NTNU

Norwegian University of
Science and Technology

# Modeling and data assimilation for improved ultrasound measurement of blood velocity fields

Thomas Grønli

Master of Science in Physics and Mathematics
Submission date: March 2017
Supervisor: Catharina de Lange Davies, IFY
Co-supervisor: Lasse Løvstakken, ISB

Norwegian University of Science and Technology
Department of Physics

# Abstract

Cardiovascular disease is the leading cause of death worldwide, and echocardiography stands as a fundamental tool in assessing congenital or developed heart failure. A cornerstone of echocardiography is the ability to obtain robust quantitative and qualitative blood flow measurements with methods such as pulsed-wave Doppler or color flow imaging and the development of these techniques has had a major impact on the quality of cardiovascular diagnostics worldwide. Modern vector flow imaging has opened the estimation of 3D intracardiac blood motion providing a detailed map of the flow structures inside a beating heart.

However, vector flow imaging techniques suffer from limited signal-to-noise ratio, and the overall accuracy is highly sensitive to noise corruption in the measurements, decreasing the overall robustness of these methods.

In this project we investigate the possible benefits of using a model-based filter to improve the quality of blood flow estimates through data assimilation by a full numerical simulation of blood motion based on the Navier-Stokes equations. We implement a Smoothed Particle Hydrodynamics simulator suitable for intracardiac flow conditions to perform the forecast stage of the assimilation cycle.

The model was evaluated towards a computational fluid dynamics phantom and showed comparable fields when attempting to forecast the phantom flow. Finally, the model was compared to a static regularisation based on a penalised B-spline grid and was found to be of comparable performance.

We conclude that the obtained particle simulator is plausibly fit to model blood motion in a data assimilation context and is ready to be employed as system dynamics in an information fusion filter capable of operating under arbitrary dynamics such as the extended or unscented Kalman filter. We hypothesise that this application will improve the quality of such filters compared to simpler models seen so far.

# Sammendrag

Hjerte- og karsykdommer er den ledende dødsårsaken i verden, og ekkokardiografi er et grunnleggende verktøy for å vurdere medfødt eller utviklet hjertesvikt. En hjørnestein i ekkokardiografi er evnen til å oppnå robuste kvantitative og kvalitative blodstrømsmålinger med metoder som pulset Doppler eller fargedoppler. Utviklingen av disse teknikkene har hatt stor innvirkning på kvaliteten på hjertediagnostikk over hele verden. Moderne vector flow imaging har åpnet for estimering av intrakardiale blodstrømninger i 3D, noe som gir et detaljert kart over strømningsstrukturer inne i et bankende hjerte.

Vector flow imaging-teknikker lider imidlertid av begrenset signal-til-støy-forhold, og nøyaktigheten til estimatene er sensitive for støy som forstyrrer målingene, noe som reduserer den totale robustheten.

I dette prosjektet undersøkte vi de mulige fordelene ved å bruke et modellbasert filter for å forbedre kvaliteten på blodstrømsestimater. Dette ble gjort gjennom dataassimilasjon av en full numerisk simulering av blodets bevegelse basert på Navier-Stokes ligninger. En glattet partikkel-hydrodynamikk-simulator egnet for intrakardiale strømningsforhold ble implementert for å utføre prediksjonssteget i assimileringssyklusen.

Modellen ble evaluert mot et numerisk fluiddynamisk fantom, og i forsøkene på å forutsi strømningene i fantomet ble det funnet sammenlignbare felt. Til slutt ble modellen sammenlignet med en statisk regularisering basert på et P-spline, og også i dette tilfellet var resultatene for de to metodene sammenlignbare.

Vi konkluderer med at den oppnådde partikkelsimulatoren er skikket til å modellere blodstrømninger i en dataassimilasjonkontekst, og er klar for å anvendes som systemdynamikk i et informasjon-fusjon-filter som er i stand til å operere under vilkårlig dynamikk, slik som et utvidet eller unscented Kalman filter. Vi hypotiserer at anvendelse av denne modellen vill forbedre kvaliteten på slike filtre sammenlignet med tidligere, enklere modeller.

# Preface

This work finalises the formal requirements for fulfilment of the degree of Master of Science at the Department of Physics at Norwegian University of Science and Technology (NTNU), Trondheim. The work has been carried out at the Department of Circulation and Medical Imaging.

## Acknowledgements

I direct an immense gratitude to my supervisor Prof. Lasse Løvstakken for invaluable guidance throughout this project. Being able to learn from someone who is at the absolute frontier of research in his field has been an immensely maturing experience. I also want to thank my supervisor at the Department of Physics Prof. Catharina de Lange Davies for her central administrative part in the completion of the project. For the last year and a half I have spent most of my time at the ultrasound group. Seeing all the impressive research and international acknowledgement this group receives makes me immensely proud and honoured to be welcomed into such an exceptionally skilled group of people and I look forward to the next years. To all my friends at the department I have only one thing to say; shooters!

I want to thank my mother and father for their endless support in my pursue of an academic career. The times when the way ahead has seemed long and difficult, knowing I make them proud with what I do has been a strong motivating force. To my brother and sister, who always make home feel like home during the holidays.

At last, to my dearest Silje, for your unending support and love, you make every day brighter for me.

# Contents

# List of Tables

# List of Figures

# Acronyms

**ABI** Application Binary Interface. 39

**BST** Blood Speckle Tracking. xiii, 39, 40, 43, 47, 48, 56, 57, 62

**CFD** Computational Fluid Dynamics. 2, 21, 39, 40, 43–46, 52, 60, 61, 69

**CFI** Color Flow Imaging. 1, 6, 59

**CFL** Courant–Friedrichs–Lewy. 17, 19, 61

**CPU** Central Processing Unit. 19, 22, 24, 32, 62

**CSR** Compressed Sparse Row. 39

**CVD** Cardiovascular Diseases. 1

**EKF** Extended Kalman Filter. 62

**EOS** Equation of State. 15, 19, 27, 33, 62

**FD** Finite Differences. 30, 31

**FEA** Finite Element Analysis. 20–22

**FIR** Finite Impulse Response. 6

**FLIP** Fluid Implicit Particle. 8, 9, 33, 68

**FUSK** Fast Ultrasound Imaging Simulation in K-space. 40, 43, 47, 48

**GPU** Graphic Processing Unit. 2, 19, 20, 22, 24, 25, 27–32, 38, 39, 63, 66, 67

**GT** Ground Truth. 40, 50, 52

**IISPH** Implicit Incompressible Smoothed Particle Hydrodynamics. 63

**IQ** In-phase Quadrature. 6, 40

**ISPH** Incompressible Smoothed Particle Hydrodynamics. 8, 19, 20

**LS** Least Squares. 22–24, 34, 36, 38

**MLS** Moving Least Squares. 14

**NNPS** Nearest Neighbour Particle Search. 18

**ODE** Ordinary Differential Equation. 14, 31

**PCISPH** Predictive Corrective Smoothed Particle Hydrodynamics. 20, 63

**PIC** Particle in Cell. 8, 33

**PSF** Point Spread Function. 40

**PW** Pulsed-Wave. 1

**SNR** Signal-to-Noise Ratio. 1

**SOI** Scale of Interest. 12

**SPH** Smoothed Particle Hydrodynamics. xi, xiii, 2, 8–11, 13, 15, 17–22, 33, 40, 43, 49, 50, 52, 54–56, 59–63, 65, 66, 69

**UKF** Unscented Kalman Filter. 62

**VFI** Vector Flow Imaging. 6

**WCSPH** Weakly Compressible Smoothed Particle Hydrodynamics. 8, 19, 20, 27, 63

# 1 Introduction

## 1.1 Motivation

Cardiovascular Diseases (CVD) is the single highest cause of death worldwide (MPN+11). It is believed that 90% of all cases of CVD can be prevented by life-style choices and earlier diagnostics can help before acute stages of the diseases are reached and chronic heart failure occurs. There has been signs that many of these diseases indeed can be reversed by proper post-diagnostic measures(WSL+07).

It is particularly believed that the vortex structures inside the heart may contain important information for unlocking new insight into cardiovascular pathology(PLCAT14). This urges the need for robust quantitative methods for flow field estimation in CVD-diagnostics.

For a long time Doppler has been a robust *de facto* standard for blood flow, and methods such as Pulsed-Wave (PW) Doppler and Color Flow Imaging (CFI) have been significant factors in increasing the accuracy of cariovascular diagnostics.

Recent advancements in ultrasound have enabled the The measurement of 3D blood motion using angle-independent speckle tracking or vector-Doppler approaches with promise for providing more detailed measurements of blood velocity patterns. Indeed, many links between flow abnormalities and CVD have already been discovered with such methods(MNW+09). However, due to a reduced Signal-to-Noise Ratio (SNR) related both to the ultrafast acquisition scheme employed as well as inherent estimator properties, the measurements can be highly corrupted by noise and require substantial spatial and temporal averaging. Further, required clutter filtering for causes measurement dropouts during parts of the cardiac cycle.

## 1.2 State of the art and related work

Many methods for improving blood flow estimates have been developed in recent years. We mention notably the B-spline regularisation grid due to Gomez(GdVJ+15) adapted by Grønli(GSN+16). Tura(TSGL99) and lastly a model-based filter by

Høgenes(HWN+16).

The Kalman filter displayed promising results using a simplified advection model based on the Navier-Stokes equation and forms the basis for the thesis project.

## 1.3 Statement of contribution

This project will investigate the potential benefits of model-based blood velocity estimation using a data assimilation approach. An unscented Kalman filter combines measurement sources and outputs a model-based estimate, but its performance is highly dependent on an accurate blood motion model. In this project the motion model will be replaced with a full Computational Fluid Dynamics (CFD) simulation using a fast computational approach based on SPH. This is a mesh-less and highly parallelisable approach which allows for flexible definition of boundary conditions. Given the data assimilation approach with a relatively high support of measurements, we hypothesise that the accuracy of SPH will be sufficient in this context. We further hypothesise that the incorporation of a SPH model will increase the accuracy and speed of 3D model-based estimation and regularisation of flow velocity fields.

## 1.4 Scope of the project

In this project, we aim to develop a fast 3D smoothed particle hydrodynamics simulator suitable for representing cardiac ventricular flows and boundary conditions. The simulator will have high support for data driven system evolution through information obtained through continuously updated input from estimators such as speckle tracking, (vector-)Doppler and B-mode.

The simulator should be tailored towards a Kalman filter application of the provided model. Further, the model should be thoroughly evaluated the SPH approach used as a model for predicting blood motion in the Kalman filter context

The project will have focus on obtaining a performant SPH code in order to investigate a (near-)realtime bedside feasibility given the hardware installed in clinical scanners today. This should be achieved by targeting high performance Graphic Processing Units (GPUs), which is well known to greatly accelerate parallel codes such as SPH.

For comparison to established method, the project will partly involve extending a 2D B-spline based framework for blood flow regularisation developed earlier. This framework has already been applied in a study in collaboration with the Sick-Kids Hospital in Toronto and should serve reasonably well as a background of the current state-of-the-art blood flow regularisation.

## 1.5 Outline

The thesis is organised into the following chapters

**Background** The relevant background material providing the basis for the method is presented in this chapter.

**Methodology** Implementation details surrounding the chosen model is presented along with the simulations setups used to evaluate the model.

**Simulations** The chapter presents the results of the simulations performed with the particle simulator along with a comparison towards a B-Spline smoother.

**Discussion** A discussion on the achieved results is presented along with an insight into the current challenges in the implementation. Suggestions for further work on this project follows and concludes this thesis.

# 2 | Background

In this chapter, an overview of the relevant background information and theory is provided. The topics will be covered in sufficient depth to motivate the methods used in this project, but we refer to the relevant literature for a more thorough treatise on the individual topics.

## 2.1 Ultrasound modality

Ultrasound is the concept of image formation through insonification of a volume with acoustic energy. Ultrasonography has long been a valuable tool for clinicians in a wide range of medical fields, most notably obstetrics, but also cardiology, where the concept of using ultrasound is commonly known as echocardiography. The operational costs of the modality is low compared to many other imaging modalities, and the procedure is quick to perform.

The basis for ultrasonography is the reflection of acoustic waves at interfaces of different bulk modulus. These impart a change of direction or reflection on acoustic waves propagating through the medium. To obtain an image, the region of interest is excited with acoustic energy from a piezoelectric or capacitive transducer. Reflected waves are detected by the transducer after reflection and the depth of origin of the reflection is computed from the measured delay from transmit to receive. From this data, a map of the reflective landscape inside the volume can be obtained. Interfaces are easily recognisable by experienced clinicians as distinct anatomical features, allowing them to examine the patient in a non-invasive and non-ionising manner.

### 2.1.1 Vector flow imaging (VFI)

The single most limiting factor of the standard Doppler postprocessing sequence is that solely the radial component is estimated. Recent developments Vector Flow Imaging (VFI) such as vector-Doppler, transverse oscillations and speckle tracking methods have enabled angle-independent measurements for blood flow fields.

A recent review Jensen *et al.*(JNAG16a)(JNAG16b) provides an exhaustive presentation and discussion of the state of research on recently developed vector velocity estimators and underlying acquisition schemes.

Sub-wavelength scatterers suspended in the plasma creates interference patterns in the wave-data seen as a speckle pattern. Through the movement of the scatterers in blood, the speckle pattern undergoes a continuous deformation detectable in the ultrafast acquisition scheme. This transformation can be estimated by cross correlation to obtain beam angle independent estimates for the blood velocity.

Intracardiac feasibility of a 3D speckle tracking estimator was demonstrated by Wigen and Løvstakken(WL16). The estimator forms the basis for the flow estimation in this project.

