Reconstruction of Bubble Trajectories and Velocity Estimation

by

Michael Krimerman

B.Sc, Technion – Israel Institute of Technology, 1995

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

The Faculty of Graduate Studies
(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA
(Vancouver)

February 2013

Abstract

This thesis addresses the problem of tracking bubble trajectories and velocities in transparent fluids, in particular we deal with water. It belongs to the area known as fluid imaging. Using long exposure images with the aid of computerized tomography, we are able to track the trajectories of micro–bubbles by first reconstructing the fluid volume containing the bubbly flow. From this volumetric data we then extract bubble trajectories by following the streaks left behind by each moving bubble. Consequently, these streaks are used to estimate bubble velocities. All of this is achieved using a limited number of consumer quality video cameras. In addition, we show how similar data that might be obtained from tracer particles could be used to calculate the velocity vector field within the fluid interior.

Table of Contents

List of Figures

Glossary

PIV Particle Image Velocimetry

PTV Particle Tracking Velocimetry

TOMO-PIV Tomographic Particle Image Velocimetry

CT Computed Tomography

CAT Computed Axial Tomography

ART Algebraic Reconstruction Technique

OPT Optical Projection Tomography

NSERC Natural Sciences and Engineering Research Council of Canada

PSM Physical Simulation and Measurement

BOS Background Oriented Schlieren

Acknowledgments

I would like to thank my supervisor, Prof. Wolfgang Heidrich, for coming up with the project idea as well as for guiding me throughout my work. His keen observations led to fresh new ways of looking at familiar problems. Special thanks go to the following Physical Simulation and Measurement (PSM) lab members for their support: to James Gregson for his finite–differences code and instrumental guidance, and to Brad Atcheson for sharing his camera–calibration work and creative thinking. My work would have not been the same without the assistance of the aforementioned colleagues. Thanks go to all the PSM lab members who helped by asking questions and brainstorming.

On a final note, I would also like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) scholarship, whom I owe gratitude for supporting me during my first year at the UBC.

Chapter 1

Introduction

In rivers, the water that you touch is the last of what has passed and the first of that which comes; so with present time. — Leonardo da Vinci

The study of fluids, known as fluid mechanics has occupied some of the greatest minds of all times. In ancient Greece Archimedes studied buoyancy. Later, during the early 16th century Leonardo da Vinci worked on flow visualization. He demonstrated deep understanding of the cardio–vascular system to the point of being able to describe vortex formations in blood flow. It is believed that da Vinci constructed a glass model to perform experiments on fluid flows [Gharib et al., 2002]. Other great scientists researched different areas of fluid mechanics: Isaac Newton worked on viscosity, Blaise Pascal on hydrostatics, Daniel Bernoulli published *Hydrodynamica* in 1738 which lay the foundations for fluid dynamics and countless others followed.

Hydrodynamics, or the study of fluid motion, is critical to many fields of science. The objective is to describe temporal fluid flow, be it laminar (as parallel streamlines) or turbulent flow. Numerous methods have been implemented to study flow, among the more popular is Particle Image Velocimetry (PIV). PIV relies on measuring and visualizing particle flow within media. Particle motion captured by PIV enables analysis of the surrounding fluid dynamics. The technique has many real world applications in areas such as medical, mechanical, aerodynamic, chemical, nuclear and food engineering.

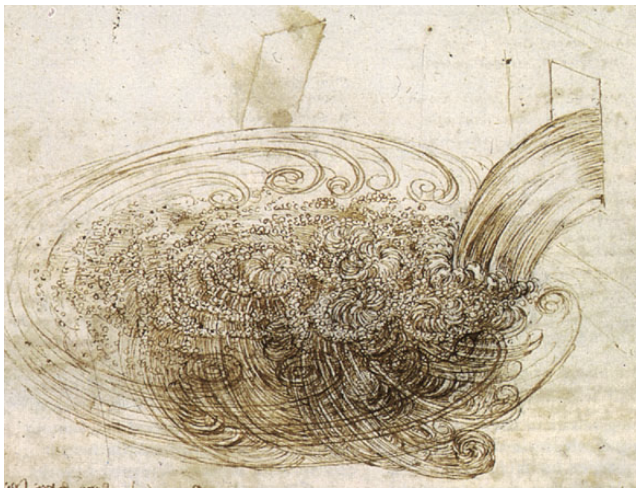


Figure 1.1: Leonardo da Vinci, Water Eddies, 1513. Image courtesy of wikimedia.org (no copyright required, work is in the public domain)

This thesis pertains to bubble dynamics in fluids. In contrast to classic PIV methods that seed the fluid with tracer particles, we examine natural bubbly flows. The study's goal is measuring and understanding bubble trajectories and velocities as they travel up in the liquid interior. In this work we are concerned with a method that is not traditionally used for particle tracking and has its roots in the medical field. Building on standard image acquisition techniques, such as X–Ray radiographs, Computed Tomography (CT) allowed medical imaging take a huge leap forwarded by constructing internal organ structure in three dimensions. We use an optical variant of CT called Optical Projection Tomography (OPT) than enables 3D particle trajectory tracking in transparent liquids. Light waves penetrating the transparent liquid that surrounds a bubbly flow are observed by an array of cameras. The images recorded by the cameras are used to reconstruct the liquid volume and analyze bubble motion. In contrast to classic CT where the detector is rotated around the specimen, our camera array is static. Unlike the rotating detector setup, the camera array facilitates capturing dynamic systems.

1.1 Tracking and Visualization Methods

1.1.1 Particle Image Velocimetry

The PIV classical implementation uses a laser to illuminate a thin sheet of fluid seeded with marker particles. Light reflected from these particles is captured using a camera. When the particle number is low, one can obtain a good estimate of particle position and velocity to the extent of being able to track individual particles (Particle Tracking Velocimetry (PTV)). Using more sophisticated techniques ([Prasad, 2000, Hori and Sakakibara, 2004]), fluid velocity in 3D can be obtained. These methods produces good results when dealing with a low number of particles and are generally applied in 2D or using limited depth.

1.1.2 Computed Tomography

Arguably the most common use of CT nowadays are the Computed Axial Tomography (CAT) devices, also known as *CAT–scanners*, primarily used in hospitals. In this case a stationary patient is instructed to lie down motionless while being inserted into the scanner. The scanner revolves around the patient's body emitting X-Rays on one end, collecting them at the detector on the other. The detector registers the accumulated density along each ray. This information is later used to reconstruct the internal organ structure.

1.1.3 Optical Projection Tomography

Unlike X–Ray absorption based computed tomography which requires an expensive lab setup, OPT uses cameras and light sources to acquire data needed for reflectance or absorption based tomographic reconstruction. The choice of using OPT depends largely on the objects targeted for reconstruction. When the medium is transparent, as it is in our case, or the objects are small enough to allow high intensity light penetration then OPT becomes a viable alternative to X–Ray tomography.

1.2 Our Method

A mandatory requirement for the methods above is the need for specimen to be motionless during capture. This is essential to avoid motion related artifacts such as blur, which impact the reconstruction quality. In order to produce sharp images PTV requires fast exposures. Typically, PTV uses temporal image–sets to track particles by performing a spatial search for neighbouring particles. In order for the search to be successful the particle density and velocity should be low, otherwise particle matching between images in the sets is hard to ascertain.

Our method takes a different approach: instead of trying to minimize exposure times, we use long exposures to capture motion. Unlike common PTV where the short exposure causes tracer particles to appear as specks, long exposures have the effect of bubbles appearing as elongated streaks. These streaks simplify trajectory tracking: we no longer need to perform inter frame search in order to follow the temporal bubble positions, instead we have a complete streak representing the bubble's path. Since we are dealing with rolling–shutter video cameras and we wish to obtain evenly illuminated images, the exposure time should last at the minimum for the entire duration of a single video frame, otherwise frames will be unevenly exposed (see Figure 3.11). Thus the video standard determines our minimal exposure duration. We have at our disposal both short and long exposures. Short exposures are possible by using single video frames, whereas long exposures are produced by averaging several successive frames. Measuring streak length in the volume reconstructed from the short exposure images offers a hint to bubble velocity, whereas trajectory path is determined from the volume reconstruction using the long exposure images.

There are several algorithms for tomographic reconstruction, the choice is determined by the problem domain. Unlike CAT where rays are projected co–axially or in parallel, ray formation in our case dictates the usage of Algebraic Reconstruction Technique (ART) or a derivative for the reconstruction algorithm. A preliminary calibration step is necessary to obtain ray projections, after which the projections along with the recorded image–sets are used as inputs to the reconstruction algorithm.

To summarize: we employ OPT to reconstruct bubbly flows in transparent me-

dia, using consumer quality cameras with long exposures. The recoded image–sets depict streaks that allow us to reconstruct the volume and track bubble trajectories and velocities.

1.3 Problem Statement

Given a transparent cylinder containing a bubbly flow, we wish to determine the trajectory and velocity of visible bubbles. The method should handle typical examples like fish–tanks where bubble density is high. The results can be used to simulate bubbly flows for special effects as well as study the flow when designing bubble column reactors [Ranade, 1997, Deckwer and Schumpe, 1993].

1.3.1 Assumptions

In order to achieve this goal the following assumptions are made: The cylinder used is transparent with good optical qualities, i.e. minimal distortions. The contained liquid is transparent. Transparency is necessary since we need bubbles to be visible by all cameras without occlusions. The bubbly flow is composed of fine bubbles with a diameter of about $1 - 2mm$, rising in the centre of the cylinder. We also assume the bubbly flow density and bubble number to be similar to ones found in modern fish tanks. Very high bubble concentrations might affect reconstruction quality. Hardware components used are described in more detail in Chapter 3.

1.4 Thesis Outline

Chapter 2 is dedicated to related–work. This is followed by Chapter 3 on the capture process, dealing with the calibration setup and capture. Chapter 4 details both the theoretical and practical work for the tomographic reconstruction part. Next, Chapter 5 talks about velocity estimation. We then conclude in Chapter 6 which also addresses future work.

Chapter 2

Related Work

In this chapter we offer a glimpse into several works that are related to our problem domain. Most of these are situated around the Algebraic Reconstruction Technique (ART) as originally described by [Gordon et al., 1970].

2.1 Particle Image Velocimetry

PIV is a significant flow field measurement and reconstruction technique in many areas of research as illustrated below. In their book [Raffel et al., 1998] describe PIV as a generally non obtrusive method: it does not require inserting probes into the observed medium. This is achieved by means of observing the locations of tracer particles, while data collection happens through cameras arranged around the medium. The technique is of high importance in the aeronautics industry where it is used for flow and turbulence measurements around aircraft models. Here, complex vortical structures that form around the wings and body are meticulously examined. The observations have impact on the shape and performance of aircraft as well as regulations regarding takeoff/landing angles and velocities, and separation between successive landings [Brossard et al., 2009].

Classical implementations utilize a laser to illuminate a thin sheet of tracer particles. These particles are selected to match fluid properties so that they can be used as flow indicators. The ratio of particle to fluid refractive indices determines the light scattering efficiency. Light reflected from the tracer particles is observed

by the cameras arranged in multiple angles around the media.

With the increasing power of modern hardware, Tomographic Particle Image Velocimetry (TOMO-PIV) gained popularity in the last decade. An implementation very similar to our own is described in [Elsinga et al., 2006, 2008]. This work later led to the investigation and reconstruction of vorticity patterns of cylindrical wakes by [Scarano and Poelma, 2009]. Here, the authors used laser sheet illumination with smaller and slower tracer particles. Tomographic reconstruction was performed on the acquired data with a small number of cameras which is sufficient due to the relatively limited depth offered by the laser-sheet illumination. [Atkinson and Soria, 2007] provide an overview of ART based algorithms that are applicable to TOMO-PIV.

