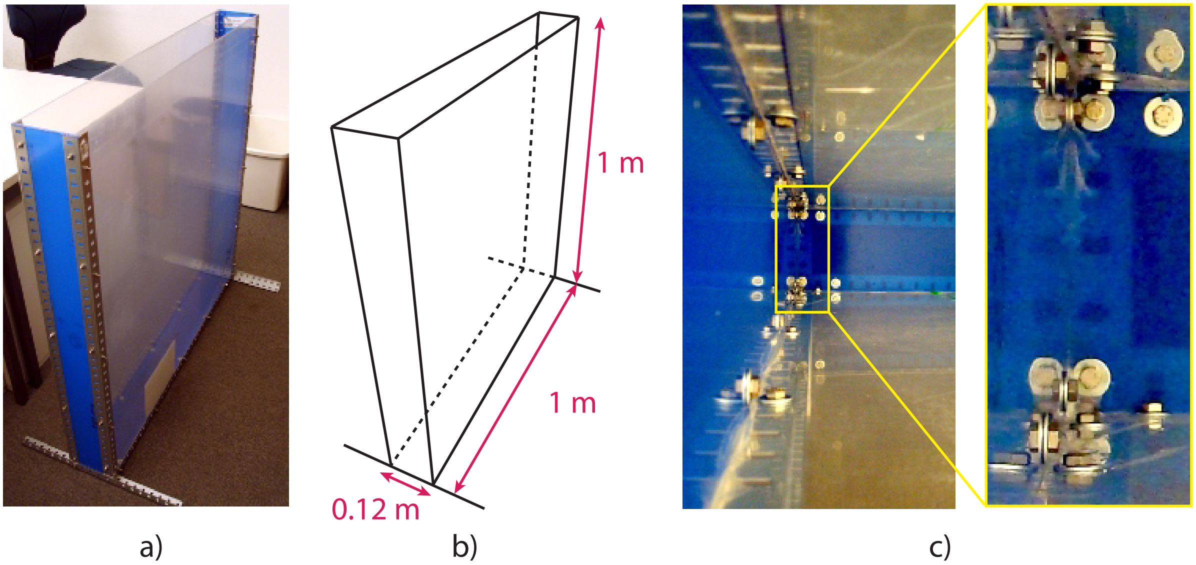


Figure 3.6.: *Images a) and b) show the original tank and its dimensions. Image c) illustrates the narrow conditions on the bottom of the original tank and also the difficulty to reach it properly due to the tank's height.*

the experiments.
Before building the tank that is used for the experiments and described in section 3.1.1, we built a bigger tank with four times the volume (*1 m* height × *1 m* width × *12 cm* depth) shown in Figures 3.6a and 3.6b. Now in retrospect, this first tank was a prototype on which many difficulties and issues crystallized. Two main difficulties were experienced during the construction of the bigger tank which later helped to build the tank that was used for the experiments:

- **Hindered screwing and sealing:** The choice of using screws and sealing substances is not evident, since also melting the corners together might work.
  However, it turns out that when one tries to melt acrylic glass it is difficult to control the melting area, since the material itself starts to burn. Besides the problem of confining the flame, the boards do not connect properly with each other and produce a carbonized coarse surface shown in Figure 3.7.
  Since melting the acrylic boards to seal it and to make the construction stable does not work, we decided to use screws and silicone. The following challenges were encountered:

  - **Screw tightening:** The tank dimensions of *1 m* height × *1 m* width × *12 cm* depth make it difficult to reach the bottom in order to tighten the screws and seal off the edged and corners.
    To be successful in tightening the screws it is important to find an order of assembling and tightening the screws. Unfortunately, there was no order of assembly that makes it easy to do, therefore at least along one edge of the tank it is very cumbersome to tighten the screws. Here, it was done in such a way that one wall was left to be closed as the last one and the tank was positioned horizontally with only the horizontal screws left to be tightened. The screws were pushed through the screw holes from the inside to the outside with a long metal stick and carefully tightened with a screw nut from the outside. The difficult part was to attach the screw nut on

  the screw without letting it drop inside, since it was not possible to hold the screw inside at the same time. If, despite the diligence, a screw dropped inside, every other screw on that side needed to be untightened and the dropped screw needed to be repositioned.

  - **Silicone application and gadgets:** To apply the bath silicone to the edges one needs to use a silicon pistol, also seen in Figure 3.5, which is very handy to use in open wide spaces, but rather difficult in a narrow setup as it is in the tank's case.
    For the silicone to be effective it needs to be smeared in a clean fashion so that there is no air gap between the acrylic glass and the silicone itself. This means that the silicone cannot just be put on the edges, it would not fill the edges due to its high viscosity, it needs to be pressed against the corner with a device to fill them completely. A self-constructed stick with a wooden circle, as seen in Figure 3.7b, was used to press the silicone as good as possible against the corners.
    The silicone takes, depending on the applied amount, at least a week to completely dry in a well-aerated room.

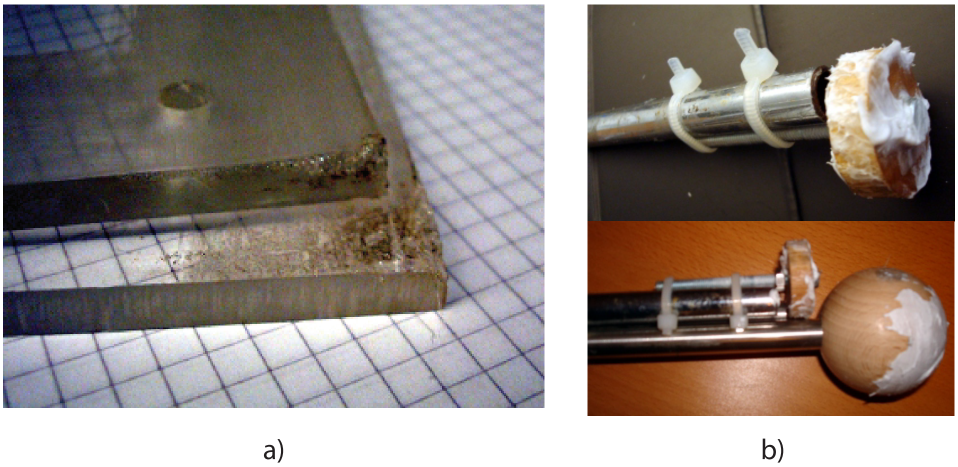


Figure 3.7.: *Picture a) shows the carbonized surface produced by the attempt to connect acrylic glass through melting. The pictures in b) show the tools used to apply the silicone in the first tank.*

- **Size and material mismatch:** After the tank was assembled and sealed with the proposed materials and dimensions it went through a stress test were it was filled with about *35 liters* of water. It turned out that the water pressure is easily underestimated and that the used materials were not able to hold these forces. Note the tank deformations in Figure 3.8a. At this point the water level was at about *35 liters* and the structure started to make cracking noises. After emptying the tank many cracks and even a broken corner of a wall were encountered as it can be seen in Figure 3.8b.

Lessons Learned:

- **Avoid corners overcrowded with screws:** Overcrowding the corners of the tank, as depicted in Figure 3.6c, only weakens the corners and complicates the building process of the tank. Just a few screws along the (long) edges are enough to stabilize the structure.

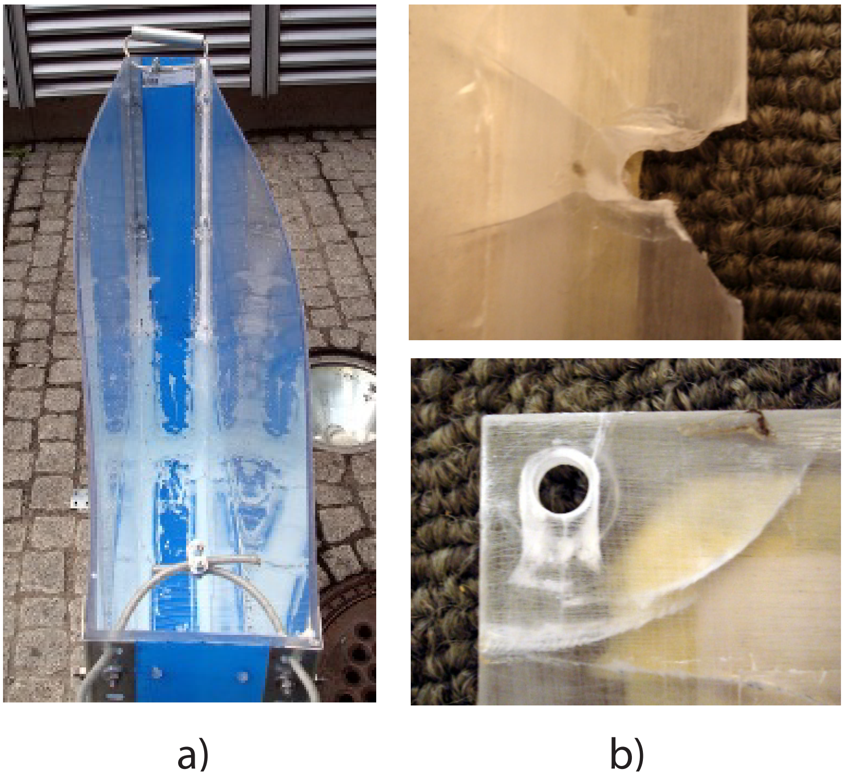


Figure 3.8.: *Image a) shows the deformations on the former tank due to water pressure. The images in b) show the discovered cracks and broken edges after disassembly.*

- **Small setups do not need many support elements:** Common fish tanks usually do not need any supporting structure along the edges and therefore using special adhesives would have sufficed to have a stable tank. Nevertheless, adding support structures helps with the glueing process and allows to add additional elements to the tank. The utilized L-shaped steel bars can be used to mount the tank onto a surface in order to fix it, as shown in Figure 3.3a.
- **Multiple layers of sealing:** Silicone is difficult to apply in narrow conditions such as the inside of a narrow tank. Utilizing different layers of sealing products improves the chances of successfully sealing the tank. An Internet-based investigation showed other possible candidates. There are glues that are supposedly "cold-welding" the acrylic glass. Cold-welding in this case means that the acrylic glasses are not only glued to each other but a much stronger *joining takes place without (melting) fusion at the interface*[8] of the substances. Obtaining the originally recommended adhesive[9] was not possible due to supply difficulties. Similar cold-welding products[10] were obtained in local hardware stores.
- **Respect the water pressure and adapt the dimensions:** The water stress test triggered an investigation in aquarium forums. It clarified that a stable tank with a comparable

[8] http://en.wikipedia.org/wiki/Cold_welding

[9] Acrifix®192

[10] UHU plast and UHU plast special

volume and height[11] would need at least *2 cm* thick acrylic glasses[12]. This fact would make the main walls as heavy as *24 kg*[13] each, which is impossible to manually treat with the provided tools and to assemble by a single person. Therefore, it was necessary to adapt the dimensions of the tank to *50 cm* × *50 cm* × *12 cm* to match the acrylic glass' thickness (*5 mm*) to the water pressure.
Additionally, adapting the dimensions of the tank to human-reachable measures is also advantageous, otherwise the sealing process becomes infeasible.

3.2. Wedge and Guide Rails

The water tank makes up an essential part of the whole setup, but it is still missing a device that is creating interesting water phenomena. The goal of producing *any* water phenomena (in this case: splashes) is simple by just throwing an object into the water. The difficult part is to be able to reproduce the phenomena one encounters or at least to get similar results. In order to do so the additional device to the tank needs to be controllable and adjustable. Controllable in the sense that the object which is colliding with the water surface should not fall in a random fashion. Additionally, the apparatus should be adjustable in order to create different splashes depending on angle of incidence and drop height.

3.2.1. Wedge

Inspired by the setups of naval research a wedge is constructed. But in contrast to them the wedge itself is not of central interest but rather the effects it creates. A wedge is usually a symmetrical, V-shaped structure. In this form it would also create a symmetrical splash which is double the number of desired splashes. Since also the available tank space (especially the width) is limited, it is useful to only capture one side of the wedge. In this manner it is not only possible to create bigger splashes but also to see more details. The wedge itself does not need to be fully in the picture and therefore there is more space for the splash itself.
For this purpose a "half" wedge compared to the usual form is used which can be seen in Figure 3.9. Besides having a wedge form it is desired to have a variable angle for the point of incidence in order to produce splashes which have different length, height and structure. The wedge foot, which is the wedge's movable part, is constructed in such a way that it can be exchanged with different sized wedge feet which also cause splashes with different characteristics. The utilized hinge, seen in Figure 3.10c, is the joint between the wedge foot and the wedge back. It uses a pin, seen in the center of Figure 3.10c, connecting one part of the hinge attached to wedge foot (left part in Figure 3.10c) and the other part which is connected to the wedge back (right part in Figure 3.10c). The pin is removable which makes it possible and easy to replace the foot with another different sized one, depicted in Figure 3.10d. Two of the wedge feet, shown in 3.10d have pieces of black rubber attached to them. These help to bridge the gap between the wedge and the acrylic board in order to only let as little water as possible through the gap.

[11] *1 m* height × *1 m* width × *12 cm* depth

[12] according to http://www.regalplastics.net/aquarium.htm

[13] $23800g = 100cm \times 100cm \times 2cm \times 1.19\frac{g}{cm^3}$, where $1.19\frac{g}{cm^3}$ is the density of acrylic glass

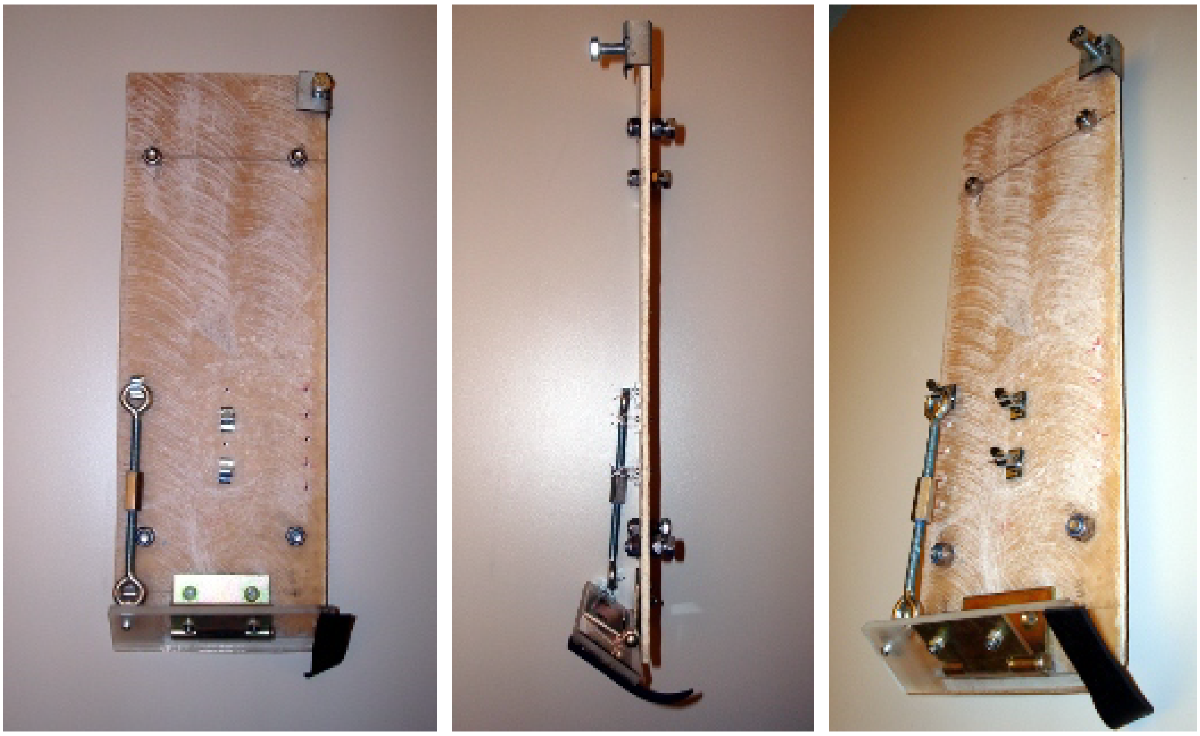


Figure 3.9.: *The images above show different perspectives of the adjustable wedge that produces the splashes.*

The wedge's back is elongated and has several clamps attached on one side (cf Figure 3.9 and 3.10a). These clamps and specially shaped pins have the purpose to fix the wedge foot's position relative to the wedge back in order not to get displaced on impact. The specially shaped pins consist of three parts, shown in Figure 3.10b. On both ends there are screw-like pins which are bent to the form of an "O" and are connected with an elongated screw nut. The screw nut makes it possible to adjust the length of the connected pins. Figure 3.10a shows a clamp holding the **O**-formed pin, Figure 3.10b depicts a partially disassembled pin.
On the wedge back's flip side there are four screws, seen in the second image of Figure 3.9. They are adjusted in such a way that they are able to move a little amount within the wedge-back-plane but not in the direction perpendicular to that plane. These screws are used to attach the wedge to the guide rails which are going to be mentioned in the next section.

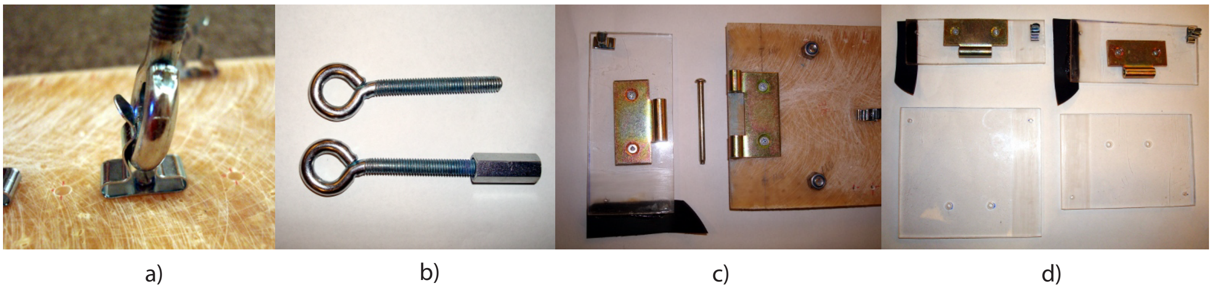


Figure 3.10.: *In a), a clamp is holding a specially formed pin which is also seen in b), in order to stabilize the wedge foot. c) and d) show different sized wedge feet which can be exchanged by removing the pin at the hinge.*

The choice of having a variability in the construction also brings disadvantages. The wedge is not a wedge in the classical sense as it is not made of one single piece. The little cavities and gaps give rise to irregularities where, for example, the water can pass through during an experiment. These irregularities can introduce errors during subsequent segmentation steps where water (, wedge) and air need to be separated.

An important part of the wedge is a gadget that allows to release the wedge for an impact. For that purpose the two small constructions seen in 3.11a and 3.11b were designed. The gadget in Figure 3.11a is designed in such a way that it can be fastened anywhere on the side to the wedge back. The counter part to the first gadget is the piece in Figure 3.11b. It is a fixture made out of three glued acrylic glass pieces containing a hole through which a metal pin passes. By attaching it to the acrylic board on top of it, as shown in 3.11c, it can hold the screw from the first gadget and once the user is ready to release the experiment, simply the metal pin has to be drawn. Since the gadget with the screw is detachable it can be positioned at any feasible point on the wedge enabling different drop height of the wedge.

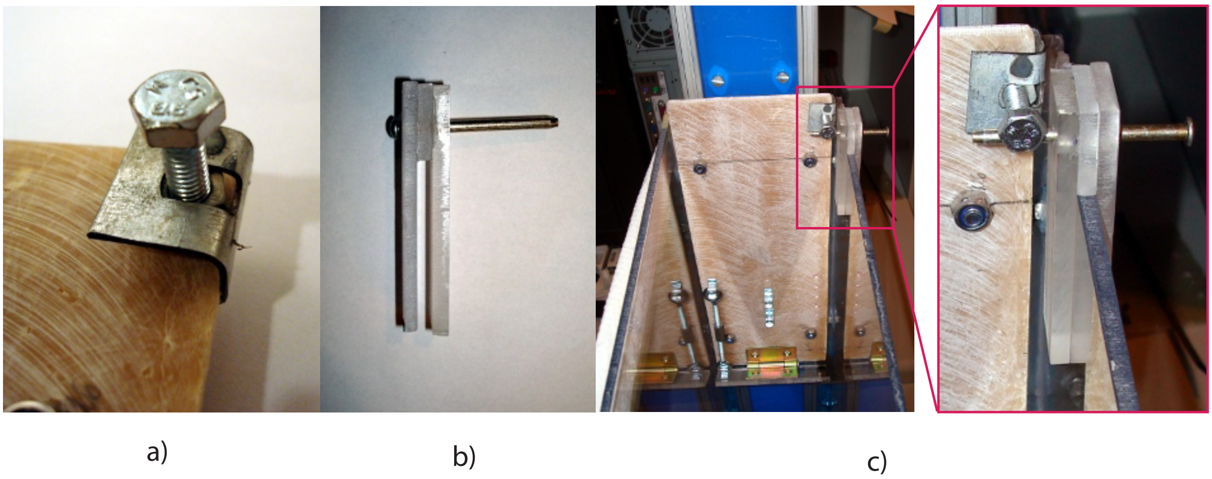


Figure 3.11.: These pictures show the two-piece gadget that enables the release of the wedge during the experiments.

3.2.2. Guide Rails

The guide rails, shown in 3.12a-c, are meant to control the wedge's trajectory during (almost) free fall from a chosen height. They comprise an oblong piece of acrylic glass, drapes, a sponge and a gadget which fixes the rails to the short side of the tank that can be seen in 3.12d. The drapes are attached to the acrylic glass with tiny screws and serve as guidance for the wedge. The four screws attached to the back of the wedge, seen in the second image of Figure 3.9, are meant to take course through the drapes. In order not to let the wedge take damage once it completes its course along the guide rails there is a sponge attached to the end of it. Surprisingly, the guide rails do not need to be much higher than the tank itself since already a few centimeters (about *10 cm*) are enough to create interesting splashes. In fact, letting the wedge drop from a very high position creates very fast and strong splashes which are too fast for the *200 fps* camera and therefore are blurred. Also, the splash's velocity is coupled with the distance that the splash particles travel and hence their trajectories usually do not end within the tank.