### 2.1.2 The Navier-Stokes equation

An integral component of the SPH particle transport is the equations of transport. For fluids this is the Navier-Stokes equation(Whi10),

$$\rho(d\mathbf{V}/dt) = \rho\mathbf{g} - \nabla p + \mu\nabla^2\mathbf{V}, \tag{2.1}$$

or the more simpler inviscid Euler equation

$$\rho(d\mathbf{V}/dt) = \rho\mathbf{g} - \nabla p, \tag{2.2}$$

where $\rho$ is the density, $\mathbf{V}$ is the velocity, $\mathbf{g}$ is the acceleration, $p$ is the pressure and $\mu$ is the viscosity.

Both equations applies for the motion of all fluids in general, and modified Navier-Stokes equations have previously been used to regularise incompressible fluid motion fields, e.g the motion of blood(TSGL99).

## 2.2 Smoothed particle hydrodynamics (SPH)

SPH is a model for simulating fluid dynamics. It was first developed independently by Gingold and Monaghan(GM77) and Lucy(Luc77) for the purpose of studying astrophysical models.

It has later been successfully adapted to handle a variety of fluid flows through modifications such as Weakly Compressible Smoothed Particle Hydrodynamics (WCSPH), Incompressible Smoothed Particle Hydrodynamics (ISPH), and (XSPH) and addressing free surface problems arising in the original formulation by various means (BT07)(Mon94).

### 2.2.1 Relation to other models

SPH is the most established and mature method in a range of Lagrangian hydrocodes.

Being a Lagrangian method, convective acceleration is eliminated, this is often a nontrivial source of error in the Eulerian methods. A few hybrid Lagrangian-Eulerian methods exist to eliminate this problem. We mention in particular the Particle-in-Cell method that was first published in 1963 by Harlow(HW65). The principle behind the method involves a periodic mapping by interpolation between particles and grids, where the advective transport is performed on the particles and the physics of the system, including pressure projection, viscosity and boundaries is solved on the grid. The excessive interpolation is a major source of numerical dissipation in this model, and besides its value in plasma physics, the method is of primarily historical interest.

A modification to the Particle in Cell (PIC) scheme called Fluid Implicit Particle (FLIP) was proposed by (BKR87). In this model, particle properties are no longer interpolated from the grid, but rather their rate of change. This way, the compounded smoothing over time is removed from the system and the numerical dissipation becomes virtually zero.

In contrast to these, SPH was developed as a purely Lagrangian mesh-less method; there is no particle-to-grid transfer taking place, and the differential equations of the system is solved individually for each particle. This carries the same benefits from the FLIP model, in that numerical dissipation is low. Higher dimensional problems increases the degree of freedom for particles. However, this problem is apparent in shared by SPH and FLIP.

A notable alternative to these is the Lattice Boltzmann model(HL97).

### 2.2.2 Basic principles

The idea behind SPH is to use a series of interpolating points along with a smooth kernel to interpolate continuum properties and move the points according to the fluid properties. In one sense, the particles represent fluid parcels, i.e units of constant mass moving with the fluid and deforming appropriately. To determine the advective transport of the fluid elements, differential equations for the dynamics of the system are solved at the position of the fluid elements to evolve the system according to the governing equations, e.g. Navier-Stokes, diffusion, heat transfer or magneto-/electrodynamics.

To obtain the equations of SPH, we use the kernel approximation

$$\langle A(\mathbf{r})\rangle = \int_{\Omega} A(\mathbf{r}')W(|\mathbf{r}' - \mathbf{r}|)\mathrm{d}\mathbf{r}', \tag{2.3}$$

where $W$ is the interpolating kernel and $\mathbf{A}(\mathbf{r})$ is some field function. If the kernel is symmetric and normalised, this approximation achieves second order accuracy. Further,

$$\lim_{h\to 0} W(x,h) = \delta(x), \tag{2.4}$$

where $\delta$ is the Dirac delta function.

To evaluate the field function $A$ at the location of the particles we rely upon the principles of Monte-Carlo theory, i.e a kernel estimation of probability densities through discrete sampling of the kernel;

$$A_a = \sum_b A_b W(r_{ab}), \tag{2.5}$$

where $A_a = A(\mathbf{r}_a)$ and $W_{ab}$ is shorthand for $W(|\mathbf{r}_a - \mathbf{r}_b|)$. To reinforce the fluid analogy and make the concept of a fluid element meaningful, a scalar density field $\rho$ is introduced to permeate the sampling domain

$$\langle A(\mathbf{r})\rangle = \int_{\Omega} \frac{A(\mathbf{r}')}{\rho(\mathbf{r}')}\rho(\mathbf{r}')W(|\mathbf{r}' - \mathbf{r}|)\mathrm{d}\mathbf{r}'. \tag{2.6}$$

Using $\mathrm{d}m = \rho \mathrm{d}r$, the kernel sampling becomes

$$A_a = \sum_b m_b \frac{A_b}{\rho_b} W_{ab}, \tag{2.7}$$

Spatial derivatives of approximated continuum properties have exact analytical closed form in the SPH framework. The inverse product rule

$$\frac{\partial A}{\partial x} = \frac{1}{\Phi}\left(\frac{\partial(\Phi A)}{\partial x} - A\frac{\partial \Phi}{\partial x}\right) \tag{2.8}$$

with a test function $\Phi$ transforms in the kernel estimation to become

$$\left(\frac{\partial A}{\partial x}\right)_a = \frac{1}{\Phi_a}\sum_b m_b \frac{\Phi_b}{\rho_b}\left(A_a - A_b\right)\frac{\partial W_{ab}}{\partial x}. \tag{2.9}$$

It was shown that the test function $\Phi = \rho$ increases the accuracy of the simulation (Monaghan's Second Golden Rule(Mon92)).

Using this, the equations of motion for the particles can be derived by coupling to the Navier-Stokes equations and solving the differential equations using kernel estimation of continuum properties and derivatives.

There are two different interpretations of the kernel estimation,

**Scatter** Particles are imagined to scatter their properties onto surrounding particles in a radius determined by their kernel support radius $h_a$.

**Gather** Particles gather properties from neighbouring particles within the kernel support radius of those particles $h_b$.

The variant that is depends on which smoothing length is used in the kernel. However, in order to obtain symmetric force pairs, a combination of the interpretations have to be used; $\bar{h} = (h_a + h_b)/2$. This is required to satisfy force symmetry in the constitutive equations.

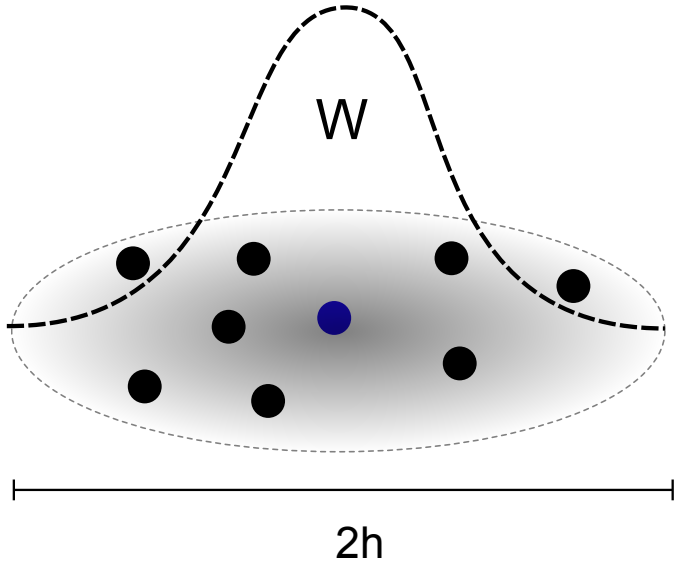


**Figure 2.1:** Illustration of the kernel approximation. The particle under consideration marked in blue gathers and scatters information onto the particles within the kernel support of a radius $2h$.

During the course of a SPH simulation the particles exhibit varying degrees of disorderliness, changing the Probability density estimation analogous, and Gingold and Monaghan hypothesised that the kernel estimation error should be obtained

from the Monte-Carlo estimate $A_i^{\text{err}} \propto 1/\sqrt{N}$. Although subsequent papers discovered a discrepancy in the the estimated error(GM78)(GM79) where the error was significantly lower than expected. This was explained as the probability density interpretation in Monte-Carlo theory allow variations not consistent with the system dynamics, whereas in SPH, the reality is that although particles are allowed to become disordered, they do so in an *orderly* fashion(Mon05).

### 2.2.3 Kernels

An important aspect of a SPH implementation is the choice of smoothing kernel. The choice should be a Gaussian-class kernel in order to carry any potential physical interpretation (Monaghan's First Golden Rule(Mon92)). One of the choices among these is the normal-distribution kernel

$$W(r,h) = \left(h^2\tau\right)^{-\frac{\nu}{2}} \exp\left\{-\frac{1}{2}\left(\frac{r}{h}\right)^2\right\}, \tag{2.10}$$

where $\nu$ is the dimensionality and $h$ is the Scale of Interest (SOI). This kernel lacks compact support, however, a truncation at $\delta$ standard deviations can be achieved as

$$W^{\text{trunc}}(r,h,\delta) = \begin{cases} \frac{W(r,h)}{\Phi(+\delta)-\Phi(-\delta)} & r \leq \delta \\ 0 & r > \delta, \end{cases} \tag{2.11}$$

where $\Phi$ is the cumulative distribution function of the standard normal distribution.

The Gaussian kernel exhibits an off-center inflection point. This can lead to undesirable artifacts such as reduction in interparticle pressures. Under high pressure conditions where particles are forced together, particle pairs may be compressed closer than the inflection point of the kernel, gradually reducing the repellent pressure force between the particles as they are pushed close together in an effect called *tensile instability*. The clustering of particles due to this effect reduces the effective resolution of the fluid int that clusters of particles behave as one. Monaghan(Mon00) suggested a tensile correction factor to the pressure,

$$1 + R_k \Psi^n, \tag{2.12}$$

where $\Psi$ is the ratio $W(r)/W(r_0)$ with inflection at $r_0$ and $n$ is some numerical coefficient, e.g 4. It has been suggested, however, that such effects should be resolved by alternative kernels with differentiability properties such as the spiky kernel

$$W^{\text{spiky}}(r,h) = \begin{cases} \frac{15}{\pi h^6}(h-r)^3 & r \leq h \\ 0 & r > h, \end{cases} \tag{2.13}$$

with gradient

$$\nabla W^{\text{spiky}}(r,h) = \begin{cases} -\frac{45}{\pi h^6}(h-r)^2\hat{r} & r \leq h \\ 0 & r > h, \end{cases} \tag{2.14}$$

with increasingly repulsive pressure as particles approach each other. Similarly, the viscosity kernel

$$W^{\text{visc}}(r,h) = \begin{cases} -\frac{r^3}{2h^3} + \frac{r^2}{h^2} + \frac{h}{2r} - 1 & r \leq h \\ 0 & r > h \end{cases} \tag{2.15}$$

with the positive definite Laplacian

$$\Delta W^{\text{visc}}(r,h) = \begin{cases} \frac{45}{\pi h^6}h - r & r \leq h \\ 0 & r > h, \end{cases} \tag{2.16}$$

ensuring non-negative contributions to avoid exciting the system through viscosity.

However, it can be argued that these artifacts should be corrected by addition of new physics to the system, the responsibility lies not on the kernel. Conflicting research, general consensus generally favours mixing kernels appropriate to the type of computation to avoid adding extra physics to compensate for what is essentially an artifact of the smoothing kernel.

Other popular kernels include the B-spline kernels with analytical compact support and high order differentiability and smoothness. Also, quintic Wendland kernels and the $M_4$ kernel have been used frequently in SPH codes, demonstrating that a wide range of kernels can be chosen depending on the desired properties of the smoothing.

### 2.2.4 Lagrangian mechanics in SPH

The equations of motion for SPH can be derived from the Navier stokes equation (2.1) using Lagrangian variational frameworks.

In fact, a limitless number of SPH schemes can be devised, and the constitutive equations change depending on the particular variational framework through which it has been derived. The particle state occupy a $6N$ phase space, where $N$ is the total number of particles, and the generalised coordinates $q_i$ and generalised momenta $p_i$ The literature concerning $N$-body problems is extensive, and the particle kinematics in the context of the constitutive equations is well understood.

Some central concepts in the theory of such systems include action functional, variational calculus Stat-mech?

### Conservation of mass

In SPH there are two distinct formulation for the estimation of the fluid density $\rho$. The first approach, often called the *summation density* method, evaluates $\rho$ through a direct usage of (2.7), revealing

$$\rho_a = \sum_b m_b \frac{\rho_b}{\rho_b} W_{ab} = \sum_b m_b W_{ab}. \tag{2.17}$$

This approach is robust and guarantees normalisation of the kernel estimation of other parameters. In this classical SPH sense, mass is always guaranteed to be exactly conserved, which is an attractive property of the method.

The other method derives the variation of $\rho$ through the mass continuity equation,

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot (\rho \mathbf{u}). \tag{2.18}$$

Using the result from (2.9) with (2.18), we can write the rate of change in $\rho$ as

$$\frac{\partial \rho}{\partial t} = \sum m_b v_{ab} \nabla W_{ab}. \tag{2.19}$$

With this method the density field is created with some initial value, usually taken to be the prototypical density as computed with (2.17) for a particle with a filled neighbourhood.

(2.19) carries some advantages over (2.17), most notably its improved handling of free surfaces or other fluid boundaries. (2.17) will experience a lack of interacting neighbours near a boundary, creating a significant incompleteness in the kernel estimation. This causes the computed density to drop at the free surface and thereby introducing spurious behaviour in these regions.

Another benefit is the reduced computational cost in using (2.19), since all differentials can be computed in one pass, whereas (2.17) requires the estimation of $\rho$ before any other property can be computed at each timestep, effectively making it a two-stage process.