Tracking and measuring bubbly flows is an active research area which commonly employs PTV. Popular implementations utilizes laser illumination [Matsumoto et al., 2012, Bröder and Sommerfeld, 2000] while others base captures around X–Ray [Seeger et al., 2001].

2.2 Computed Tomography

The tomographic reconstruction method is based on work pioneered by Radon from 1917 [Radon, 1986]. The basic principle of his work was to show how information can be reconstructed from different projections. Based on Radon's work, in 1937 Kaczmarz[1] showed that by solving a large linear system one can obtain an approximation of the original information. ART, as presented by [Gordon et al., 1970] is an iterative technique that was re-introduced to solve the same linear system initially proposed by Kaczmarz. ART and many of its variants are widely used today.

2.2.1 Blood Flow Field

Measuring blood flow is vital for the study of cardio–vascular diseases. The need to perform measurement on living patients mandates the use of X–Rays to penetrate tissues. This, in most cases, limits reconstruction to 2D. In a recent paper [Dubsky et al., 2010] used X–Rays to measure in vivo PIV blood flow. The measurements are

[1] http://en.wikipedia.org/wiki/Kaczmarz_method

preformed by rotating the sample while capturing a frame sequence at each angle with a high speed camera. This system is an example of a transition from limited angle PTV to multiple angle views. The transition enables the use of tomography to reconstruct the three dimensional blood flow field. While enabling 3D tomographic reconstuction, the setup has the limitation of non overlapping temporal captures for each angle, and is therefore less suitable for faster fluid flows.

2.3 Optical Projection Tomography

The traditional means of acquiring data to be reconstructed by tomography was using X–Rays. The scanners for performing this acquisition utilize a fan beam projector. OPT on the other hand uses light waves. Here, rays have the form of parallel lines. In either case, ray formation has a direct implications on the applicable algorithm collection, thought the methods used in OPT are usually more flexible when it comes to ray formation.

OPT has been used to a lesser extent than X–Ray tomography due to its limited application when dealing with opaque objects because visible light hardly penetrates if at all. The power and simplicity of OPT shines when concerning transparent and translucent media. That is the case when capturing microscopic samples where a high intensity light permeates the delicate tissues.

2.3.1 Visualizing Embryo Anatomy

When OPT was introduced in 2002 by [Sharpe et al., 2002, Sharpe] the first application was visualization of embryo anatomy in outstanding clarity. In these papers the authors captured a young mouse embryo and also mention the method is applicable for specimen of up to 15*mm* thick. The method proved to be instrumental in developmental biology. The specimen was placed in a cylinder that is capable of rotation around the central axis. The cylinder was sited inside a microscope equipped with a CCD camera. Capture was performed while rotating the cylinder producing a collection of 400 images from 360° of the embryo. In order to maintain sharpness of imaged details, the authors used a confocal microscope with narrow illumination cones which minimized noise from neighbouring pixels.

2.3.2 Transparent Object Scanning

[Trifonov et al., 2006] used OPT to reconstruct the structure of transparent objects. By immersing the object in a liquid tank, unwanted refractions can be reduced by means of matching the refractive index of the liquid to that of the glass object to be scanned. Rotating the cylindrical tank around the central axis enables capturing of multiple angle views. This set of views is then fed into a tomographic reconstruction algorithm that reconstructs the immersed object shape. This approach works well for static objects: the captured object formation is not affected by the temporal shift caused by the delay between successive captures from multiple views.

The common requirement to this procedure and the one in Section 2.3.1 is that both are observing stationary objects. The simplest way to perform a multi–angle capture is to use a single camera while rotating the specimen. A computer controllable rotation platform allows capturing numerous views in precise angle increments. However when capturing dynamic objects the same configuration cannot be applied. This usually calls for replacing the rotation platform and single camera with a camera array encircling the objects as is described in Section 2.3.3 below.

2.3.3 Gas Flow Reconstruction

A similar setup to ours appeared in a work by [Atcheson et al., 2008] on scanning non-stationary gas flows. In this project the same camera array was used to capture gaseous flows. The authors made use of Background Oriented Schlieren (BOS), which utilizes a high frequency background image for enabling the measurement of per pixel deflection introduced by the gaseous volume. The recorded images represent the integral of displacement that light waves undergo due to the refraction caused by the gas volume. A tomography algorithm is then used to reconstruct the refractive indices in this volume. Our setup utilizes some of the same hardware components as well as the same camera synchronization code.

Chapter 3

Capture Process

This chapter discusses the equipment setup and procedure used to capture the image–set that is later used as a basis for reconstruction. The steps involve a preliminary calibration stage where all the cameras are properly aligned and ray projections are deduced. Following that, we capture some support images such as background (for background subtraction) and then perform the bubbly flow capture.

After describing the capture process we elaborate on the post capture data processing steps.

3.1 Hardware Setup

Most of the hardware involved relies on low–cost and easy to obtain off the shelf components. The camera array is composed of 16 HD quality Sony camcorders. To generate bubbly flow we are using a simple air–pump and an air–diffuser that were purchased from a pet store. The cylinder is a regular laboratory clear glass beaker with a custom opening at the bottom for the air tube. The calibration target structure was printed using a 3D printer to fit in the cylinder, while the pattern itself was printed on a transparency using a regular laser printer.

3.1.1 Fine Bubbly Flow

The setup requires fine bubbly flow which is produced by a common 3" wooden air–diffuser[1] of the type usually found in modern aquariums. Our goal was to produce a fine bubbly flow giving bubbles with about $1-2mm$ in diameter. We have experimented with several diffusers, both of the common sort found in pet stores and industrial types, such as sintered metal (a porous material used in industrial filters). All of these produced large bubbles with varying diameters which would imply adding support for bubble light reflection to our model. [Puleo et al., 2004] contains an overview of how filter materials affect bubble sizes. Our experience showed that wooden diffusers indeed produce fine homogenous bubble curtains. We have not been able to obtain similar results with metallic diffusers, which according to [Puleo et al., 2004] can generate even finer bubbles. Finally, a 3" wooden air diffuser was selected for our setup. The diffuser is placed at the bottom of the cylinder and fed by an air tube connected to a small electrical pump. Air flow was regulated by a tap.

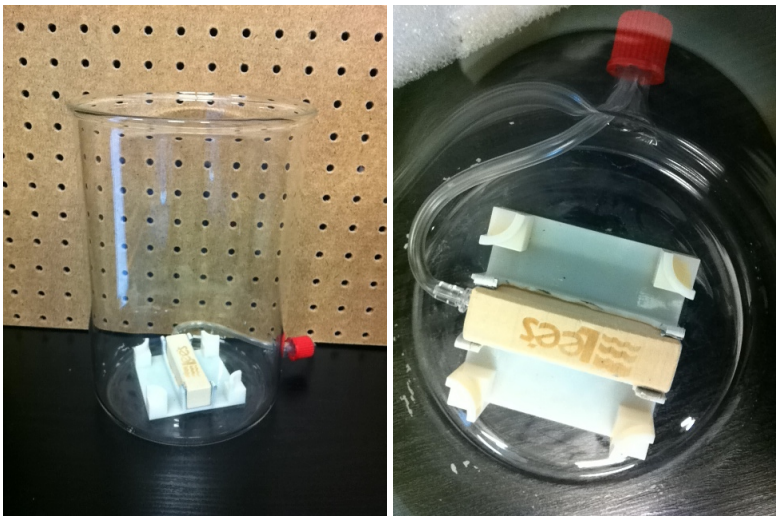
In order to achieve good reconstruction results we make sure the bubbly flow remains near the central axis of the cylinder where all cameras have good coverage. Water is stirred to produce more complex bubbly flows.

In the interest of producing bubbly flow in the centre of the cylinder, the volume that is covered well by all cameras, we tried blocking parts of the diffuser to restrict the flow. This however caused the flow regime to change so that bubble sizes became non–uniform. Since varying bubble dimensions is not desirable, we opt for using the diffuser as is with the drawback that some of the bubble trajectories will be leaving the ideal reconstruction volume and travelling within areas with partial camera coverage.

3.1.2 Strobe Lights

In this setup strobe lights serve a dual purpose: they provide overhead illumination causing light to reflect from bubbles, and they acts as a secondary synchronization mechanism for the camera array.

[1] http://www.amazon.com/Lees-Wooden-Diffuser-3-Inch-2-Pack/dp/B0002QQN4Y/



(a) Side view **(b)** Top view

Figure 3.1: Diffuser placement in bottom of cylinder

The Arduino[2] programmable strobe controller supports the specification of custom on/off durations. The controller is programmed to repeatably switch the strobes on for the duration of several NTSC frames (each NTSC frame length is 29.97μs) and then off for the duration of a single NTSC frame. During the on period, the cameras are exposed to the incoming light and the bubble traces are recorded. Then the strobes are switched off for a single NTSC frame to allow for a later frame synchronization as described by [Bradley et al., 2009].

3.1.3 Camera Array

The camera array is comprised of sixteen Sony HD consumer video cameras arranged in a 160° semicircle around the cylinder. This arrangement has the cameras about 10° apart with the angle $(\alpha_0, \alpha_1, \ldots, \alpha_{15})$ of each camera measured relative to one master camera, which for convenience was chosen as α_7. Camera orientation measurements are mandatory for the calibration capture (Section 3.2.1) as well for ray coordinate transform (Section 3.3.1).

Camera orientation alignments do not need to be highly accurate since the cali-

[2] Arduino website: http://arduino.cc

Figure 3.2: Camera array centred around cylinder. Dark cloth was used to minimize reflections from glass cylinder.

bration procedure is tolerant to angular shifts. However, once the angles have been determined, it is important for the motorized rotary stage (Section 3.1.5) to rotate to these angles precisely.

The cameras are not required to be coplanar, some cameras are positioned on higher mounts than others. Coplanarity is not mandatory as each camera undergoes a separate calibration procedure. Intuitively, the fact they are not collocated in the same elevation should improve coverage for reconstruction but we have yet to verify this claim.

The semicircle arrangement is necessary to avoid cameras viewing each other through the transparent cylinder. Each view has only dark background visible behind the cylinder. Further background elimination is described in Section 3.2.2.

3.1.4 Camera Control

Rudimentary camera control is achieved through an Arduino based programmable controller board. Control is performed by sending LANC[3] commands to all cameras *almost* simultaneously. The controller supports the following commands: turn cameras on/off, start recording, pause and toggle focus between manual and automatic. Unfortunately, the LANC protocol was not designed for multiple synchronous camera control, therefore a secondary mechanism is needed for frame synchronization. This is performed by utilizing the strobe lights (see Section 3.1.2).

Figure 3.3: An earlier setup showing the platform holding the calibration pattern near the glass cylinder. The platform is inserted into the cylinder for calibration. Here the pattern is seen in the front slot.

[3] An unofficial reverse engineered protocol description can be found at: http://www.boehmel.de/lanc.htm

3.1.5 Calibration Platform and Pattern

Calibration is performed using a 3D printed calibration pattern mounted on a platform. The pattern structure is described in [Atcheson et al., 2010]. The pattern contains tiles with unique markings on each tile as can be seen in Figure 3.4. The platform and pattern are inserted into the cylinder and rotated together to face each camera at angle α_i in turn, Figure 3.5). The platform structure was printed using a 3D printer and was designed to fit tightly inside the cylinder to eliminate potential calibration errors.

The cylinder is placed on a motorized rotary stage[4] controlled via USB by a computer which also sends commands to the strobe and camera controllers via USB.