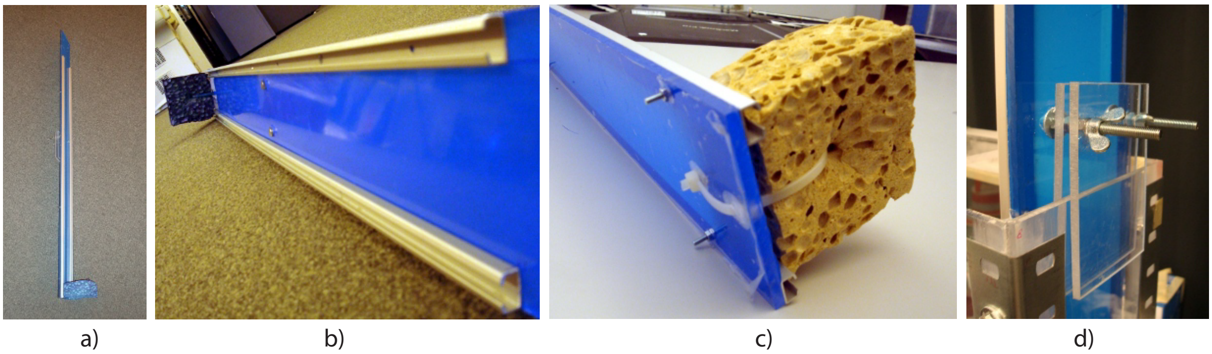


Figure 3.12.: *The images above show the guide rails attached to the oblong shaped acrylic glass in a) and b), a sponge to stop the impacting wedge in c) and the gadget that fixes the guide rails to the tank in d).*

4

Camera and Lighting

The previous chapter described a self-made setup which consists of a tank that contains the liquid to be observed and a wedge that is creating the desired splashes. In order to profit from this setup the liquid needs to be captured in a useful way. This chapter will focus on the extension of this setup by a camera device and various illumination devices to improve the digitalization of the observed scene. Besides the setup itself, encountered advantages and disadvantages with the utilized camera and lighting will be shown to account for the final form of the setup.

4.1. Capturing Device

When observing a water drop falling into a puddle when it is raining outdoors, one notices quickly that the seen splashes are very fast moving. A human eye observes its environment with a limited rate of about 24 impressions per second[1], which seems to be quite a lot, but considering the water drop example it is obviously not enough to seize all details. Common video cameras are matched to the users visual needs. In order to perceive the recorded data as a fluent movement the recording rate should be at least the same or higher than the human eye's. To analyze the effects of an impacting object on a water surface it is desirable to have a camera with a higher rate. The higher the rate, the less differences will be noticeable between two subsequent images which, in turn, improves the temporal coherence of the result and simplifies the tracking of interesting objects.

Generally, we are used to visually perceive the world in color, which is crucial for many everyday matters, but in digital processing color, compared to "intensity only", means more data to handle. For the experiments described in this thesis color was not a critical factor which lead

[1] http://www.grand-illusions.com/articles/persistence_of_vision/

to the reason a black-and-white camera[2] was used to record the desired phenomena. With a black-and-white camera, captured images are represented as two-dimensional 8 bit[3] intensity maps, where the absence of light results in an intensity value of zero and a fully illuminated sensor issues an intensity value of 255.
A color camera maps the infinite range of colors of the observable environment[4] on three color components *R (red), G (green), B (blue)*. Each of these components are fed by sensors sensitive around the according wavelength which usually also issue intensity values between 0 and 255, similar to the black-and-white sensors. Composed, these three components yield any color displayable by a common monitor. Thus, the usage of a black-and-white instead of a color camera saves us $\frac{2}{3}$ of bandwidth, which is important once the recording speed is considered.

Furthermore, the provided software that comes with the camera allows to use a few different modes in which can be shot. Each recording mode has a specific resolution and a recording speed. The movements of a splash are admittedly fast, hence only fast recording rates are interesting. For that purpose, the two most important and adequate modes are *Mode 0*[5] and *Mode 1*[6]. Besides the frame rate and the resolution there are other attributes which the user is able to alter. Directly on the camera one can modify:

- The *zoom* controls how the camera frames the object.
- The *aperture* controls how much light is allowed to pass through to the photo-sensitive sensors.
- The *sharpness* controls the focus of the perceived picture.

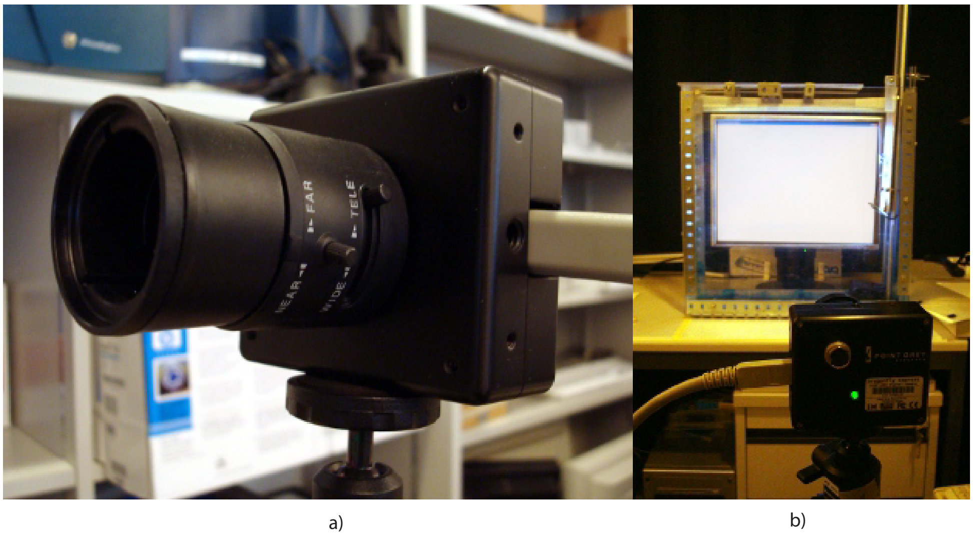


Figure 4.1.: *Image a) shows the utilized Dragonfly Express and image b) depicts the setup including camera, tank, wedge and flat screen.*

The most obvious way to position the camera is perpendicular to the front pane of the tank,

[2] Dragonfly Express, 200-350 fps, Black & White

[3] Ranging from 0 to 255, because $2^8 = 256$

[4] Color depends on the wavelength of light. This wavelength can have an arbitrary positive value and therefore results in an infinite range.

[5] Mode 0 supports 200 fps with an optical resolution of 640×480 pixels

[6] Mode 1 supports 350 fps on an optical resolution of 320×240 pixels

aligned with its center. A picture of the setup can be seen in Figure 4.1. Additionally, the camera parameters on the camera are to be adjusted in such a way that the water looks as two-dimensional as possible. Maximizing the zoom handle towards the configuration *"tele"*[7] forces the *depth-of-field*[8] to be as small as possible. A small depth-of-field has the effect that the perspectivity becomes as small as possible leading to an image with "less three-dimensionality".
More interesting is the question at which height to place the camera. If the liquid level is not on the same height as the tank's center, the camera's height should be adjusted as well. The central placement allows to capture the whole range of the tank.
Whilst the camera's lense is pointing towards the liquid surface, it appears perfectly two-dimensional as the liquid surface is coplanar with the camera's focus. However, once the wedge enters the liquid its level is going to rise due to the additional mass that enters the volume and the water gains back its three-dimensionality. Because the camera cannot focus on more than one spot at a time its height needs to be adjusted to have most of the movement around the action's center where the three-dimensional "artifacts" appear the fewest (cf. Figure 4.2). For the conducted experiments a height of *1 cm* above the still-water-level yielded the best results.
Also, the camera should be able to capture as much as possible of the events within the tank, by placing and adjusting the camera accordingly. Whether the wedge needs to be partly or entirely in the picture is up to the user's decision.

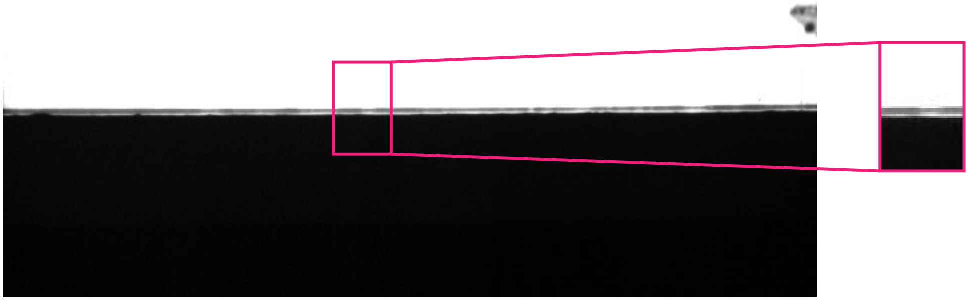


Figure 4.2.: *This picture shows the three-dimensional "artifacts" when resting water is recorded. The tip of the wedge foot can be seen in the right upper corner.*

4.2. Illumination

The goal of the lighting in this context is to provide a high contrast relative to the liquid that is being filmed. An ideally illuminated setup would have a uniform, bright and diffuse background whereas the liquid is completely dark and therefore easily distinguishable from the background. This way the segmentation step would take place without any hassle.

In search of the closest lighting to the ideal one a couple of illuminations were tested. A distinction has to be made for two types of experiments. Both types depend on lighting and on the colorant used to dye the liquid. The colorants are discussed in section 5.1:

[7] *Greek*, translates to "far", i.e. the camera focuses on objects in the distance

[8] The portion of a scene that appears sharp in the image.

- The first type utilizes the classic bright background illumination with a dark liquid generating a high contrast.
- The other type uses a fluorescent substance dissolved in the liquid. The fluorescent substance needs an ultra-violet (UV) light source that excites it to make it emit visible light. Therefore, the liquid itself provides the light in an otherwise dark room also creating a high contrast in that combination.

In the end, we decided to use an ordinary flat screen with an appropriate size for the experiments with dark water. The experiments with the fluorescent substance achieve the best results with ultra-violet neon tubes.

Often illumination is desired to be diffuse. Light, that comes from a light source directly into the camera causes the camera to adapt its aperture, otherwise objects in the picture will be over-shone. However, to have a *background* lighting the light source needs to be positioned directly in front of the camera. In order to be able to conduct useful experiments sheets of paper are clamped to the backside of the tank to yield a diffuse background. Hence, for all types of background illumination except the flat screen thin sheets of papers are used.
The following section describes all used light sources (see Figure 4.3 and Figure 4.4) and the issues which were discovered by using them. Methods used to try to rectify these issues follow right after in section 4.4:

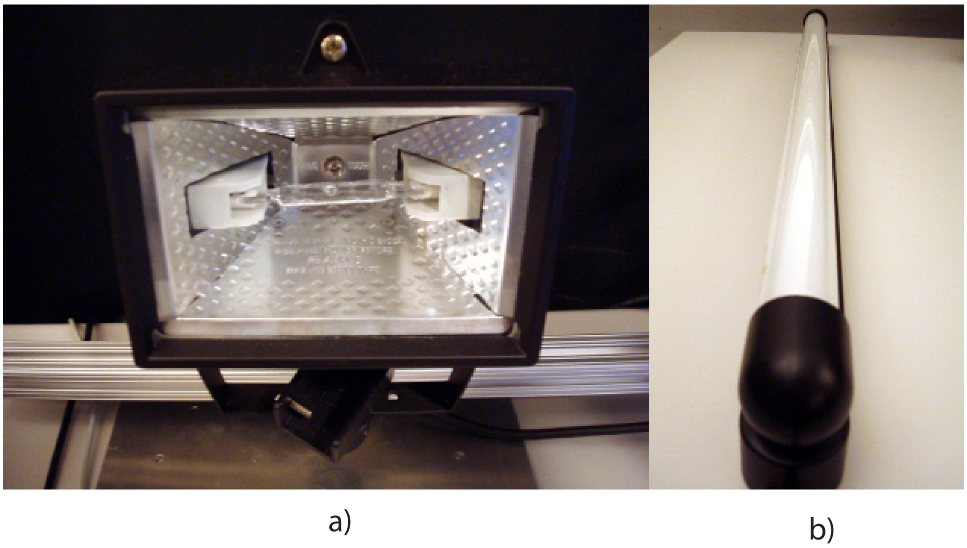


Figure 4.3.: *These images show the tested halogen bulb and the neon tube.*

- **Ambient light:** Before applying any dedicated lighting to the setup, simple ambient light was tested in a closed room. There are two difficulties with this approach:
  First, with high frame rates there is usually not enough light for every single frame to have a sharp image without dedicated lighting.
  Secondly, no light from any particular direction, i.e. from the liquids back side, is prevailing which means that all kind of reflections are visible on the glasses. Even the camera itself might present itself in the resulting picture. Hence, it is not possible to capture an obstacle-free image.
- **Daylight:** The natural extrapolation of the ambient light idea is to record at daylight. On

a sunny day there is more than enough light to record at 200 fps. Still, there are a couple of reasons why working at daylight is not ideal:
The first reason is similar to the one in ambient light, there is no light prevailing from any particular direction, if not recorded at direct sunlight. All kind of reflections are visible on the glasses. Even worse, since sunlight is much stronger there might even be all sorts of optical effects as shadows and caustics which make the segmentation in a subsequent step a lot more difficult.
An other reason is a more practical one: Carrying the setup with the corresponding computer for every measurement outdoors is cumbersome and once outside, obviously leaving the setup there is not an option either due to unstable weather conditions or possible theft.

- **Table lamp:** The first setup with a dedicated lighting constitutes a simple table lamp positioned behind the paper-covered acrylic glass. The lamp emits light in a (half) spherical manner, enough to prevail over ambient light. However, this setup unfolds various basic issues which are also encountered with other illumination techniques. These issues and their cause are explained in detail in section 4.3:
  - **Non-uniformity:** The non-uniformity of the background intensity in space.
  - **Non-uniform emitting frequency:** The non-uniformity of the background intensity over time.
  - **Paper structures:** Fine structures of the paper introduce shadows upon illumination. These shadows do not belong to the liquid.
  - **Dark background vs bright liquid:** When dark background areas are mistaken for bright liquid areas.
- **Halogen lamp:** The halogen bulb, depicted in Figure 4.3a, emits a much stronger light compared to the table lamp. Its bulb case's ratio roughly matched the tanks ratio enabling a more controlled distribution of light to the backside of the tank as compared to the table lamp. The bulb inside the case is oblong, tube-like and measures about *5 cm*. The case's inner surface is coated with a reflective material used to redirect backward going light improving the efficiency in terms of light yield. Additionally, this illumination has a high energy consumption with *150 W* and therefore also heats up a lot. Basically, the same issues arise with this illumination techniques as it is the case with the table lamp, except that they are more pronounced due to higher intensities.
- **Neon tube:** Neon tubes, shown in Figure 4.3b, usually have a oblong shape, therefore they emit light in a cylindrical manner which is different to the way the before mentioned light bulbs emit their light. Cleverly arranged[9], neon tubes are able to illuminate the scene more uniformly than ordinary bulbs. However, commercially available neon tubes with *18 W* are not as powerful as, for example, the halogen bulb. Lower intensities make the background of the liquid darker and lower the quality of the images due to introduced noise.
- **Flat screen:** A flat screen[10] provides an almost perfect uniform lighting on its screen. The screen pixels are too small to be recognized by the camera and therefore the background

[9] Parallel to the acrylic rear glass.

[10] NEC MultiSync LCD 2070NX

of the scene looks like a perfectly white area. The screen is positioned as close as possible to the rear glass without the sheets of paper covering the glass. The top of the screen is covered with bibulous material such as toilet paper to protect it from potential splash outliers. For all further experiments with dark liquids, except for comparison reasons, the screen is used for providing illuminating to the scene.

- **UV tube:** UV neon tubes basically show the same properties as the common neon tubes except that their emitted range in the light spectrum differs. Interestingly, the Dragonfly Express seems to perceive more of the spectrum of the UV tube than humans since the tube is recognizable as a bright bar in an image, whereas humans only perceive the UV tube as a dark glow, cf. Figure 4.4a and 4.4b).
- **UV bulb:** UV light bulbs, seen in Figure 4.4b and 4.4c, also show similar properties as the common light bulbs. Especially, the frequency issue is noticeable as well. With *75 W*, the light bulb consumes about four times as much energy as a UV tube. However, apparently most of the energy is transformed into heat instead of UV light making the UV bulb less efficient than the UV tube.

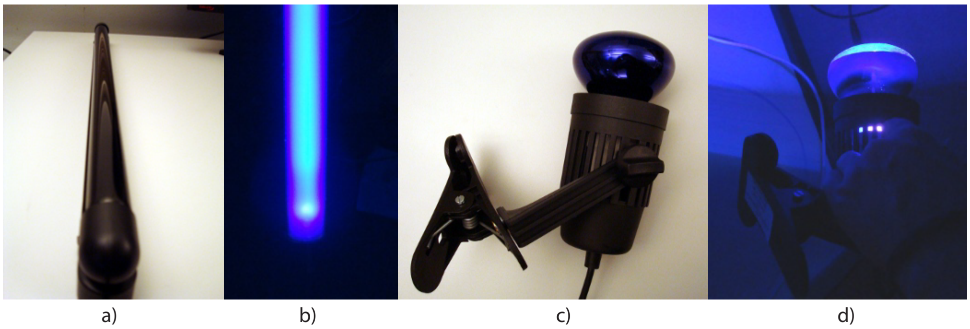


Figure 4.4.: *Images a) and b) show the UV tube, images c) and d) the UV light bulb, turned off and on, respectively.*

4.3. Challenges

There are a couple of issues which were mentioned earlier in section 4.2. These issues arise when a high-speed camera is used in combination with ordinary illumination to record image sequences. Additionally, further issues are listed as well, which were encountered during the course of the experiments:

- **Non-uniformity:** The main issue with common light bulbs as well as with halogen light bulbs in our work is the non-uniform illumination of the background. Ideally, all parts of an empty background, such as the corners and the center, would be illuminated with the same perceived intensity. However, these light sources emit their light spherically and this property is mapped to the background. There is literally a white spot in the center of the image (see Figure 4.5a and Figure 4.5b) and the farther away a point is located from the center the darker it becomes.
  Additionally, the halogen bulb emits a very strong light that is able to pass partly through the very darkly dyed water (cf. section 5.1 for explanation on colorants). The fact that the

light is able to pass through the dark liquid makes the camera record the water not as homogeneous area but an area with changing intensities. Moreover, the bright water regions now have about the same intensity as dark background regions due to this non-uniformity. Consequently, segmentation algorithms show difficulties in handling this problem as can be seen in Figure 4.5.

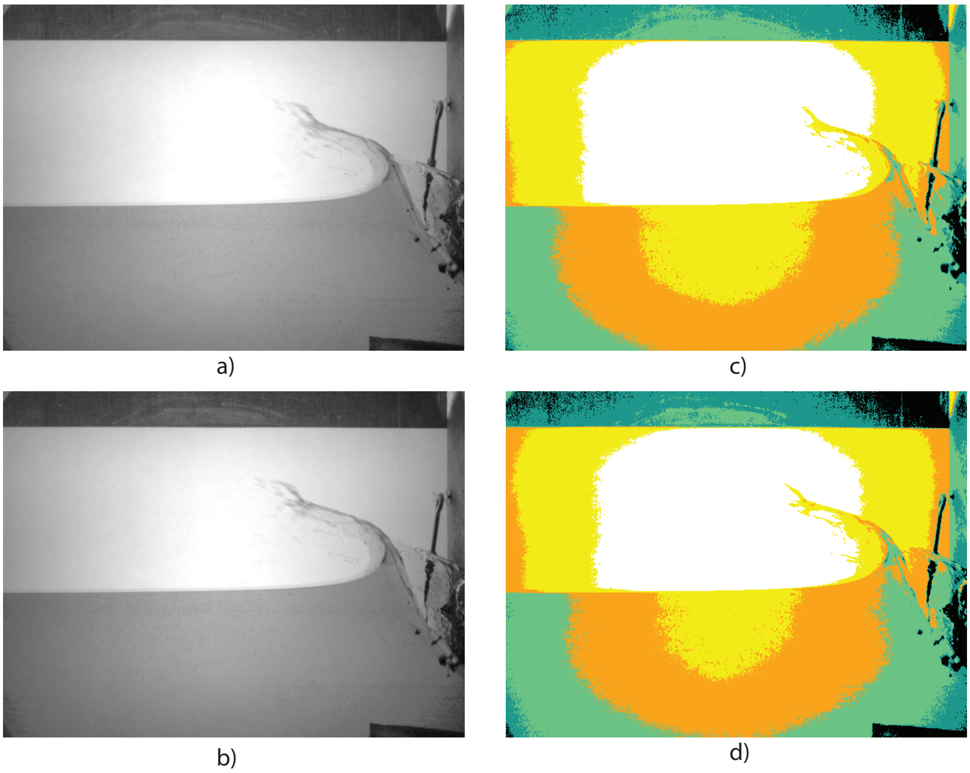


Figure 4.5.: *Images a) and b) are subsequent, c) and d) their according segmentation with k-means. Same colored areas are supposed to belong to the same class. The segmentations should be almost exactly the same. However, for example, the yellow spot in the center is clearly larger in c) than in d).*

- **Non-uniform emitting frequency:** Most of our everyday visual equipment, such as cameras or TVs, are adjusted to our eyes and only provide information as fast as they are comfortable with it. This also holds for ordinary room lighting. A commercially available light bulb works with alternating current on *220V* and *50Hz* (in Switzerland). These attributes therefore mean, that voltage reaches *220V* and *-220V* alternately with a frequency of *100Hz*. If the bulb does not contain any electrical modifier it will peak with the highest brightness at *100Hz*. The human eye simply does not perceive the light fast enough to see the *100Hz* flickering.
  Letting the camera run with *200 fps*, it provides eight times more frames per second than an ordinary video camera. This high number of fps makes the flickering of the light visible and notable as alternating brighter and darker images. A more precise look at the images reveals that bright images do not have all the same brightness and dark images the same darkness, respectively. This inconsistency originates from the fact that the camera's phase and the light's phase are decoupled, meaning the camera catches the intensities at different positions on the sine wave. The lighting is run with the frequency that comes

from the power outlet and probably also has a tolerance for the delivered frequency. Additionally, the camera does not shoot the images as constantly as one might think. For example, if there is already one more image per second, than the predicted *200 fps*, the phases will not be in sync anymore.
The frequency issue impinges on the segmentation quality. Two subsequent images, which ought to have similar intensities are segmented noticeably different. See Figure 4.5c and Figure 4.5d for some examples.

- **Paper structures:** Sheets of paper are clamped to rear glass in order to achieve a diffuse background illumination, in case the flat screen was not used. On closer inspection, one notices that paper has a fine visible structure. The changing light[11] intensity causes these structures to cast shadow patters which change minimally with the light intensity, possibly unnoticeable for humans. See Figure 4.6 for a visualization. These patterns can affect two subsequent steps: First, "false" shadow patterns may mistakenly be considered as part of the splash by segmentation algorithms or inversely, parts of the splash are not distinguishable from these background structures anymore. Secondly, illumination correction algorithms described in section 4.4 are no longer effective if the background is "dynamic".

- **Absorption law:** The absorption law is discussed in detail in section 5.1. It is the reason, why it is not possible to compare images solely depending on average image intensity values to come to a decision for certain operations which are, for example, used in the *background subtraction* technique discussed in section 4.4.