(BK02) To improve the stability of the density calculation, an auxiliary particle filter known as the Shepard filter may be applied to the density calculation(PS99)(BL99). The Shepard filter is a zeroth order correction to the density estimate,

$$\rho_i^\star = \frac{\sum_j m_b W_{ab}}{\sum_j \frac{m_b}{\rho_b} W_{ab}}. \tag{2.20}$$

Other approaches such as the first order Moving Least Squares (MLS) filter which is capable of restoring a linear density error. (OYG14) investigates the use of a density smoother based on the kernel estimator.

#### Kinematic pressure

At each timestep, the relevant physics are computed and the system is evolved according to a set of Ordinary Differential Equations (ODEs).

To calculate the kinematic pressure gradient experienced by the particles we refer to the inviscid Euler equation (2.2). Following Monaghan's second Golden Rule (Mon92) to include the density in computations of field variables we arrive at

$$\rho_a \nabla P_a = \sum_b m_b (P_b - P_a) \nabla_a W_{ab}. \tag{2.21}$$

However, this form of the pressure contribution yields asymmetric force pairs in the constitutive equations, violating the conservation laws. An alternative derivation of the pressure gradient exploits the identity

$$\frac{\nabla P}{\rho} = \nabla \left( \frac{P}{\rho} \right) + \frac{P}{\rho^2} \nabla \rho. \tag{2.22}$$

Turning the Gingold-Monaghan crank, we arrive at

$$\frac{d\mathbf{v}}{dt} = -\sum_b m_b \left( \frac{P_b}{\rho_b^2} + \frac{P_a}{\rho_a^2} \right) \nabla_a W_{ab} \tag{2.23}$$

This formulation is symmetric for particle pairs and conserve linear and angular momentum exactly.

Since the pressure gradients are computed analytically from a discrete kernel sampling, particle disorderliness may create unstable gradients if the neighbourhood is sparse. Some gradient stabilising methods have been proposed similar to the Shepard kernel correction, although these no longer conserve linear and angular momentum exactly.

Solving the Euler simulation requires a coupling to an Equation of State (EOS).

### 2.2.5 Equation of state

Depending on the properties of the fluid of interest, an appropriate EOS has to be selected. There is a multitude of choices for EOS, including the classic compressible SPH for an isothermal gas using

$$p = c^2 \rho, \tag{2.24}$$

where $p$ is the pressure and $c$ is the speed of sound, or polytropes using the Murnaghan EOS suggested by Batchelor(Bat00)

$$p = \frac{\rho_0 c^2}{\gamma} \left( \left( \frac{\rho^\star}{\rho_0} \right)^\gamma - 1 \right) + p_0, \tag{2.25}$$

known as the *Tait equation*, where $\rho^\star = \rho$, or optionally the Hughes-Graham corrected

$$\rho^\star = \begin{cases} \rho & \text{if } \rho > \rho_0 \\ \rho_0 & \text{if } \rho \leq \rho_0, \end{cases}$$

where $\rho_0$ is the reference density, usually taken as the prototypical particle density. The Tait equation is often chosen in cases where incompressible flow is desired.

The correction considers pressure fluctuations asymmetrically to discard negative pressure from particle vacuums while penalising particle compression, making it suitable for free surface flow and spray modeling.

### 2.2.6 Artificial viscosity

In section 2.2.4 we considered the inviscid Euler equation to compute the advective transport. To include the viscosity from the full Navier-Stokes treatment we consider the following term

$$-\mu\nabla^2\mathbf{u}, \tag{2.26}$$

where $\mu$ is the kinematic viscosity quantifying the shear stresses between lamina. Applying (2.9) twice to velocity,

$$\nabla^2 = \mu \sum_b m_b \frac{\mathbf{v}_b - \mathbf{v}_a}{\rho_b} \nabla^2 W_{ij}. \tag{2.27}$$

Gingold and Monaghan proposed the artificial viscosity tensor

$$\Pi_{ij} = -\nu \left( \frac{\mathbf{v}_{ab} \cdot \mathbf{r}_{ab}}{r_{ab}^2 + \epsilon \bar{h}_{ab}^2} \right), \tag{2.28}$$

where

$$\nu = \frac{\alpha \bar{h}_{ab} \bar{c}_{ab}}{\bar{\rho}_{ab}} \tag{2.29}$$

and $\epsilon \sim 0.01$ is some small parameter to avoid singularities.

This viscosity tensor has many attractive properties, such as Galilean invariance, and cancellation in rigid rotation. Many extensions of this viscosity have been devised to deal with some of its limitations(Bal95), but it remains as the most widely used formulation for artificial pressure due to its simplicity.

### 2.2.7 Time evolution

Multiple integration schemes are applicable in , ranging from The most commonly applied scheme is leap-frog, or the equivalent velocity Verlet integration which allows for variable time steps. This is a second order explicit (time-marching) integration updated according to the scheme

$$\mathbf{v}^{n+\frac{1}{2}} = \mathbf{v}^{n} + \frac{\mathbf{F}^{n}}{m}\frac{\Delta t}{2} \tag{2.30}$$

$$\mathbf{x}^{n+1} = \mathbf{x}^{n} + \mathbf{v}^{n+\frac{1}{2}}\Delta t \tag{2.31}$$

$$\mathbf{v}^{n+1} = \mathbf{v}^{n+\frac{1}{2}} + \frac{\mathbf{F}^{n+1}}{m}\frac{\Delta t}{2}. \tag{2.32}$$

This geometric(HLW+03) integrator reflects attractive symmetric properties of the Hamiltonian, e.g. reversibility in absence of frictional forces and symplecticity, leading to improved conservational features(LP96)(MQR99), maintaining stability at larger time steps. It can be argued, however, that with dispersive effects and pseudo-Rayleigh dissipation from XSPH and the Gingold-Monaghan tensor respectively, the Hamiltonian of the system is not strictly conserved after all, so the arguments promoting use of a symplectic integrator are void. Second order Runge-Kutta or modified Euler methods may be better suited in light of this. Another popular choice is predictor-corrector integrators.

In the weakly compressible formalism, the fluid acceleration is the limiting factor in determining the timestep due to Courant–Friedrichs–Lewy (CFL) criterion. Increasing the number of particles also reduces the allowable timestep by decreasing the characteristic scale of the system, i.e. the kernel extent. The CFL criterion is a necessary(but not sufficient) condition for stability in a system of that ensures information propagation through the fluid never jumps the discretisation in one time step.

Goswami and Pajarola discuss heuristics for achieving global stability at larger timesteps(GP11) yielding significant performance increase compared to using the simple time step case.

### 2.2.8 XSPH

Altough the Gingold-Monaghan tensor (2.28) provides realistic viscosities, it introduces significant numerical dissipation. Monaghan suggested to include another term in the position update

$$\frac{\mathrm{d}\mathbf{r}_i}{\mathrm{d}t} = \hat{\mathbf{v}}_i = \mathbf{v}_i + \epsilon \sum_j m_j \mathrm{v}_{ji} W_{ij}, \tag{2.33}$$

i.e. to move the particle with its self-momentum corrected with the parameter $\epsilon$ to the neighborhood-averaged velocity. This reduces the amount of artificial smoothing needed to keep the particles orderly, generating less numercial dissipation. This is more in the nature of the fluid element interpretation and reduces particle penetration and chaotic gas behaviour without the need for excessive artificial viscosity. To ease the interopability with the leap-frog scheme, Schechter and Bridson(SB12) suggested to execute the XSPH-correction at the velocity level,

$$\mathbf{v}_i^\star = \hat{\mathbf{v}}_i, \tag{2.34}$$

so that the immediate particle update can be done using the XSPH-corrected velocity directly $\mathbf{r}_i^{t+1} = \mathbf{v}_i^{t+\frac{1}{2}}\delta t$. The dissipative effects of this approach remains to be fully documented.

### 2.2.9 Boundary conditions

The problem of robust and mathematically rigorous boundary handling remains an open problem in SPH. One of the earliest variants suggested by Monaghan(Mon92) involved placing immovable particles at the boundary exerting a repulsive force on the according to some interparticle potential, e.g a Lennard-Jones potential. This method is simple to implement, and is a popular choice for quick SPH codes.

Another method that has been used with success is the ghost particle method(SB12). In this treatment, a virtual particle is mirrored over the boundary when a particle approaches the edge. The virtual particle is identical to the real particle in all aspects with the notable exception that the velocity is inverted. This creates a repulsive force when particles approach the boundary and can create a no slip effect if desired. This method is mathematically sound and guarantees a normalised vector function for plane boundaries. However, since the mirroring depends on incident angles, the complexity quickly escalates for non-trivial boundaries. In cases where particles are located near a corner, the problem of overcounting also arises, and corrective measures have to be applied.

In a scenario with rapidly changing boundary boundaries this introduces significant computational overhead in the dynamic repositioning of virtual ghost particles and correcting for overcounting effects.

The boundary particle deficiency is a well-studied problem in SPH, see e.g.(BTT09) for an elegant solution to this problem

### 2.2.10 Nearest neighbour particle search

A crucial component, often playing a significant role in the overall theoretical performance a SPH implementation can achieve is the computation of nearest neigh-

bour particle pairs.

In a naïve *all pair* implementation, all interactions from all other particles within the influence must be considered and for kernels without compact support this means that interactions between all particle pairs must be computed. This causes the problem to scale as $\mathcal{O}(N^2)$, where N is the total number of particles in the system. For kernels with compact support, such as spline kernels and quintic Wendland kernels, the neighbouring particles have to be found in order to reduce the total number of computations to include only the strictly nonzero interaction pairs.

The problem is closely related to the concept of broad-phase collision detection, i.e. to perform candidate reduction to only include potentially interacting candidates. Many different strategies for Nearest Neighbour Particle Search (NNPS) have been devised. We mention the original linked list approach suggested by Monaghan(Mon92) and the related Verlet list method for which a sensitivity study conducted by Viccone *et al.*(VBC08).

Other methods include boundary volume hierarchical trees, octrees (quadtree in 2D) and kd-trees. Some methods for tree construction on the GPU have shown promise, notably Zhou(ZHWG08). However, tree based structures are not a natural fit for GPUs due to the lack of recursive features for tree traversals(HKK07). Hegeman *et al.*(HCM06) proposed a fixed topology tree that allows the tree processing to be performed on the GPU, but the most common method is to maintain the tree on Central Processing Unit (CPU) and perform the interaction calculation on the GPU.

Methods that have seen a lot of attention in the transition to GPU are spatial hashing methods.

### 2.2.11 Weakly compressible SPH

One of the major drawbacks of SPH for modeling incompressible fluids is non-negligible compressibility artifacts arising from the governing equation of state.

WCSPH was suggested as a method to model incompressible fluids by Monaghan (Mon94) by using the Tait EOS along with a continuity equation approach for computing the density.

The fluid is considered to be incompressible if the flow velocity is low compared to the speed of sound in the fluid. It can be shown that density fluctuation is given by $\Delta\rho = vL/(c^2\tau) \propto M^2$, where $M$ is the mach number. That is, in order to achieve density fluctuations lower than $1\%$, the speed of sound in the fluid must be at least ten times greater than the expected maximum flow velocity.

This can be achieved by increasing the bulk modulus of the simulated fluid, however, the resulting differential equations in turn become very stiff. Although the explicit WCSPH scheme facilitates extremely quick particle updates and computations, the small timestep required to propagate the particles accurately in this formulation from the CFL-condition limits the theoretical performance in the near-incompressible regimes.

There has been a lot of research on methods allowing larger time steps, so called ISPH schemes. These do, however, introduce problems of their own. For example, the nonelliptic governing equations of WCSPH do not translate to a truly incompressible scheme. The solution of the implicit equations requires costly solver steps at each particle update, for example by Jacobi relaxation(ICS+14). Projection approach to enforce incompressibility by performing a Helmholtz-Hodge decomposition to project the solution onto an divergence free field as suggested by Colin(CEL06). Incompressibility can also be incorporated by the Gibbs-Appell equations.

A very successful method to enforce incompressibility was published by Solenthaler and Pajarola(SP09). They presented a Predictive Corrective Smoothed Particle Hydrodynamics (PCISPH) scheme to predict a density fluctuation and do a backwards propagation of pressure to counteract the target density fluctuation. This way the simple and explicit equations from WCSPH can be used at larger timesteps.

An advantage of WCSPH is the ability to adapt to multiple flow situations through the modification of the fluid parameters

Multiple comparisons between the ISPH and WCSPH models have been published, see for example (LMX+08) or more recently (DGB+16). Hughes and Graham compared WCSPH and ISPH schemes and found that when tuned correctly, WCSPH performs on par with ISPH, and in some situations even outperforms ISPH (HG10).

### 2.2.12 Advantages and disadvantages

SPH is a model that exhibits superb conservational properties, such as conservation of mass, and linear and angular momentum.

The distinguishing feature of SPH compared to mesh-based methods is the computation of pressure forces and coupling with an equation of state and an integrator instead of reliance on solving systems of linear equations.

This makes the model conceptually simple and additional physics can be added by simply encoding them as interaction terms to be computed at every update. Because of this, an elementary SPH simulator may readily be implemented at first,

and further extended with more complex physics added at a later stage to obtain research quality flows.

The simulation grid is encoded in the particles, so unlike mesh-methods there is no requirement to explicitly adapt to changing boundaries or perform free surface tracking, the simulation will self-resolve these situations. For this reason in particular it is often the case that Finite Element Analysis (FEA) is faster and more accurate for fixed, closed domains, whereas SPH may provide significant performance benefits in problems that require free surface tracking and/or frequent regridding due to continuously changing boundary conditions. Due to the Lagrangian nature of the method, the grid follows the particle, making the method self-adaptive in resolution.

Another advantage of the SPH model is the high degree of inherent parallelism, making it suitable for easy implementation on massively parallel architectures, including GPUs. At every update, the internal and external forces can be computed in parallel for each particle before they are collectively updated. By construction, SPH avoids the problem of unphysical negative pressures arising in other numerical CFD models.