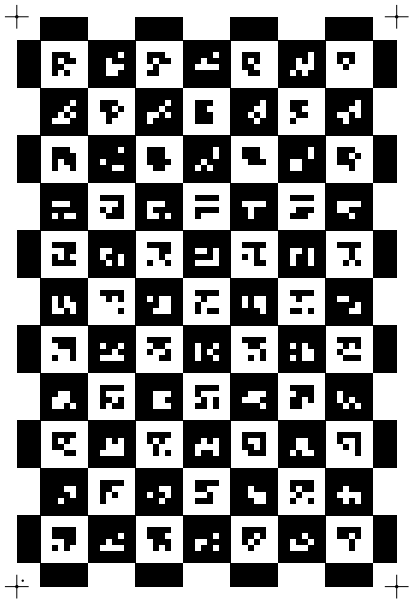


Figure 3.4: Calibration pattern. Note that each tile contains a unique code which helps locating grid coordinates and inferring plane orientation.

[4]Zaber T-RS: http://www.zaber.com/products/product_group.php?group=T-RS

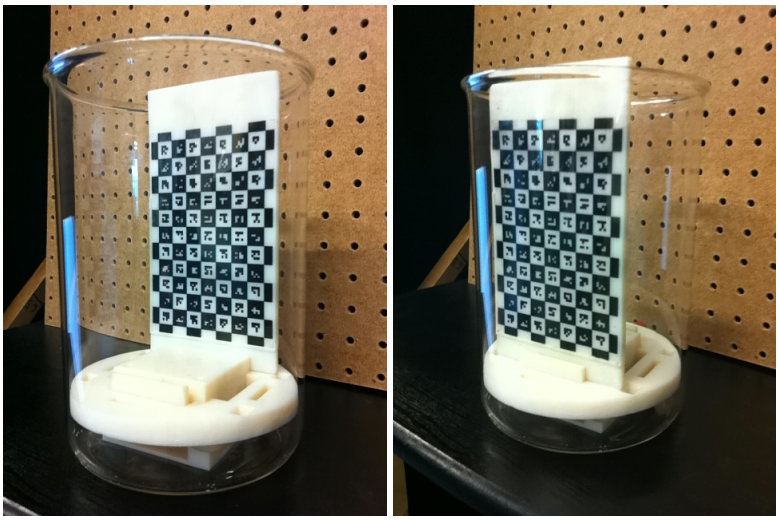
3.2 Capture Procedure

The capture sequence is composed of two main parts. In the first we perform careful camera calibration with the final goal of generating a list of ray coordinates. In the second part we capture the bubbly flow for reconstruction. We begin with the calibration related capture description.

3.2.1 Calibration Related Capture

The cylinder and the calibration platform in it are rotated together to face each of the sixteen cameras using a computer controlled rotating stage. Once facing a camera at angle α_i, the camera controller sends a start recording command to that camera, followed by a pause command after a second. This produces a short static video sequence which is later used to extract a single frame. We opted for video frame extraction versus the snapshot functionality in order to keep the resolution as well as other settings equal to those used in the main capture. Calibration capture happens in two sequences. In the first sequence the pattern is placed in the front of the platform and rotated to face each camera in turn. Once all cameras have captured the pattern, it is moved to the rear of the platform and the capture sequence is repeated. After the calibration capture step is done, each camera contains recordings of both front and rear views of the pattern facing it.

Once calibration capture is over, the platform is removed from the cylinder. The bubble diffuser, situated underneath the platform is now exposed and able to release bubbles for the volumetric capture.



(a) Rear **(b)** Front

Figure 3.5: Calibration platform holding the calibration pattern at the two positions

(a) Rear **(b)** Front

Figure 3.6: Calibration pattern as seen by the leftmost camera in the two positions

3.2.2 Background Capture

In order to remove as much background noise as possible, a background image (short video sequence) is captured by each camera. To minimize inconsistencies, this capture is performed in the same manner as bubbly flow capture Section 3.2.3 without turning on the air pump.

3.2.3 Bubbly Flow Capture

Now the air pump is switched on and the water is stirred. Flow capture begins by switching all cameras to record mode simultaneously. At this point in time the strobes are turned off. Once all cameras have started recording the strobes are switched on to work in the predetermined frequency. The cameras are now recording the bubbly flow in frame sequences separated by a single dark frame. After recording for a couple of seconds the cameras are paused. More bubbly flows are captured following the steps above after which the capture sequence is complete.

3.3 Data Processing

In this step we perform all the post capture data processing to prepare the captured images for reconstruction. This includes calculating ray equations and generating an image–set for tomographic reconstruction. The ultimate purpose of calibration data processing is to generate and transform the equations of all the rays observed by each camera into this global viewer coordinate system.

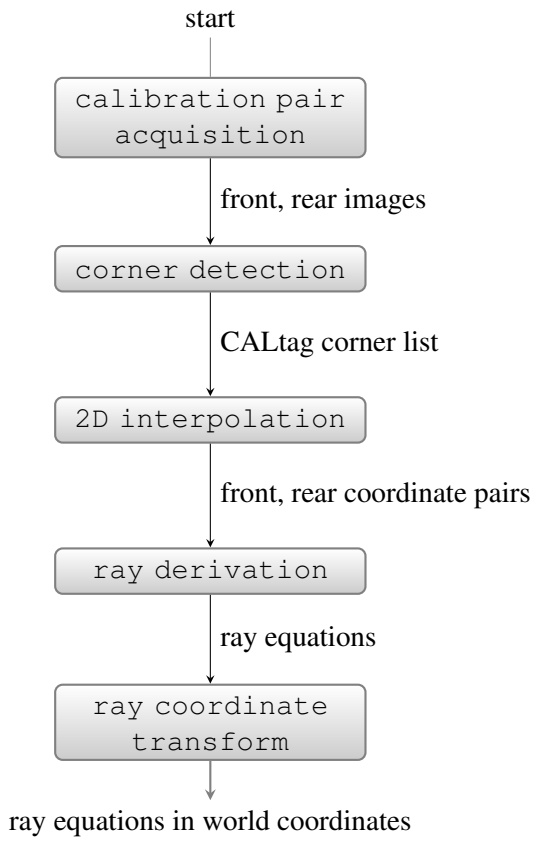


Figure 3.7: Calibration procedure. After successfully obtaining a pair of front and rear images, we follow these steps to produce the ray equations in global viewer (master camera) coordinates.

3.3.1 Deriving Ray Projections

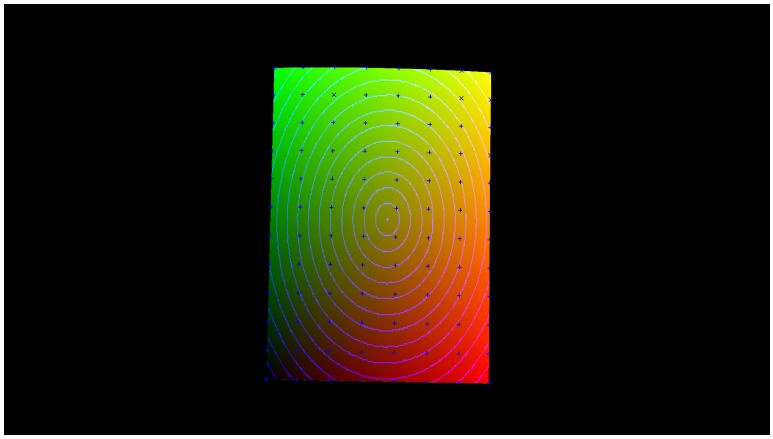
The reconstruction algorithm projects rays that penetrate through the volume. The goal of obtaining a front and rear image pair is to construct the ray equations from corresponding point pairs.

Tile Corner Coordinates

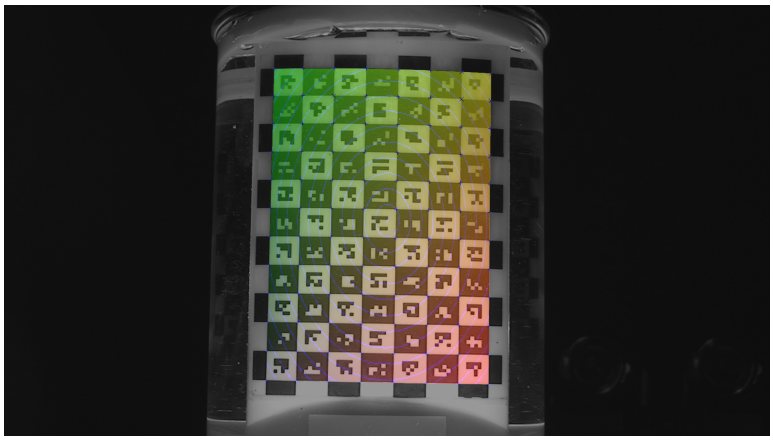
The first step is to locate the coordinates of each corner of the tiles in Figure 3.4. For each camera CALTag, see [Atcheson et al., 2010], produces a sparse mapping from pixel coordinates to world coordinates on the pattern plane.

Coordinate Interpolation

Once all tile corner coordinates are located, we produce the two dimensional interpolated gradient pattern, see Figure 3.8a. This is implemented using a bilinear interpolation of an OpenGL texture over the tile coordinates, then sampling the pixels in between using an offscreen buffer. Both front and rear images undergo the same interpolation process.



(a) Tile corner interpolation



(b) Superimposed interpolation

Figure 3.8: Front calibration pattern interpolation for the leftmost camera. Crosses represent the detected tile corners. Green is used for the interpolation along the X axis, red represents Y axis interpolation. The concentric circles are a visual aid for verifying the 2D interpolation.

Deducing Ray Equations

Using the front and rear interpolated planes calculated above, we select all pixel pairs that have both front and rear interpolated values. Pixels that appear outside the tiled area in both or one of the planes do not provide the complete information required to produce rays.

Since the distance between the two planes is known we determine the ray as a 3D line that passes through the two coordinates $(x,y,z)_{front}$ and $(x,y,z)_{rear}$. These rays live in the coordinate system of the camera that contained the pixel pair.

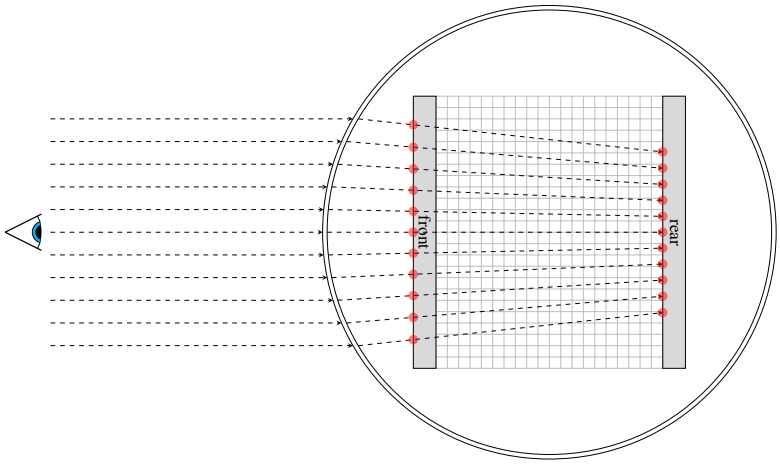


Figure 3.9: Ray projection through reconstruction volume using calibration point pairs. Assuming a remote viewer, incident rays can be considered to be parallel. Rays undergo refraction when hitting a material boundary. By locating a pair of points on each ray (red dots) we can determine through which voxels the ray passes.

Ray Coordinate Transform

Camera rays are transformed to the global viewer coordinate system by rotating the rays according to the respective camera angle α_i (relative to the global viewer direction α_7), see Figure 3.10. The pattern rotation angle α_i for each camera was determined during the camera array setup Section 3.1.3.