- **Distinct sensors and noise:** Operating the Dragonfly Express at low intensities brings along two coupled issues:
  - All CCD and CMOS based cameras introduce noise at low recording intensities, which is expected.
  - However, unexpected is the fact that the Dragonfly Express seems to have two distinct sensors sharing the photo-sensitive area. One of them introduces a lot more noise than the other half. As can be seen in Figure 4.6 the left side of the picture is clearly more affected by the noise than the right side, making it difficult to distinguish interesting features in the image.

The images in Figure 4.6 were generated out of 400 subsequent images each. The left image uses a neon tube and the right image a flat screen for illumination. One such image represents the sum of differences of all subsequent images of the sequence. Dark areas mean that there is no great change of intensity value over 400 pictures at these particular pixels. The brighter a pixel, the more it changed over time. The areas around the water surface are expected to be bright due to the moving water surface. Two issues are visualized in these images[12]:

- In both images the left side of the image is brighter than the right side, indicating that there is more noise on that particular side.

[11] Due to the non-uniform emitting frequency.

[12] Another interesting fact, which is not relevant to the issues though, is that one can see the reflection of the camera setup with tripod in front of the tank in the left image, which is quite fascinating, because one does not notice them when looking at a single frame.

- The border of the left images shows bright structures which originate from structures of the paper. As explained before, due to the frequency of the light the illumination is not constant, hence the shadows of the structure change with the intensity of the light. The center of the left image does not show much change because the photo-sensors of the camera at these positions do not notice the difference of intensity. They are fully saturated over the 400 images.

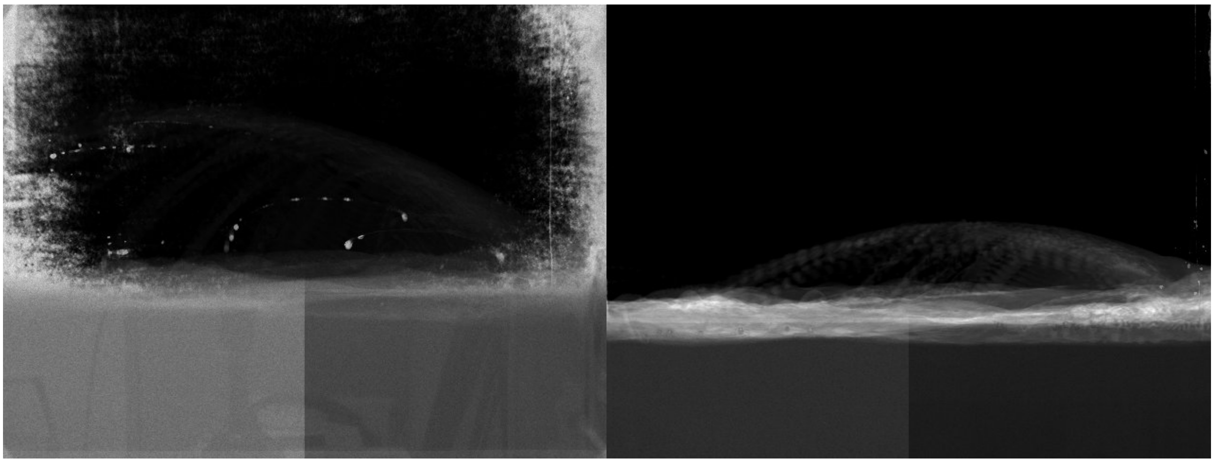


Figure 4.6.: *The left image shows the picture of differences taken of an neon tube illumination and the right picture accordingly with a flat screen illumination.*

4.4. Software Correction

Different techniques are presented to tackle the issues described in section 4.3. However, none of the techniques are satisfactorily successful which is most probably due to the interplay between all of the issues.
In the following section these notions will be used: A pixel p with position (x, y) of the i-th frame will be denoted as $p(x, y, t_i)$. t is used for the frame's index since every frame is part of the temporal progress of a certain recorded event. Reference images taken for correction purposes will be referred to as *correction images* and images taken in the course of the experiments are referred to as *experimental images*.

- **Background subtraction:** The idea behind this technique is to be able to detect moving objects within a scene in a simple manner by subtracting the *static* background of a moving object.
  For this particular case one regards the paper structure and the patterns that arise from the inhomogeneous illumination as a background which should not change over time, in the ideal case. Hence, it is possible to remove the background structure from the image which has immersed moving objects in it and obtain a picture with only the differences, i.e. the moving object(s).
  To achieve this goal the following steps are taken:

  1. A *reference picture* of the static scene is taken. "Static" is meant here in two senses:

   - Everything that moves does not belong to the background but to the observed object, hence static.
   - The properties of the illumination stay the same during all frames of a sequence, i.e. its position and emitted intensities.

   The depicted scene contains the empty tank, the sheets of paper and the lighting, an example is shown in Figure 4.7a.

2. Then, the image sequence of a water impact is shot.
3. Now, one uses the reference picture to subtract it from every picture of the sequence and obtains the difference between the static scene and the moved sequences, which means that moving object will be made visible.

Effectively, for each frame of a sequence every pixel is subtracted from the according pixel p_{ref} in the reference image and taken the absolute value of the result. Pixels which did not change their intensity values will have a value equal to zero or close to it:

$$p_{\text{res}}(x, y, t_i) = |p(x, y, t_i) - p_{\text{ref}}(x, y)|,$$

where p_{res} denotes the resulting image's pixel intensity, $i = 1, \ldots, N$ and N being the number of pictures in the sequence.

The problem with this technique is the assumption that the background in the moving pictures is static. This assumption does not hold for our setup. The reason is the before mentioned frequency issue. The small changes in the amount of light that reaches the paper also changes the little shadows cast by the paper's fibers which in turn change the image. Since the background is not static the resulting corrected image is not very satisfying since this technique works on pixel basis (cf. Figure 4.7c).
The background is not close to plain black and detected movements of the water have similar intensity values. The similarity makes the segmentation as hard as it is without using the background subtraction technique.

To rectify this issue, not only one correction image can be taken but several, with the idea to produce *matching* correction images for experimental pictures with different background intensities.
The before proposed solution to use several correction images leads to another problem of how to choose matching correction pictures for according experiment captures. This particular problem is connected to the absorption law mentioned in section 4.3. The law, discussed in section 5.1, states that the absorption of light that is passing through the liquid depends *non-linearly* on the distance it covered. Therefore, the difference in average intensities of two images may not only be due to lighting (frequency issues) but also due to areal water redistribution caused by the splash itself. Hence, it is not clear how to define a measure that correctly account for the intensities in order to match correction images with experimental images.
A possible strategy to choose the matching correction image for an experimental image might be to only compare a subset of the image pixels of both images instead of the whole image. This subset can be chosen such that only the background is depicted in that area. However, such a patch would need to be localized first and secondly, there is always the

risk of comparing a region where a water drop slipped in, instantaneously darkening the according average intensity value of the patch.

Yet another problem that comes up with the determination of proper correction images is whether to take create the correction image with a empty or half-filled tank:

- An experimental image corrected with a correction image of an empty tank will notice all of the liquid upon application of the subtraction method, which is basically correct. However, this behavior does not serve our purpose since the correction itself introduces new structures into the resulting image. The structures come from the correction image where sheets of papers are used to diffuse the light. The paper patterns are not existent when trying to correct the experimental image, since the liquid is plain dark.
- A half filled tank has a certain water level. Once the wedge touches the water, additional mass is introduced into the water body which elevates the level. During the evolution of the splash, the water surfaces goes several times over and below that level due to movement of the water. The subtraction of a still water level from a moving water level results in different colored areas despite them being part of the same water volume. Hence, complicating the acquisition of clear images, instead of improving them.

Basically, working with a light source which does not have a proper lighting frequency leads to many difficulties which need to be solved. It would be much more convenient to not even have to tackle this problem algorithmically, but to find an appropriate illumination in the first place. Finding another solution would also avoid the repeated acquisition of correction image for every new session. Each session involves pumping out the water, adjusting the scene, shooting the images, and pumping the water in again and all without displacing any object within the scene.

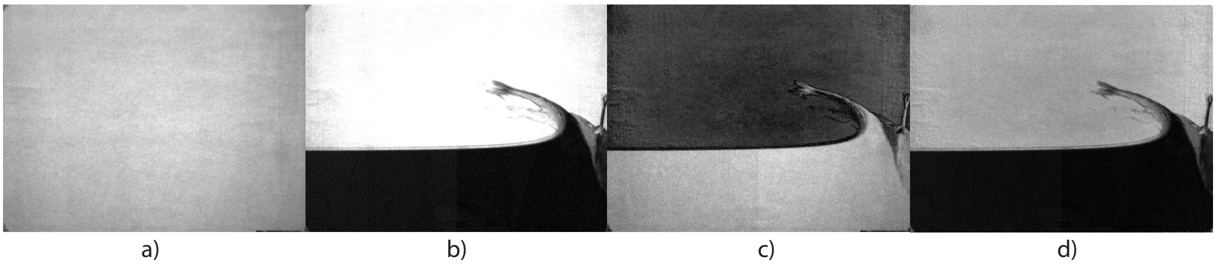


Figure 4.7.: *Image a) shows the correction image without water. The experimental picture is shown in image b). Image c) uses the background subtraction method whereas image d) utilizes the correction of inhomogeneous illumination approach. Note that after the correction, the background should be completely uniform.*

- **Correction of inhomogeneous illumination:** This technique is very similar to the previously described background subtraction. It is also used to remove shadow patterns caused by the paper and the inhomogeneous illumination, as the name method's states it. The idea is to use a beforehand taken image of the inhomogeneous illumination of the scene and to remove it from the experimental images, thus resulting in an homogeneously illuminated image.

In this setup it can basically be used to do the same thing as before with the background subtraction method but with a different mathematical operation per pixel. Instead of subtracting and taking the absolute value of every pixel, one divides a pixel through a reference pixel and by normalizing to the $[0, 255]$ interval the image results in a, in some sense, "normalized" image.
A resulting pixel $p_{res}(x, y, t_i)$ with frame number t_i at position (x, y) can be computed as follows:

$$p_{\text{res}}(x, y, t_i) = p(x, y, t_i)/p_{\text{ref}}(x, y),$$

where $p_{\text{ref}}(x, y) \neq 0$. As mentioned, an inflating of the obtained values to the interval $[0, 255]$ needs to be performed since they are in a much narrower band:

$$p_{\text{res,n}}(x, y, t_i) = \frac{(p_{\text{res}}(x, y, t_i) - \min\{I(t_i)\}) \cdot 255}{\max\{I(t_i)\} - \min\{I(t_i)\}}, \tag{4.1}$$

where $p_{\text{res,n}}(x, y, t_i)$ is the inflated (normalized) value, $\min\{I(t_i)\}$ and $\max\{I(t_i)\}$ the minimum or maximum intensity pixel value of an image I at temporal position t_i, respectively.
Here as well, the same issues as before in background subtraction are encountered and the same countermeasures are applied which turn out to be unsuccessful. Creating the correction images is done in the exact same manner as in the background subtraction technique, in fact, the same images were used. A comparison of both techniques can be seen in Figure 4.7.

- **Elevation of sequence intensities:** With this approach, the frequency issue of the bulbs is tried to be solved by aligning the average intensity values of every image in a sequence to a common level. As some images are evidently brighter than others, their average intensity per image must also be higher than the darker images. By adjusting the sequence these differences, which make the segmentation more difficult, are possibly alleviated.
More precisely, one chooses an image of the sequence to be the reference image, usually the first one, and calculates the average pixel intensity per image in the sequence. Then one elevates each image's average by the difference between the current image and the reference image by adding each pixel the difference. t_0 is chosen to be the frame number of the reference image:

$$\hat{i}(t) = \frac{\sum_{i=1}^{w} \sum_{j=1}^{h} p(i, j, t)}{w \cdot h}$$
$$e(x, y, t) = p(x, y, t) + \left(\hat{i}(t_0) - \hat{i}(t)\right),$$

where $\hat{i}(t)$ is the average intensity value over all pixels of an image in frame t, w and h the image width and height in pixels and $e(x, y, t)$ the change per pixel which has to be applied whereas $t \neq t_0$ must hold.

The approach is based on a good idea, but it fails to be useful and produces seemingly unchanged images. Basically, because the images with a higher average value only have brighter pixels at non-liquid positions. The reason probably lies in the before mentioned non-linear absorption law which causes the dark liquid to change its brightness only imperceptibly for the camera. Therefore, the change in average is mostly dominated by bright pixels, but when calculating $e(x, y, t)$ the total sum of differences is distributed

over all pixels, the brighter and darker ones. In applying the (positive or negative) elevation, the dark pixels consume some of the "total energy", which was only meant to be for bright pixels.
A further strategy that can be applied is to use a thresholding on the pixel intensities to select them. The average of the selected pixels is then used to cross-compare with other images. A point which would exacerbate the challenge is the fact that the body of water gets redistributed over the whole visible surface, adding and removing non-linearly behaving dark pixels. The consequence is that the amount of pixels which is affected by the frequency issue is not constant over time. Hence, the energy to distribute would need to be adapted in a, still to be defined, appropriate way. This and similar approaches would present yet another workaround.

- **Normalization of images:** This is a standard image processing technique to improve contrast in a picture. By stretching the histogram of an image the whole bandwidth of gray values from 0 to 255 is used. Though, if the image is already well illuminated and contrasts are high, there is no notable difference between the input image and the output image.
  This technique was used to see if it helps to get rid of the fluctuation of the intensities caused by the emitting frequency of the light bulbs. However, the images already cover almost the whole intensity spectrum with the dark liquid and bright background and show a high contrast. The differences of images in a sequence manifest themselves in the distribution of the values in their histograms. This means a normalization only stretches the slopes of their histograms, not really tackling the problem of the fluctuating intensities.
  The normalization can be calculated by determining the minimum and the maximum intensity value of an image and by simply mathematically stretching this interval to the interval $[0, 255]$ as it was already done in equation 4.1.

5

Liquid and Colors

Previously, the construction of a tank and the setup of a camera which allows to capture liquid phenomena was portrayed. The setup of a camera and its according lighting that enable a proper recording of the scene to depict was described. Within that setup, attention was paid to find a lighting which creates the highest uniform contrast between the liquid and its background.
Now, the attention shall be drawn to the medium that is going to be filmed. We chose water for our experiments. Instead, oil or other chemical substances could be used, see section 9.2 for further ideas. Evidently, water is transparent in a clean and liquid state which is barely ideal for analyzing video recordings where a luminous background is chosen that shines through the medium that is to be observed. The logical conclusion to prevent the light from passing through the water was to use a colorant to dye the water. Of course, a substance needs to be picked such that no other physical properties of the water are changed except transparency (for example its viscosity can change due to excessive dyeing). In the following, different coloring methods and substances are evaluated and their advantages and disadvantages are discussed.

5.1. Coloring

This part of the setup, the liquid and the inside of the tank, needs to be seen in a larger period of time than just the few minutes during the recording of the events. Before taking any pictures the water needs to be filled into the tank. After all, the amount of water that is used is considerable and needs to be transported into the room where the tank is standing. Spilling water in a room with many electronic devices is not tolerable, hence minimizing the number of liquid transports a good resolution to keep. Ideally, the liquid is stocked to save water, colorant and time.
The dodge with stocking water is that it evaporates slowly and in doing so it leaves stains on the surfaces it touches. Plain water leaves limescale. Water with color leaves colored stains. This

can be observed with all the colorants that we have tested. To confine the largest part of the vapor the stocking basin should be covered in order to let the water condensate on the covering material or just a lockable canister can be used. A good piece of advice is to not keep the water in the tank where the experiments are carried out. Otherwise, the acrylic glass gets coated with a color stain. For this setup an additional bucket was kept ready for pumping the water out of the tank with a little water pump.

There is an immense variety of colorants available. To narrow down the choices a few points were considered when choosing a colorant. Pricing, easiness of availability, and, depending on the used amount, also environmental friendliness. The pricing usually correlates with the availability and the needed quantity, and the environmental friendliness partly with the quantity. Besides these general points, one specific factor was also important: suitability.

Generally, one can say that a faint coloration of the water is quickly visible when adding a colorant. But a strong darkening takes a considerable amount of colorant. This observation is reflected in the so-called *Beer-Lambertian law of absorption*[1], illustrated in Figure 5.1, which relates the incoming light passing through a material, in our case the colored water, with the outgoing light and is for liquids usually denoted as:

$$I_1 = I_0\, 10^{-\epsilon c l} = I_0\, 10^{-\alpha\, l},$$

with:

- I_0: The intensity of the incoming light
- I_1: The intensity of the transmitted light
- ϵ: Molar absorptivity of the absorber
- c: Concentration of the absorbed substance in the fluid
- l: Distance, the light travels through the fluid
- α : Absorption coefficient of the substance

- **Common Painting Colors:** This is the easiest product to obtain. The cheaper and environmentally friendlier, the better, because the content of the tank will need to be emptied more than once. Therefore, standard paint, which is available in Do-It-Yourself stores, can be used. Once a small amount is dissolved in a bottle, one notices quickly that it has a quite strong dyeing acquirement. The dyeing's goal is to let as little light as possible shine through, which works quite well with this color.
  However, there are several disadvantages. First, the color particles of this paint are not water soluble, that is to say if one keeps a close eye on the water the particles are still visible, also for the camera. Secondly, if the water evaporates the color gets distributed along the rim of the water surface and sticks tenaciously to the walls. If one considers that the water may stay a longer period in the tank then it is not a preferable property since in that case each time it would need cleaning. Furthermore, for mingling the dyeing color with the water one needs to shake or stir it thoroughly which can cause small bubbles. These bubbles, in turn, cause a delay in the whole recording process since pictures with bubbles are not desirable and they do not burst for quite a while.

[1] http://en.wikipedia.org/wiki/Beer-Lambert_law

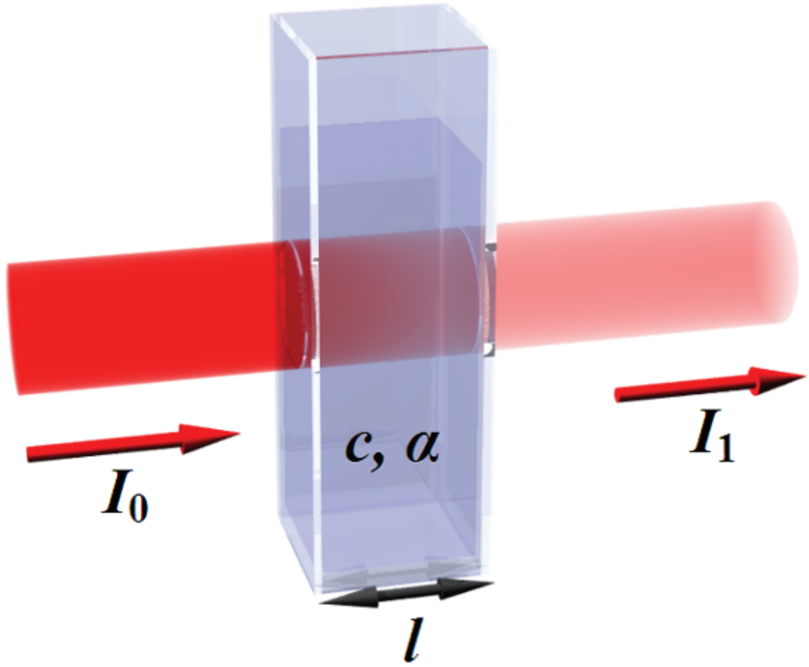


Figure 5.1.: The incoming light gets attenuated by the medium and results in a weaker outgoing light (Source: http://en.wikipedia.org/wiki/File:Beer_lambert.png)

- **Food Coloring:** This coloring is obtainable from local grocery stores and is probably one of the environmentally friendliest colors one can find and is nonhazardous for humans. However, it is also relatively expensive because its intended use for cooking. The tank has a capacity of about *25 liters* whereas only half of it needs to be filled for the recordings. For not only having a considerable dyeing but a dark dyeing of the *12.5 liters* one needs quite a lot of this kind of colorant. Unfortunately, it is only sold in little portions which makes the whole issue more cumbersome.

- **Common Ink:** Ink is very easily obtainable. One only has to browse through larger stationery shops for not only encountering the small cartridges for fountain pens but the bigger refillable cartridges. Generally, one can say that a faint dyeing is quickly observable once a drop is let into the water, but to obtain a deep coloring it needs a lot of ink. Ink is inoffensive and is even given to little children to write and play with, hence the environmental friendliness issue is not a concern. The fact that colorant within the ink is already dissolved in water already ticks off the water solubility issue. The main disadvantage is, again, the needed amount for being effective.

- **Methylene Blue:** This chemical substance with the chemical formula $C_{16}H_{18}N_3SCl$ is a salt. The acquisition is difficult since it is only sold in special chemical stores. Methylene blue is considered harmful when swallowed[2] which is the reason that it is rather difficult to get access to and not available in pharmacies. A visit to the university's chemical department was then the final resort. Its price of about *1000* $\frac{\text{CHF}}{\text{kg}}$ appears to be a lot, but once one has seen the incredible dyeing acquirement of this substance it becomes clear

[2] http://en.wikipedia.org/wiki/Methylene_blue or http://de.wikipedia.org/wiki/Methylenblau

that a very little amount is necessary to serve the purpose of darkening the water. About *1 g* dissolved in *12.5 liters* of water attained the darkest coloring of all colorants, which was perceivable as almost black. In contrast to the other colors, this substance was obtainable in a pure powder form which accounts for the deep dyeing. Its dyeing acquirement is both, its advantage and disadvantage. Inevitably, small stains occur on tables and hands which are relatively tenacious and difficult to clean and therefore it is recommended to handle it carefully.Figure 5.3a shows the flacons.

- **Fluorescein:** Among others, fluorescein has applications in biology where it is also used to dye the cytoplasm of organic cells for obtaining a better visibility and distinguishability under the microscope. Thus, fluorescein is rather inoffensive[3] for living organisms as opposed to methylene blue. It is also only available in special stores.
  The fluorescein as powder has a reddish orange color which can be seen next to the dark greenish methylene blue in Figure 5.3a. The more it gets diluted with water the less one sees of the red color and the more it blends into a greenish yellow. The interesting part about this substance is, that when it gets irradiated by ultra-violet light it emits light in the yellow part of the visible spectrum. The idea behind the usage of this substance is to exploit the self-luminous property to get rid of the external (visible) background illumination so that in a dark room only the luminous water needs to be recorded. Hence, the color cannot be used as darkening element but contrariwise as scene illumination.