Although SPH provides accurate results in 1D and 2D for relatively small numbers of particles, the high degree of freedom in 3D requires a significant particle number in order to maintain stable gradients and derived pressure forces. SPH also tends to introduce dissipation in strong shocks and damp post shock oscillations smaller than the smoothing length $h$.

It is also sensitive to the initial distribution of particles, and special care has to be taken to make sure initial conditions do not affect the simulation results. In cosmological computations, a small overdense cloud may trigger the accretion of a structure spanning lightyears! Lucy recognised this problem and addressed it with a quiet start solution in his original paper(Luc77).

From lack of effective screening of the gravitational force, self gravitating systems can be extremely demanding to compute in SPH ($\mathcal{O}(N^2)$). Regarding convergence, if a numerical singularity occurs, SPH will continue running and give incorrect results. On the contrary, most of the other will terminate early with some tolerance error. Finally, accuracy is generally considered to be poorer for SPH than rivalling methods. It is well known that SPH is not the best tool for high velocity shock simulations or other situations with high Mach numbers, however SPH is a fast and reliable simulator for rather placid flow.

Even if the model may appear deceptively simple, many deep questions surrounding convergence, consistency and stability in SPH are still not fully understood,

which may be explanatory of the overall lack of adoption in comparison to mathematically sound FEA methods. For example, it was long hypothesised that the free surface conditions were implicitly satisfied in the model, but a basis for this was just recently established through a rigorous analysis in the treatise by Colagrossi *et al.*(CALT09) for which they received the Monaghan award.

SPHERIC Steering Committee identifies five *Grand challenges* in the SPH method. As of 2017, these are

1. Convergence, consistency and stability
2. Boundary conditions
3. Adaptivity
4. Coupling to other models
5. Applicability to industry

In the end, although SPH and FEA are very dissimilar methods, comparable results can be achieved by both methods as proved in a comprehensive study by Durisen and Gingold(DG86).

## 2.3 B-spline regularisation

Part of the work has been directed towards further work Grønli *et al.* (GSN+16), although the 2D CPU implementation has been covered in detail, it is useful to give a brief overview over the basics of penalised B-spline regularisation here in order to motivate the details of the 3D GPU extension of this work. A large part of the effort was directed at implementing a fast iterative Least Squares (LS) matrix solver on the GPU to solve the exponentially demanding 3D system.

The following section is adapted from Grønli(Gr6) and revised to reflect the advancements achieved in this project. The benefits of least square smoothers with custom Tikhonov regulariation matrices are well studied(Eil03)(Gar10), in particular divergence penalties have shown great promise in blood flow regularisation (GPS+13)(GdVJ+15)

### 2.3.1 Formulation of splines

A spline $S$ of degree $n$ is a piecewise polynomial real function $S : \mathcal{D} \mapsto \mathbb{R}$

$$S(t) = \begin{cases} p_1(t) & t \in [t_0, t_1) \\ p_2(t) & t \in [t_1, t_2) \\ \dots & \\ p_k(t) & t \in [t_{k-1}, t_k) \end{cases} \tag{2.35}$$

where the set $\{t\}_i$ composes a partition of the domain $\mathcal{D}$ of definition, and $\forall i : p_i \sim \mathcal{O}(t^n)$ in the Bachmann-Landau notation, that is, none of the polynomials has a degree larger than $n$. Furthermore, the spline also is of differentiability class $C^{n-1}$ and therefore enjoys $n-1$ differential continuity at each of the internal joining nodes, also called the *knots* of the spline. Combined with the infinitely smooth interstitial polynomials, this ensures exceptional smoothness of the curve at all points. To achieve this, special care has to be taken when selecting the polynomials $\{p\}_i$, and algorithms have been devised for proper coefficient selection.

### 2.3.2 Basis splines

All splines of a chosen order and domain partition compose a function space $\mathcal{F}$, and from linear theory there exists linear subspaces $B \subseteq \mathcal{F}$ such that $\mathcal{F} = \text{span}\{B\}$, i.e. $B$ forms an orthogonal basis for the function space. One example of such a basis is the subspace composed of splines that have minimal support with respect to the chosen order and spline domain partition. This particular subspace is aptly named basis splines or B-splines, and all splines can be written as a linear combination of this basis in accordance with the fundamental theorem stated by de Boor(DB78) in his seminal treatise,

$$S^k = \sum_i c_i B_i^k. \tag{2.36}$$

Given a domain partition and spline order, the B-spline basis can be recursively calculated by the means of the Cox-de Boor algorithm,

$$\beta_i^k(t) = \begin{cases} \frac{t-t_i}{t_{i+k-1}-t_i}\beta_i^{k-1}(t) + \frac{t_{i+k}-t}{t_{i+k}-t_{i+1}}\beta_{i+1}^{k-1}(t) & k > 1 \\ \Theta(t-t_i) - \Theta(t-t_{i+1}) & k = 1 \end{cases} \tag{2.37}$$

where $\Theta$ is the Heaviside step-function. It is apparent from this recursive definition that the base case sets a limit on the possible extent of the k-th order spline.

### 2.3.3 Nonlinear least squares spline regression with B-spline basis

Because the high degree of smoothness and the finite support of the spline function, this makes a B-spline an attractive basis for nonparametric polyvariate regression problems. The idea behind spline regression is to partition the regression domain with a suitable number of subintervals and find the B-spline coefficients $c_i$ in (2.36) that minimise the squared error of the fitted spline $S$, i.e

$$c = \arg\min_x ||\mathbf{S}x - b||^2, \tag{2.38}$$

where $\mathbf{S}$ is the coefficient matrix and $b$ is the values vector. For simplicity the nodes may be placed on a uniformly spaced grid in the regression domain. Such configurations of spline nodes are known as uniform, or *cardinal*, spline grids. Because of the uniform partitioning of the basis domain the Cox-de Boor algorithm simplifies greatly and compact closed form expressions for the functions can be found easily. In the case of cubic splines, we obtain

$$\beta_i^4(t) = \begin{cases} \frac{2}{3} - \frac{1}{2}|t|^2(2 - |t|) & |t| < 1 \\ \frac{1}{6}(2 - |t|)^3 & 1 < |t| < 2 \\ 0 & \text{otherwise.} \end{cases} \tag{2.39}$$

In figure 2.2, an example regression on a set of scattered data points is shown. It displays the regression basis at each node of the uniformly partitioned domain and how each node must be weighted to fit the data points in the LS sense.

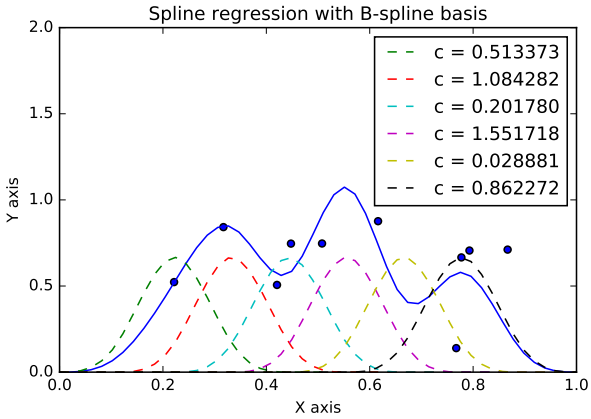


**Figure 2.2:** Illustration of a spline fit of a scattered data point set, represented as blue circles, using a cardinal cubic B-spline basis. Each node has its corresponding function drawn in a separate color and correct LS weighting of the nodes is indicated. The reconstructed spline is drawn as a solid blue line.

## 2.4 The General-Purpose Computing on Graphics Processing Units (GPGPU) programming paradigm

### 2.4.1 Emergence

Recent advancements in microprocessors technologies primarily aimed at high performance game graphics have been increasingly utilised for heavy computations, e.g. simulations, forecasting and machine learning.

The early adoption of this involved redesigning the algorithms from a sequential to a parallel and by writing custom compute fragment shaders which required in-depth insight into an esoteric language built on graphics primitives jargon. Since the launch of NVIDIA's parallel computing platform CUDA in 2006 and later OpenCL from the Khronos group, the entry barrier for computationally intensive research has been lowered considerably, making its usage widespread in the numeric community. Even though the algorithms still need to be translated into parallel forms, there is less overhead in transforming idea to code. This has lead to an explosive boost in limitations on previously compute bound problems.

Compared to a CPU, a GPU contains several thousand cores , threads are extremely lightweight, virtually no context switching overhead

# 3 Methodology and implementation

This chapter concerns the implementation details of the SPH model developed over the course of this project. Further, the extension of the LS-smoother is covered in detail. We also present the methods that were used in the validation process.

All of the methods were implemented using the CUDA parallel computing platform and run using the PyCUDA Python library(KPL+12).

## 3.1 SPH

We implement the WCSPH scheme using the EOS due to Batchelor(Bat00). The method employs the many of the techniques described in(Mon94) and (GGRDC10) along with a novel boundary handling that has not been discussed in the literature. The computational approach the method chosen here bears some resemblance to the GPU approach studied by Goswami(GSSP10) with some deviations.

Normally we advocate the use of summation density (2.17), however, since long term stability is not a concern for the intended application, we opted for the continuity equation formulation (2.19) for evolving the fluid density.

Many of the attractive properties with this formulation align well with the intended application. In particular, the formulation eliminates potential model noise resulting from an agitated initial state that hasn't undergone sufficient relaxation procedures. It also facilitates dynamic reseeding of particles near inlets and outlets and avoids boundary particle deficiency caused by a not fully developed boundary model.

### 3.1.1 Computational model

The computational model is set up in such a way that the system tracks two vectors $q$ and $p$, the generalised coordinates and the generalised momentum densities

respectively.

$q$ is a 4-component vector containing the components $(\mathbf{x}, h)$, whereas $p$ contains $(\mathbf{p}, \rho)$, hence the label *momentum density*.

The generalised coordinates are scaled down to the unit domain $\Omega^3$, where $\Omega = [0, 1]$. Velocities are scaled accordingly, as is the speed of sound in the simulation.

The variables are stored on an equispaced 3D grid in cells able to store up to $2^D$ particles in each. In linear memory, this looks like a blocks of $2^D$ particles stacked end to end.

In this work, we employ strategies found in broad-phase collision detection algorithms to locate a neighborhood $\mathcal{N}_i$ of a particle $i$, and only consider interactions from this localised group. This is done by a cell sorting algorithm where the particles are spatially sorted into the equispaced cells across the simulation domain, and the sorting is done by 3D rasterisation of the particle position onto this grid. The individual cells have space allocated to hold up to `cellSize` number of particles. A subdivision of the warp into full-warp, half-warp or quarter-warp compute units that process 32, 16 or 8 particles respectively, so that the number of particles per cell can be any power of 2.

The computational model in this implementation employs the warp shuffling and voting intrinsics provided by the Kepler architecture and forwards on NVIDIA GPUs to achieve a fast parallel reduction scheme for the particle interactions.

Interactions are computed by using equation (2.7) for the continuum property of interest by launching thread blocks with execution configuration size $n_p^2$ where $n_p$ is the number of particles in each cell. The kernel smoothing length is not allowed to surpass one cell spacing in every direction to ensure all particles have the same amount of coverage kernel coverage. The kernel computes the current `warpId` and the values $q_a$ and $p_a$ respectively hold the position state and the momentum density state for the particle located in cell $k$ at position `warpId`. In turn, the memory locations of neighbours are found through the three dimensional index of cells adjacent to the leader. The states of the particles in a neighbouring cell $k$ are assigned to variables $q_b$ and $p_b$ in the current working set.

The total interaction is computed using butterfly warp reduction over the lane pairings of particles determined by the combinations of `warpId` and `laneId`. This is repeated until all adjoining cells are searched and the particle pair interactions have been computed before the leader lane finally pushes the result to global memory to reduce frequency of global memory writes.

This approach was chosen because of its inherent parallel nature which should be

able utilise the GPU architecture in an effective manner.

The average number of neighbours per particle can be computed by considering a particle centred in a unit cube representing the cell. The smoothing length includes particles located within a sphere of unit radius surrounding this particle. This leads to the mean particle number $N_p = \frac{V_{\text{sphere}}}{V_{\text{cube}}} n_p = \frac{4}{3}\pi n_p$

The fluid parameters struct is stored in a global device symbol, and a field update from the host code triggers a memory transfer to the symbol, allowing the fluid simulation parameters to be changed on-the-fly.

From the scripting code, we perform prepared calls on frequently launched kernels to remove the cost of repeated argument marshalling.

### 3.1.2 Particle rebinning

Periodically, the particles has to undergo rebinning to observe the correct neighbourhood. We readdress each particle by examining particles in cell $\mathcal{B}$ and computing the target cell. A warp mask is generated to signal which particles are scheduled for a readdressing. The warp lanes cooperate to detect which slots in the target are free and a (primitive pseudo-) random offset determines the target slot of each particle transfer. The marked lanes race to reserve a slot and the random offsets help reduce collisions. The atomic intrinsics provided in the CUDA instruction set helps detecting whether a cell reservation was successful or not.

The collective velocity update aided by XSPH reduces the target cell disparity. With this method, the process is then able to transfer multiple particles in parallel to a single target cell and the warp divergence factor is reduced.

A benefit with this sorting is that total overhead incurred by the rebinning is virtually zero when particles are nearly sorted.

### 3.1.3 Boundary forces handling

We suggest a classical potential field approach where boundaries are nonholonomic rheonomous constraints represented as regions of elevated potential energy w.r.t. the reference level, such that the local conservative force

$$\mathbf{F}^U = -\nabla U = -\left(\frac{\partial U}{\partial x}, \frac{\partial U}{\partial y}, \frac{\partial U}{\partial z}\right), \tag{3.1}$$

where $U$ is the potential field. In this formulation, the domain can be thought of as a finite potential well, where the energy level determines the stiffness of the wall. This level set approach greatly simplifies the calculation of domain boundaries in that boundary forces can be directly computed locally for each particle in

an extremely effective manner by using highly specialised linearly interpolating texture references available on GPUs. The gradient can be numerically estimated by the use of Finite Differences (FD) enabled by the linear texel interpolation. To avoid plateauing of particles penetrating the boundary gradient, a 3D morphological distance transform operator may be applied to strengthen the localisation of the particles.