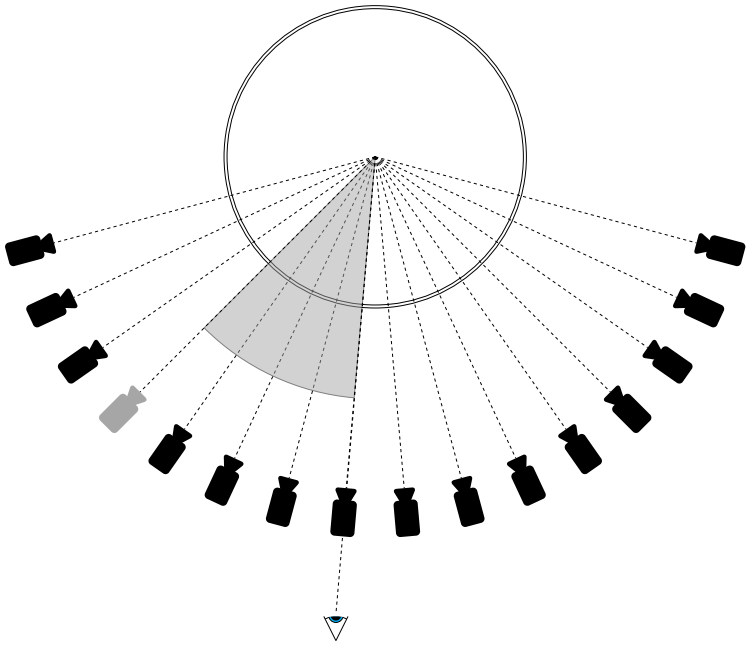


Figure 3.10: Approximate camera arrangement around cylinder. This setup reduces background noise by making sure cameras do not lie on each others direct line of sight. One angle is defined as the global view–point and all camera rays are rotated to the viewer coordinate system.

3.3.2 Frame Extraction

The first step is to extract frames from videos. Next we search for the offset of the first frame that is exposed by the strobe–lights, this is done with a program that locates a frame with a sharp increase in intensity. The result is a sequence of images per camera for the duration the strobe was on.

Background images obtained in Section 3.2.2 are subtracted from each frame in the sequence to reduce artefacts from background and reflections on the glass cylinder. This has the effect of removing everything that is static in the image, leaving (almost) only the bubble traces.

3.3.3 Frame Synchronization

The challenge when using consumer quality cameras with a rolling–shutter is that they are very hard to synchronize. The controller issues the *start–recording* mes-

sage to all cameras sequentially with a short delay in between the messages. That along with the non–deterministic camera phase does not permit exact simultaneous synchronization. The result is temporally shifted frames. [Bradley et al., 2009] have addressed this issue by stitching frame segments to produce temporally matched frames.

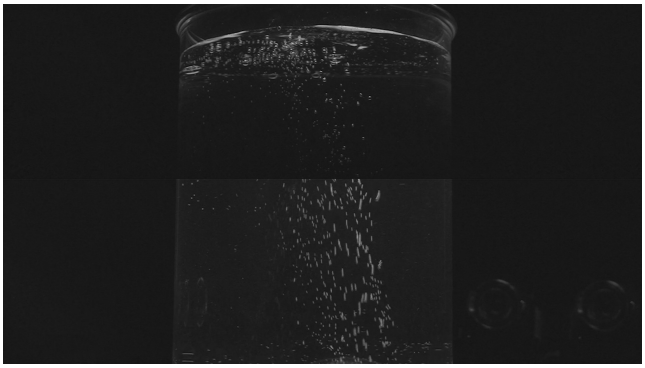


Figure 3.11: A typical frame artifact caused by the rolling shutter effect with the temporal strobe exposure beginning mid–frame.

3.3.4 Radiometric Correction

Each camera frame–sequence is exposed to different amounts of light due to the reasons listed below. This can have a large impact on our ability to reconstruct the volume as the reconstruction algorithm treats all the cameras equally and cannot handle illumination variance. We therefore need to perform a radiometric–correction before reconstruction takes place so that views are not corrupted by exposures differences.

There are several reasons for the different camera exposures, the most prominent one is due to the inability to programmatically set the exposure/gain levels of the cameras we use. The only way to set the exposure is by manually rotating a knob until all views appear to be visually similar. Needless to say, this method is highly inaccurate. Another reason for the different exposure levels is due to the distance of each camera from the centre of the cylinder, this distance varies both horizontally and vertically. A third reason is the location and shape of the strobe light, which has varying light path distances to each camera.

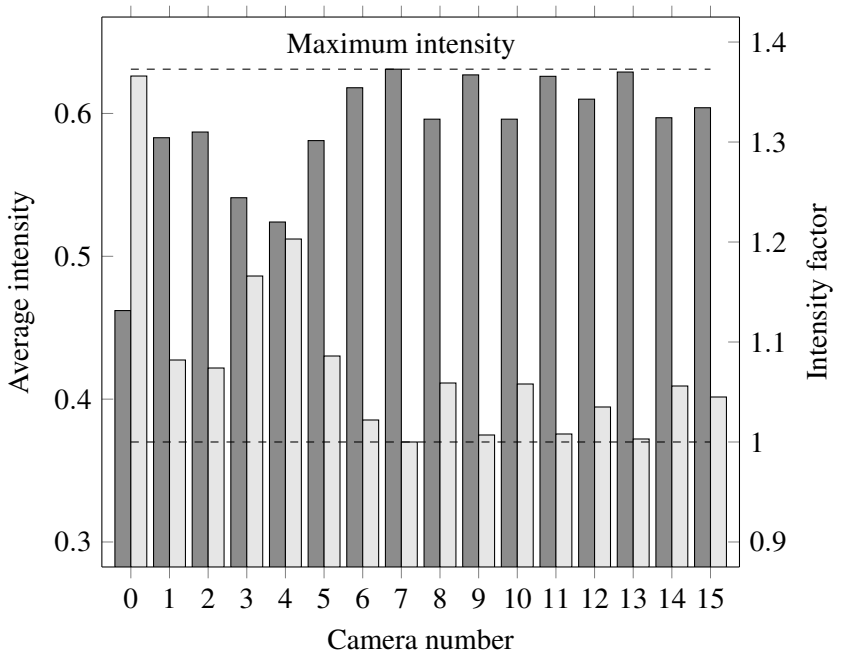
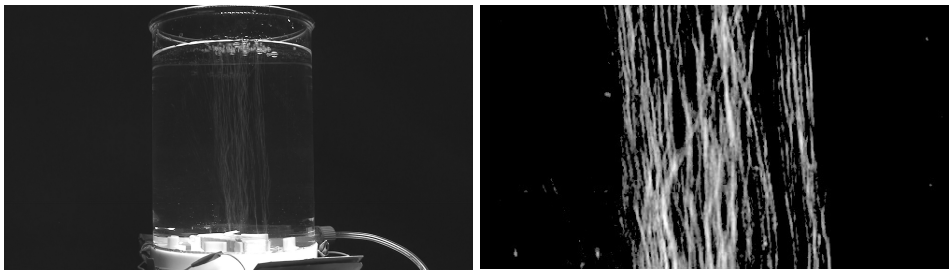


Figure 3.12: Average normalized intensity (dark bars) of the same patch (a rectangle containing all pattern tiles) as seen by all cameras and radiometric normalization–factor (light bars). Camera #7 shows the highest average intensity. The views for all cameras are multiplied by the corresponding factor (max intensity to camera average intensity ratio $f_c = I_{max}/I_c$).

3.3.5 Frame Averaging

Video cameras exposure duration is determined by the video standard (NTSC in our case). Using this exposure duration bubbles appear as short lines in video frames. In order to visualize longer bubble trails we add up several consecutive frames together then average to maintain original intensity (energy) levels. When this is done bubbles appear as elongated streaks as can be seen in Figure 3.13. The streak image–set serves as a good basis for bubble trajectory reconstruction. However it is not suitable for bubble velocity estimation. For velocity estimation we prefer the smaller streaks that can be used to deduce instantaneous bubble velocity which cannot be accurately estimated from the elongated streaks.



(a) Streaks **(b)** Enhanced detail

Figure 3.13: Bubble trajectories appearing as elongated streaks when averaging several frames together.

3.3.6 Visual Hull Mask

As can be noticed in Figure 3.13, much of the volume contains water with no bubbly flow. Since we are dealing with an underdetermined problem and the reconstruction algorithm is quite time intensive, it is preferable to reduce both the number of required unknowns and calculations as much as possible. Using a visual–hull mask also reduces ghosting since a large portion of the voxels are marked as empty and thus do not play a role in the algebraic reconstruction. This has the effect of a dramatic reduction in the number of the unknowns (voxels) in our linear system. Although using a visual–hull mask also decreases the number of rays (equations in the linear system), the impact on reduction of unknowns is much larger: $O(n)$ unknowns reduction per single visual–hull ray. The overall effect on the system is that it becomes better conditioned.

This can be achieved by demarcating blank or empty rays that are not penetrating any bubbles, in other words their corresponding pixel value is (close to) zero. A simple threshold can do the trick and reduce computational efforts. We set the threshold manually to a relatively low number to avoid losing important data (false positives). A typical threshold image such as Figure 3.14 contains two types of values: black means the ray goes only through non–bubble voxels that should be masked out, and white means the ray potentially penetrates bubbles and should therefore be considered in the reconstruction process.

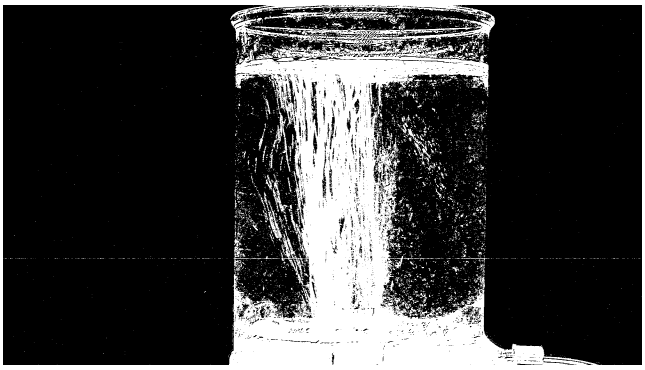


Figure 3.14: Threshold image used as a visual–hull mask. Black pixels are known not to contain any bubbles. This information assists in decreasing the reconstruction runtime.

This chapter dealt with the type of data we collect, how we collect it and how it is processed in preparation for reconstruction. In the next chapter we look at the reconstruction procedure, how all the data collected so far are used to produce a three dimensional volume representation of the bubbly flow.

Chapter 4

Reconstruction

The previous chapter dealt with the acquisition sequence, all the steps necessary to prepare a set of images for the reconstruction of the water volume with bubbly flow. Now that we have the image–set we proceed to reconstruction. The current chapter also covers several additional post tomographic reconstruction steps to find the bubble trajectories.

4.1 Ray Projection

In this section for simplicity we shall be dealing with a 2D description of the problem, the real–world 3D approach is very similar.

A ray is parametrized as a straight line, $r_{\theta,t}$, with angle θ and distance from origin t on the $X-Y$ plane. Cartesian coordinates have the following relation:

$$x\sin(\theta)+y\cos(\theta)=t \tag{4.1}$$

A function $f(x,y)$ represents the presence of objects on the same $X-Y$ plane. In CT this function is interpreted as the density of the object while in our case it is the reflectance of light from bubbles. We can now define the total reflectance measured along ray $r_{\theta,t}$ as:

$$p_{\theta,t}=\int_{r_{\theta,t}} f(x,y)\,\mathrm{d}s \tag{4.2}$$

Figure 4.1: The complete reconstruction procedure.