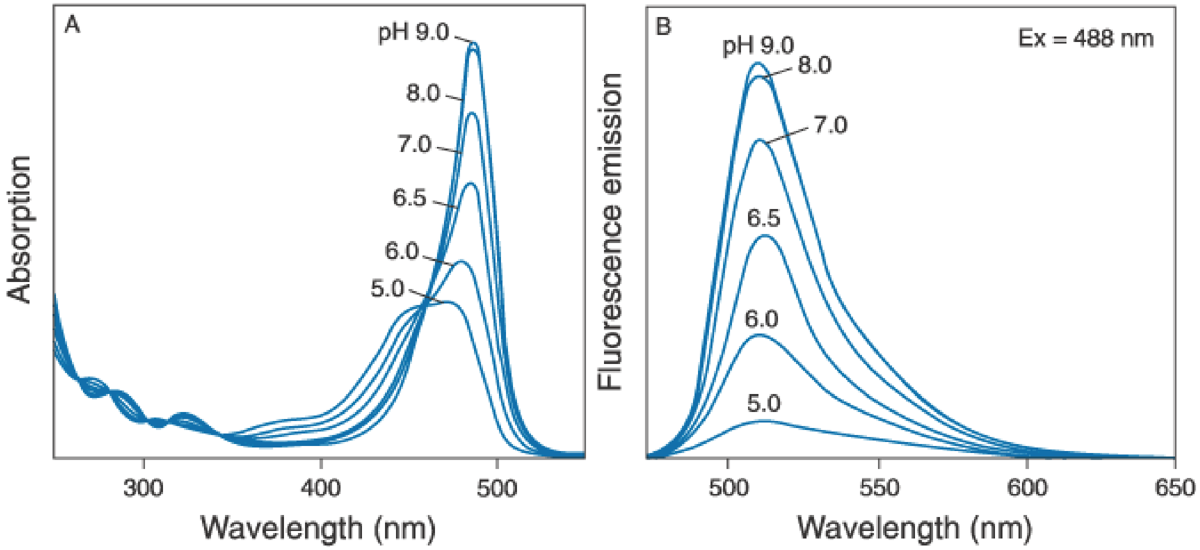


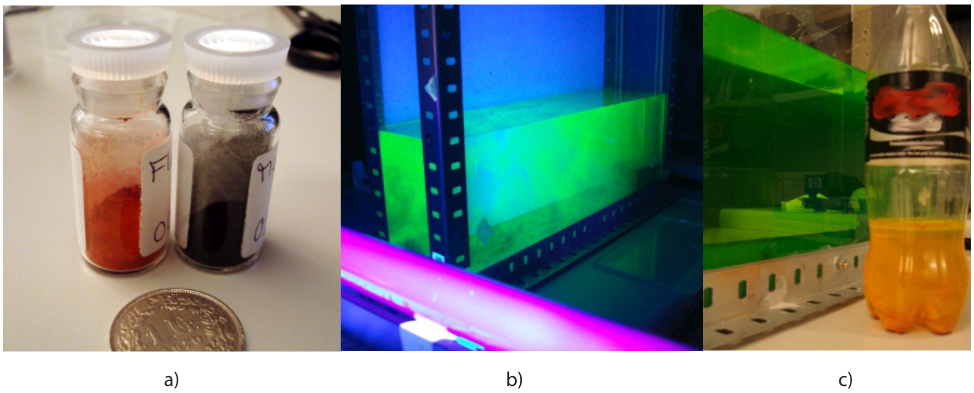
Figure 5.2.: *On the left the absorption spectrum of fluorescein is depicted. The absorption and emission are dependent on the pH value of the medium. The right graph shows its*[4] *emission spectrum.*

A little elaboration on the idea: Ultra-violet light is outside the spectrum which humans can perceive. One can think of the fluorescein to be a light wave converter. It absorbs certain wavelengths and emits light with a different wavelength. Therefore, one could use a camera that does not see the absorbed UV light but the emitted visible light and have an optimal illumination of the scene from everywhere without having (visible) reflections.

[3] Only irritating to eyes.

However, there are a few drawbacks to this:

- UV light, is by definition, invisible. Commercially available UV lights, which are also used in dance clubs, are not only emitting UV light but also visible light, which is seen by the camera, as shown in Figure 4.4b and 4.4d. This can be verified by just looking at the bulb, since humans perceive it as weakly glowing. Hence, the UV illumination cannot be positioned somewhere close to the field of view of the camera because it would diminish the quality of the picture. Also, the liquid is then not the only source of light which draws us back from the optimal illumination.
- Although recordings are possible with the glowing fluorescein, in our setup it is not strong enough to generate a low-noise image at *200 fps*. Adding additional UV light sources is only limitedly useful, because there is a physical limit to the emitted light that comes from one fluorescein molecule which depends on its quantum efficiency. The fluorescence is a quantum phenomenon which means that it is discrete and restricted in some way. In fact, there is a certain number of molecules in a portion of fluorescein and each molecule has a certain number of electrons that can produce the glowing phenomenon. Once these are saturated the surplus on intensity does not change the outcome anymore.


a)

b)

c)

Figure 5.3.: *Image a) shows the flacons in which fluorescein and methylene blue was obtained. For comparison a 1 CHF coin is positioned in front of them. Image b) illustrates the fluorescent property of fluorescein. Image c) shows twice the same substance (water with fluorescein) but in different concentrations and different colors.*

5.1.1. Verification of the Absorption Law

To verify the Beer-Lambertian law a series of measurements were conducted; sample images are shown in Figure 5.4a. Methylene blue was obtained as powder, and since the used quantity is difficult to weight without adequate instruments it was dissolved in a bottle of water prior to carrying out the measurements. The idea is to qualitatively measure the attenuation of the background light depending on the amount of colorant that is put into the tank. At first, a picture of the tank filled with plain water was taken. In every subsequent step a constant amount of dissolved, but still quite concentrated, methylene blue was put into the tank. Then a picture was taken and an area of the image where only liquid prevailed was chosen. Finally, the average of intensities of those pixels in that image patch was taken. The average intensity values are depicted in the graph shown in Figure 5.4b. As one can see the curve roughly follows the Beer-Lambertian law.

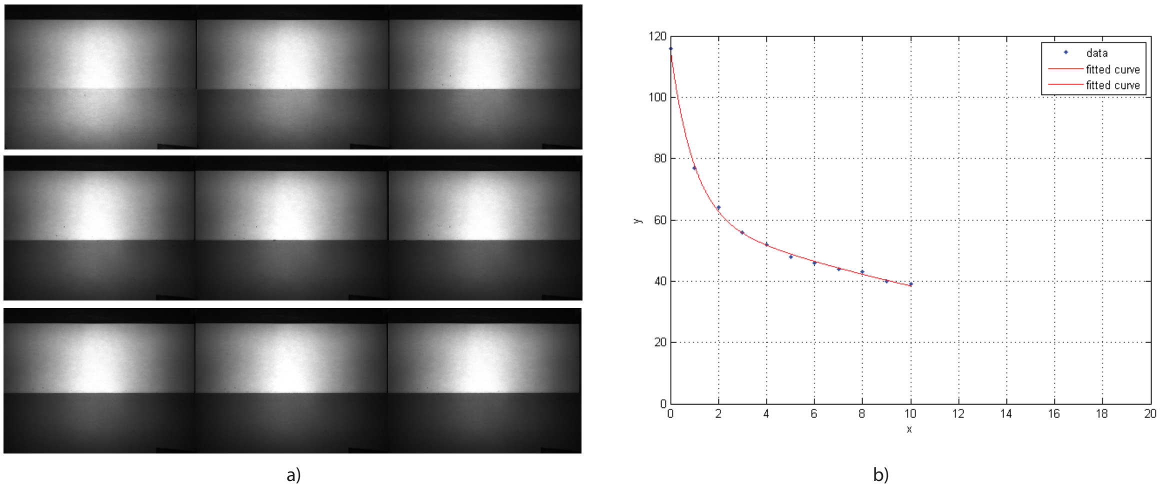


Figure 5.4.: *The nine pictures on the left illustrate the visual darkening of the water by adding methylene blue, where the water in the upper left picture does not contain any colorant. The right graph shows that the curve actually behaves exponentially, where the x-axis is the number of samples and the y-axis the measured intensity value.*

The finding that the intensity is exponentially attenuated as it is described by the absorption law is not only important for determining the colorant amounts for dyeing the water sufficiently, but also supports the prior reasoning mentioned in section 4.3 and 4.4 that one cannot determine the amount of distributed water due to an impact from the measured average image intensity.

6

Segmentation

To make the liquid appear as two-dimensional as possible a few steps were taken as it was described in chapters 3, 4 and 5. Despite the effort put into these steps such as narrowing the tank, choosing a uniform background and coloring the water, some remaining three-dimensionality persists when taking pictures of the liquid in action. The ideal image to work with would look such that the liquid has a complete opposite color than the air or background, independent from the size or shape of the water and water drops. Until now, it was tried to solve this problem on its core by applying the mentioned measurements. Now, algorithmic approaches are studied to complement the previous efforts to finally convert the image with some ambiguous regions into a two-dimensional image. Ambiguous regions are regions where it is difficult to tell if a pixel belongs to the liquid or the background. These emerge from shadows on areas where no shadows should be or from reflections which make parts brighter than they should be.

By segmenting an image one decides according to some criteria to which class any pixel belongs to. In our case, there are two classes - the background and the liquid. However, segmentation can be used for any data set and can have an arbitrary number of classes. Even in this case, it sometimes makes sense to define more than two classes as it will be seen for the *k-means clustering* algorithm. There is a big number of such algorithms out there, choosing the appropriate one entails quite some investigative work. Besides *k-means*, two other segmentation algorithms, namely *Region Growing* and *Graph Cut*, were implemented and tested. Finally, k-means was decided to be the most suited algorithm for this problem due to the good segmentation results and its processing speed and algorithmic simplicity.

Having applied a segmentation algorithm means having eliminated optical ambiguities which does not imply that the image is correctly separated into background and liquid. Generally, one can say if a pixel looks ambiguous for a human than there is also a high risk that this particular pixel is incorrectly labeled by an algorithm and therefore there are false positives and false negatives with respect to membership in either of the two classes.

6.1. Algorithms

In the following subsections all implemented and evaluated algorithms are presented.

6.1.1. Region Growing

This algorithm tries to group "similar" pixels into largest possible patches of the same label. Once there is no unlabeled pixel left it terminates.
It starts off by choosing a threshold T and an arbitrary pixel anywhere in an image i, either randomly or by user input. This pixel will be referred to as the seed pixel s and labeled l_1. In every iteration, for every neighbor of a l_1-labeled pixel it is decided by using a similarity constraint whether it is put into the same class or it will be labeled l_2. If it already has a label then it will not be changed. Once all similar neighbors are united under the same label and there are no pixels left that are similar, a new seed point is chosen and the algorithm is repeated until there are no pixels left without a label.

Now, the mentioned similarity constraint still needs to be further specified. There are several different choices of how to do so. Suppose a similarity constraint function $S(a, b)$ takes two pixels a and b and returns a value in the interval $[0, 1]$. If the resulting value is closer to one, the pixels a and b are similar and dissimilar otherwise. To decide whether those two pixels are similar where b is the neighbor of a to be inspected, the result of $S(a, b)$ has to be compared with the previously chosen threshold T. Therefore, if $S(a, b) \leq T$ holds, then b will be added to the group of pixel a and will be labeled accordingly.
The comparison can be done in three ways:

- **Seed point comparison:** When deciding whether a neighbor b belongs to a group of pixels the other pixel a in $S(a, b)$ is the original seed point of this particular region. Meaning all pixels within a region are compared with the same pixel giving them a common comparing ground. All are similar to each other. However, if the seed pixel is chosen badly, such as an edge of a depicted object, it will only result in a small solution patch.
- **Neighbor comparison:** Instead of always checking back with the seed pixel one can compare the new pixel with its direct neighbor that led the growing patch there. Hence, a and b in $S(a, b)$ are always neighboring pixels. Obviously, this gives quite some flexibility depending on the chosen threshold T. The flexibility gives rise to the question how similar the seed pixel and any other pixel within the same region after a few iterations really are.
- **Statistical comparison:** Since the algorithm sweeps over all pixels of a region it can collect statistics about the encountered pixels. For example, one could imagine to compare pixels with a *moving average*[1] of the pixels which are already included in the region. Such a moving average function could look similar to:

  $$\mathrm{ma} = \frac{p(i) + p(i-1) + ... + p(2) + p(1)}{i},$$

  where $p(i)$ is the i-th pixel added to the region. Also, one could only choose to take the last n of those added pixels. There is a huge variety of possibilities to choose from.

[1] http://mathworld.wolfram.com/MovingAverage.html

The segmentation does not need to be fully automatic. There is still a user who can judge whether a labeling result is acceptable or not. Therefore, a solid choice of the seed pixel can be assumed. Finally, assuming a solid seed pixel selection we choose the first of these three similarity constraints to be utilized for our region growing algorithm.

The constraint function $S(a, b)$ itself uses a simple normalized difference of values:

$$S(a, b) = \frac{|a - b|}{255},$$

where a and b are intensity (gray-scale) values of two given pixels. $\frac{1}{255}$ is the normalization factor to fit the result into the desired $[0, 1]$ interval, where 255 is the greatest difference that can appear between intensity values of two pixels.

As for the neighborhood that is being examined in this algorithm, there are two possible definitions shown in Figure 6.1.

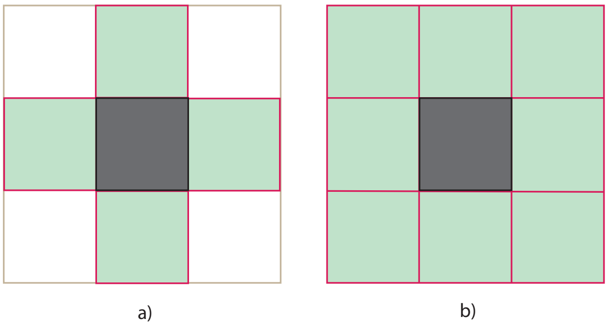


Figure 6.1.: *Image a) shows the utilized 4-cell neighborhood (von Neumann) and b) the alternative 8-cell neighborhood (Moore). In 3D the number of neighbors are accordingly higher: 6 and 26, respectively.*

For the result it does not really make a difference which of the neighborhoods is used, as both of them can reach any part of a connected region. We choose the first one, because most of the times at least one part of an examined pixel's neighborhood is in the region where the labels are already set, as illustrated in Figure 6.2. Minimizing the number of checks for unset labels is therefore preferable.

6.1.2. 3D Region Growing

The data that is taken from the experiments will be in sequential form, two-dimensional intensity arrays over a certain period of time. This data can be seen as an evolution of the liquid over time. Hence, all data can be arranged in a three-dimensional grid where the dimensions can be addressed over the according coordinates x, y and t. The region growing algorithm can be extended in such a way that it does not only grow in the (x, y)-plane but also into the t-dimension. By doing so, one seed point can be chosen which will grow over the whole sequence connecting all neighboring parts.

A disadvantage of the two-dimensional version is that droplets will either not be recognized or they will belong to another class than the main body of water, depending on the number of

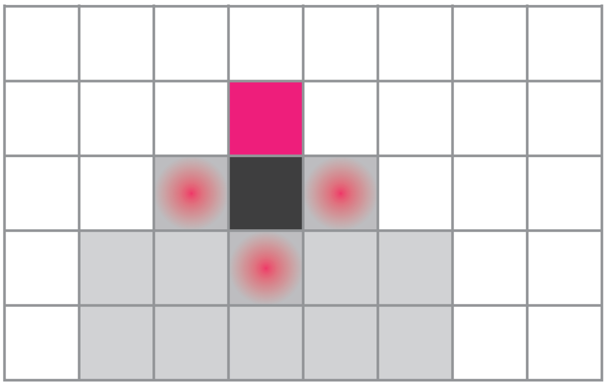


Figure 6.2.: Region Growing: The black cell is the currently considered cell. Light gray cells are already given a label. Effectively, only one out of four neighbors has to be checked in this particular case (pink).

classes that are demanded by the user. The idea behind the three-dimensional growing lies in the fact that in the beginning every piece of water droplet in a sequence (over time) will be in the main body of water, united. All droplets that eventually evolve, evolve out of the main body and thus connecting the droplet not over space with the main body but over time. The example in Figure 6.3 was taken with a moving hand. It show the difference between a 2D grown sequence and a 3D grown sequence. The reason why an example with the hand is depicted but not with the liquid is that the liquid is moving too fast. The concept of growing over time only works properly when the according objects to connect also "touch" each other in the time-dimension, see Figure 6.4 for an example. In the case of little water drops which move more pixels per time step than they are in diameter, the algorithm just cannot connect them. The only way to correct this issue would be to use a camera with a higher frame rate. See Figure 6.4 for an illustration. A general remark that applies to two-dimensional and three-dimensional region growing is that it needs a lot of parameter tuning by the user until a reasonable result is obtained.

Algorithm

The following algorithm applies for 2D and 3D region growing. The only differences are the considered neighborhood and the additional dimension:

1. Choose a threshold T.
2. Choose a seed pixel s, either manually or randomly.
3. Label s with l_1.
4. Put s's neighbors into a list L if they are not in L yet.
5. Compare s with the first pixel b in L according to a similarity measure $S(s, b)$ and remove b from the list L.
6. If $S(s, b) \leq T$ give b the label l_1 otherwise l_2.
7. If b is labeled l_1 add its neighbors to L if they are not in yet.
8. Go back to item 5. if L is not empty.

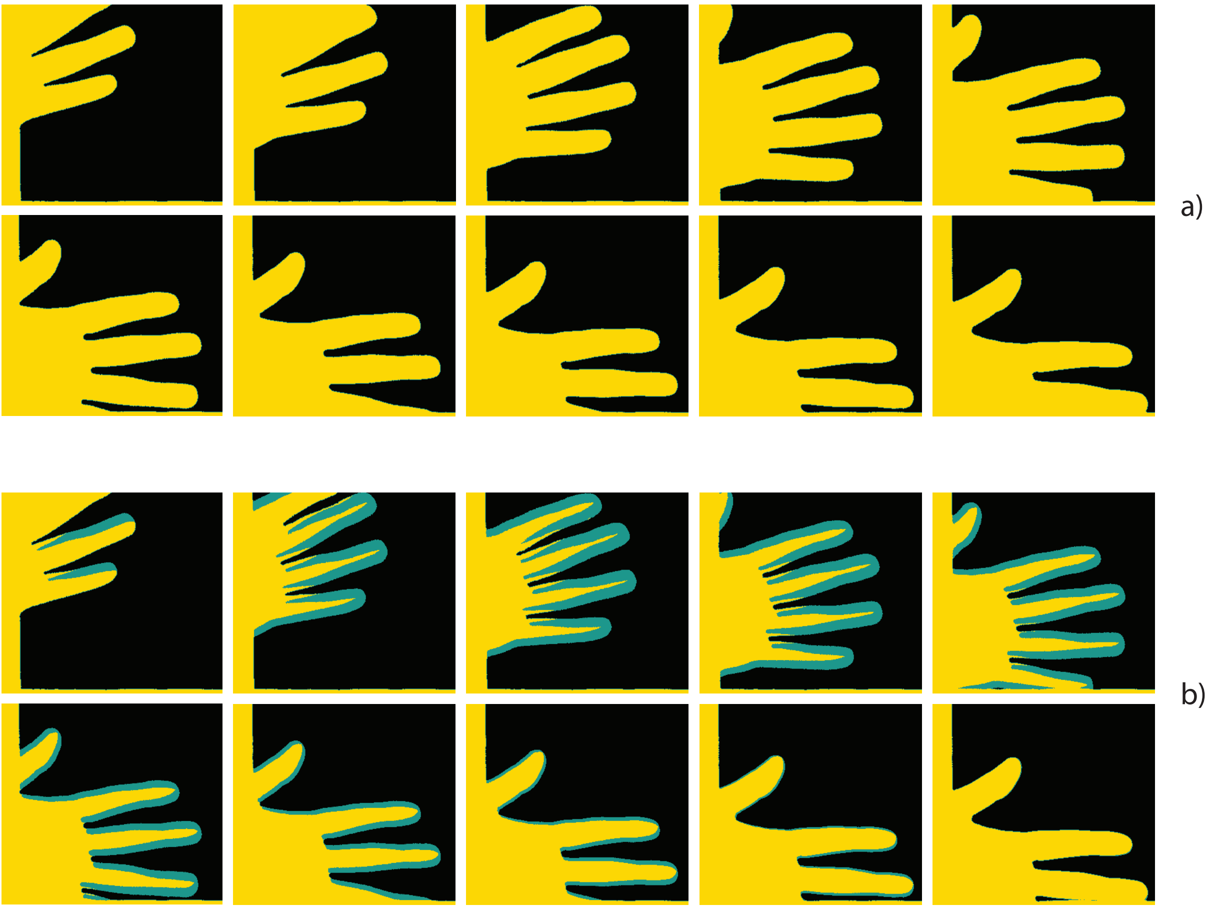


Figure 6.3.: *Sequence a) shows the segmentation with 2D region growing whereas b) uses 3D region growing. In sequence b) the blue areas are the border regions, which are newly acquired from one frame to the other. Especially interesting are the images on the far right. In 3D growing parts of the area between the fingers are recognized, whereas 2D region growing counts the background to the same class as the hand.*

2D and 3D region growing are rather simple algorithms with limited abilities. The choice of the seed pixel is a crucial point in the algorithm. Depending on the position of the seed the outcome might differ. If the seed is accidentally chosen to be on a crest between dark and bright areas the segmentation result will look rather poor.

The way the algorithm is currently implemented, it only recognizes either main water body and bubbles or main water body and water drops. Extending the algorithm to segment into more classes would need user intervention to determine and assign the associated classes. For example, let us say the main water body and the contained bubbles are labeled in a first iteration, the second iteration would randomly choose a further seed point and this one happened to be within a water drop. Let us further assume that all further seed points are, by mischance, chosen to lie in water drops, therefore resulting in an image with as many classes as there are water drops, most probably more than two.