One of the drawbacks of this method is the need for a fine-grained texture in order to sufficiently resolve curved and sloped boundaries. This anisotropic effect can induce angle dependent artifacts in fluid simulations sensitive to boundary noise, such as Poiseuille flow. This criterion increases the amount of data flowing across the PCI-e bus in simulations with rapidly changing boundary conditions leading to increased bus congestion, particularly if frequent state fetching is neccessary in the application.

This can be somewhat resolved by reducing the potential strength, allowing a small penetration of the boundary and effectively smoothing out the jagged potential observed by the particle. In this softer version of the boundary, particles undergo a harmonic spring like motion near the boundary under the influence of a conservative force derived from the potential. This means that, with sufficiently small timesteps, no energy is lost in the reflection. On the contrary, the finite difference step must be chosen as to not let an approaching particle skip the discretisation distance and gaining energy by jumping the gradient without the potential doing any work on the particle.

This method is, however, susceptible to boundary particle deficiency depending on the choice of density formulation. In the summation density (2.17) formulation this will lead to decreased particle densities near domain boundaries unless this is corrected for in an extended formulation of this boundary handling.

To implement a no slip condition to simulate the fluid boundary layer, a retardation of the velocity component in the normal plane to the potential gradient can be applied. The reduction coefficient may also be a function of the local potential allowing for realistic contact-dependent frictional forces.

$$\mathbf{p}^{\star} = \mathbf{p} - \xi f(U) \left( \mathbf{p} - \frac{\mathbf{p} \cdot \nabla U}{||\nabla U||^2} \nabla U \right), \tag{3.2}$$

where f is some function increasing along with the local potential, e.g. $f(U) = 1 - e^{\frac{U - U_0}{T}}$.

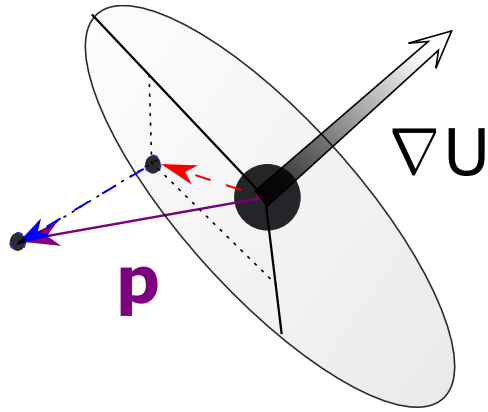


**Figure 3.1:** Illustration of the component reduction described by (3.2). The velocity component of $\mathbf{p}$ in the plane with normal vector along the potential gradient $\nabla U$ is reduced by some coefficient depending on the local potential $U$.

The boundary forces to keep the particles enclosed in the domain is computed using the potential gradient method from (3.1). The scalar potential is initialised using a level set obtained through segmentation methods or by direct creation.

For the host to device transfer, a pitched buffer is allocated on the GPU and the resulting buffer is bound to the volume texture reference. The texture is set to use linear filtering on normalised coordinates so that the scalar potential can be evaluated at arbitrary particle positions by using the linear interpolation capabilities of the graphics texture illustrated in figure 3.2. The normalised coordinate access of the texture makes the gradient lookup compatible with the unit scale simulation domain, as per section 3.1.1.

Using the spatial caching of the texture, gradients can quickly be evaluated numerically by a FD scheme and using the linear interpolation between texels for groups of particles at any point within the domain. The spatial caching capabilities of the texture memory reduces global memory fetches by utilising the compactness of the points being processed in parallel.

In this, the domain is represented by a potential cavity. In situations where the domain changes over time, two textures are uploaded and the transient potential is calculated from linear interpolation between the two textures. In the case of constant boundary both textures are bound to the same underlying array

The boundary force is applied directly in the ODE update step similar to body forces.

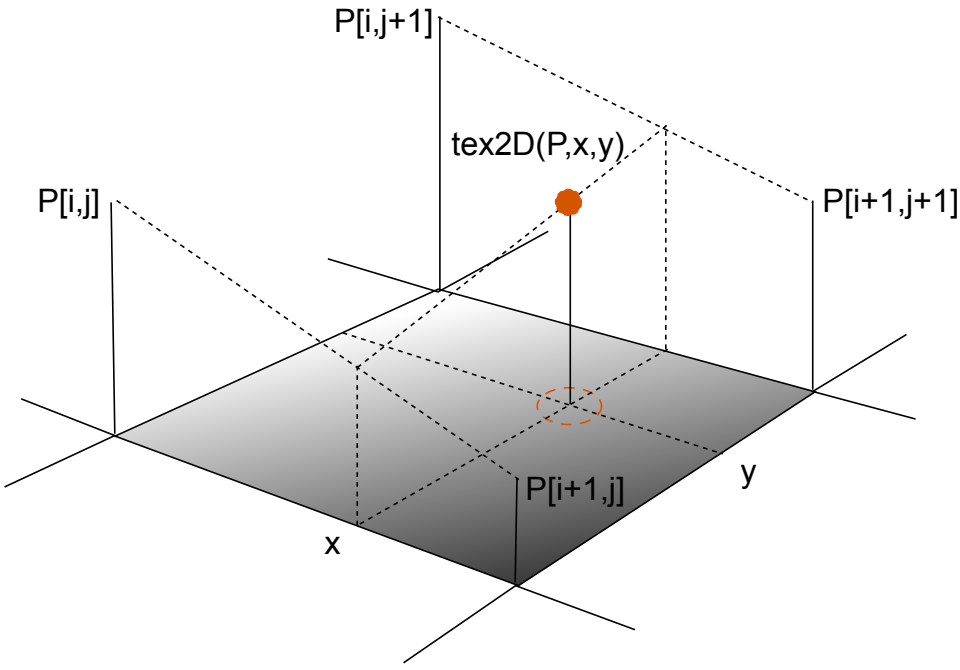


**Figure 3.2:** Illustration of the texture interpolation on GPUs. The 2D texture in this image is provided an array defining the grid values. Using normalised coordinates, the texture lookup can interpolate the value of the texture at any point inside the unit domain.

### 3.1.4 Seeding

The initial distribution of particles is computed by iterating over each cell and on a thread basis compute a random position in the unit cube. The potential level set is evaluated at the generated position through interpolation to find the local potential $U$. Further, the Boltzmann-factor $f(U;\ U_0) = e^{\frac{U-U_0}{T}}$ is computed and compared to a sample drawn from the standard uniform distribution. The particle is accepted if the condition holds, rejected otherwise. This helps distribute particles according to the rules of statistical mechanics.

The random samples are drawn from the cuRAND library on the device path so that a parallel initialisation of particles can be performed in a GPU kernel, and additionally, no CPU to GPU transfer is required in order to setup the system; the whole lifecycle of the particles is confined to the GPU memory. These two advantages significantly reduces the setup period of finely resolved simulations involving millions of particles.

Note also that no expensive post-initialisation sorting is required, and particles are more evenly distributed with less variance, suggesting a less random initial state. Studies on the initial distribution of particles show that it is beneficial to have a somewhat ordered disorder to capture random fluctuations in the numerics.

Similar methods using sampling by levelset(SB12) have been quite successful.

### 3.1.5 Leap frog integration

To evolve the system in time, we implement the Leap frog integration scheme. The initial particle positions $q_i^{-\frac{1}{2}}$ and initial momenta $p_i^0$ is initialised through the seeding procedure and by imposition of an external velocity field.

The update proceeds as described in algorithm 1.

**Data:** $q^{-\frac{1}{2}},\ p^0$
**Result:** Particle transport
**while** $t < t_{end};\ ++n$ **do**
  **for** *parallel i* **do**
    $\rho_i^{n+\frac{1}{2}} := \rho_i^{n-\frac{1}{2}} + K(\{q^{n-\frac{1}{2}}, p^n\}_i)\Delta t;$
    $q_i^{n+\frac{1}{2}} := q_i^{n-\frac{1}{2}} + \frac{p_i^n}{m_i}\Delta t;$
    wall reflect;
    rebin;
    $p_i^{n+1} := p_i^n + F(\{q^{n+\frac{1}{2}}, p_i^n, \rho^{n+\frac{1}{2}}\}_i)\Delta t;$
    xsph update;
    **if** *++k > filter period* **then**
      shepard filter;
      $k := 0;$
    **end**
  **end**
**end**

**Algorithm 1:** Fluid update procedure

Strict consistency demands that the density drift in 2.19 is calculated from the XSPH corrected position differential. However, in the modified XSPH suggested by Bridson employed here this is implicitly satisfied; all particle movement is consistent with the intrinsic momentum of the particle.

### 3.1.6 State imposition and fetching

A major component of this model related to the Kalman coupling at a later stage involves methods for imposing a regular velocity grid onto the particle fluid with a given grid uncertainty and mapping the current fluid flow onto a regular grid of some specified dimension. SPH is an excellent interpolating framework, indeed, half of the logic behind SPH is interpolation. Leveraging this

For each cell, the grid points influencing the particles is computed, and the grid points are weighted according to the kernel and uncertainty estimate to compute the velocity estimate for particle $i$. An alternative and arguably simpler solution could be trilinear interpolation very often used for the corresponding operation in PIC/FLIP calculations.

### 3.1.7 Physics

The physics in the model uses the Tait EOS along with a small stabilising pressure to smooth the gradients. The pressure gradients are computed from 2.23,

and the kernel used is the truncated Gaussian (2.11). Further, the boundary model discussed in this chapter along with the continuity form of the density 2.19 The artificial viscosity from the Gingold-Monaghan tensor is used along with XSPH corrections at velocity level. To maintain an orderly density distribution by correcting for sporadic particle motion caused by kernel incompleteness or numerical singularities we periodically apply Shepard density filtering. The filter is applied at a regular interval every 50 steps.

The model uses simple pressure sources to generate motion according to the valve dynamics. These are computed from the potential.

$$V_i = \sum_j \frac{Q_j}{r_{ij}}, \tag{3.3}$$

revealing

$$\nabla V_i = -\sum_j \frac{Q_j}{r_{ij}^3} \vec{r}_{ij} \tag{3.4}$$

## 3.2 B-spline regularisation

The parts of the following section is adapted from Grønli(Gr6) and revised to reflect the theoretical advancements achieved in this project.

### 3.2.1 Formulation of the velocity reconstruction

Let $\mathbf{v}(x)$ be the reconstructed field. By (2.36) we have that

$$\mathbf{v}^\gamma(x) = \sum_{d_0} \sum_{d_1} \dots \sum_{d_{D-1}} c^\gamma_{d_0 d_1 \dots d_{D-1}} \beta^{(0)}(x^0)\beta^{(1)}(x^1)\dots\beta^{(D-1)}(x^{D-1}), \tag{3.5}$$

In order to obtain the coefficient tensor $c^\gamma_{d_0 d_1 \dots d_{D-1}}$ by LS estimator methods, we need to minimise the cost function

$$\mathcal{L}(\mathbf{c}) = \sum_\gamma \sum_i ||\mathbf{v}^\gamma(x_i) - v_i^\gamma||^2, \tag{3.6}$$

where $x_i$ is the position of the value $v_i$. From (3.5) and (3.6) we can setup the sampling matrix $\mathbf{S}$ by noting that if we let

$$\mathbf{S} = \underbrace{\begin{bmatrix} \mathbf{S}_s & 0 & \dots & 0 \\ 0 & \mathbf{S}_s & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{S}_s \end{bmatrix}}_{D}, \tag{3.7}$$

with

$$S_{ij} = \beta^{(0)}(x_i^0 - n_j^0)\beta^{(1)}(x_i^1 - n_j^1) \ldots \beta^{(D-1)}(x_i^{D-1} - n_j^{D-1}), \tag{3.8}$$

i.e the product of the B-spline functions in each direction called with the difference between point $x_i$ and node $n_j$, we can write $\mathcal{L}(\mathbf{c}) = ||\mathbf{Sc} - \mathbf{v}||^2$, with

$$\mathbf{v} = [\mathbf{v}^0\ \mathbf{v}^1 \ldots \mathbf{v}^{D-1}]^T. \tag{3.9}$$

In principle, the individual velocity components can be solved independently as a scalar field solution from the lack of cross-coupling between the velocity components. However, addition of regularising terms coupling the solutions requires, as will be shown, a simultaneous solving of the complete system, i.e. an extended version of $\mathbf{S}$.