The next step is to discretize the problem, first the integral is replaced by a sum. A new factor $w_{\theta,t,x,y}$ has to be introduced since part of the discretization is to produce a grid G with a finite number of cells. This weight depends on the inverse of the distance of the cell at x,y from the ray $r_{\theta,t}$. Typically a Gaussian shaped

function is used for the weights. Thus most of these weights are practically zero and only cells in the ray's vicinity posses weights greater than zero.

$$p_{\theta,t} = \sum_{x,y \in G} w_{\theta,t,x,y} f(x,y) \tag{4.3}$$

For the discretized problem, $p_{\theta,t}$ represents the intensity of a pixel corresponding to ray $r_{\theta,t}$. This intensity is governed by the sum of individual bubble intensities in the ray's close neighbourhood.

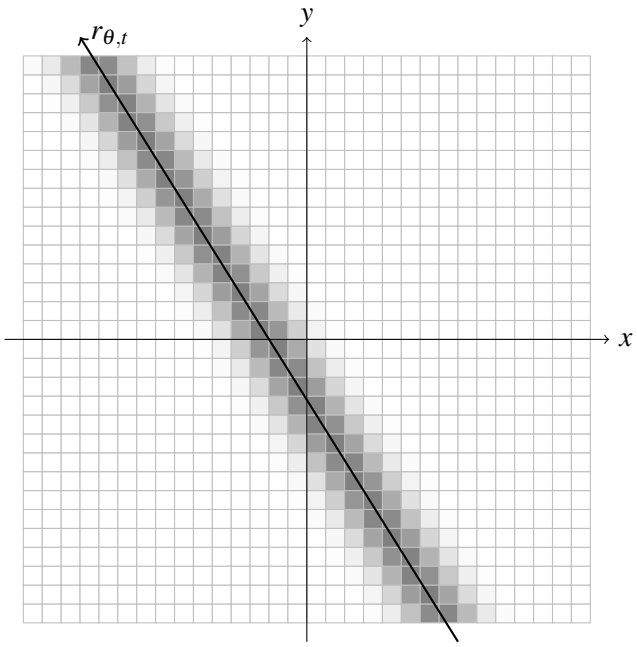


Figure 4.2: A typical voxel weight allocation for a single ray. Darker voxels indicate higher weight values.

4.2 Reconstructing the Original Function

In Equation 4.3 the function $f(x,y)$ is used to compute the projection for a ray. In our case we measure the projections as pixel intensities and the rays are known from the calibration. We do not know what the original $f(x,y)$ is and that is exactly what we are interested to find out from the information obtained. This is called

an *inverse problem*: given some observations we wish to reconstruct the original function that corresponds to these observations. The problem depends largely on the number of equations M, versus number of unknowns N (for the exact values see the sections below). The bubbles we use are quite small, requiring a high resolution reconstruction (in other words a large number of voxels) implying a large number of unknowns. On the other hand, the number of rays depends on camera resolution and number of cameras, dictating the number of available equations.

This leaves us with the following set of equations (note the change of notation: for brevity we use i to index the rays instead of the pair (θ,t), and j to index the voxel at (x,y)):

$$\begin{array}{rcccc} p_0 & = & w_{0,0}f(0)+\ldots+ & w_{0,j}f(j)+\ldots+ & w_{0,N}f(N) \\ \vdots & = & \vdots & \vdots & \vdots \\ p_i & = & w_{i,0}f(0)+\ldots+ & w_{i,j}f(j)+\ldots+ & w_{i,N}f(N) \\ \vdots & = & \vdots & \vdots & \vdots \\ p_M & = & w_{M,0}f(0)+\ldots+ & w_{M,j}f(j)+\ldots+ & w_{M,N}f(N) \end{array}$$

We are looking for a solution to this set of M equations that agrees with our observations. Next we will look into how tomographic reconstruction helps us find a valid solution corresponding to our measurements.

4.3 The Algebraic Reconstruction Technique

Tomographic reconstruction can be divided into two major categories: Fourier based and algebraic. The latter is suitable for problem as it lends itself to dealing with different ray geometries and makes it possible to add a–priori knowledge about the reconstruction domain. Algebraic Reconstruction Technique (ART) is an iterative approach for solving a large linear system. In our case the matrix size is number of voxels times number of rays, which is, for practical resolutions, too large to keep in memory. ART can be implemented as a matrix free method thus keeping memory requirements minimal. The technique was originally developed by [Gordon et al., 1970] in the early seventies.

4.3.1 Problem Statement

Given a set of M rays $\{r_0, r_1, \ldots, r_M\}$, with each ray r_i having a corresponding measured intensity value p_i, and a set of N voxels $\{v_0, v_1, \ldots, v_N\}$ find the unknown intensity value x_i of each voxel v_i.

In matrix notation we need to solve:

$$\boldsymbol{A}x = p \tag{4.4}$$

where $\boldsymbol{A}$ is a matrix of coefficients a_{ij} representing how much of ray r_j passes through voxel v_i.

$$A_{m,n} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{pmatrix}$$

In the notation above, the 3D voxel cube has been "flattened out" into a long voxel list. Thus, the number of voxels N, is actually the product of the number of voxels along each axis: $N = N_x \times N_y \times N_z$. It is usually convenient to work with cubic volumes where all dimensions are similar, as it is in our case. Therefore we use $N = n^3$, where n is the same number of voxels along each of the dimensions.

4.3.2 Typical Matrix Dimension

There are several techniques to solve equation Equation 4.4, most rely on the availability of matrix A. The matrix can be precomputed or computed on the fly when an element is accessed. Both ways of accessing the matrix lead to poor performance as the matrix is too large to fit in memory. The relatively small diameter of the bubbles (around $1-2mm$) while the reconstruction volume is quite large (around $150mm \times 150mm \times 150mm$). This calls for volumes of at least 300^3 voxels, and for better results we use up to 600^3 voxels. The cameras we use produce around 6 million rays. The resulting matrix is too large for modern computers to fit in memory.

4.3.3 The Algorithm

A camera obtains the cumulative reflections from bubbles along a ray as a pixel intensity:

$$p_i = \sum_{j=1}^{K} w_{ij} x_j \tag{4.5}$$

or the weighted sum of intensities of K voxels along ray r_i producing a pixel intensity of p_i. And w_{ij} representing the weight of ray r_i at voxel v_j produced by our kernel–function, a Gaussian kernel. This can be observed by expanding the product of a single row of $\boldsymbol{A}$ and the set of voxels from Equation 4.4.

The above is a somewhat simplified model as bubbles that are further away from the camera or light source will appear less bright. However these differences are minor and can therefore be ignored.

We begin with an initial guess, in our case an empty volume. This implies setting each voxel–intensity to zero:

$$x_i = 0 \qquad i = 1, \ldots, N$$

Next we begin iterating until convergence as specified below. Each ray is projected through the volume, summing up the voxel intensities for the K voxels that lie close to the ray. This step is called the *forward projection*:

$$\widehat{p_i} = \sum_{j=1}^{K} w_{ij} x_j \tag{4.6}$$

Now we compare the calculated intensity $\widehat{p_i}$ with the measured one p_i, and write down the difference between the two:

$$\Delta p_i = \widehat{p_i} - p_i \tag{4.7}$$

The back–projection step is to distribute the above residual between all the voxels along the ray according to each voxel's weight.

$$\widehat{x_j} = x_j + \lambda \frac{w_{ij}}{\sum_{k=1}^{K} w_k} \Delta p_i \qquad k = 1, \ldots, K \tag{4.8}$$

Δp_i can be negative if other rays/iterations with common voxels assigned values that produce an intensity sum larger than the measurement p_i. When this happens, the intensity for all participating voxels is reduced.

For the residual distribution we are using a parameter λ to control how much of the residual we wish to incorporate into the next iteration $\widehat{x_j}$. This factor affects the convergence rate. From time to time, due to the iterative nature of the algorithm, an updated voxel value $\widehat{x_j}$ might be negative. In this case it is set to zero, as negative values are meaningless.

When the algorithm has run for a predetermined number of iterations or total residual is less than a specified threshold the algorithm terminates and we have our result.

4.3.4 ART Variations

Over the years many flavours of ART have been developed. For the sake of brevity we will not go over the details of the various flavours. For more details on ART variants see [Herman, 2009, Slaney and Kak, 1987, Andersen and Kak, 1984]. The variants differ mostly in the way voxels are updated: for each ray or after every iteration, the way weights are assigned, and whether the relaxation factor is constant or dynamic. Also some of the algorithms have special treatments for reducing noise by using bilinear elements instead of pixels, as well as many other heuristics for improving the reconstruction results and accuracy.

4.3.5 An Underdetermined System

Our setup can be categorized as *limited view tomogoraphy*, which generally produces more unknowns than equations. This situation is called an *underdetermined system*, meaning the problem has more than one possible solution. Historically, some effort has been done to improve solution quality by using priors, see [Hanson and Wecksung, 1983, Hanson, 1993] but we have not pursued this path. A future direction for improving reconstruction quality can be to apply priors on bubble

shape, dimension, density etc.

4.3.6 Ray Extension

Each ray is associated with a single camera pixel, the measurement of pixel intensity represents the sum of all voxel intensities that lie on the ray's path. Since rays inside the cylinder travel in straight lines, we extend the ray coordinates from the front and rear planes until they intersect the cylinder coordinates along the same straight line equation.

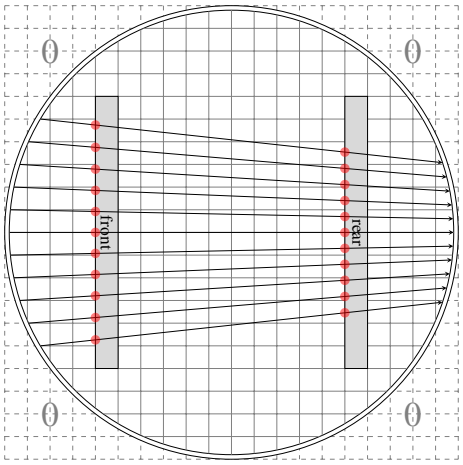


Figure 4.3: Ray extension within cylinder bounds. Given two points (in red) for each ray representing the intersection with the front and rear planes, we extend the rays to intersect with cylinder boundaries. Any voxel within the cylinder may contain information while voxels outside of the cylinder are known to be empty.

4.3.7 Visual Hull

The reconstruction algorithm takes a considerable time to converge. To alleviate this process we use the visual–hull that specifies which rays and voxels contain relevant information and which do not. The process begins by selecting all the rays which produce a zero intensity sum $p_i = 0$. All voxels v_i that lie on the ray's path $(x,y) \in r_i$ must contain an intensity value of zero since the total intensity sum is zero and $\forall i, v_i \geq 0$. These voxels' values are set to zero $f(x,y) = 0$, and they are

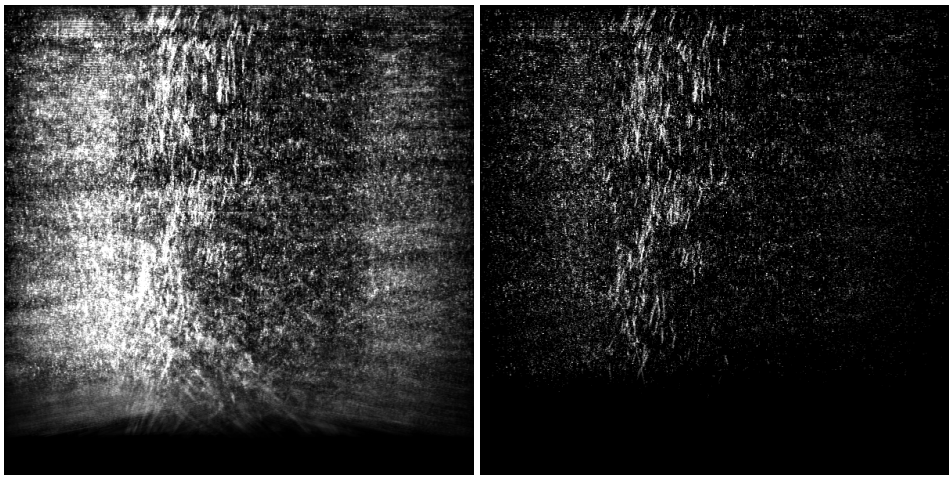
marked as non participating in the back–projection process, so that residuals are not distributed to them. This enables faster convergence and reduces artifacts since all marked voxels remain empty throughout the iterations.