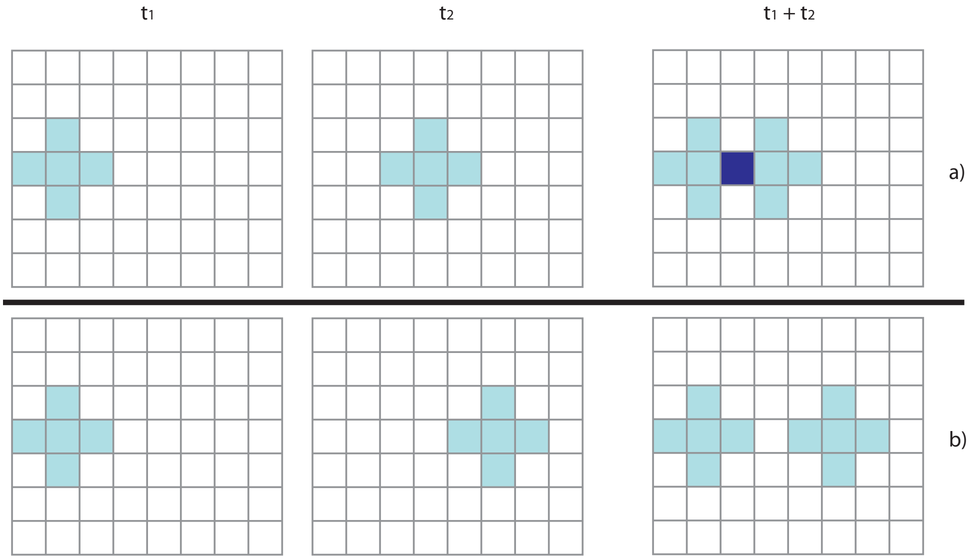


Figure 6.4.: *Example a) shows two "water drops" in two subsequent frames. They can only be grown with 3D growing if they have each other as neighbors, i.e. if they "touch". Example b) is a case where 3D region growing does not work.*

6.1.3. Thresholding and *k*-Means

The k-means algorithm is basically a generalization of the thresholding algorithm. Although thresholding is separately implemented it will not be discussed in further detail except for some occasional remarks. The k-means clustering algorithm was presented in 1966 by James MacQueen in [M$^+$66]. It is an iterative algorithm that gets more and more accurate with every iteration and can work on multi-dimensional data. The k stands for the number of desired classes into which the algorithm segments the data. k has to be provided by the user. There are extensions of the algorithm where it tries to estimate an optimal k before applying the actual clustering, but this step is not guaranteed to deliver a good choice. Hence, we trust the user to choose an adequate k.

The algorithm strives to group similar data into the same group. As mentioned, it can work on multi-dimensional data, but in our case it is only used on intensity values meaning that similar colored pixels will belong to the same group, whereas their position within the image will be irrelevant.

One could consider the position in the image too for similarity calculations, but the question to ask is whether it makes sense to do so. For example, let us consider two pixels with the same intensity value, $p(x_1, y_1, t) = p(x_2, y_2, t)$, which are part of the same liquid body but which have very dissimilar positions. This makes sense since a liquid may be distributed on the lower half of an image. Therefore, let us assume (x_1, y_1) is on the lower left corner and (x_2, y_2) is on the lower right corner of the image. If the similarity does not only depend on the intensity value, but also on the distance, then these two pixels will have a high chance of not belonging to the same group of pixels although they are part of the same liquid. Hence, the similarity measure for pixels was chosen to only work on pixel intensity values for our case.

After the user has chosen a suited k, k random points m_1 to m_k within the working domain are chosen. A data point m_i is a vector with d dimensions in the general case. In our case $d = 1$

being the intensity value. If d is chosen to be greater than 1 and the dimensions do not work on the same domain, then the vector should be normalized in order to have an equal weighting. For example, $v_1 = (i, x, y)$ could be composed of an intensity part which can take values in the interval $[0, 255]$ and a position where x can take any value within the width of the image, say $[0, 639]$ and y within the height of the image, say $[0, 479]$. Obviously, a change by one in the i-dimension does not have the same meaning as the same change in x-dimension.
Then, for every data point v_j, it is decided to which of the m_l it is the closest by giving it its label l. Once all data points have been labeled, the average of all points of a certain label is calculated and assigned to the according m_l, which is the center of gravity of all the points with the same label. All of the m_l now have a new value and the procedure begins from the start again. These steps are repeated until there are no more changes to the values of m_l which means they converged to a *local optimum.*

This brings us directly to one of the disadvantages of this algorithm. It is not guaranteed that the global optimum is found, which is also connected to the initialization of using random points. One could chose to fix the initial points but then, if the global optimum is not found, it will never be reached. By using a random initialization there is a chance to find the global optimum when applying the algorithm several times. There is, in fact, a newer version[2] of the k-means algorithm that raises the chances of finding the global optimum, but it also increases the computation time. For the purpose of this thesis, the original algorithm is used since it is very fast and there is a user who has to approve the segmentation anyway. It is reasonable to let the user wait for one or two more applications of the algorithm until a convenient solution is delivered.

The similarity measure is defined by the (normalized) squared euclidean distance of two points:

$$dist(a, b) = \sum_{i=1}^{d} (a_i - b_i)^2,$$

where a and b are two points with d dimensions.

By choosing k greater than two, one can differentiate between strong features in an image and fainter ones. In a further step, it can then be decided which labels are put into one class that represents the background and which are put into the other class that represents the liquid body.

Algorithm

1. Determine k, the number of classes to separate to
2. Choose k random points m_1 to m_k
3. For all pixels p_i determine the closest m_l, where $i \in [1, width \times height]$ and $l \in [1, k]$
4. Calculate the average of all points that belong to a class and assign it to the according m_l
5. Repeat steps 3. and 4. until there is no difference between the old and new m_l

K-means is a fast as well as a simple algorithm. However, it only relies on intensity values meaning that no information about connectivity is included in the classification process. Its

[2] Global k-means and Fast Global k-means

random nature in the initial steps gives it an uncertainty about the outcome. It is not guaranteed to return the global optimum. Removing this randomness by choosing fixed position to start, one takes away the flexibility of the algorithm. A poorly segmented image is quickly identified and is easily replaced by a user-demanded re-computation.

6.1.4. Graph Cut

In the field of graph cuts there are many candidates to choose from. The work of Boykov et al. [BVZ01] called our attention. The chosen algorithm uses graph cuts to minimize energy functions. The energy describes how well, in some sense, the input matches the output image, i.e. the picture from the experiment and the segmented image, respectively.
Each pixel p in an image has to be given a label. Which label p receives depends on the result of the energy function $E(f)$, which is to be minimized with respect to f:

$$E(f) = E_{\text{smooth}}(f) + E_{\text{data}}(f), \tag{6.1}$$

where f is a selected labeling configuration for an image. If an image is pictured as a height field then the pixel intensities tend to be piecewise smooth hills, except at object boundaries, where abysses will be seen (i.e. when changing from liquid to background and vice versa). $E_{\text{smooth}}(f)$ measures the degree to which the labeling f is not piecewise smooth. $E_{\text{data}}(f)$ rates the difference between the configuration f and the actual image.

Determining the smoothing function E_{smooth} is a delicate issue and influences the outcome strongly. There are some examples given in [BVZ01], such as

$$E_{\text{smooth}} = \sum_{\{p,q\}\in N} V_{\{p,q\}}(f_p, f_q),$$

where N is the set of pairs of adjacent pixels and $V(f_p, f_q) = |f_p - f_q|$. With this particular example no noteworthy results were achieved. An additional MATLAB wrapper written by [Bag06] uses a smoothing function defined for every pixel that was generated by using the given image and producing another picture containing the edges with the help of horizontal and vertical *Sobel filters*. That idea was transfered to our implementation.

The data cost function is defined in the publication [BVZ01] as follows:

$$E_{\text{data}} = \sum_{p\in P} D_p(f_p),$$

where P is the set of pixels and f_p the label chosen for pixel p. D_p is a function that determines how appropriate a label f_p for a pixel p is. Typically, $D_p(f_p) = (f_p - i_p)^2$ is chosen, as it was the case in our implementation. i_p is the observed intensity of pixel p and f_p is the label chosen for p. However, it is not specified how these two variables are subtracted from each other, since f_p does not have a connection with an intensity value per se. To solve this issue, f_p can be given the average value of all pixels labeled with the same label.

To minimize $E(f)$ the graph cut approach is used. The used graph cut algorithm itself was already implemented in the framework taken over from [Bag06] and it was not elaborated on it.

Therefore, only a summary of the general idea is given in the following paragraph, for further details the reader is encouraged to consult the according literature.

A graph cut $\mathcal{C} \subset \mathcal{E}$ in a (weighted) graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$, where $\mathcal{V}$ are its vertices and $\mathcal{E}$ its edges, is a separation of the vertices into two disjoint subsets. In computer vision, the graph cut idea manifests itself as follows: The graph nodes $\mathcal{V}$ are single pixels connected with each other. Additionally to the pixel nodes, there are two more distinct nodes called *terminals*, t_a and t_b. All pixel nodes are connected to both terminals. A proper cut removes edges from $\mathcal{E}$ such that two disjoint subsets are created *and* none of the terminals are completely separated, resulting in graph $\mathcal{G}(\mathcal{C}) = \langle \mathcal{V}, \mathcal{E} - \mathcal{C} \rangle$. Figure 6.5 shows an illustration for a very simple example with 4 pixels.

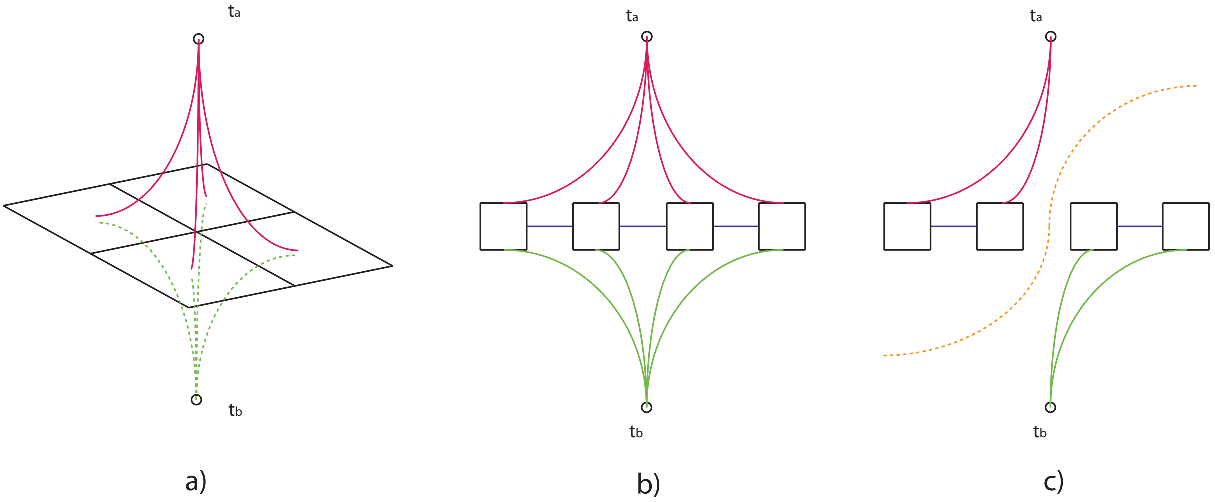


Figure 6.5.: *Graph Cut: Image a) shows the four pixels in a 2D array. All pixels are connected to both terminals, t_a and t_b. In image b) and c), the pixels are illustrated as 1D array for the sake of simplicity. In image c), the dashed line illustrates the cut and hence also the segmentation of the image.*

The minimization problem consists of finding the minimum cut $\mathcal{C}$ with the lowest cost, where the cost $|\mathcal{C}|$ is defined by the sum of its edges. The costs of the edges, in turn, depend on the function $E(f)$ that is to be minimized.

Although, the results obtained using the graph cut approach are satisfactory the algorithm is not further used in the workflow of acquiring data. The reason is due to the comparable results of the aforementioned k-means algorithm and its algorithmic simplicity and speed. Compare with section 6.2 for a comparison between segmentation algorithms.

6.2. Comparison of Techniques

Table 6.1 summarizes the advantages and disadvantages of each utilized algorithm:

	RG 2D	*RG 3D*	*k-means*	*Graph cut*
Approx. calc. time for an image	0	0	0	0
Easy to implement	yes	yes	yes	no
Has random factor	yes, seed	yes, seed	yes, init	no
Considers connectivity among pixels	yes	yes	no	yes
Needs parameter tuning	yes	yes	no[3]	yes[4]
Can/could consider temporal data	no	yes	yes	yes

Table 6.1.: *Comparison of the utilized algorithms. ([3] Only k. [4] The right functions need to be found.)*

6.3. Methylene Blue and Fluorescein

The presented algorithms were mainly evaluated on pictures which were taken of water with methylene blue. As mentioned before, k-means scores well under these conditions, see Figure 6.6, but if the pictures are replaced with the pictures taken of water with fluorescein all algorithms score badly. The reason is due to the immense amount of noise introduced by the fact that the pictures were shot in a dark environment. What makes fluorescein interesting is that details within the water, such as small bubbles, are visible due to the color of the water. Since the water remains mostly transparent (and luminous) bubbles generated by the impact are easily seen, by a human. The problem with recognizing them with algorithms is that the size of those bubbles is comparable to the size of noise in the image, which makes it difficult to differentiate them. Figure 6.7 shows a couple of images of the experiment with fluorescein and the resulting segmentations.

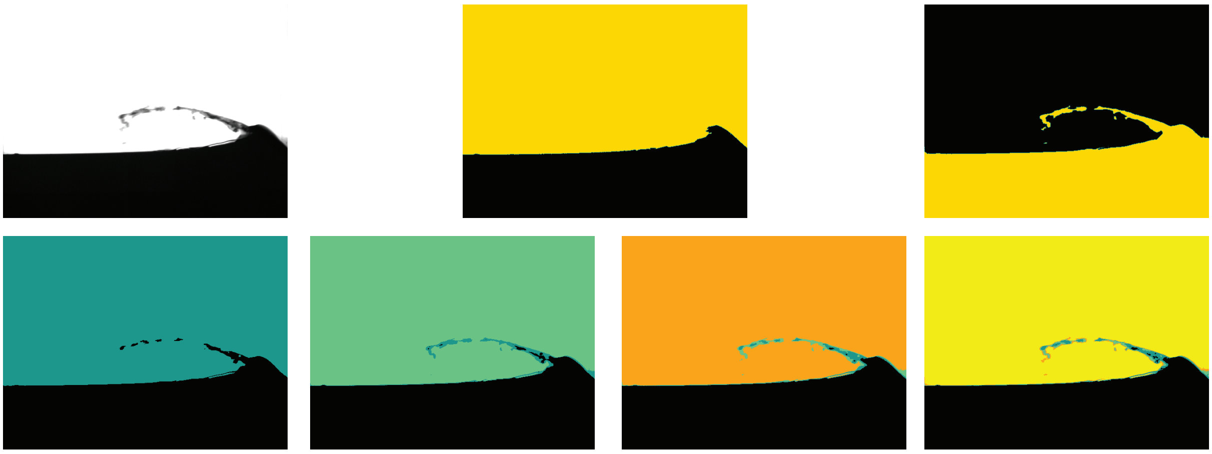


Figure 6.6.: *The top left image is the original experimental image. The other images on top are segmentations with the region growing algorithm. Once with the seed point in the fluid (central image) and once in the background. The lower images are all segmented by k-means with 2, 3, 4 and 5 classes, from left to right accordingly. The results obtained with the graph cut algorithm are omitted since they are do not show any observable difference from the k-means results.*

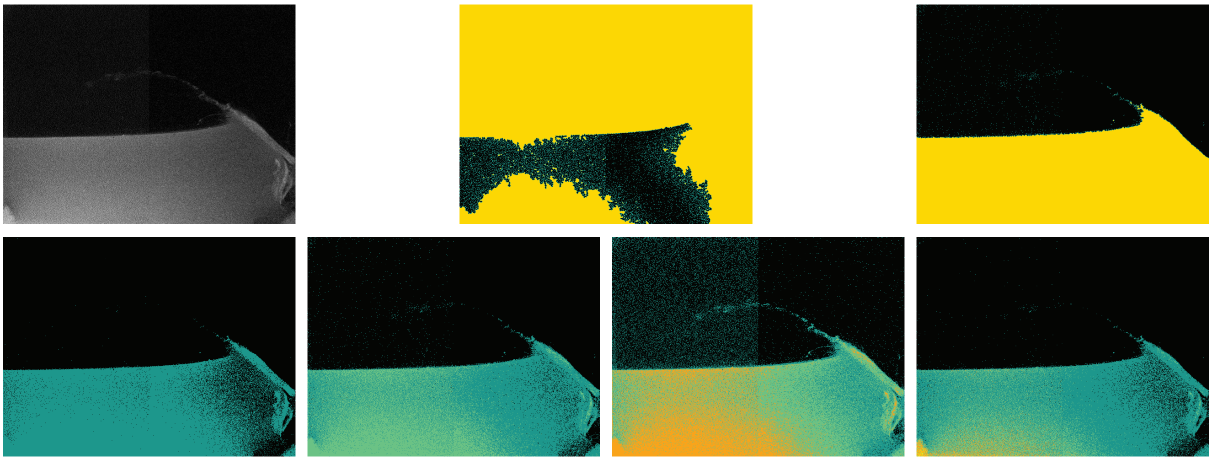


Figure 6.7.: *The alignment of the images are the same as in Figure 6.6. All examples show that the segmentation algorithms have difficulties to clearly detect the liquid. Due to noise, the segmentation results are very chaotic.*

7

Higher Semantics

With the segmentation step, the processing of the images has come to the point where the liquid can easily and unambiguously be distinguished from the background. The next step is to extract the data contained in the image and attach a meaning to it. To this end, everything that is measured and computed is in form of pixel data. This chapter presents approaches to introduce higher semantics on this data. Going one step higher than solely pixel data allows us to analyze movements over time, to find complete or incomplete trajectories. Incomplete trajectories may originate from unrecognized water drops which temporarily had a too low intensity value to be recognized or segmented by k-means. Besides completing trajectories it is also possible to remove persistent noise. This noise is not the same as image noise, which, for example, was especially encountered in the fluorescein experiments, but elements which are existent in real world but not desired. Such elements can be water drops which were created in an early stage of an impact and are adhering to the acrylic glass. If the person who is watching the sequence with such a "high level noise" knows that there is a wall between the water and the camera, then it might make sense to see them. However, if that knowledge is not present, this noise does not fit in and is bothersome.

First, tackling the problem using *optical flow* is presented. Then, the algorithmic approach based on a graph data structure into which all extracted elements of an image are pushed into is introduced. It achieves better results and is implemented in two versions: an automatic and a manual assignment.

Primarily, the contents of an image need to be extracted, meaning that all occurring elements will be identified as either main water body, air bubble or water drop, and they will grant access to their spatial and temporal positions. By applying two algorithms, which are described in Appendix A.2, these details are extracted and made accessible.

7.1. Computer Vision Approach: Optical Flow

Determining the optical flow of a sequence of images means to approximate the apparent motion of a depicted object in a scene relative to the camera. It returns a vector field of velocities for every pixel between two images. This vector field tells how fast each pixel must have approximately moved and to which direction it moved. This piece of information can be used to detect objects that are moving, i.e. flying water drops following their trajectory or non-moving objects, i.e. drops adhering to the acrylic glass. Once these objects are identified it can be decided whether to keep them if they are valid or prune them otherwise.

The basic difficulty with the optical flow approach is to identify interesting objects at all. Interesting objects should appear as a similar bunch of vectors indicating how fast intensity values have moved. Basically, two kinds of vector collections are of interest:

- The first kind are vectors indicating that an object has moved with a certain velocity into a certain direction. Unfortunately, outliers with enormous velocity values soil these collections. Outliers make it difficult to determine an automatic threshold to separate interesting vectors from the common ones.
- The other kind of vectors are those that are not moving at all, indicating that the object is adhering to the glass and is a "noise object". Since the biggest part of the main body of water is not moving, i.e. the bottom part where no intensity change appears, it is very tricky to distinguish these two cases from one another.

Optical flow was not implemented in the created software framework, but some examples were tested with a MATLAB implementation of the Lucas-Kanade-Algorithm (LKA)[1] to see some results and know what can be expected from it. Therefore, this subsection will not go much into mathematical and algorithmic detail about optical flow. Lucas-Kanade's algorithm was picked because it is one of the most popular algorithms for optical flow calculations and, hence, an implementation is easy to find.
Normally, optical flow algorithms differ in one point. They all assume a different constraint for solving the aperture problem. The *LKA* assumes the flow of the local neighborhood around the central pixel to be constant.

In Figure 7.1, some examples of the applied LKA are shown. Figure 7.1c is showing the velocity field between two subsequent images. It is important to note that the optical flow is only an approximation meaning, that after calculating the optical flow O_i from image i_a to image i_{a+1} in the sequence, O_a cannot be used to perfectly reconstruct i_{a+1} by moving the pixels of i_a [2], which is illustrated in Figure 7.1e. The orange pixels did not get a value after applying O_a. The reason is that O_a applied on i_a is not necessarily an injective function, where i_a is the preimage and i_{a+1} is the image, in a mathematical sense.

[1] cf. *An iterative image registration technique with an application to stereo vision* [LK81]

[2] trying to exploit an **in**existent equality as: $i_a + O_a = i_{a+1}$

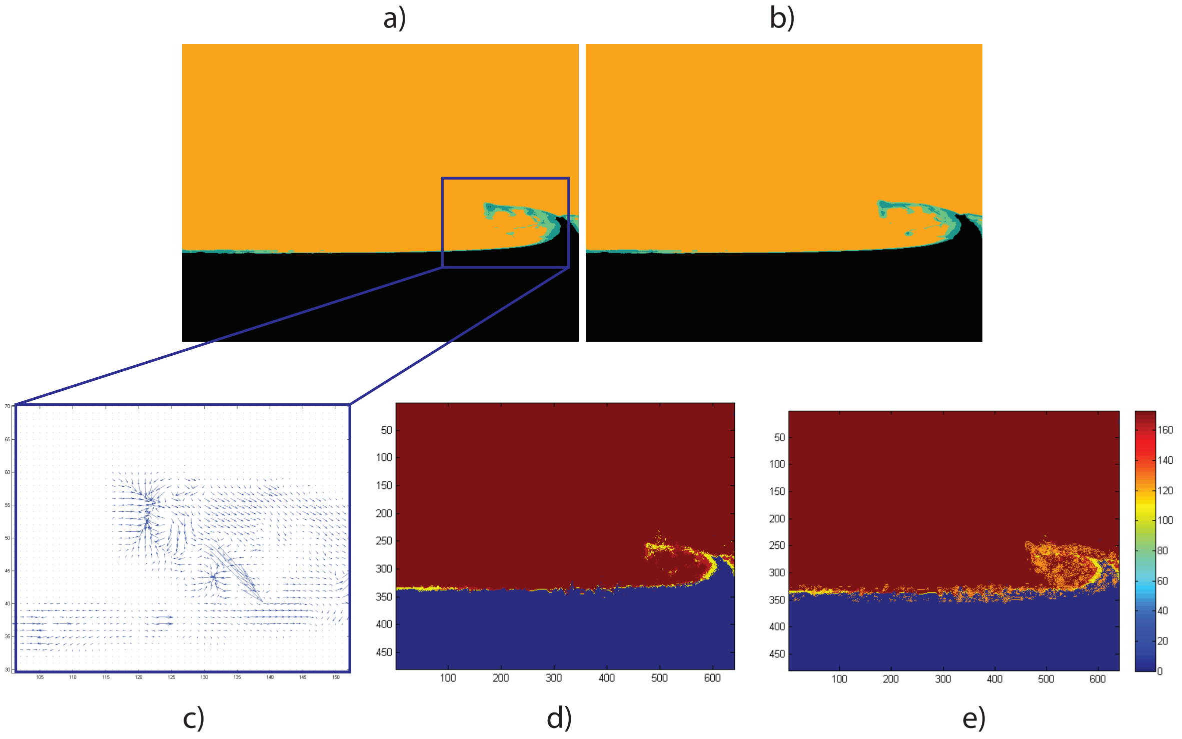


Figure 7.1.: The optical flow using the Lucas-Kanade algorithm is computed between the image a) and b). Image c) is the visualization of the velocity field of the marked rectangle. Image d) is the result of adding the velocity field from c) on the picture b). The difference between d) and e) is that the orange, noisy pixels were substituted by pixels of a) if they did not get moved by the velocity field (because the function is not necessarily injective).