### 3.2.2 The divergence regularisation term

In order to impose mass conservation under the incompressible flow assumption, it is useful to include another term in the minimisation problem

$$\mathcal{L}(\mathbf{c}) = (1-\lambda)\mathcal{L}_{\text{fit}}(\mathbf{c}) + \lambda\mathcal{L}_{\text{div}}(\mathbf{c}), \tag{3.10}$$

where $\lambda$ is a tuning parameter to adjust the ratio between fitting the data and obtaining a divergence free solution. The energy term

$$\mathcal{L}_{\text{div}} = \sum_i ||\nabla \cdot \mathbf{v}(x_i)||^2 \tag{3.11}$$

can be calculated analytically by using that from (3.5), we have

$$\partial_\gamma \mathbf{v}^\gamma(x) = \sum_{d_0}\sum_{d_1}\ldots\sum_{d_{D-1}} c^\gamma_{d_0 d_1 \ldots d_{D-1}} \beta^{(0)}(x^0)\beta^{(1)}(x^1)\ldots\dot{\beta}^\gamma(x_i^\gamma)\ldots\beta^{(D-1)}(x^{D-1}), \tag{3.12}$$

yielding

$$\nabla\cdot\mathbf{v}(x_i) = \sum_\gamma \partial_\gamma \mathbf{v}^\gamma(x_i) = \partial_0\mathbf{v}^0(x_i) + \partial_1\mathbf{v}^1(x_i) + \ldots + \partial_{D-1}\mathbf{v}^{D-1}(x_i). \tag{3.13}$$

Note that, in general, it is not required that points $x_i$ in (3.11) correspond to the points $x_i$ in (3.6), one may for example consider evaluating the divergence on a coarser grid. By creating the derivative sampling matrices identical to (3.21),

$$\dot{S}^\gamma_{ij} = \beta^{(0)}(x_i^0 - n_j^0)\beta^{(1)}(x_i^1 - n_j^1)\ldots\dot{\beta}^\gamma(x_i^\gamma - n_j^\gamma)\ldots\beta^{(D-1)}(x_i^{D-1} - n_j^{D-1}). \tag{3.14}$$

we note that the divergence energy term (3.11) including the cross terms in matrix form can be written as $\dot{\mathbf{S}} = [\dot{\mathbf{S}}^0\dot{\mathbf{S}}^1, ..., \dot{\mathbf{S}}^{D-1}]$ is a row block vector of derivative matrices. Minimising (3.10) requires a simultaneous minimisation of the two energy terms using the matrix formulation

$$c = \arg\min_x \left((1-\lambda)||\mathbf{S}x - b||^2 + \lambda||\dot{\mathbf{S}}x||^2\right) \tag{3.15}$$

$$= \arg\min_x \left\| \begin{pmatrix} (1-\lambda)\mathbf{S} \\ \lambda\dot{\mathbf{S}} \end{pmatrix} x - \begin{pmatrix} (1-\lambda)b \\ \mathbf{0} \end{pmatrix} \right\|^2 \tag{3.16}$$

To solve this, the tall matrix

$$\mathbf{S} = \begin{bmatrix} \mathbf{S}_s & 0 & \dots & 0 \\ 0 & \mathbf{S}_s & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \mathbf{S}_s \\ \dot{\mathbf{S}}^0 & \dot{\mathbf{S}}^1 & \dots & \dot{\mathbf{S}}^{D-1} \end{bmatrix} \tag{3.17}$$

is constructed along with the r.h.s vector $\mathbf{v} = [\mathbf{v}^0 \ \mathbf{v}^1 \ ... \ \mathbf{v}^{D-1} \ \mathbf{0}]^T$, with the blocks weighted according to the regularisation parameter. The resulting system is solvable with specialised iterative linear LS solvers such as LSQR/LSMR.

### 3.2.3 Wall regularisation

Similar to the divergence regularisation a second regularising term is added to the total objective function (3.10),

$$\mathcal{L}(\mathbf{c}) = \mathcal{L}_{\text{fit}}(\mathbf{c}) + \lambda\mathcal{L}_{\text{div}}(\mathbf{c}) + \kappa\mathcal{L}_{\text{wall}}(\mathbf{c}), \tag{3.18}$$

where $\mathcal{L}_{\text{wall}}$ is an energy term enforcing no penetration and free slip conditions close to a wall segment. The velocity component normal to the wall can be computed as

$$\mathcal{L}_{\text{wall}} = \sum_i ||\hat{\mathbf{n}}_i \cdot \mathbf{v}(x_i) - v_{i,\perp}||^2, \tag{3.19}$$

where $\hat{\mathbf{n}}_i$ is the unit normal vector of a wall segment positioned at $x_i$ and $v_{i,\perp}$ is the corresponding wall velocity along the normal vector. Following the derivation in(3.13),

$$\hat{\mathbf{n}}_i \cdot \mathbf{v}(x_i) = \sum_\gamma \hat{\mathbf{n}}_i^\gamma \mathbf{v}^\gamma(x_i) = \hat{\mathbf{n}}_i^0\mathbf{v}^0(x_i) + \hat{\mathbf{n}}_i^1\mathbf{v}^1(x_i) + ... + \hat{\mathbf{n}}_i^{D-1}\mathbf{v}^{D-1}(x_i). \tag{3.20}$$

We are now in a position to set up the matrix components of the wall regulariser by following the argument from (3.21),

$$S_{ij}^{\gamma} = \hat{\mathbf{n}}^{\gamma}\beta^{(0)}(x_i^0 - n_j^0)\beta^{(1)}(x_i^1 - n_j^1)\ ...\ \beta^{(D-1)}(x_i^{D-1} - n_j^{D-1}), \tag{3.21}$$

$$c = \arg\min_x \left\| \begin{pmatrix} \mathbf{S} \\ \lambda\dot{\mathbf{S}} \\ \kappa\mathbf{S}_{\perp} \end{pmatrix} x - \begin{pmatrix} b \\ \mathbf{0} \\ \kappa\mathbf{v}_{\perp} \end{pmatrix} \right\|^2 \tag{3.22}$$

Although strict consistency requires $v_{\perp}$, the value zero can often be used with acceptable results.

The temporal algebraic extension of (3.22) is trivial, however in practice, the additional dimensions causes memory requirements to grow outside of the feasible regime for currently available hardware. Gomez(Gom13) suggested a patch based divide-and-conquer algorithm to overcome this problem at the cost of increased computational complexity. This strategy may be well worthwhile if some time-dependent physical constraint can be added, such as conservation of volume throughout the cardiac cycle.

### 3.2.4 CUDA implementation

In order to set up the problem for efficient computation, a sparse matrix implementation was chosen. Naively, the approach would be for each point to calculate spline functions of the distance to every node in each direction, so that the memory requirements become $|X| \cdot |N|$, where $X$ is the set of regression points and $N$ is the set of grid nodes. In meaningful data, both these sets are typically exponential in the dimensionality $D$, $|X| \sim |N| \sim (\cdot)^D$. With increasing dimensionality, the memory footprint of the computation rapidly outgrows the capability of any modern computer in the dense matrix format.

To create the sparse sampling matrices $S$, we note that each point has exactly $n+1$ nonzero-valued splines within its support, and by taking advantage of this convenient locality of the regression basis, only splines functions within the range $\mathcal{R} = \left[-\frac{n+1}{2}, \frac{n+1}{2}\right]$ in every dimension have to be calculated and stored. Therefore, the memory requirement is reduced to scale as $|X| \cdot (n+1)^D$ in order and dimensionality. The direct benefit of this is that node density no longer becomes a limiting factor in terms of storage and we can tune the amount of smoothing without penalty.

### 3.2.5 LSQR solver

To solve the demanding 3D system, a fast LSQR solver for sparse GPU matrices was developed. This was achieved by interfacing the cuBLAS and cuSPARSE libraries in the CUDA Toolkit. LSQR is an iterative sparse linear LS solver that uses the Lanzcos process to perform a bidiagonalisation. The derivation of the algorithm is presented in (PS82). Algorithm 2 provides an overview of the algorithm.

**Data:** A,b
**Result:** LSQR
$\beta_1 u_1 = b,\ \alpha_1 v_1 = A^T u_1,\ w_1 = v_1,\ x_0 = 0$
$\bar{\phi}_1 = \beta_1,\ \bar{\rho}_1 = \alpha_1$
**for** $i = 1, 2, 3...$ **do**
  // Bidiagonalisation
  $\beta_{i+1} u_{i+1} = A v_i - \alpha_i u_i$
  $\alpha_{i+1} v_{i+1} = A^T u_{i+1} - \beta_{i+1} v_i$
  // Orthogonal transformation
  $\rho_i = (\bar{\rho}_i^2 + \beta_{i+1}^2)^{1/2}$
  $c_i = \bar{\rho}_i / \rho_i$
  $s_i = \beta_{i+1} / \rho_i$
  $\theta_{i+1} = s_i \alpha_{i+1}$
  $\bar{\rho}_{i+1} = -c_i \alpha_{i+1}$
  $\phi_i = c_i \bar{\phi}_i$
  $\bar{\phi}_{i+1} = s_i \bar{\phi}_i$
  // Update $x$ and $w$
  $x_i = x_{i-1} + (\phi_i / \rho_i) w_i$
  $w_{i+1} = v_{i+1} - (\theta_{i+1} / \rho_i) w_i$
**end**

**Algorithm 2:** Fluid update procedure

The solver is modified to handle a weighting vector in the weighted LS minimisation

$$c = \arg\min_x ||\mathrm{diag}(w)(\mathbf{A}x - b)||^2, \tag{3.23}$$

This weighting can in principle be applied directly to the sampling matrix $A$ by a costly row-wise matrix multiplication. This process bakes weighting into $A$ which could be undesirable in cases where the matrix $A$ can be reused for a modified set of values such as a grid interpolation. In our modified method, we apply the weighting on vector level inside the bidiagonalisation procedure by noting that if

$W = \mathrm{diag}(w)$ and $\tilde{A} = WA$,

$$\tilde{A}v_i = WAv_i = w \odot (Av_i) \tag{3.24}$$

$$\tilde{A}^T u_{i+1} = A^T W^T u_{i+1} = A^T (w \odot u_{i+1}), \tag{3.25}$$

where $\odot$ denotes the elementwise vector product. This means that the weighted system using $\tilde{A}$ is solved equivalently by replacing the appropriate operations in algorithm 2 with (3.24) and (3.25). All sparse matrix routines are provided by cuSPARSE and the dense vector routines are contained in the cuBLAS library. These libaries provide extremely fast numerical methods on the GPU for dense and sparse linear algebra.

From the computational setup $A$ is stored as a Compressed Sparse Row (CSR) matrix. Due to the particular way the matrix is laid out in memory in this compression format, the operation $A^T u_{i+1}$ is significantly slower than an equivalent non-transposed multiplication. For this reason, the matrix is transposed and stored as a separate copy prior to entering the convergence loop following (HWLC12). The transposed matrix is used direction in the faster non-transposed matrix-vector multiplication routine. This significantly improves performance in cases there where strict stopping tolerances forces many loop iterations before convergence is achieved, since the transpose computation is only performed once for the whole loop. The doubling of storage requirement for storing both matrices can become significant for larger system, forcing a fallback to the standard loop.

Due to the lack of temporal spline fitting in the framework, a temporal gaussian blur is applied to the coefficient tensor, tuned to some specified parameter.

## 3.3 Blood speckle tracking (BST) framework interface

To obtain the blood velocity estimates, we interface a CUDA accelerated Blood Speckle Tracking (BST) library developed at Department of Circulation and Medical Imaging, NTNU. The hand-tailored Foreign function interface (FFI) utilizes the *ctypes* Python library to natively call the exposed C functions through the Application Binary Interface (ABI) with the tracking parameters provided in a C `struct`. From the tracking library, we obtain blood flow estimates along with computed speckle correlation to be used as point-wise weighting input.

## 3.4 Evaluation of method

The model is validated towards a CFD model of a neonatal heart developed by Joris Van Cauwenberge and Abigail Swillens at Ghent University This model should has high enough accuracy to be considered a Ground Truth (GT) of the flow inside the specified domain(SLK+09).

The CFD phantom will be used in two distinct ways. Firstly, the phantom will be used to validate the general accuracy of the particle transport in the SPH model. The particles are propagated over a range of prediction intervals determined by chosen measurement rates. The ventricle is modeled as a truncated prolate spheroid with an elliptical mitral inlet and circular aortic outlet. Please refer to (VC14) for a full description of the flow model.

Secondly, the flow fields from the phantom are used to simulate an ultrasound image suitable for BST. This is performed using the Fast Ultrasound Imaging Simulation in K-space (FUSK) software to simulate moving scatterers in a velocity field and imaging the interference field by a Point Spread Function (PSF) convolution. The PSF is the spatial impulse response of the ultrasound acquisition system. Baseband demodulation yields the In-phase Quadrature (IQ) signal. The resulting signal is passed through a clutter filter to simulate the procedure of tissue rejection in the data. The frequency response of the filter is shown in figure 3.3.

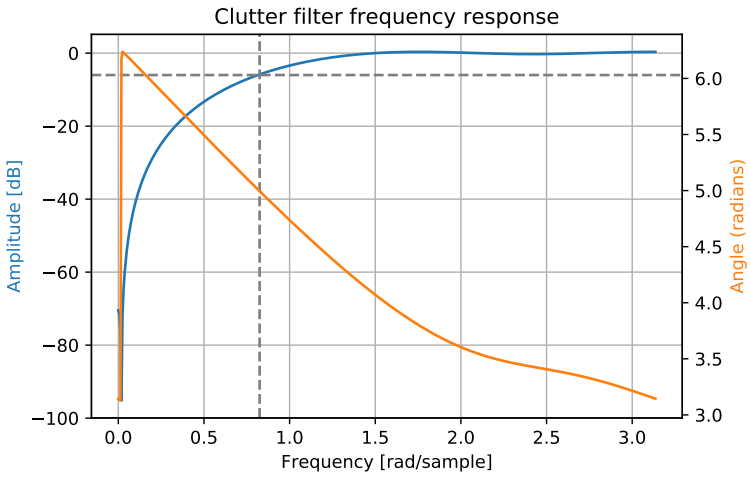


**Figure 3.3:** The response of the Finite Impulse Response (FIR) clutter filter used to simulate tissue rejection.

**Table 3.1:** The parameters used in the speckle tracking procedure.

Speckle tracking parameters

| Parameter | Value |
|---|---|
| Maximum velocity | [1, 1, 1] m/s |
| Kernel size | 1mm x 1mm x 1mm |
| Kernel dims | 7 x 7 x 7 |
| Tracking grid density | [1,1,1] $mm^{-1}$ |
| Kernel minimum samples | [3,3,3] |
| Subsampler | Parabolic |

In order to obtain the wall definition for the B-spline smoother, the boundary points were extracted through binary morphology and the points sent to the qhull library for computation of the Delaunay triangulation of the point set.

All methods were run on a NVIDIA TITAN X graphics card.