4.4 Post Reconstruction

At this point we have obtained a three dimensional reconstruction of the bubbly flow. We have a volume where each voxel contains an intensity value. Our next task is to trace the trajectories. Trajectories can be interpreted as connected voxels whose intensity level is above a certain value indicating the presence of a bubble's path.

4.4.1 Applying a Threshold

The first step therefore is to apply a threshold to the volume voxels in order to get rid of reconstruction artifacts. The threshold zeros out voxels containing low intensity values which are left over from the iterative nature of the reconstruction procedure. Unfortunately, according to our knowledge, there is no automatic method to decide upon the desired threshold value that can safely reduce artifacts while maintaining reconstruction fidelity. Thus we have to visually inspect the volume and decide which value seems good enough for the streaks to be visible while reducing the artifact noise to a minimum. As is always the case when applying a threshold, care must be taken not to use a value which is too high as this will cause important voxels (non noisy ones) to be dropped out (false negatives). The value we chose is one that reduces noise considerably while preserving connected streak patterns as can be seen in Figure 4.4.

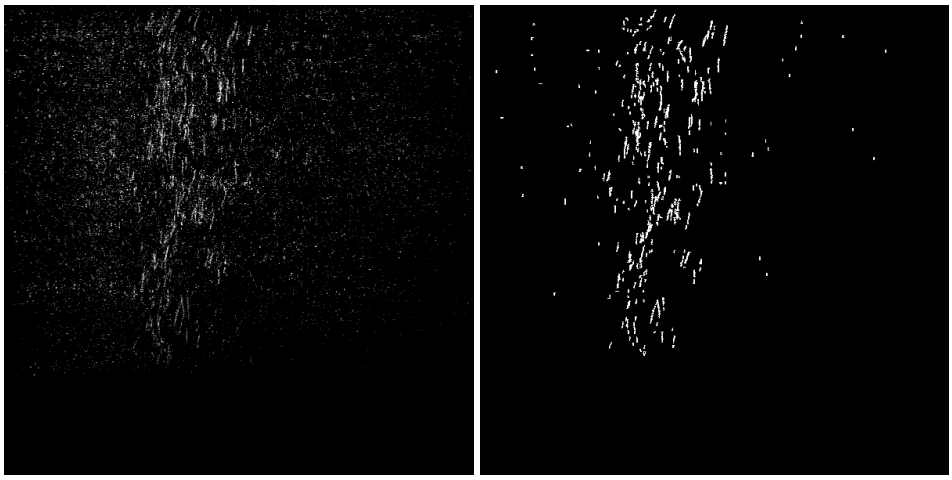


(a) Original reconstruction **(b)** Top 0.1% high intesity voxels

Figure 4.4: Reducing low intensity reconstruction noise by thresholding. Figure 4.4b still contain some noise, this will be reduced later on.

4.4.2 Connected Components

The next step is to associate neighbouring voxels as belonging to the same streak. This is done by running a connected component labeller on the thinned streaks. The labeller issues a distinct numeric value to each streak. Theoretically, two or more streaks can intersect and will thus be labelled as a single streak but this rarely happens in our setup. Had this been a real concern we could have analyzed the intersection point and separate the streak into distinct streaks by comparing the gradients before and after the intersection.



(a) Before **(b)** After

Figure 4.5: The output of running the connected-components filter (on the right). The results are showing components with 10 pixels or more.

4.4.3 Thinning

Once the threshold has been applied to the volume and noise reduced, the streaks appear as thick elongated blobs. The next step is to perform 3D thinning so that only the core of the streaks remain. As this functionality was originally missing from our toolkit, it was consequently implemented based on the work of [Ma and Sonka, 1996, Wang and Basu, 2007]. We have looked at several solution to the 3D thinning problem and selected this one as it seemed to handle most cases more thoroughly than the rest. The method and the improvement boil down to an iterative scheme that removes voxels by looking–up their neighbourhood in a predefined dictionary. Neighbourhoods that are found in the dictionary cause the voxel to be set to zero. This continues until the last iteration completed did not have any voxels that were zeroed, signalling a convergence and the termination of the thinning algorithm.

4.5 Implementation

The implementation of most of the code mentioned above was done in C++ using an in–house toolkit called MDA. MDA provides functionality to manipulate arrays

of any dimension. Much of the support code for the functionality in this chapter had to be implemented as it was lacking from MDA. More efforts went into completeness and correction than into performance tuning. Many of the above routines can be parallelized as well as modified to take advantage of GPUs, but this was not our goal. The main goal was producing a complete reconstruction workflow leaving fine–tuning for a later stage.

In this chapter we have seen how the raw data obtained in the calibration and capture steps are used to reconstruct streak patterns. Next we shall look at the problem of how to determine velocity from the reconstructed data.

Chapter 5

Fluid Velocity Estimation

The previous chapter completed the reconstruction process and left us with three dimensional representations of streaks. In this chapter we produce velocity estimation based on the reconstructed streaks. Our problem lives in the domain of two phase flows: gaseous (air bubble) flow and liquid (water) flow. The complete fluid dynamics of our system are characterized by both the bubbly flow and fluid flow. The two are related but not identical.

Instantaneous bubble velocities are obtained from data gathered during trajectory reconstruction. On the other hand, the fluid velocity field is typically determined by using tracer particles. In this chapter we show how fluid interior velocity is estimated from measurements on such passive advected particles. Since our current data collection only provides bubble tracking, we use the available bubble velocity vectors as if they were tracer particle velocities to reconstruct the fluid interior velocity. We acknowledge the fact that bubbles do not behave as passive advected particles, thus the resulting fluid velocities are inaccurate. However, once measurements are performed using proper tracer particles, applying the exact same procedure will produce the sought after results. At this point the two phase velocity field reconstruction is complete.

5.1 Fluid Velocity

One of the key goals of PTV systems is to measure or infer the fluid velocity within the medium where the particles are travelling. For this to be achieved a good control of particle density and density uniformity is required. These are typically needed in order to match particles between frames to determine particle trajectory, which is in turn used to calculate the velocity.

5.2 Bubble Velocity

Our setup differs from other PTV setups (for instance the recent implementation in [Matsumoto et al., 2012]) since bubble control is quite limited. In addition to that bubble velocity is not under our direct control as we have no means to apply pressure to the water cylinder. Thus, bubbles are flowing freely from the diffuser at the bottom to the water surface top. The consumer quality video cameras also pose restrictions on the exposure duration. The end result is that we are unable to produce sharp looking bubble images, where bubbles appear as separate points. Instead we obtain images where bubbles appear as streaks. This temporal blur can be used to our advantage: instead of locating matching bubbles as points between image sets, each bubble appears as a separate streak and bubble velocity is manifested as streak length. In contrast, PTV locates the positions of the same bubble in two successive frames and uses the displacement information to calculate velocity.

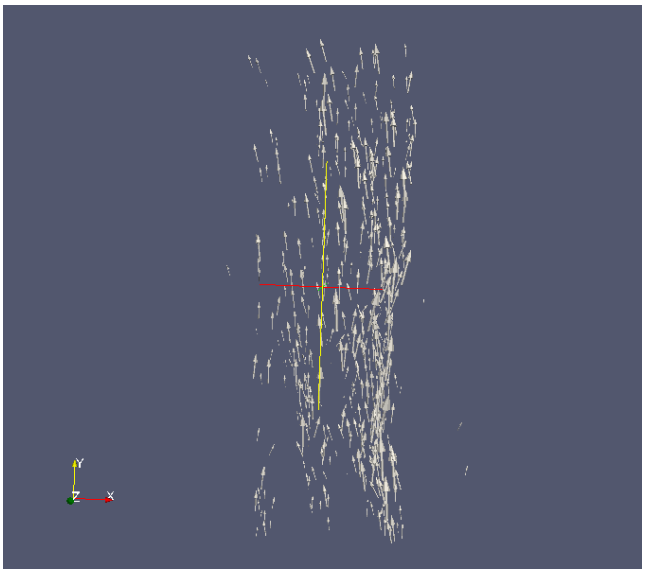


Figure 5.1: Velocity vectors in volume. This vector field represents bubble velocity magnitudes constructed from 4 consecutive frames. Note the imperfections in air diffuser as it is releasing air from several clusters (bottom) instead of an 'air curtain'.

5.2.1 Estimating Fluid Velocity from Particle Velocity

In order to obtain the complete dynamics of a two phase flow system, we need to track both the bubbles and the fluid itself. Tracking the fluid is traditionally done by using smaller tracer particles that move along with the fluid flow (see [Bröder and Sommerfeld, 2000]). [Seeger et al., 2001] used X–ray in conjunction with regular cameras to obtain the fluid dynamics using X–ray absorbing particles.

Unfortunately, our setup does not currently allow tracking tracer particles. We have tested different materials but did not find any of them suitable for water flow tracking: most would either float or drown too quickly. Since we preferred to capture fast moving bubbles, we did not consider other transparent fluids such as Glycerin, which is more viscous and has a wider availability of tracer particles. Therefore we have limited our captures to tracking bubble trajectories only. In the remainder of this chapter we treat the bubble samples as if they were samples of tracer particles. There is non negligible difference between fluid and gas veloci-

ties due to the density differences and surface tension between the gaseous/liquid layers. However, the same procedure applies for both when building the velocity vector field. Of course the resulting velocity field obtained from bubbles will not be equal to the one we ideally wish to reconstruct, however it can provide us with a clue of how fluid dynamics work.

Fluid close to the particle is being dragged along with the particle by the wake created by the bubbly flow moving upward. Tracking the trajectories of tracer particles allows reconstructing the neighbouring fluid velocity. Using this data as boundary conditions we can estimate the overall fluid velocity field everywhere in our volume.

5.2.2 Flow Theory

Flow is usually categorized as either laminar, turbulent or in transition. Flow type is governed by the Reynolds Number, which is defined as:

$$Re = \frac{\rho U_t d_0}{\eta}$$

where ρ is the fluid density, U_t is the terminal velocity of the bubble, d_0 is the bubble diameter and η is the liquid viscosity. Given our measurements, we obtain large Reynolds Numbers (magnitude of 10^5) indicating highly turbulent flows. This result implies that we cannot reliably estimate fluid velocity from bubble velocity alone.

5.3 The Minimization Problem

Taking a note of our incomplete fluid measurements, we now continue to describe our implementation for velocity estimation in the volume interior based on the sparse bubble measurements. The same implementation would apply had we obtained proper particle trajectories.

Since we are dealing with incompressible flows, we assume the flow is divergence free or $\nabla \cdot U = 0$. Instead of estimating the velocity directly, we use the velocity potential ψ which is defined as $\nabla \times \psi = U$. Applying a curl operator here has the effect of restricting the velocity field to be divergence free.