7.1.1. Assumption Violation

The vector field image in Figure 7.1c looks rather chaotic. Basically, using the standard optical flow algorithms is not the ideal choice in order to track liquids. Most of the assumptions made for optical flow are here often violated and hence do not lead to proper solutions.

A water droplet, that is traversing the image, has a relatively dynamic surface and form which causes it to redistribute its water masses during trajectory. Thus, redistributing the masses also shifts the colorant forcing the droplet to look brighter or darker in certain parts.

This fact already violates the first assumption of optical flow expressed through:

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t). \tag{7.1}$$

Equation 7.1 implies that the measured pixel intensity I at the spatial-temporal coordinates (x, y, t) is the same intensity as at the coordinates $(x+\delta x, y+\delta y, t+\delta t)$. This assumption does not hold in every case when applied to liquids.

By generally assuming the displacement that needs to be tracked to be small, the following Taylor series is used to develop equation 7.1.

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + \text{ H.O.T.}$$

Additionally, making the assumption that moving liquid objects only travel a short path per frame is delicate, since water drops can move over several pixels between two subsequent images. Exactly these fast moving objects are the interesting ones and these are most likely not to adhere to the assumption.

Looking at these drawbacks of optical flow it seems that it is ill-posed with respect to the liquid tracking problem. Although, the field of optical flow within computer vision is vast and features many adapted algorithms, these were not further analyzed due to missing promising intermediate results and time constraints.

7.2. Algorithmic Approach

Analyzing the movements of liquids with mathematical methods (optical flow) turned out to be rather infeasible. Instead of relying again on pixel intensity information, the next presented approach will solely work with previously extracted positional information. Although, using velocity fields that were computed using optical flow might come in handy for assisting velocity estimations of water drops, it will be left for future work. Without having a reliable velocity information one might end up editing wrong elements of the image and possibly removing interesting features.

Everyday, one experiences how liquid behaves in certain situations. Whilst one is standing in the shower, the water that comes from the pipes goes through two states. Before reaching the shower head it is part of a greater whole, a water body. Passing through the shower head splits it into millions of little particles only to be united again when reaching the drain.
During this little journey, water drops were separated from the *main* water body, then on the way they possibly merged with another water drop forming a bigger instance, perhaps separating again, and so forth until they reach the tub floor.

This little description shows that a single droplet can have a very diverse life. These actions also happen similarly on drops (as well as on bubbles) which are evolving caused by impacts on the water surface. From an algorithmic point of view these states can be characterized as:

- connected to the main water body
- separated from the main water body

If an impact is seen through an objective of a camera, further states need to be added in order to describe all perceived actions on objects. Sometimes, a water drop is seen in frame i_a of a sequence but is missing in frame i_{a+1}, thus leading to incoherent sequence of appearances. Therefore, the new state must account for not recognized objects by the segmentation algorithm due to suboptimal conditions.

To cope with these diverse "evolution" possibilities a graph data structure is introduced; an illustration can be seen in Figure 7.2. Nodes correspond to the three classes that exist: the main water body, bubbles and water droplets. Note that in the coming sections these three classes can be referred to as "nodes", representing a general term for them. Actually, each node in the graph contains all information about its real instance, such as spatial and temporal position.
To be able to navigate and orient oneself within the numerous nodes of such a graph, two

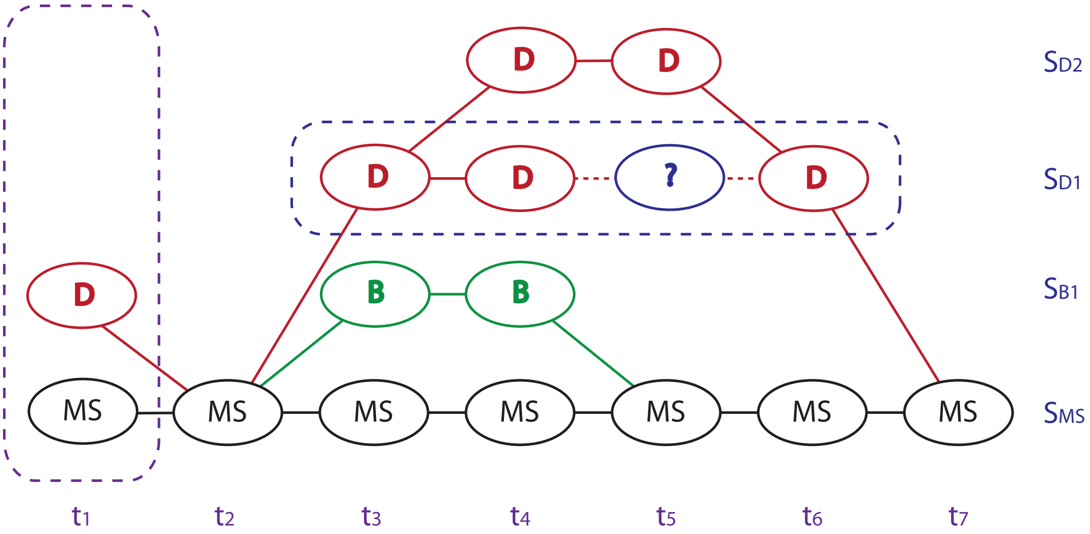


Figure 7.2.: *This figure illustrates the graph data structure used for a splash event.* ***MS****-nodes denote the main surface,* ***B****-nodes bubbles and* ***D****-nodes water drops. t_i are frame numbers and S_{xy} are stripes made out of the subscribed elements. The question mark denotes a node, which was not detected during segmentation and hence was not extracted.*

further collections are used. All pictures belonging to a specific frame are collected in *framelist* structures, enabling the differentiation of the nodes in a temporal manner. The second collection represents the trajectory of a node, by imagining them in a graph they look much like *stripes*, thus the given name of this list. Obviously, stripes can share certain nodes, for example when two water drops merge.

The challenge now is to correctly find corresponding nodes and to connect them by adding them into stripes. Two approaches were taken for that purpose: an automatic and a manual. An automatic creation of such stripes is desirable and will be explained in the next subsection. Subsequently, the subsection 7.2.2 will present the manual, more reliable, way of creating these stripes.

7.2.1. Automatic Graph Creation

The automatic approach basically goes through two steps:

- First, the nodes of a sequence of images are composed in a first stripe collection using simple categorization algorithms. By statistically analyzing this collection some valuable information is gathered such as maximum node speed and so forth.
- Second, based on the above analysis a second stripe collection is created, that replaces the one created in the first step. The gathered information from the first step narrows down the number of choices in every composing step of a stripe. Additionally, prediction algorithms are used to help to find the appropriate successor node.

The extraction algorithm presented in Appendix A.2 solely returns surface positions of water drops, bubbles and the main water body in frames. Since there is only one main water body it is trivial to create a stripe for it. Usually, finding corresponding water drops is more intricate than finding corresponding bubbles. The latter are not traversing the image in the normal case, they just appear after an impact at a certain position and usually shrink until they burst and disappear. Water drops, on the other hand, are all produced at about the same location in the image and emerge from there, split up and travel through space quite closely to each other with distinct speeds, maybe merge and interact with other drops on their way and finally join the main water body again.

Only the frame number, type of node and the position of the border pixels per node are known. With this knowledge the stripes are built. Obviously, water drops cannot change their type and vice versa, therefore when searching for subsequent nodes the search is only performed among the same type. A good assumption to make, at least for a first iteration, is to only look for nodes in preceding or subsequent frames.

First Step

The first step itself is composed of two sub-steps:

1. By evaluating a score for each possible candidate that depends on at most three functions, the successor node is determined. These functions assess the possible successor candidates by looking at different characteristics of the node, such as center of mass, size or brightness. This step is repeated until all nodes are assigned to a stripe.
2. After having all nodes distributed into stripes four values are determined: direction of movement of the splash, the maximum speed of a water droplet, the height of the main water body surface and the gravitational acceleration.

FIRST SUB-STEP

The utilized score and the three functions are discussed first. The score s_c for a candidate c is defined as a sum of weighted functions:

$$s_c = \alpha_1 \cdot f_1(c) + \alpha_2 \cdot f_2(c) + \alpha_3 \cdot f_3(c), \tag{7.2}$$

where f_1, f_2 and f_3 are the assessing functions and α_1, α_2 and α_3 their according weights.

- $f_1(c)$ assesses the distance between two nodes. Since, there is no information about the future position yet only one assumption with respect to the original position of a node can be made. The closer a candidate node in a successor frame $i + 1$ is the more probable it is to be the correct successor node of the original node.
  However, taking the closest position can go wrong in two ways:
  First, not all droplets emerge from the same pixel in the image, they are generated with a few pixels of spacing close to the position where the wedge impacts on the water surface and can rather be regarded as a moving and spreading point cloud. Additionally, if these droplets do not all have the same velocity it is possible that a droplet d_1 has taken over the position of another droplet d_2 in i+1, therefore being chosen by the function because it is closer to the original d_2's position.
  A second way of wrongly identifying can happened when for example two droplets have about the same x-position in the frame f_i, but in f_{i+1} one of them has traveled a few pixels

less far than the other one. Hence, one droplet being the closer one for both droplets which were observed in f_i.

The function $f_1(c)$ itself will now be extended to take also the current node n as input and is defined as the Euclidean distance between two nodes:

$$f_1(c, n) = \text{dist}(c, n) = \sqrt{\sum_{i=1}^{d}(c_i - q_i)^2},$$

where d is the number of dimensions, which is usually $d = 2$ for images, c_i and n_i are the according vector components of the position of a node.
In order to compute the Euclidean distance the position of a node has to be defined first. Due to their surface tension water drops and bubbles have a roundish to spheric shape which they strive to rest in. The obvious way to specify a droplet d_i's position is to take its center of mass c_i, which can be approximated by taking the average of all border points v_j:

$$c_i = \frac{\sum_{j=1}^{n} v_j}{n},$$

where border point $v_j = (x_j, y_j)$ is composed of its x and y-coordinate within the frame. Although, this is the simplest way to work with positions, it is not always the optimal choice due to deformations of droplets during their trajectory, especially when they are rather big they may split up and assume interesting shapes, for example a "C"-shape. Shapes differing from a circular form can have a center of mass outside their body, hence, leading the trajectory into a drifted off direction. But for the now, this definition of a position serves the purpose.

- The characteristic that is evaluated when using function $f_2(c)$ is the size of the node. Similarly to the preceding item in this list, the size has to be defined first as well. There are two ways to specify a node's size: taking the circumference or the area of a droplet. Neither of the possibilities are optimal since the volume of a three-dimensional water drop is being reduced to a two-dimensional projection. An unfortunate capture of an apparently small droplet might turn out to be misleading when the third dimension, which was omitted, contains the most relevant part of the volume. For the calculations, the first option is used. The circumference is defined as the number of border points (pixels) of a droplet. Hence, leading to the definition of $f_2(c)$, which is also first extended to take two arguments and return the difference of sizes of two given nodes a and b:

$$f_2(a, b) = \text{diff}_{\text{size}}(a, b) = |\#\text{border points}(a) - \#\text{border points}(b)|.$$

- The third function $f_3(c)$ assesses the average brightness value of a node. The average brightness value is defined as:

$$\text{avg}_b(c) = \frac{\sum_{j=1}^{n} c.i_j}{n},$$

where $c.i_j$ denotes the j-th intensity value of the node c. Taking the difference of average brightnesses of two nodes leads to the definition:

$$f_3(a, b) = \text{diff}_{\text{brightness}}(a, b) = |\text{avg}_b(a) - \text{avg}_b(b)|.$$

This function makes only sense for droplets but not for bubbles. Bubbles are only visible when they are not concealed by (dark) water hence making them as bright as the background.

After having defined the three evaluations functions, the score function $s_c(a, b)$ from equation 7.2 can be rewritten as:

$$s_c(a, b) = \alpha_1 \cdot \mathrm{dist}(a, b) + \alpha_2 \cdot \mathrm{diff}_{\text{size}}(a, b) + \alpha_3 \cdot \mathrm{diff}_{\text{brightness}}(a, b).$$

The current implementation omits the last function $\mathrm{diff}_{\text{brightness}}(a, b)$. Therefore, the last weighting factor α_3 drops out and the other two weights can be redefined to be dependent on each other out of practical implementation reasons:

$$\alpha_1 := \alpha \quad \text{and} \quad \alpha_2 := 1 - \alpha,$$

leading to the equation:

$$s_c(a, b) = \alpha \cdot \mathrm{dist}(a, b) + (1 - \alpha) \cdot \mathrm{diff}_{\text{size}}(a, b),$$

where $\alpha = 0.8$ was considered to return the best results.

By applying score $s_c(a, b)$ to all possible successor candidates of a node a the nodes are entered in a ranking. The "best" node in this ranking is supposed to be the successor node of a. It should be noted that both, the distance and the difference in size should be normalized to a certain interval, preferably $[0, 1]$, otherwise one measure might uncontrollably outbalance the other due to its range and the weighting factor α will be in vain.

Preferably, a certain threshold t should be included for the above measures, such that all candidate nodes are considered fulfilling

$$s_c \leq t.$$

Defining a certain t has the advantage that in case a specific droplet d is unrecognized in a frame i the algorithm is not forced to take a distant neighbor as a replacement for d in $i+1$. Taking this neighbor would degrade the trajectory result. Of course, the difficulty here is also to determine an adequate value for t.

SECOND SUB-STEP

Now, the score defined in equation 7.2 is utilized to create a first collection of stripes. This collection now contains correctly and wrongly classified stripes. Although, completely correctly classified stripes are rare, there can still be gathered information about properties of the sequence of images. These pieces of information give help to reduce the number of possible candidates and give hints about which node suits best as successor.
The following itemization discusses four important values. Some of them are needed in order to be able to apply curve fittings, which are used to predict the trajectory of water droplets. Others are useful for determining general facts about the particular splash being examined:

- **Direction of movement:** The direction of movement is determined by comparing the number of stripes evolving into the positive and the negative x direction and taking the direction of the majority. Depending on the quality of preliminary stripe classification the wrong direction can be chosen. For that case, the user can predefine this value. Since, all droplets are moving to the same direction, one can safely assume that they do not change their direction. Determining this attribute gives the advantage of knowing into which direction the algorithm has to search for successor nodes. Effectively, this attribute reduces the searching area to a half-space, whereas the current node halves the image space vertically at its position.

- **Maximum droplet x-direction speed:** For determining the *most reliable* maximum velocity in x-direction of a droplet it is assumed that the velocity in x-direction stays constant for a droplet on its trajectory. This assumption is made due to the physics of trajectory calculations of projectiles. Hence, the most reliable value is said to be the value coming from the stripe with the smallest empirical standard deviation on its velocities. Additionally, the standard deviations are weighted with the number of nodes their stripes contain. Therefore, yielding the effect that the more nodes a stripe contains and the smaller its standard deviation is, the more reliable its velocity value is considered. Some exceptions are made, though, by excluding stripes with less than three nodes. Since, these stripes do not have a standard deviation (there is only one velocity vector) and the direction of their trajectory can practically point anywhere.
  The velocities v_x in x-direction are defined as differences of subsequent droplet positions d_i:
  $$v_{x,i} = d_{i+1} - d_i,$$
  for $i = 1, \dots, n-1$ and n being the number of nodes in a stripe. The empirical standard deviation std is defined as:
  $$\mathrm{std}^2 = \frac{1}{m-1} \sum_{i=1}^{m} (v_i - \overline{v})^2 ,$$
  where $m = n - 1$ is the number of velocity vectors. In order to weight the standard deviation, std only needs to be divided by the number of stripe nodes n yielding: $\widetilde{\mathrm{std}} = \frac{\mathrm{std}}{n}$. After having determined the stripe $s^* = \{d_1, \dots, d_n\}$ with the most reliable sequence of velocities, its average velocity $\overline{v}_{s^*}$ can be calculated with:
  $$\overline{v}_{s^*} = \frac{(d_2 - d_1) + (d_3 - d_2) + \ldots + (d_n - d_{n-1})}{n-1} = \frac{d_n - d_1}{n-1},$$
  where d_i are two-dimensional droplet positions.

  Determining this attribute helps to exclude candidates with too far distances, meaning the droplet would travel with a higher velocity than the determined $\overline{v}_{s^*}$ to reach that particular candidate node. Hence, applying this constraint reduces the area to search to a circle. If the first condition (direction of movement) is included as well, then the resulting area shrinks to a half circle.

- **Main water body surface height:** This piece of information is used for determining the point of incidence of the wedge. For that, it is assumed that the first picture of a sequence does not contain a splash. Hence, the main water body is at rest. The main water body's border pixels are calculated by the extraction algorithm presented in Appendix A.2. Taking the average of these border pixels yields its height h. Additionally, it is assumed that the wedge enters the water close to the right rim of the image. Therefore, the origin of the splash being at the (mathematical) coordinates (picture width, h). Determining the height of the surface is used for a prediction step, which will be explained in the next item.

- **Gravitational acceleration:** The estimated gravitational acceleration g_e is needed, to compute a predicted position for the successor node based on the *trajectory-of-a-projectile* equation presented in the next subsection.
  Estimating g_e instead of taking the usual $9.81\frac{m}{s^2}$ is necessary due to the mismatch of unit systems. Values in our system are calculated based on pixel positions for spatial and frame numbers for temporal distances, whereas the physical units are given in meters and seconds. How many $\frac{pixels}{frame^2}$ in average does the earth accelerate the wedge?
  In order to determine the acceleration the n appearances of the wedge in the first few images are used. By measuring the subsequent positions $\{p_1, \ldots, p_n\}$ of the wedge and computing the differences the velocities v_j and accelerations a_k are estimated:

  $$v_j = p_{i+1} - p_i \quad \text{and} \quad a_k = v_{j+1} - v_j, \tag{7.3}$$

  where $i \in [1, n]$, $j \in [1, n-1]$ and $k \in [1, n-2]$. The difference vector v_j between positions p_{i+1} and p_i is also the approximated velocity vector since the difference is taken over one unit, which is one frame.
  To obtain a robust estimate for g_e the measurements need to be smoothed cleverly. The above calculations are meant to be applied for every distinct sequence of images. However, the current implementation does not support this idea yet. Nonetheless, in order to have a rough estimate of g_e a precomputed "constant" value is determined. The constant g_e can be used as a fall-back solution in case the "dynamic" estimation fails. A "dynamically" determined g_e is more desirable since it depends on pixel positions. Pixel positions, in turn, depend on the position and adjustments of the camera, which might change between two subsequent sequences of images.

  Determination of a constant g_e:

  To compute a *constant* g_e a sequence of images containing a wedge needs to be selected. Then, the following steps are taken:

  – In every image where the wedge appears separated from the water, it is recognized by using a segmentation algorithm.

  – By extracting the wedge's border pixels using the algorithm in Appendix A.2, its center of mass is determined. The center is calculated in the same way as it was explained in section 7.2.1, since the wedge is also recognized as a water drop.

  – Using the formulas in 7.3 on the center of masses approximated velocities and accelerations are obtained.

  – Due to shape irregularities of the wedge, the segmentation of it does not look the same in all frames. Hence, the border positions differ from frame to frame, which in turn dislocates the center of mass. These irregularities need smoothing in order to obtain a constant gravitational acceleration g_e. Next, MATLAB's polynomial fit function[3] is applied on the positional data with a second degree polynomial as well as a polynomial of first degree on the velocity data. On these smoothed data the formulas in 7.3 are applied to determine the acceleration g_c.

[3] POLYFIT(x, y, degree)

Second Step

The first step creates a first categorization of the droplets and bubbles and provides the above described information. In a next step, the collected statistics are used to create more reliable stripes than before. The advantage of using these gathered information is to help to reduce the number of choices and to help to choose the right successor or predecessor of a node in a stripe. Due to good assignment of bubbles into stripes in the first step only water drops undergo the second step. The good assignment of the bubbles is based on two reasons:

- Bubbles tend to move rather slowly during a splash. This property provides reasonable results by using the distance function defined in section 7.2.1.
- Also, their quantity is not as large as the droplets'. Therefore, there are not as many competitors for a successor position of a bubble as it is the case for water drops.

The second step of the automatic graph generation can be roughly summarized into three sub-steps:

1. Choose a droplet and calculate its successor in the next image by using the four determined estimates in the previous step of the automatic graph generation. If there is no successor close to the predicted position, add a *fake* droplet based on the assumption that the water drop on this position must have been unrecognized.
2. Having two nodes already, a third node position can be predicted using the same prediction method as before but with the a different start position than before.
3. This sub-step requires to already have three nodes in a stripe in order to be able to use Least Squares (LS) fitting and predict future droplet positions. This step will be applied until the predicted position reaches the main water surface or it is out of the frame range.

Here, it will be elaborated on the single sub-steps to give a more detailed image of the happenings in the second step of the automatic algorithm. All sub-steps include a part where a predicted water drop position is computed. Inevitably, these positions are afflicted with an uncertainty, thus there will rarely be a water droplet at this exact predicted position. To account for this uncertainty a check is performed. Before a predicted position is inserted into a stripe as a *fake node*, it is checked whether there is any *real* node, i.e. a water droplet, that is intersecting with the fake node. If that is the case, the closest intersecting node is assumed to be the one we are looking for. Otherwise, it is assumed that the water drop at the examined position was not recognized and a *fake node* with the predicted position is inserted into the stripe.