# 4 | Simulations and results

In this chapter we present the results of the majority of simulations that contributed to the validation of this project. First the phantom flow field and the BST estimated velocities from the FUSK phantom is presented. We then give a visual verification by simulating a free surface setup demonstrating the boundary force model. Subsequently, the inherent smoothing of the model is examined before a quantitative analysis of the model quality with respect to the CFD model to determine its predictive power. Further, the model is applied to velocities obtained via the BST estimator using the FUSK simulated image to study the filtering effects of the model.

A static B-spline smoother is presented to serve as a comparison to the dynamical filter explored here and is used to provide comparable estimates in all cases but the advection evaluation.

## 4.1 CFD phantom

In this section, an overview of the simulated CFD flow phantom is presented. The phantom will be used as ground truth in following computations. Figure 4.1 shows the CFD flow field as seen in the apical plane passing between both valves, and figure 4.2 shows The flow fields are displayed at a regular interval and represented by absolute velocity and in-plane components. Figure 4.3 displays the result of the primitive segmentation as a mesh overlaid with the computed normal vectors used in the B-spline regularisation framework. The mesh is created by a Delaunay triangulation of the levelset that defines the domain in the SPH computation, obtained through an exact zero-limited velocity thresholding. (GSN+16) showed how the same measurements can be obtained in a segmentation pipeline to be used for identical computations as the method used here will not work *in vivo*.

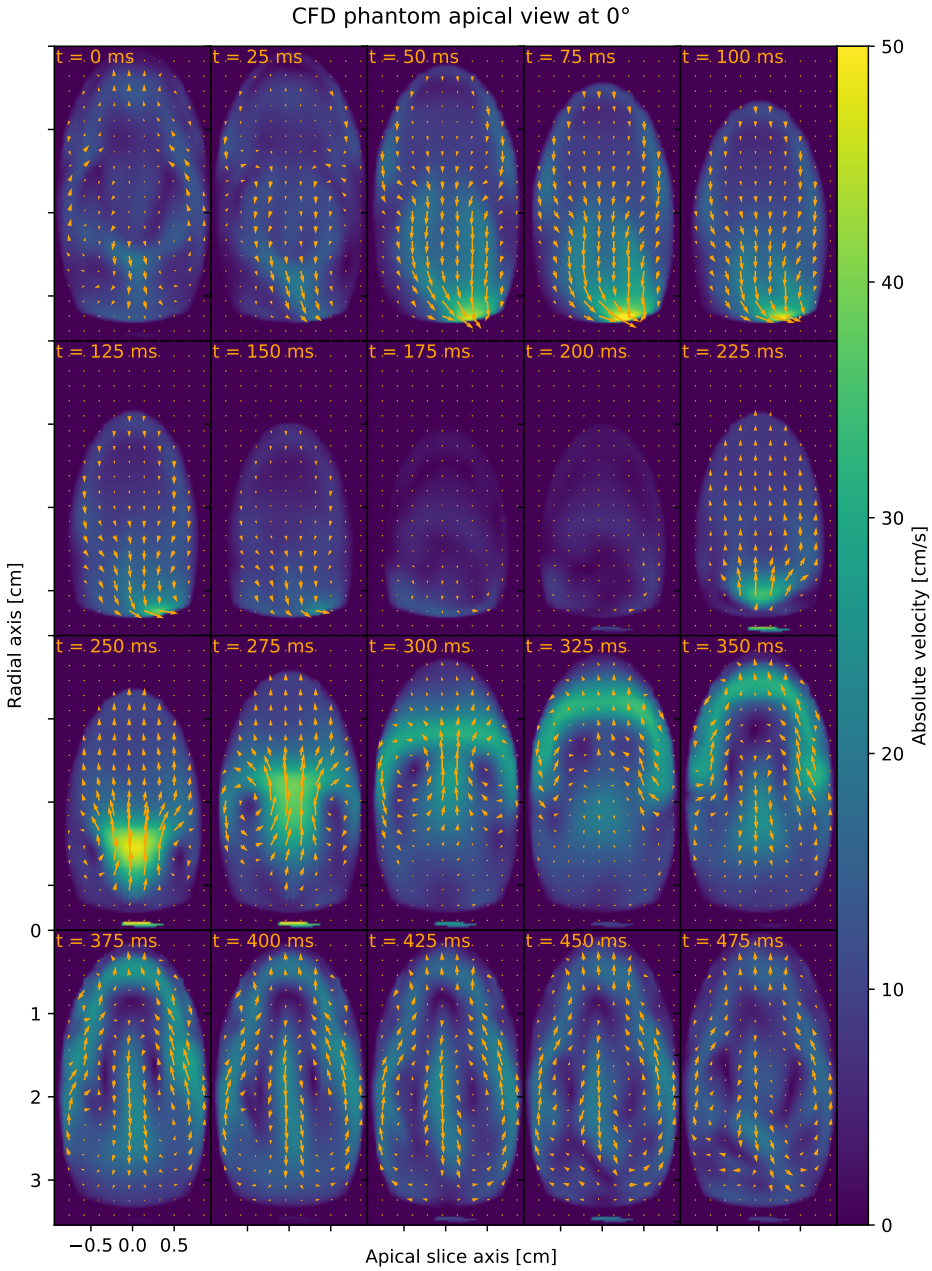


**Figure 4.1:** The figure shows the velocity field inside the CFD phantom at constant intervals of 25ms. The phantom is rotated 0° around the apical axis, and the middle volume slice is extracted. The field is displayed with an absolute 3D velocity heatmap, and a directional overlay of in-plane velocities is shown as arrows. The slice corresponds to the plane passing through apex and between the valves.

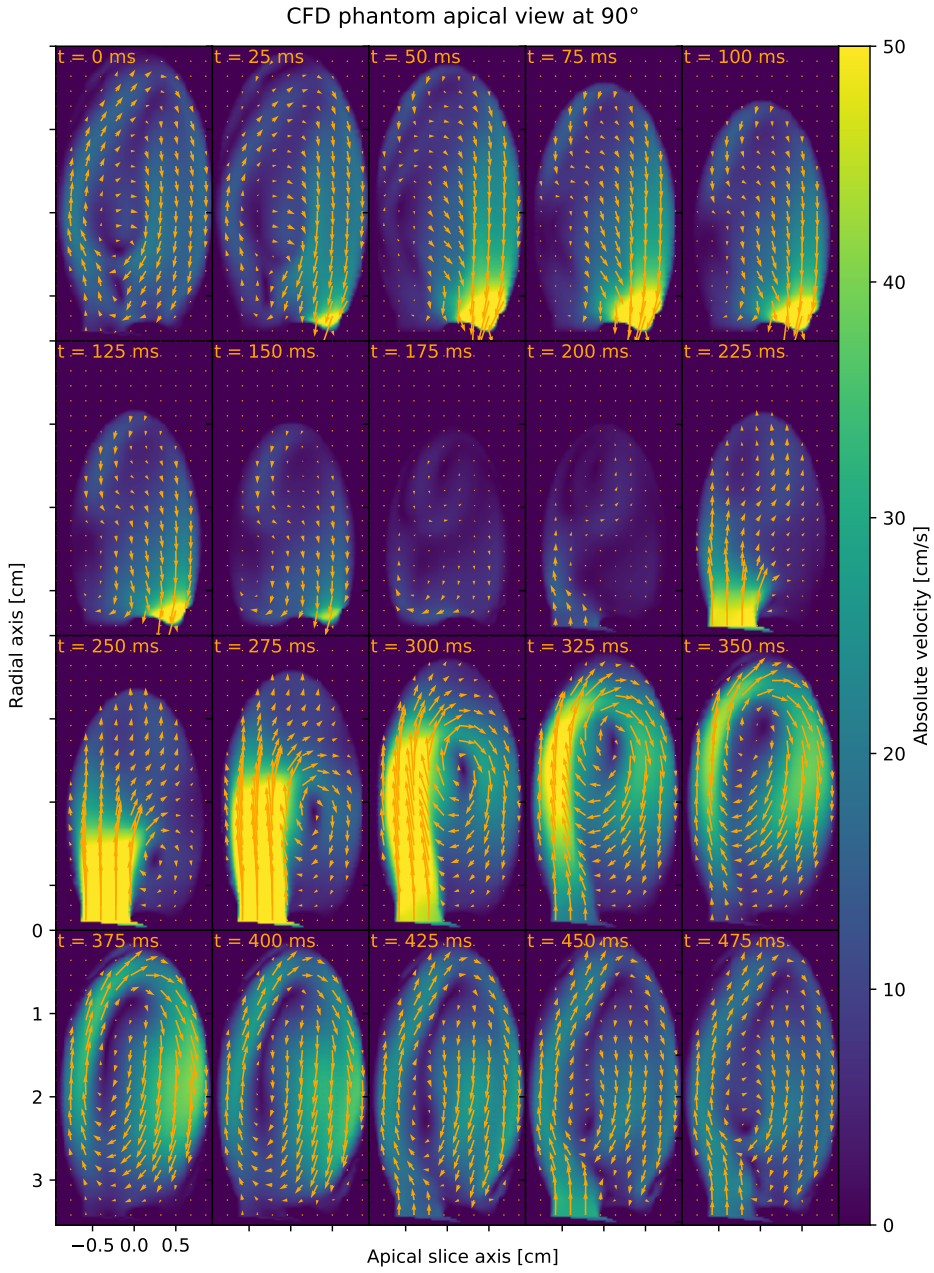


**Figure 4.2:** The figure shows the velocity field inside the CFD phantom at constant intervals of 25ms. The phantom is rotated 90° around the apical axis, and the middle volume slice is extracted. The field is displayed with an absolute 3D velocity heatmap, and a directional overlay of in-plane velocities is shown as arrows. The slice corresponds to the plane passing through apex and both valves.

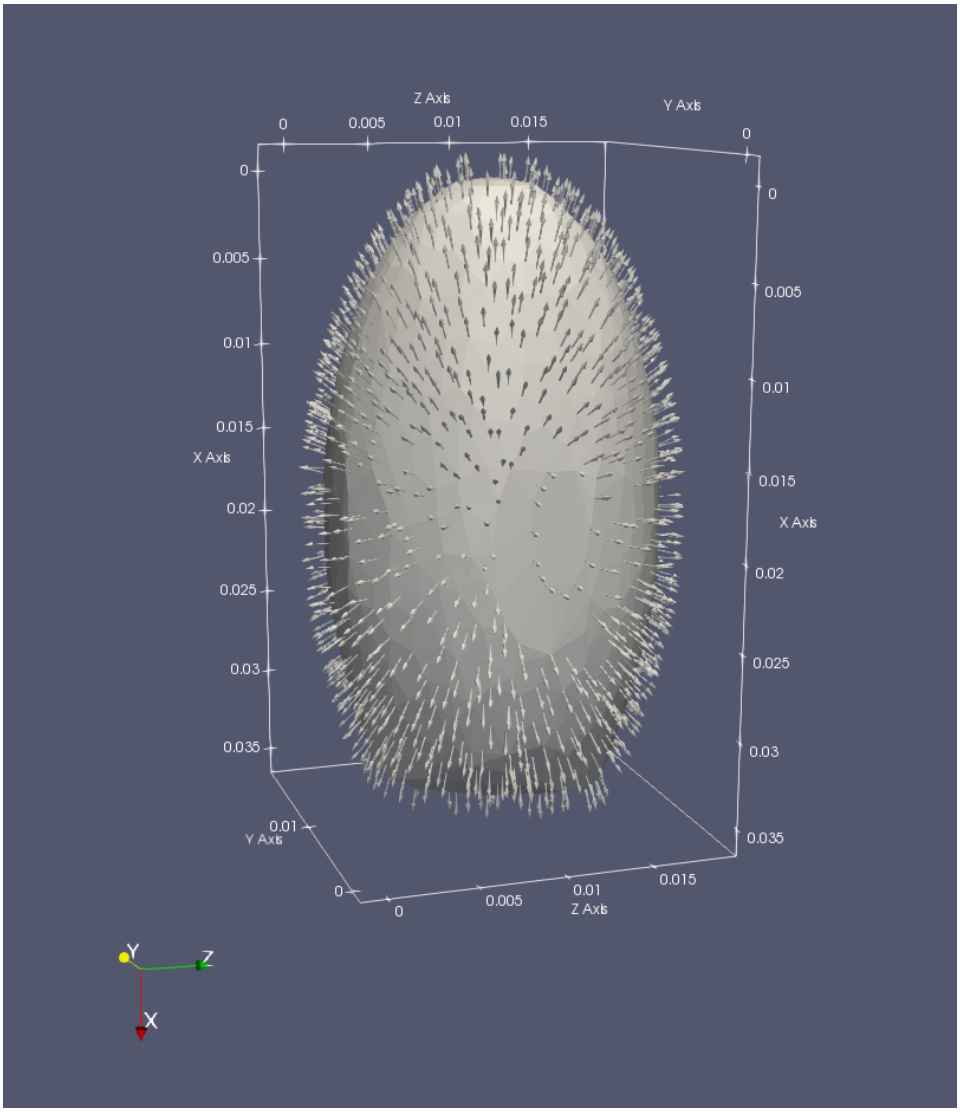


**Figure 4.3:** Result of the segmentation along with the computed normals. Segmentation is obtained through a simple velocity thresholding of the CFD flow and the mesh is created through Delaunay triangulation on the resulting domain point cloud.

## 4.2 BST estimates

Figure 4.4 and 4.5 shows the velocity estimates obtained from the blood speckle at two orthogonal apical planes. The estimates outside the boundary are removed manually to provide a clearer view of the relevant estimates.