Given a set of measured particle (bubble) velocities which are used as boundary conditions $\hat{U} = \{u_1, u_2, \ldots, u_n\}$ at positions $X = \{x_1, x_2, \ldots, x_n\}$ we want to estimate the continuous spatially varying velocity field U within the fluid. The sparse samples yield an underdetermined problem which has more than one solution. We use least–squares minimization to find the velocity field. Next we define the energy which is used for minimization. Our goal is to keep the velocity field close to the original measurements for points that contain the predetermined velocity boundary–condition vectors, thus:

$$E_{bc}(U(x)) = \int_{x \in X} \|U(x) - \hat{U}(x)\|_2^2 \, dx$$

This leads to an interpolated field that tends to be very similar to the known boundary–condition samples, however we also want the velocity field to be smooth in between the samples. We enforce smoothness across all the domain Ω by adding another component to the overall energy to be minimized.

$$E_s(U(x)) = \int_{x \in \Omega} \|\nabla^2 U(x)\|_2^2 \, dx$$

This produces smoothness across the entire domain. In order for these two energy functions to be useful for minimization we introduce a weighted linear combination of both functions. The weights can be tweaked in order to tune the algorithm to prefer sample fitting vs. smoothness.

$$E(U(x)) = w_{bc} E_{bc}(U(x)) + w_s E_s(U(x))$$

Another component we found useful is overall kinetic energy. This should take care of over–fitting noisy samples by preferring fields with low kinetic energy.

$$E_k(U(x)) = \int_{x \in \Omega} \|U(x)\|_2^2 \, dx$$

The weight associated with this component will typically be much smaller. The combined energy function is:

$$E(U(x)) = w_{bc} E_{bc}(U(x)) + w_s E_s(U(x)) + w_k E_k(U(x))$$

At this point we incorporate the divergence free constraint and change our energy functions to use the velocity potential ψ. Using the energy functions above we obtain:

$$\begin{aligned} E_{bc}(\psi(x)) &= \int_{x\in X} \|\nabla \times \psi(x) - \hat{U}(x)\|_2^2 \, dx \\ E_s(\psi(x)) &= \int_{x\in\Omega} \|\nabla^2(\nabla \times \psi(x))\|_2^2 \, dx \\ E_k(\psi(x)) &= \int_{x\in\Omega} \|\nabla \times \psi(x)\|_2^2 \, dx \end{aligned} \tag{5.1}$$

The total weighted energy sum becomes:

$$E(\psi(x)) = w_{bc}E_{bc}(\psi(x)) + w_s E_s(\psi(x)) + w_k E_k(\psi(x)) \tag{5.2}$$

5.3.1 Discretization

In order to solve Equation 5.2 we need to discretize our equations. This calls for selecting basis functions, in this case we choose finite–differences. The Laplace and curl operators are discretized using centered differences. There are of course other possibilities however our selection is relatively simple to implement. The integrals become sums over the fluid interior. The discrete curl ($\nabla\times$) is denoted as C and the Laplacian (∇_2) as L. The velocity potential function becomes $\Psi = [\psi_1, \psi_2, \ldots, \psi_n]^T$. The boundary conditions do not need discretization as they are already represented by samples. In addition to the velocity measurements we set the boundary–conditions to be zero everywhere outside the cylinder.

$$\begin{aligned} E_{bc}(\Psi) &= \sum_{i=1}^{m} \|C\psi_i - \hat{u}_i\|_2^2 \\ E_s(\Psi) &= \sum_{i=1}^{n} \|L(C\psi_i)\|_2^2 \\ E_k(\Psi) &= \sum_{i=1}^{n} \|C\psi_i\|_2^2 \end{aligned} \tag{5.3}$$

The corresponding total energy remains as it was before, a weighted linear combination of the energy components:

$$E(\Psi) = w_{bc}E_{bc}(\Psi) + w_s E_s(\Psi) + w_k E_k(\Psi) \tag{5.4}$$

5.3.2 The Solver

In order to solve Equation 5.4 we utilize the L–BFGS method. This is an iterative quasi–newton optimization method with low memory requirements, which is critical for solving large volumes. It has a theoretical quadratic convergence rate. As an input L–BFGS requires computing the energy gradient $\nabla E(\Psi)$ for every iteration taken.

The derivatives used by L–BFGS for each of the energy components are:

$$\begin{aligned} \nabla E_{bc}(\Psi) &= \sum_{i=1}^{m} 2C^T(C\psi_i - \hat{u_i}) \\ \nabla E_s(\Psi) &= \sum_{i=1}^{n} 2L^T(L\psi_i) \\ \nabla E_k(\Psi) &= \sum_{i=1}^{n} 2C^T C\psi_i \end{aligned} \tag{5.5}$$

5.3.3 Results

Fluid velocity estimation requires the use of tracer particles to track fluid motion. Because we were unable to perform tracer particles captures the results below were obtained using the bubble trajectories. Although bubble dynamics are different than fluid (tracer particle) dynamics, when it comes to velocity estimation the same procedure applies for both. We now proceed with velocity estimation based on bubbly flows, which should be treat as tracer particles.

Due to the relatively large dimensions ($150 \times 175 \times 150$ voxels) of our velocity vector grid, the L–BFGS algorithm requires a large number of iterations in order to achieve a solution that is smooth enough in the points that do not hold boundary–conditions while maintaining similar values to the points that do. Our data is quite sparse as bubbles appear to travel in very specific paths leaving large portions of the volume's interior without any measurements. We calculate the root–mean–square error every 10 iterations for points that have corresponding boundary–conditions only. The root–mean–square error defined as:

$$E_{RMS} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(U_i - \nabla \times \psi_i)^2} \tag{5.6}$$

This error measurement is calculated by averaging the sum of difference be-

tween the boundary–condition velocities to the velocity in the resulting vector–field. Only points that originally contain boundary–condition values take part in the sum.

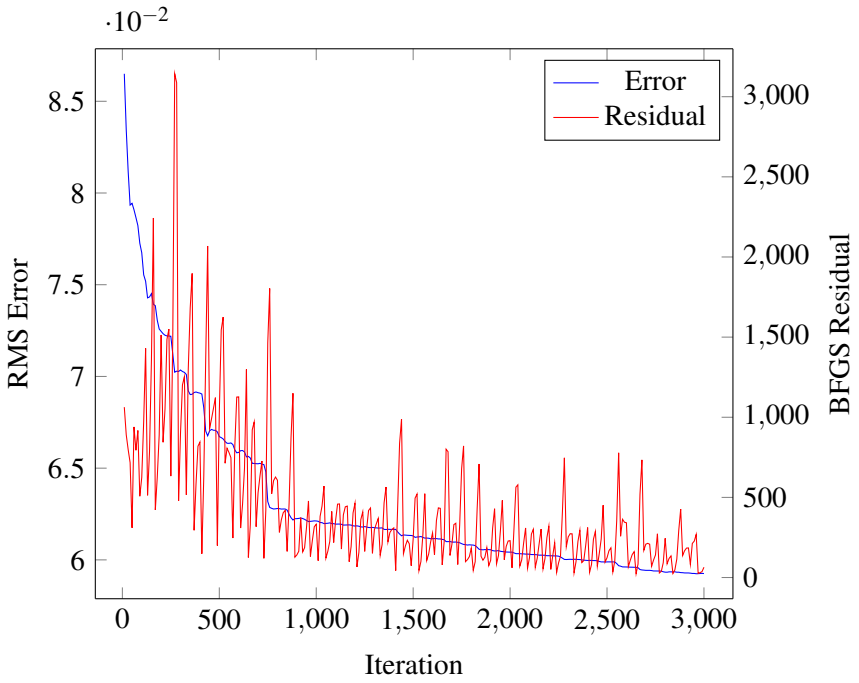


Figure 5.2: Error and residuals for a smaller grid. After 1000 iterations or so the error does not decrease as fast as it initially did. The overall residual (energy) as defined above continues to vary as the optimizer continues looking for a solution.

As is evident from Figure 5.2 the RMS error tends to stabilize after several hundred iterations. The only benefit from running more iterations is to smooth out the flow field into the fluid interior where no prior boundary–conditions were applied. With proper fluid interior velocity samples, smoothing produces the desired solution approximation. The optimizer can be stopped once a predetermined energy limit has been reached or if the fluid interior velocity vectors do not change much between iterations.

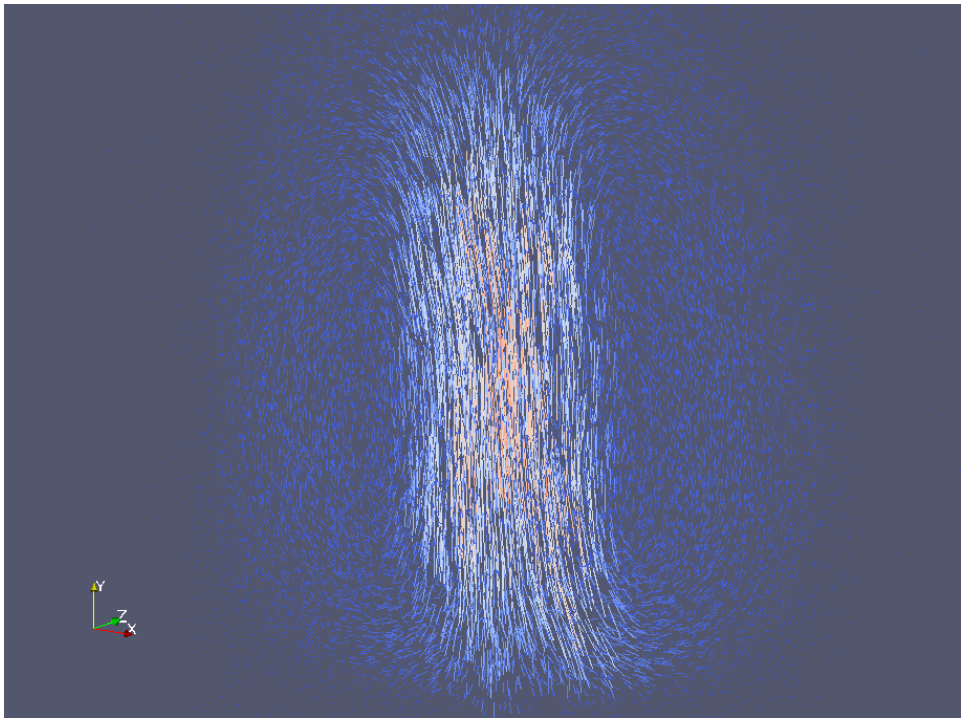


Figure 5.3: Fluid interior velocity vectors. Red for higher velocity magnitude, moving upward and blue for low magnitude, moving downward (direction indicators removed for the sake of clarity). This reconstruction from bubble velocity vectors is inaccurate overall as it was not done from tracer particle velocities but it does offer a clue to how flow in the fluid interior looks like. Note how the slower fluid flow is directed downward in parts remote from the upward bubbly flow in the centre.