1. **First and second node:** The first water drop that is inserted in a stripe becomes the first node of the stripe. The stripes are constructed in such a way that the first nodes are taken from images with the lowest possible frame number. Then the following nodes are taken from subsequent images. Therefore, there is always a first node unless there are no water drops left to put in stripes in which case the algorithm is terminated.
   Determination of the second node is a critical point in the algorithm since it decides to which direction the trajectory of a stripe is pointing. To determine the second node advantage is taken of the four before calculated values. The entry point of the wedge is assumed to be the origin of the droplet. The entry point was calculated before when the *main water body height* was computed to be at $(picture\_width, h)$. With the entry point

and the first node of the stripe two points of the trajectory are available. Exactly these two points are sufficient to be able to apply the *trajectory-of-a-projectile* (TOP) equation, where the water drops are assumed to follow a trajectory of projectile.
The TOP is used twice: First, to calculate the angle β at which the droplet is "launched". Then, by using β to predict the position of the droplet at frame $t+1$:

$$r(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} v_{0,x} \cdot t \cdot \cos(\beta) \\ v_{0,y} \cdot t \cdot \sin(\beta) - \frac{gt^2}{2} \end{pmatrix}, \tag{7.4}$$

where $r(t)$ is the vectorial trajectory-of-a-projectile equation parametrized with the time t (frame number). To determine β there are two equations available. Both are used and the average is taken as long as the values are valid, i.e. not infinite or undefined. This leads us to the following equations to determine β_1 and β_2:

$$\beta_1 = \arccos\left(\frac{x(t)}{v_{0,x} \cdot t}\right) \quad \text{and} \quad \beta_2 = \arcsin\left(\frac{y(t) - \frac{gt^2}{2}}{v_{0,x} \cdot t}\right),$$

Here is where two further pre-calculated values are used. v_0 is the launch velocity, which is assumed to equal the maximum velocity of a water drop. g, the gravitational acceleration, is assigned the estimated value g_e. The missing parameter t is determined by node's position n_{pos} that is already in the stripe and dividing it by the maximum droplet speed v_{max}: $t_{\text{first}} = \frac{n_{\text{pos}}}{v_{max}}$.
Hence, the position of the *second* node in the stripe is computed by determining its "temporal" position $t_{\text{second}} = t_{\text{first}} + 1$ and plugging it in into equation 7.4.

2. **Third node:** The third node position is determined in the same way as the second node, except that the "launch position" of the droplet is not the entry point of the wedge but the first node of the stripe.

3. **Forth and further nodes:** Having obtained the third node a different prediction method is utilized, which only considers the actually chosen point in the stripe without using any pre-computed values such as gravitational acceleration and so forth.
The trajectory of a water drop in this setup can physically be described by the TOP as mentioned before. The equations in 7.4 are parametrized. The x-component is a linear polynomial whereas the y-component is described by a second order polynomial with respect to t. Due to measuring inaccuracies and additionally the inaccuracy of determining the center of mass of a water drop it is reasonable to approximate the trajectory by fitting a polynomials into the data points, i.e. the node positions. To predict the forth or further nodes the x-components are fitted by a Linear Least Squares (LLS) method whereas the y-components are fitted by a Quadratic Least Squares (QLS) solution:

$$\min \epsilon(b_1, b_2) = \min\left(\sum_{i=1}^{n}(b_1 + b_2 t_i - d_{i,x})^2\right) \tag{7.5}$$

and

$$\min \epsilon(c_1, c_2, c_3) = \min\left(\sum_{i=1}^{n}(c_1 + c_2 t_i + c_3 t_i^2 - d_{i,y})^2\right), \tag{7.6}$$

where b_i and c_i are the coefficients to determine, $d_{i,\{x,y\}}$ the available water drop locations in a stripe and t_i the according frame numbers. The solution is found if the error ϵ is minimal for a particular configuration of the coefficients b_i and c_i. In the current implementation the equations are solved using an according method provided by the ALGLIB *project*[4].

The LS equations are formulated in a parametrized way due to the easiness of determining the predicted positions. Once the coefficients b_i and c_i are determined one only needs to plug in the desired frame number t to obtain a new position for a node. Calculating the positions in this fashion is reasonable since the movement in x-direction follows a linear and the movement in y-direction a quadratic equation. This physical property is reflected in the TOP equation 7.4.

7.2.2. Manual Graph Generation

A human-supervised algorithm still performs superior to an algorithm that automatically tracks and recognizes objects moving through a sequence of images. By exploiting this fact a better result for connections in the graph can be achieved instead of processing the image sequences with the automatic approach described in section 7.2.1.

The manual approach in a nutshell:

1. The main water body is extracted automatically into a stripe as it was also the case in the automatic approach.
2. The user selects a water drop or a bubble by clicking on an according pixel. Selecting a pixel causes the closest object to that pixel to be added to a stripe (Stripes are visible in check box lists).
3. The user needs to continue to choose two more objects of the same type in the subsequent frames.
4. From now on the LS approximation, introduced in section 7.2.1, guides the user by visualizing predicted positions of objects, as shown in Figure 8.6. The user can decide whether she wants to select an appropriate object in the frame or if no choice is fitting a fake node can be added (cf. Figure 8.6). This step can be repeated until the last frame or the user decides that the objects journey has ended.

In every user-specified step except the first one, the user has the choice of adding a new object, adding a fake node to a stripe, or removing an existing node from a stripe.
The prediction can be useful in case a water drop was not recognized in an intermediate frame by giving the user a hint of the missing node's central position, see Figure 8.6 for an example. The goal of the graph data structure, seen in section 7.2, is to give the opportunity to adjust incomplete trajectories. Therefore, in frames where a node is missing it can be added at a user-specified position, possibly aided by the prediction mechanism. With each new node that is added to a stripe, the prediction is updated showing the future positions according to the currently selected nodes.

[4] http://www.alglib.net

Two different sorts of prediction methods are used depending on the type of object:

- If the objects to add to a stripe are water drops, the above mentioned LS is used where the x-component of the location is fitted using a linear LS and the y-component using a quadratic LS.
- Otherwise, if the objects the stripe contains are bubbles linear LS is used for both components. Bubbles succumb stronger to the buoyancy effect than the impact of the gravitation. The buoyancy forces the bubbles to follow a rather linear movement than a quadratic as it is the case with flying water droplets.

8

Results

At this point, some intermediate and final results will be shown. First experiments were conducted with a small ordinary fish tank as it was explained in chapter 3. Although these examples were taken only to get familiar with the camera and equipment it is still interesting to see the differences between the images at that stage and the final results.
One part of the results of this is the tank itself. It was part of the thesis requirement and consumed a lot of time, because it had to be built first in order to conduct any experiments. There have been enough pictures of it during the past chapters therefore pictures of it will be omitted here.

8.1. Fish Tank Try Outs

As mentioned before these images are added to have a comparison between the first, rather bad setup and the final pictures. The pictures in Figure 8.1 are rather dark which is due to the bad illumination, but for demonstration purposes the contrast of the pictures was adjusted. The background is not transparent and therefore there is no light in the back that could create a high contrast between itself and the dark liquid.

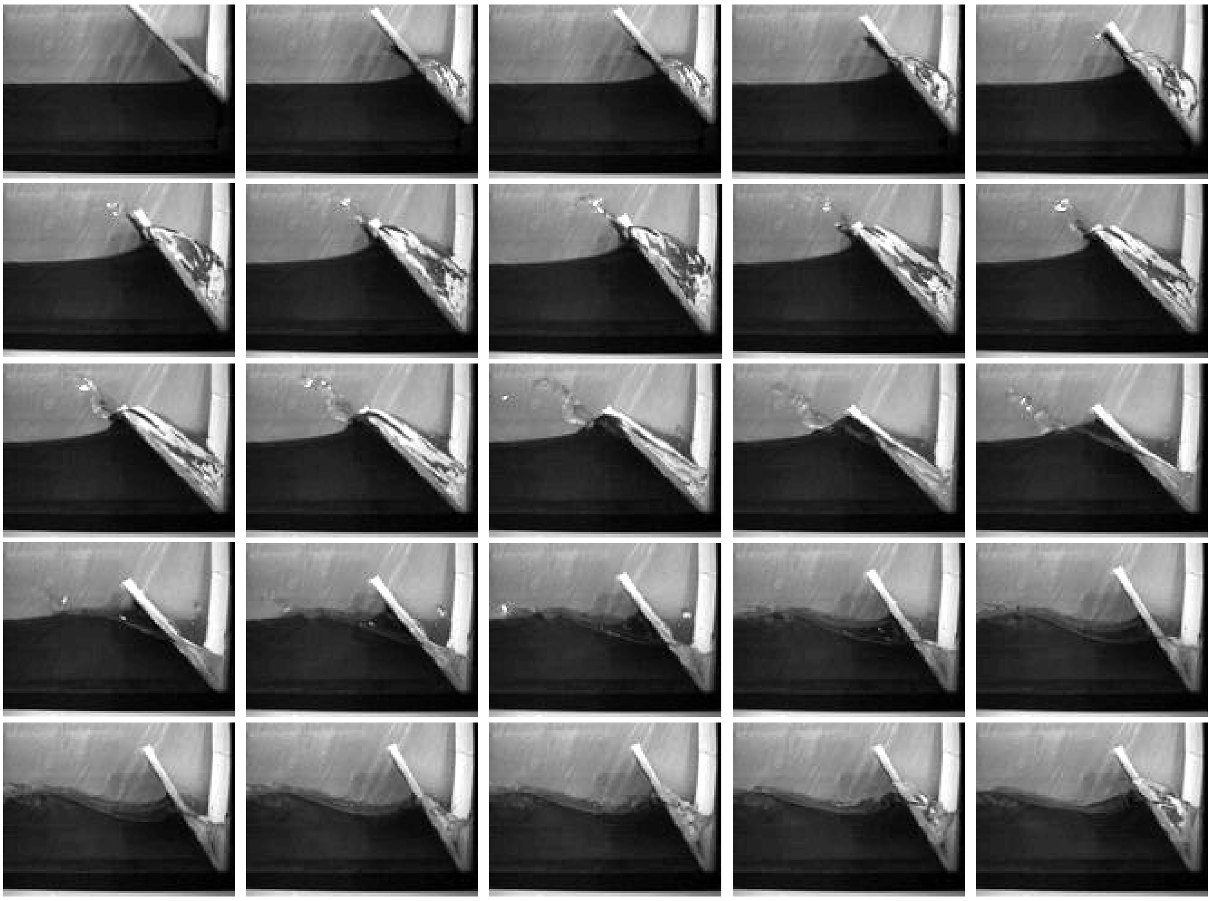


Figure 8.1.: *A sequence of images which were taken from an experiment in a fish tank. The depicted pictures show every second image of the real data, effectively with 100 frames/second. Contrast adjustments were applied.*

8.2. Methylene Blue and Fluorescein

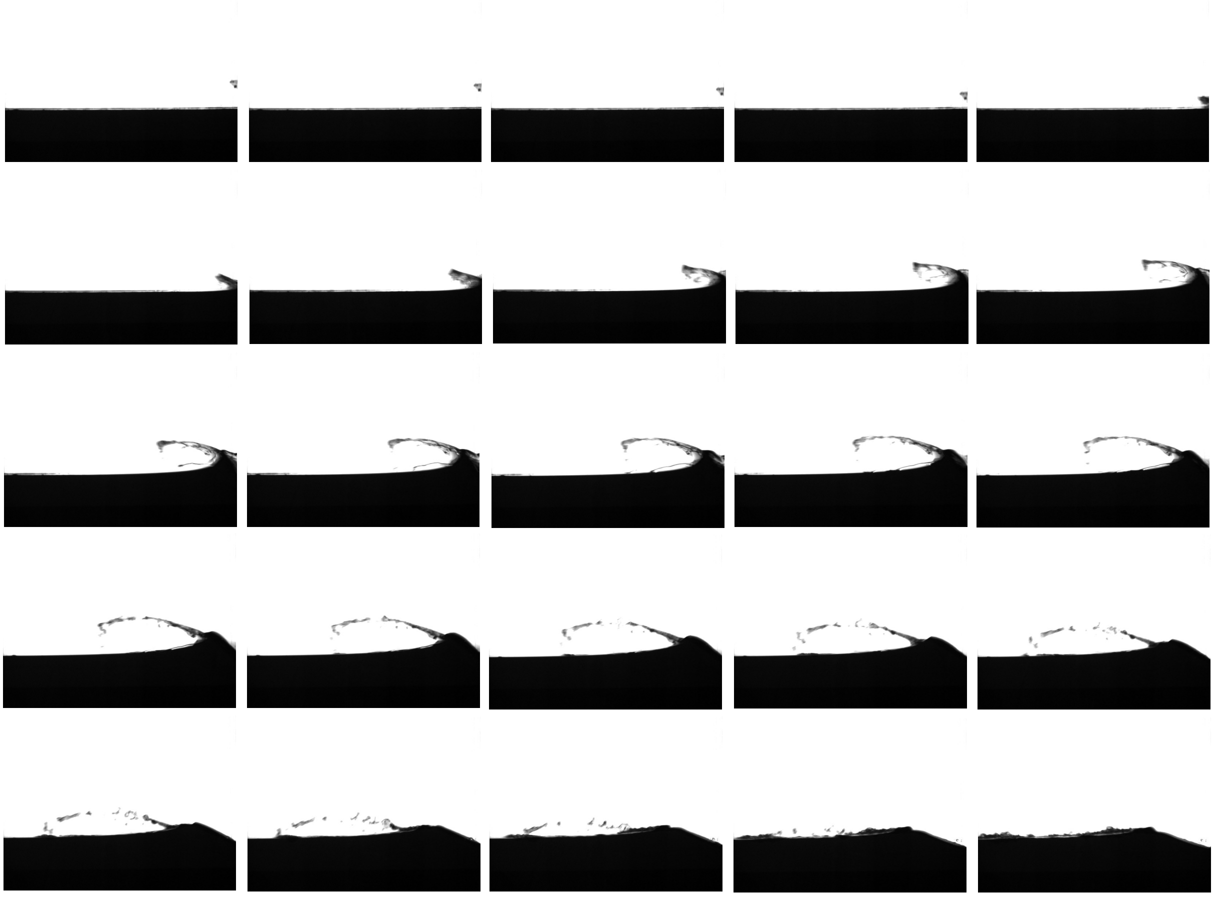


Figure 8.2.: These images show an example of the experiments conducted with methylene blue and a flat screen. Clearly, the contrast is a lot better than the contrast in the fish tank experiments. No contrast adjustments were applied for these images.

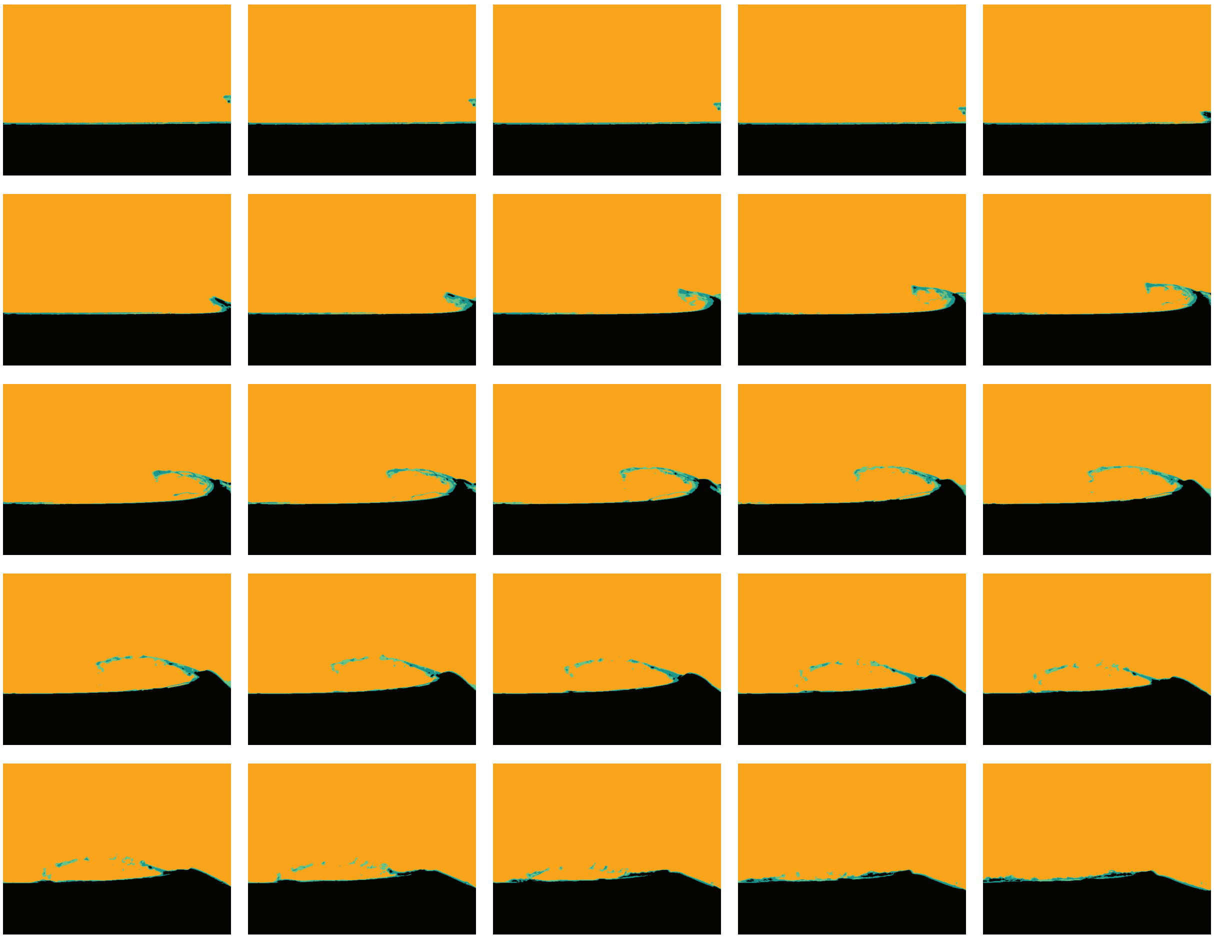


Figure 8.3.: *These images show the segmentation of the sequence in Figure 8.2 with the k-means algorithm, where is $k = 4$.*

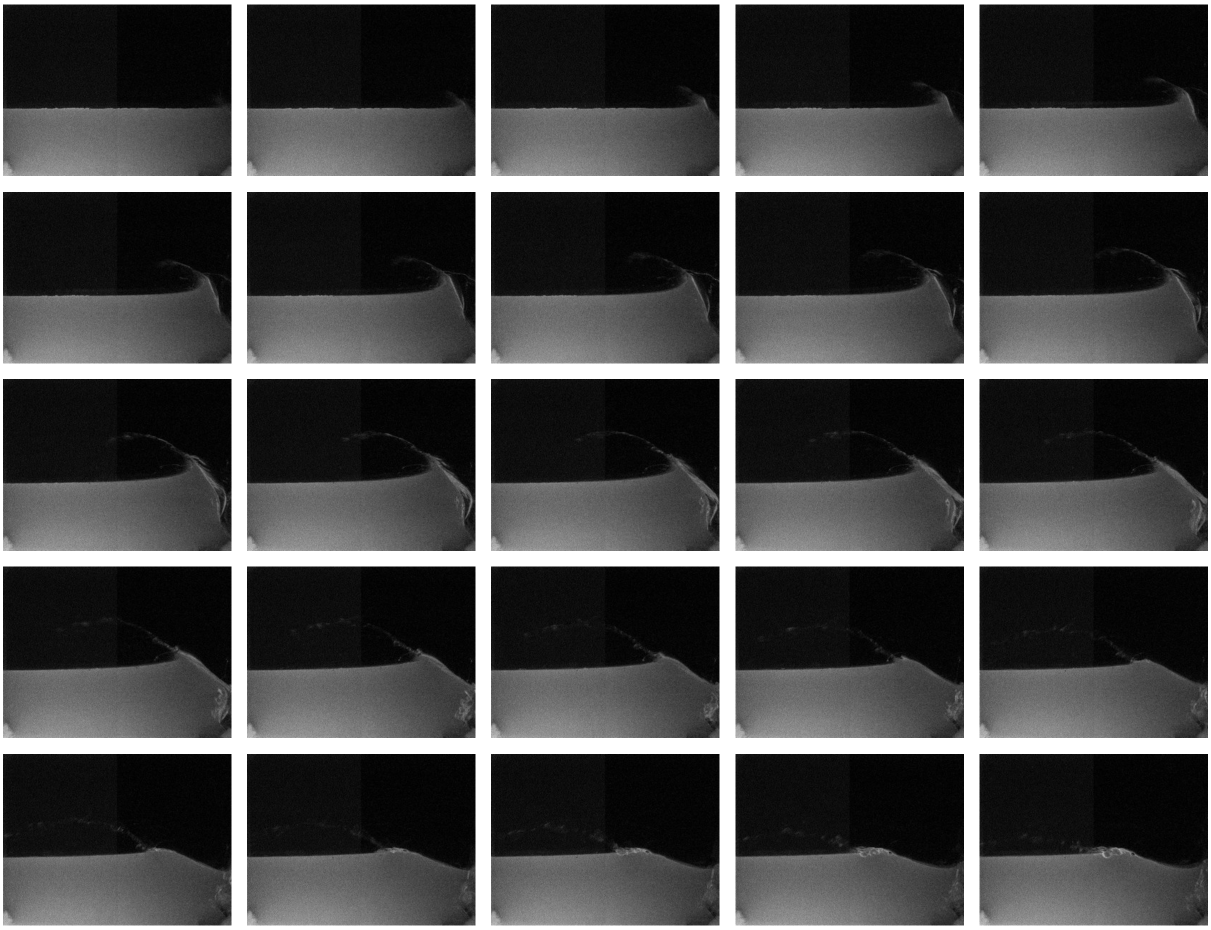


Figure 8.4.: *This sequence shows the results of an experiment conducted with fluorescein. The images are rather dark which is also the reason for the large quantity of noise. However, close to the wedge's entry point small details such as bubbles are visible, which is not the case with the methylene blue experiments.*