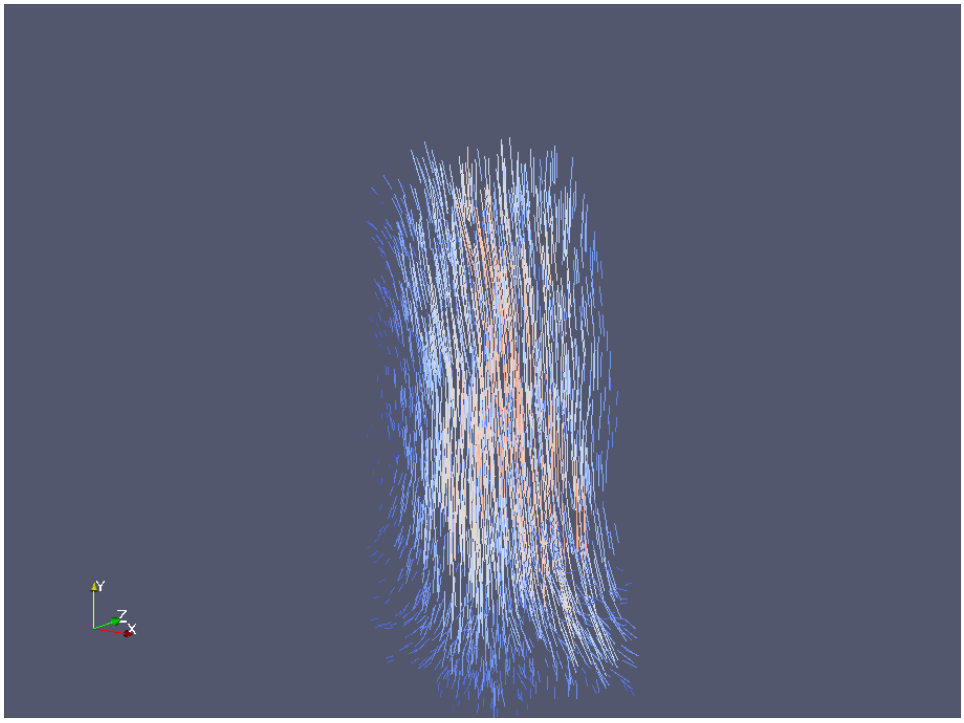


Figure 5.4: Velocity vectors restricted to bubble proximity. Velocity vectors that are further away from bubble position are masked out. If we work under the assumption that fluid in close proximity to the bubbly flows is moving in similar velocities, this vector field suggest how the fluid behaves.

Chapter 6

Conclusions

The investigation of bubbly flows is an ongoing research area. In this work we have seen how a simple consumer grade camera array can be effectively used to reconstruct bubbly flows in fluids. The bubble trajectories can be traced and bubble position and velocity can be obtained by relatively simple means. Similarly, using data obtained from capturing tracer particles we should be able to reconstruct fluid flow.

We have shown how long exposure frames, which are generally avoided due to motion blur, can be used to our advantage, to obtain both the bubble trajectories and bubble velocities. These elongated streaks can be combined to provide the full path a bubble takes, with velocity measurements along the way. The same procedure applied to tracer particles is used to obtain the fluid dynamics, then by utilizing finite–differences an optimization problem yields the dynamics of the fluid interior.

6.1 Future Work

Here we look into some improvements and research directions where this project can be taken to.

6.1.1 Fluid Tracer Particles

Fluid dynamics in a multiphase flow cannot be complete without proper measurements of fluid flow. We have covered gaseous dynamics in terms of position and velocity, by reconstructing bubbly flow from measurements. The next step is acquiring similar data for tracer particles to obtain a fluid velocity field. Although these particles are usually opaque, their miniature dimensions should not require modifications to the reconstruction procedure. The cameras we used as well as the reconstruction volume dimensions allowed a resolution of about $0.25mm$ and should be sufficient for sub $1mm$ tracer particles. The particles should have a distinctive colour so they can be distinguished from bubbles which appear as white blobs or streaks. Different colours can be used as an aid during reconstruction as was suggested by [Bendicks et al., 2011].

6.1.2 Tomographic Reconstruction Improvements

As previously mentioned, there have been several efforts in the past to incorporate various types of constraints into the tomographic reconstruction procedure in order to find a more accurate solution for this over–determined problem. Incorporating domain knowledge can improve reconstruction results albeit require considerable modifications and more sophisticated algorithms.

Incorporating Incompressible Flow Constraints

A very interesting approach has been recently applied to fluid tomography. [Nemirovsky et al., 2011] suggested incorporating flow incompressibility as a constraint into reconstruction. This, according to their paper, produces higher fidelity reconstructions. The method, Incompressible Fluid Tomography, relies on capturing successive temporal frames. They choose one frame to be the anchor frame, then step from one point in time to the next while applying density moment constraints.

Algorithmic Variants

ART has many variants, we have chosen one that produced good results. More effort can be put into a comprehensive comparison between the numerous methods.

Methods that work well for sparse samples should work best for this problem.

6.1.3 Varying Bubble Dimensions

The aerator used in this work produced bubbles of about $1-2mm$ in diameter. These were not accurately measured, the sizes were estimated from captured images. Reducing bubble dimensions is possible as long as it does not reach the capture resolution (about $0.25mm$). Increasing bubble diameter beyond a certain size will produce bubbles that are no longer a perfect sphere, they transform into a concave shape when rising up. In addition to that, partial specular light reflections will be visible due to the top illumination. All of this will make it harder to distinguish between the different trajectories.

Using a fine controlled aerator we could modify the air pressure to produce different bubble densities as well as modify the bubbly flows.

6.1.4 Verification of Flow Models

We have manually verified the resulting trajectories and velocities magnitudes are in a similar range to the ones that result from measurements. Next, it will be interesting to compare the velocities and trajectories with the theory of multiphase flow fluid dynamics.

Bibliography

M. Gharib, D. Kremers, M. Koochesfahani, and M. Kemp. Leonardo's vision of flow visualization. *Experiments in fluids*, 33(1):219–223, 2002. → pages 1

A.K. Prasad. Stereoscopic particle image velocimetry. *Experiments in fluids*, 29 (2):103–116, 2000. → pages 3

T. Hori and J. Sakakibara. High-speed scanning stereoscopic piv for 3d vorticity measurement in liquids. *Measurement Science and Technology*, 15(6):1067, 2004. → pages 3

V.V. Ranade. Modelling of turbulent flow in a bubble column reactor. *Chemical Engineering Research and Design*, 75(1):14–23, 1997. → pages 5

W.D. Deckwer and A. Schumpe. Improved tools for bubble column reactor design and scale-up. *Chemical Engineering Science*, 48(5):889–911, 1993. → pages 5

R. Gordon, R. Bender, G.T. Herman, et al. Algebraic reconstruction techniques (art) for three-dimensional electron microscopy and x-ray photography. *Journal of Theoretical Biology*, 29:471–481, 1970. → pages 6, 7, 30

M. Raffel, C.E. Willert, and J. Kompenhans. *Particle image velocimetry: a practical guide*. Springer Verlag, 1998. → pages 6

C. Brossard, J.C. Monnier, P. Barricau, F.X. Vandernoot, Y. Le Sant, F. Champagnat, and G. Le Besnerais. Principles and applications of particle image velocimetry. *Aerospace Lab journal*, 2009. → pages 6

G.E. Elsinga, F. Scarano, B. Wieneke, and B.W. Van Oudheusden. Tomographic particle image velocimetry. *Experiments in Fluids*, 41:933–946, 2006. → pages 7

G. Elsinga, B. Wieneke, F. Scarano, and A. Schröder. Tomographic 3d-piv and applications. 112:103–125, 2008. → pages 7

F. Scarano and C. Poelma. Three-dimensional vorticity patterns of cylinder wakes. *Experiments in Fluids*, 47:69–84, 2009. ISSN 0723-4864. → pages 7

C.H. Atkinson and J. Soria. Algebraic reconstruction techniques for tomographic particle image velocimetry. In *16th Australasian Fluid Mechanics Conference*, pages 191–198, 2007. → pages 7

T. Matsumoto, S. Hosokawa, and A. Tomiyama. Measurements of bubble velocity using spatial filter velocimetry. In *Proc. 16th International Symposium on Applications of Laser Techniques to Fluid Mechanics, Lisbon, Portugal*, 2012. → pages 7, 40

D. Bröder and M. Sommerfeld. A piv/ptv system for analysing turbulent bubbly flows. In *Proceedings of*, volume 10, 2000. → pages 7, 41

A. Seeger, K. Affeld, U. Kertzscher, L. Goubergrits, and E. Wellnhofer. Assessment of flow structures in bubble columns by x-ray based particle tracking velocimetry. *Proc. 4th Int. Sym. Particle Image Velocimetry, Göttingen, Germany*, 2001. → pages 7, 41

J. Radon. On the determination of functions from their integral values along certain manifolds. *Medical Imaging, IEEE Transactions on*, 5:170–176, 1986. → pages 7

S. Dubsky, R.A. Jamison, S.C. Irvine, K.K.W. Siu, K. Hourigan, and A. Fouras. Computed tomographic x-ray velocimetry. *Applied Physics Letters*, 96, 2010. → pages 7

J. Sharpe, U. Ahlgren, P. Perry, B. Hill, A. Ross, J. Hecksher-Sørensen, R. Baldock, and D. Davidson. Optical projection tomography as a tool for 3d microscopy and gene expression studies. 296(5567):541–545, 2002. → pages 8

J. Sharpe. Optical projection tomography as a new tool for studying embryo anatomy. *Journal of Anatomy*, 202(2). ISSN 1469-7580. → pages 8

B. Trifonov, D. Bradley, and W. Heidrich. Tomographic reconstruction of transparent objects. In *Proc. Eurographics Symposium on Rendering*, pages 51–60, 2006. → pages 9

B. Atcheson, I. Ihrke, W. Heidrich, A. Tevs, D. Bradley, M. Magnor, and H.P. Seidel. Time-resolved 3d capture of non-stationary gas flows. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 27(5):132, 2008. → pages 9

J.A. Puleo, R.V. Johnson, and T.N. Kooney. Laboratory air bubble generation of various size distributions. *Review of Scientific Instruments*, 75(11):4558–4563, 2004. → pages 11

D. Bradley, B. Atcheson, I. Ihrke, and W. Heidrich. Synchronization and rolling shutter compensation for consumer video camera arrays. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 1–8. IEEE, 2009. → pages 12, 23

B. Atcheson, F. Heide, and W. Heidrich. Caltag: High precision fiducial markers for camera calibration. In *Proc. VMV*, pages 41–48, 2010. → pages 15, 19

G.T. Herman. *Fundamentals of computerized tomography: image reconstruction from projections*. Springer, 2009. → pages 33

M. Slaney and A. Kak. Principles of computerized tomographic imaging. 1987. → pages 33

A.H. Andersen and A.C. Kak. Simultaneous algebraic reconstruction technique (sart): a superior implementation of the art algorithm. *Ultrasonic imaging*, 6 (1):81–94, 1984. → pages 33

K.M. Hanson and G.W. Wecksung. Bayesian approach to limited-angle reconstruction in computed tomography. *JOSA*, 73(11):1501–1509, 1983. → pages 33

K.M. Hanson. Bayesian reconstruction based on flexible prior models. *JOSA A*, 10(5):997–1004, 1993. → pages 33

C.M. Ma and M. Sonka. A fully parallel 3d thinning algorithm and its applications. *Computer vision and image understanding*, 64(3):420–433, 1996. → pages 37

T. Wang and A. Basu. A note on a fully parallel 3d thinning algorithm and its applications. *Pattern Recognition Letters*, 28(4):501–506, 2007. → pages 37

C. Bendicks, D. Tarlet, C. Roloff, R. Bordás, B. Wunderlich, B. Michaelis, and D. Thévenin. Improved 3-d particle tracking velocimetry with colored particles. *Journal of Signal and Information Processing*, 2(2):59–71, 2011. → pages 50

J. Nemirovsky, A. Lifshitz, and I. Be'Ery. Tomographic reconstruction of incompressible flow. *Review of Scientific Instruments*, 82(5):055115–055115, 2011. → pages 50