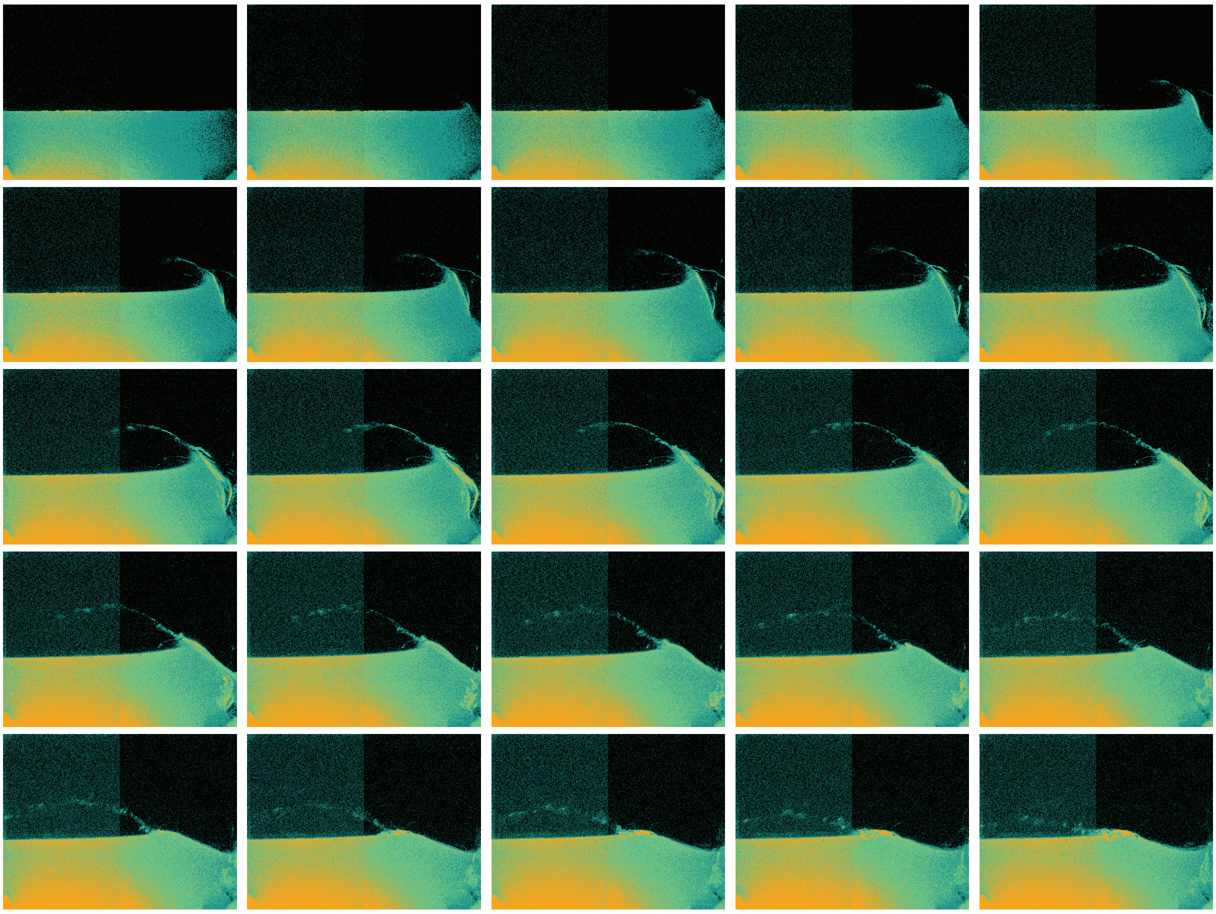
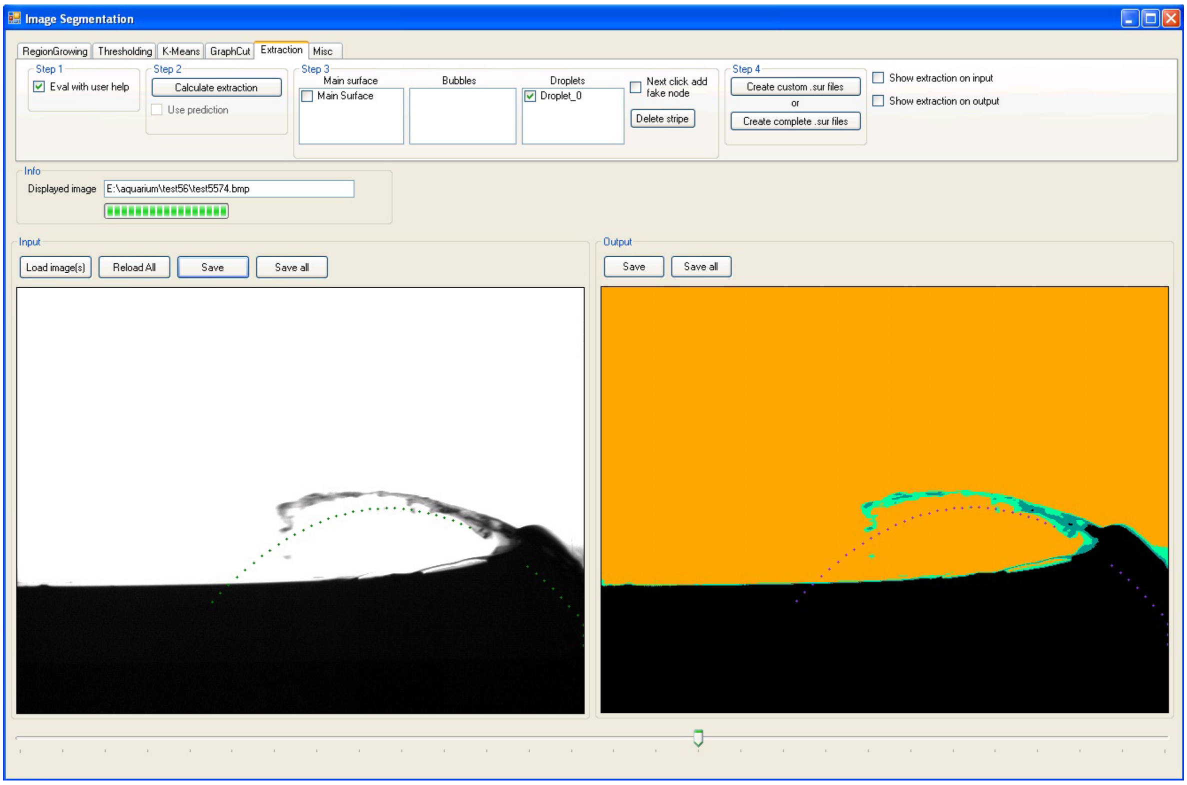


Figure 8.5.: *Accordingly, these images show the segmented results of the fluorescein experiments. The water does not have a uniform labeling due to noise. Here, the issue mentioned in section 4.3 can be observed. The left side of the images show distinctly more noise than the other sides.*

8.3. Giving a Higher Meaning



☑ Eval with user help

□ Use prediction

□ Main Surface

☑ Droplet_0

□ Next click add fake node

□ Show extraction on input

□ Show extraction on output

Displayed image E:\aquarium\test56\test5574.bmp

Figure 8.6.: *This picture shows the software that was developed in this thesis. It shows the extraction tab where segmented pictures can be loaded and processed to connect the elements, such as bubbles and droplets, in a graph data structure. Using the manual graph generation explained in section 7.2.2 so-called stripes can be created. The original and the segmented picture in the Figure show prediction curves that help the user to choose the next node in a stripe.*

9

Conclusion and Future Work

9.1. Conclusion

This work presented a system of devices and algorithms to examine two-dimensional splashes of liquids. The detailed construction of a tank capable of housing the liquid was described. Additionally to the tank, a wedge-shaped apparatus with a guiding system was shown that produced the desired splashes. The wedge is not "stiff" as in the original sense mentioned in section 2.1, but adjustable in size and aperture angle in order to produce a variety of splashes.
The utilized high-speed camera posed a challenge to the illumination of the scene due to which different inexpensive illumination techniques were examined and the most suited solution, a flat screen, was chosen.
In order to improve the contrast between liquid and background various colorants were inspected whereas methylene blue was deemed to be the most adequate. A second approach to capture splashes by changing illumination and colorant was presented as well. By adding a fluorescent substance to the liquid, namely fluorescein, and lighting the substance with a ultraviolet illumination further experiments were conducted.
To transform gray-scaled images into binary images, where the pixels representing the liquid have a certain value and pixels representing the background have the opposite value, distinct segmentation algorithms were examined and assessed. Ultimately, the k-means cluster segmentation algorithm was selected to be incorporated into the workflow.
Finally, two algorithms were presented that assign the sequences of images a higher semantics. Higher semantics in the sense that working with objects is facilitated instead of struggling only with pixel data. By utilizing a graph data structure the means were given to distinguish and track elements of a splash such as the main water body, bubbles and water drops.

This thesis occupied itself with a broad spectrum of subjects. To begin with, a great deal of the work was not even about computer science but the construction of a tank. Hence, not only mental work was demanded but also handicraft. As a student, one is usually not used to work manually. That is also why a bit more time should have been spent on researching about building water tanks. Although, investigations were carried out about aquariums the wall thickness of the tank was completely underestimated for the first tank, which also resulted in an unsuccessful first attempt. Also, when one thinks about large aquariums in general, one notices that there are rarely any screws used to fortify the walls. The intuition about fortifying the edge is misleading and is also the reason why aquarium forums were consulted rather late in the construction process. Fish tanks can be built without using any screws but for it to hold the thickness, again, needs to be correctly calculated. Due to the overestimation, the physical attributes of the materials were not checked if they matched this setup, i.e. if they would stand the pressure of the water. One aspect that was underestimated, is the incredible amount of water pressure produced by a water column of a few ten centimeters.

The broad spectrum of this thesis had both, advantages and disadvantages. Many topics were covered which gave one sort of an overview of many things, but due to time constraints not as much time as desired could be spent on every topic, resulting in a rather coarse knowledge in the specific fields. This factor showed an effect on the chosen approaches and algorithms in that the simpler approaches were preferred to sophisticated ones. Two reasons supported this behavior: Intricate algorithms usually need more time until they are completely understood, such as the graph cut algorithm mentioned in chapter 6.1.4, otherwise they might be used in a wrong fashion. Especially, this particular algorithm was a complex one to implement, because it uses certain functions which need to be minimized. These functions are essential to the algorithm's success. It turned out that choosing the functions badly has a heavy effect on the result, vice versa with a good choice. The paper [BVZ01] did not mention specific implementation details which impeded on finding an appropriate function without spending much more time on this topic.
Another aspect that supported preferring simpler algorithms was the implementation. Usually, one does or should not entirely trust implementations that can be found on the Internet and which were programmed by random people, i.e. not the authors of these algorithms. In doing so, one might not achieve the result that is promised to appear. Unless the software one finds is reliable, it was tried to implement algorithms according to publications and their general ideas. Also, sometimes the desired piece of code is not available in the desired programming language which will mean either to switch between programs to achieve certain goals which is cumbersome in the end, or to re-implement the code, which costs scarce time. Often, it was even unobvious whether an approach works on the given problem, which led to a lot of testing of different encountered implementations.
Every step towards the end of the thesis offered a lot of choices. Starting from the material of the tank, to form and size, over choice of lighting and segmentation algorithms up to the tracking gave lots of paths which led to the objective. However, the challenge was to filter suboptimal choices as good as possible.

Despite the vast amount of difficulties and issues, usable results were produced as presented in chapter 8. In fact, these difficulties made this thesis very interesting and diverse. An undeniable critical point in this thesis was the successful construction of the tank which is still stocking the water it should stock.

The depicted images in Figure 8.2 show the results obtained using methylene blue dissolved in water with a flat screen in the background. A high contrast is achieved using this combination of illumination and colorant and the segmentation is facilitated. Most of the airborne water drops are still visible although the thickness of the matter through which the light has to travel is much thinner than the main water body. However, there are still some droplets which disappear for a couple of frames or which totally disappear and are only indirectly visible by deforming parts of the main water surface. Disappearing water drops result in tracking issues where fake nodes have to be introduced to achieve temporal coherence of a water drops.

Basically, it is a very interesting idea to use a fluorescent substance. The advantage, as compared to the liquid with methylene blue, is that the liquid itself remains almost transparent which allows to identify details within the liquid, i.e. small bubbles. However, as one can notice there is a high occurrence of noise in the images impeding the quality of the segmentation and therefore the usage of fluorescein is not (yet) suitable for analyzing 2D liquid phenomena.

As seen in Figures 8.3 and 8.5 the k-means algorithm only works well if the there is no excessive noise, as it is the case with the fluorescein data.

It turns out that the automatic tracking is a very difficult problem. The implemented automatic methods only work limitedly well. Therefore, the manual graph generation is provided to be able to work and further process the collected data. The user is aided with a well working prediction system, shown in Figure 8.6.

Concluding, one can say that although there have been several difficulties in the course of this work the possibilities to produces, record and process 2D liquid phenomena are provided with the tools of this thesis.

9.2. Future Work

There are a couple of points that could be further explored within every part of this work. Construction-wise the tank could be made narrower such that the depth of the tank would only measure a few millimeters or a centimeter at most making the fluid inside look as two-dimensional as possible. By only using adhesives there would be no need for screws or any metal frame. The challenging part would consist of building an appropriate wedge.

The idea to use UV light in combination with fluoresceine is a very interesting and exciting approach. Unfortunately, the presented results are not very convincing. Nevertheless, experiments could be conducted by using stronger light sources animating the fluoresceine to emit a larger amount of visible light. This way image quality could be improved. Another source of noise to reduce would be to exchange the current camera with a camera that does not entail the difference of noise level in the halves of the image as explained in section 4.3. One problem with the current UV lights is that they do not only emit light in the UV spectrum but also in the visible spectrum. Hence, the camera perceives the light sources as well. The issue could be solved by finding such UV light sources only emitting light in the desired spectrum. Additionally, a band filter specially matched to the fluoresceine's emitted light could be used forcing the camera only see the light coming from the fluoresceine in the water.

Instead of using water, one could imagine to experiment with different fluids such as milk or oil. Acquirement, storage and disposal would probably be issues to tackle. However, these fluids have different physical properties and would hence result in slightly different observed behaviors than water.

In this work, the k-means clustering algorithm was observed to return good results. However, faint water drops or weak air bubbles are still difficult to separate from their backgrounds. More sophisticated algorithms might help to assign the right label classes to these weak elements of a picture. The examined graph cut algorithm performed as good as k-means as was presented in section 6.2 and 6.3. However, to this end it remains unclear if the components of the energy function $E(f)$ in equation 6.1 are *the* optimal choice and therefore whether the graph cut algorithm could perform better. Elaboration on this subject might help to clarify.

The 3D region growing algorithm presented in chapter 6.1.2 is an interesting thought to work on. It assumes that the whole splash is generated out of one water body, which is very reasonable. The advantage over k-means is that the connectivity of pixels are considered. Hence, pixels not belonging to the same liquid body but having a similar intensity value are less likely to be labeled the same. However, to make this approach work a camera with a higher frame rate and an appropriate lighting system is needed to enable the water bubbles to be closer to each other in subsequent frames.

Most of the extensions and improvements could be performed in tracking and semantics:
The automatic graph generation algorithm presented in section 7.2.1 consists of good ideas, but probably fails due to inaccuracies in the four pre-determined values. These four values are key parts of the described algorithm and depend on their accurateness.
There are many little adaptations that could be tested and incorporated into the automatic algorithm. However, only the following will be presented: The stripes in the graph data structure are currently created in a depth-first manner. An element of an image of a sequence (water drop or bubble) is chosen to be the first node of a stripe. Subsequent nodes are directly added to this very stripe until it is considered complete. Instead, the stripes in a frame could be built concurrently in a breadth-first manner. Some sort of voting or ranking system could be deployed to distribute the available nodes of a subsequent frame to the according stripes.

Besides improving existent algorithms, the framework of Blair and Dufresne[1] could be examined. Unfortunately, we only became aware of Blair and Dufresne's work in a late phase of this thesis. Their provided framework looks promising and is written in MATLAB though. Nevertheless, it could be incorporated into the current framework written in C# built for this thesis. Blair and Dufresne try to predict trajectories of particles by analyzing the pictures with the aid of statistical methods. Particles are usually small in size, but with some adaptations to the algorithm it could made running for water-drop-sized particles.

[1] The MATLAB Particle Tracking Code Repository, [BD]

A

Appendix

A.1. Workflow

To have an actual idea about how any picture was taken and what had to be done to obtain them, a short sketch of the workflow is given below. The workflow includes the usage of a software especially built for this work. The sketch may deviate depending on what one desires to record, but the most common way is enumerated here:

1. Position the tank on a solid table and fill it to the half with water (approximately *10 liters*).
2. Place a big enough flat screen monitor behind the tank, as close as possible to the glass.
3. Add a small portion of methylene blue into the water such that the emission of the monitor in the back does not shine through the liquid anymore.
4. Position the camera in an appropriate height and distance orthogonally in front of the tank, such that the camera captures most of the desired area.
5. Set the wedge and the pin at an appropriate height.
6. Close the tank's lid, position it as close as possible to the wedge, with just enough room for the wedge to pass through.
7. Start the capturing software and choose the modus where the camera records 200 frames per second.
8. Two to three seconds of recording should usually be enough time to capture the impact. Press the start button and release the wedge.
9. Start the segmentation software and load the pictures that were just recorded.

10. Choose the number of classes (usually four[1]) in the "k-means" tab and segment all pictures. Pay attention to the result and repeat this step if the segmentation looks wrong.
11. Change to the "Extraction" tab and choose whether you want an automatic or manual extraction. The manual extraction will be explained because the trajectories need to be created by hand. If the automatic extraction is chosen, skip the points below except number 16.
12. After pressing the extraction button, one main surface should appear in the check boxes.
13. Choose a bubble or water drop that you want to chase through the frames by clicking on it in the segmented picture. If it is not clear whether a node was recognized, click the "Show extraction" check boxes.
14. To add a new node to a stripe that *already* exists it must be enabled in the check box list. By clicking a second time on a node that has been added, it can be removed.
15. To create a new stripe uncheck any check box and click on a new node in the segmented image.
16. Once all desired bubble and droplet stripes are produced, create your custom surface files by clicking on the appropriate button.

A.2. Extraction

This algorithm is used to extract the positions of main water body, air bubbles and water drops of an image that is segmented into two classes. It is a two-phase algorithm and assumes that the main water body reaches from the right side of the image to the left side.

- **1st phase:** The algorithm goes pixel by pixel through the image and separates all encountered elements in a top-bottom left-right manner. By separating, every element gets its own bit texture. For main water body and water droplets these bits are set to 0, if the pixel represents air and 1, if it represents water. For air bubbles (also the main air body) the texture is inverted (water $= 0$ and air $= 1$).
- **2nd phase:** Every bit texture that was generated before gets examined. The main surface is extracted out of the main water body by following the edge between air-bits and the water-bits. Similarly, the border points of water drops and air bubbles are extracted.

A.2.1. 2nd Phase in Detail:

This step works for main water body, air bubbles and water drops almost the same. The difference is the termination condition:

- For the main water body the algorithm starts from the right side of the picture and follows the surface until it arrives at the opposite side.

[1] The more classes are chosen, the finer are the boundaries between classes. Finally, k classes can be merged into 2 classes representing the liquid and the background.

- For bubbles and droplets, the algorithm terminates when it reaches its starting position. This condition is based on the assumption that bubbles and droplets have closed shapes.

This algorithm returns a list L of positions (vectors). These positions indicate the border between the elements (air and water). Positions in this algorithm are not the pixel position itself, but the lower left corner of the pixel, such that a coordinate (x, y) actually lies between four pixels. Besides the bit texture, the first phase also includes an entry point which is the lower left corner of the first pixel in the texture:

1. Add the pixel to the list L.
2. Perform the calculated step. If it is the first step in the algorithm, step in direction $(1, 0)$[2].
3. Adapt the local coordinate system according to the moved direction.
4. Judge the situation:
   - If the left pixel equals 1, turn 90 degrees to the left.
   - If the left pixel equals 0 and right pixel equals 1, go straight.
   - If the right pixel equals 0, turn 90 degrees to the right.
5. Determine step and direction according to the situation.
6. Go to $1.$ as long as the start position is not found (or in case of the main water body, the right side of the image is reached).

[2] This step always works due to the way the elements are encountered in the first phase.

Bibliography

[AB94] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):641–647, 1994.

[AKS07] A. Alam, H. Kai, and K. Suzuki. Two-dimensional numerical simulation of water splash phenomena with and without surface tension. *Journal of Marine Science and Technology*, 12(2):59–71, 2007.

[Bag06] Shai Bagon. Matlab wrapper for graph cut, December 2006.

[BD] D. Blair and E. Dufresne. The Matlab Particle Tracking Code Repository. http://physics.georgetown.edu/matlab/.

[BK04] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, September 2004.

[BVZ01] Yuri Boykov, Olga Veksler, and Ramin Zabih. Efficient approximate energy minimization via graph cuts. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 20(12):1222–1239, November 2001.

[CGML09] G. Colicchio, M. Greco, M. Miozzi, and C. Lugni. Experimental and numerical investigation of the water-entry and water-exit of a circular cylinder. *IWWWFB*, 2009.

[GL83] M. Greenhow and W.M. Lin. Nonlinear-free surface effects: Experiments and theory. *MIT, CAMBRIDGE, MA(USA), 1983, 101*, 1983.

[Gre87] M. Greenhow. Wedge entry into initially calm water. *Applied ocean research*, 9(4):214–223, 1987.

[HS81] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Computer vision*, 17:185–203, 1981.

[IB98] M. Isard and A. Blake. Condensation-conditional density propagation for visual tracking. *International journal of computer vision*, 29(1):5–28, 1998.

[IGM05] I. Ihrke, B. Goidluecke, and M. Magnor. Reconstructing the geometry of flowing water. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, volume 2, 2005.

[JTP04] C. Judge, A. Troesch, and M. Perlin. Initial water impact of a wedge at vertical and oblique angles. *Journal of Engineering Mathematics*, 48(3):279–303, 2004.

[KFVI04] K.M.T. Kleefsman, G. Fekken, A.E.P. Veldman, and B. Iwanowski. An Improved Volume-of-Fluid Method for Wave Impact Problems. *ECCOMAS*, 2004.

[KOK$^+$08] H.D. Kang, S.H. Oh, S.H. Kwon, J.Y. Chung, K.H. Jung, and H.J. Jo. An experimental study of shallow water impact. *IWWWFB*, 2008.

[KWT88] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International journal of computer vision*, 1(4):321–331, 1988.

[KZ04] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? *IEEE transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159, February 2004.

[LK81] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International joint conference on artificial intelligence*, volume 3, pages 674–679. Citeseer, 1981.

[LSLD03] M. Li, I.K. Sethi, D. Li, and N. Dimitrova. Region growing using online learning. In *Proceeding of the International Conference on Imaging Science, Systems, and Technology*, 2003.

[LVJV03] A. Likas, N. Vlassis, and J. J. Verbeek. The global k-means clustering algorithm. *Pattern Recognition*, 36(2):451–461, 2003.

[M$^+$66] J.B. MacQueen et al. Some methods for classification and analysis of multivariate observations, 1966.

[MLY99] X. Mei, Y. Liu, and D.K.P. Yue. On the water impact of general two-dimensional sections. *Applied Ocean Research*, 21(1):1–15, 1999.

[Wag32] H. Wagner. Über Stoß-und Gleitvorgänge an der Oberfläche von Flüssigkeiten. *Zeitschrift für angewandte Mathematik und Mechanik*, 12(4):193–215, 1932.

[WLZ$^+$09] H. Wang, M. Liao, Q. Zhang, R. Yang, and G. Turk. Physically Guided Liquid Surface Modeling from Videos. *ACM SIGGRAPH*, 2009.

[XP97] C. Xu and J. Prince. Gradient vector flow: A new external force for snakes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 66–71. INSTITUTE OF ELECTRICAL ENGINEERS INC (IEEE), 1997.

[YDC06] E.M. Yettou, A. Desrochers, and Y. Champoux. Experimental study on the water impact of a symmetrical wedge. *Fluid Dynamics Research*, 38(1):47–66, 2006.

[ZF06] R. Zhao and O. Faltinsen. Water entry of two-dimensional bodies. *Journal of Fluid Mechanics Digital Archive*, 246:593–612, 2006.