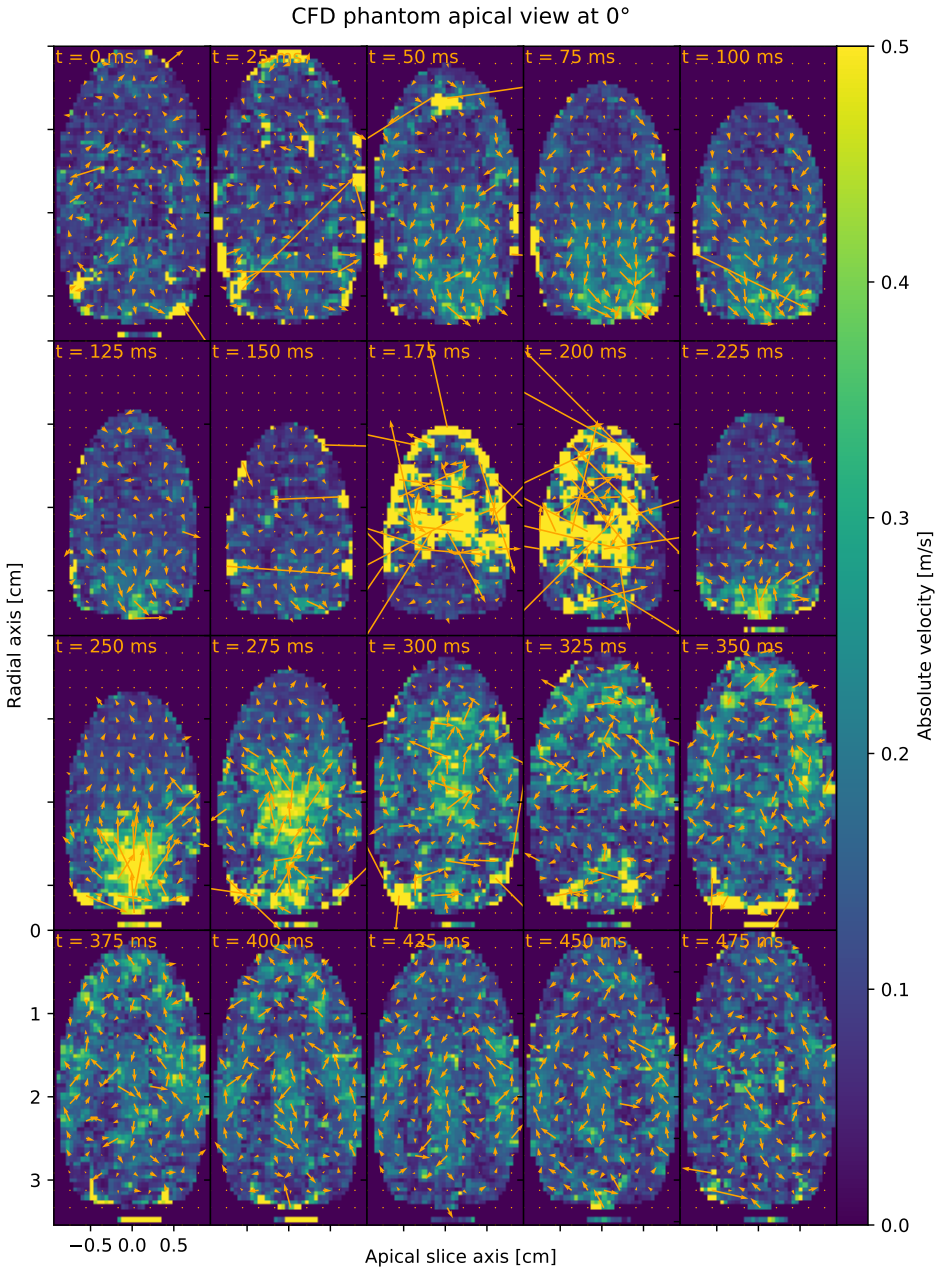


**Figure 4.4:** BST velocity estimates from the clutter filtered FUSK simulated waveform data from the phantom. The flow field is displayed at regular intervals of 25ms, rotated 0° around the apical axis, and the very noisy out-of-domain velocities are masked by the segmentation to avoid occluding the estimates.

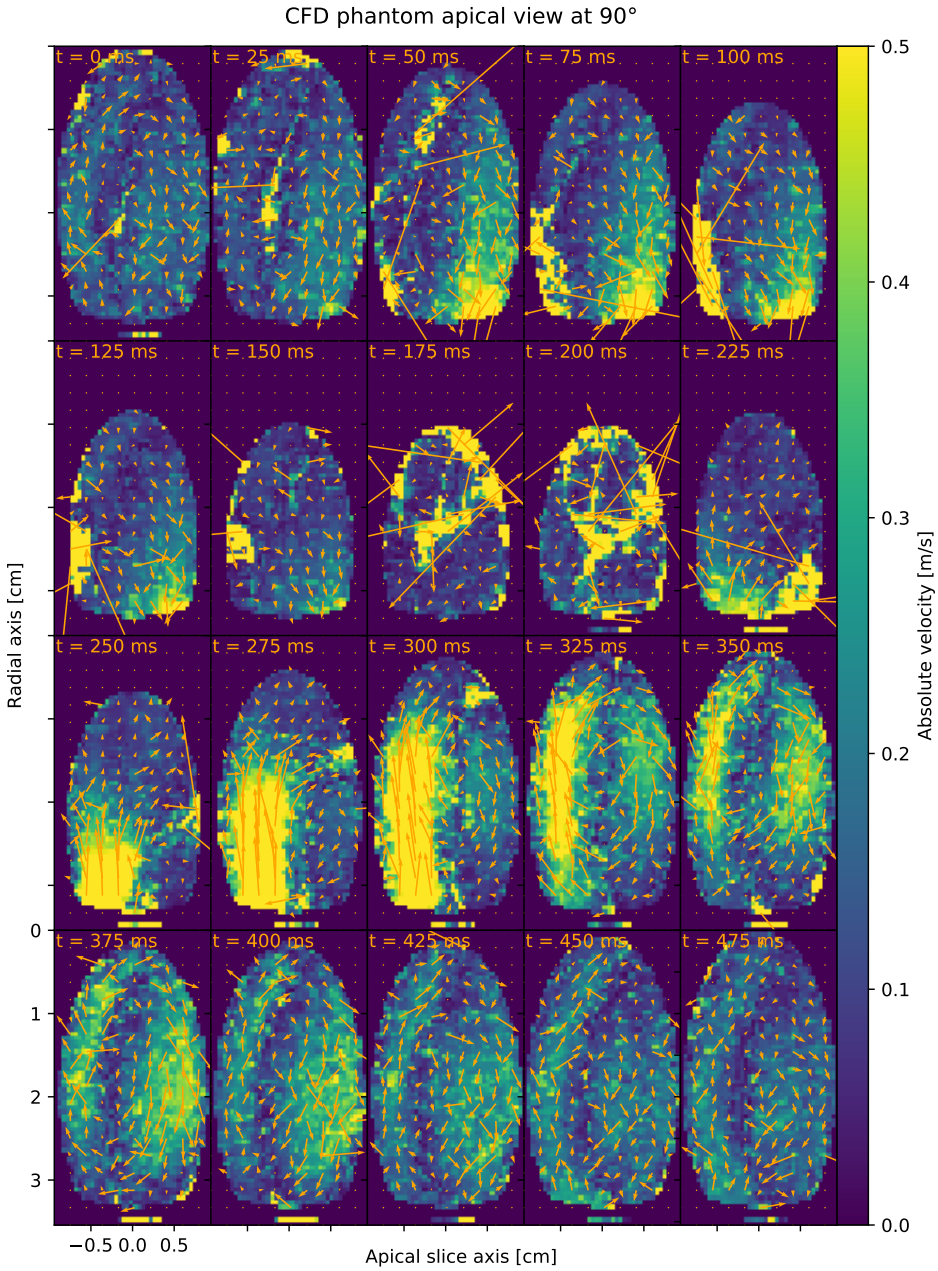


**Figure 4.5:** BST velocity estimates from the clutter filtered FUSK simulated waveform data from the phantom. The flow field is displayed at regular intervals of 25ms, rotated 90° around the apical axis, and the very noisy out-of-domain velocities are masked by the segmentation to avoid occluding the estimates.

## 4.3 Inspective verification

The classical dam break is a commonly used validation setup for free surface SPH Figure 4.6 shows a modified version of a dam break; a water column is pulled by gravity onto a potential wedge to split the flow.

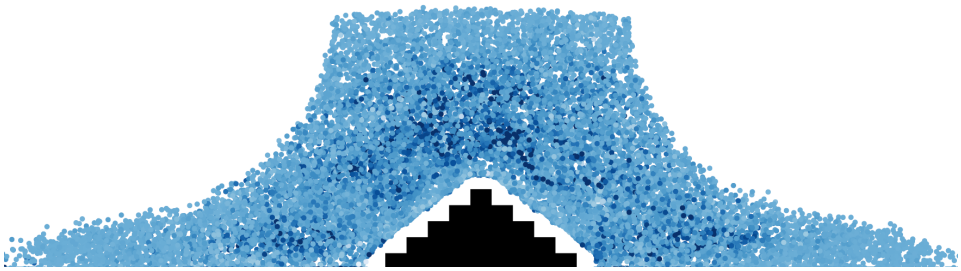

**Figure 4.6:** Illustration of the texel interpolation method for boundary force computing. A block of fluid particles are pulled by gravity onto a coarsely resolved wedge. The spacing between the wedge and the fluid particles originates in the finite difference discretisation.

## 4.4 Inherent model smoothing

Figure 4.7 shows the smoothing effect of imposing a grid of velocities with uniform weighting onto the particle domain and immediately fetching the particle states on an identical grid. The effective smoothing depends on the number of particles in the system, and the error compared to the original velocities is shown for a wide range of particle numbers. The error is evaluated for the full frame sequence and no fluid parameters are relevant for this validation.

Figure 4.8 shows the error in the smoothing provided by the B-spline grid. The grid setup performs no regularisation in this test, meaning the result is solely from interpolation error as to be comparable to 4.7.

Figure 4.9 illustrates the effect of increasing the cell grid dimension of the fluid simulation. The filtered estimates are imposed and extracted at $t = 0$ to not be perturbed by advection errors. The grid density used in this calculation corresponds to the particle numbers used to generate 4.7

Similarly, figure 4.10 shows how increasing the grid node density in the domain affects the quality of the smoothing at a single timestamp. The smoothed values are taken at $t = 0$ for comparison with 4.9.

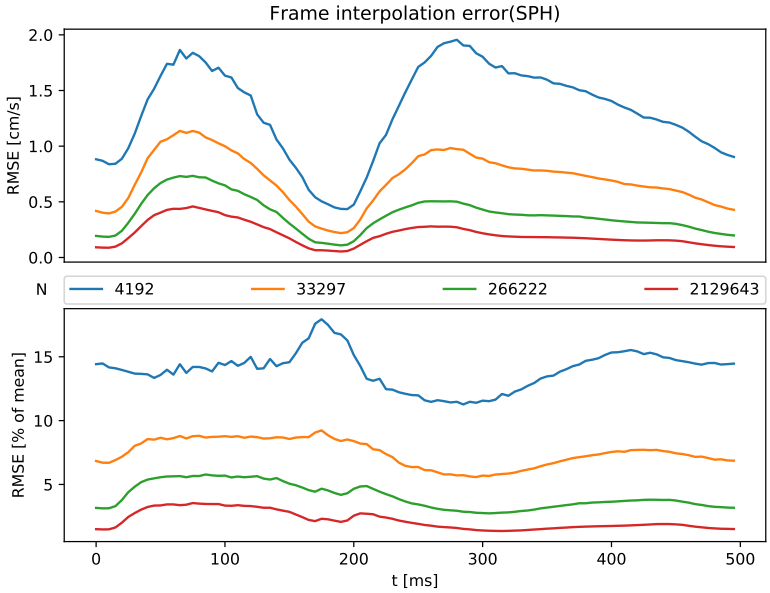


**Figure 4.7:** Frame sequence error due to interpolation artifacts in the SPH model. The interpolation is evaluated for a range of particle numbers. The top panel shows the absolute velocity error inside the flow domain, whereas the bottom panel displays the GT-mean normalised errors.

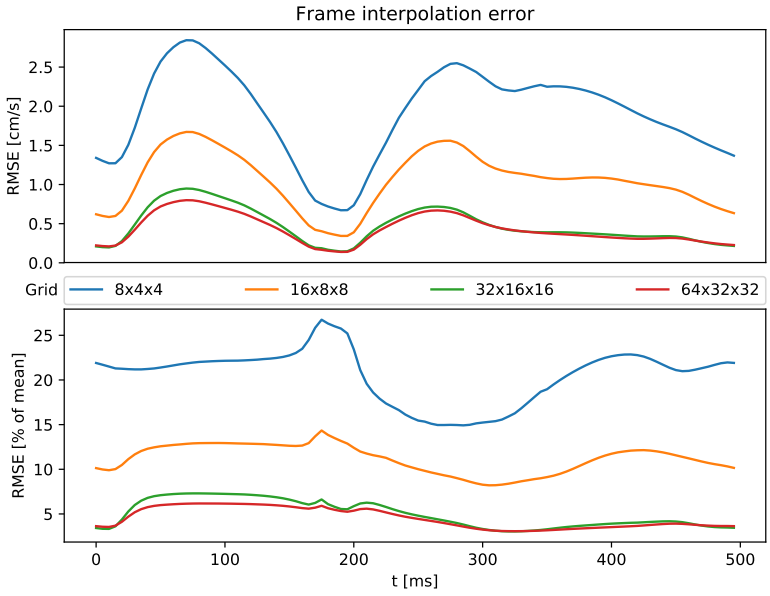


**Figure 4.8:** Frame sequence error in the smoothing provided by the B-spline grid. The smoothing is evaluated for a range of grid sizes. The top panel shows the absolute velocity error inside the flow domain, whereas the bottom panel displays the GT-mean normalised error.

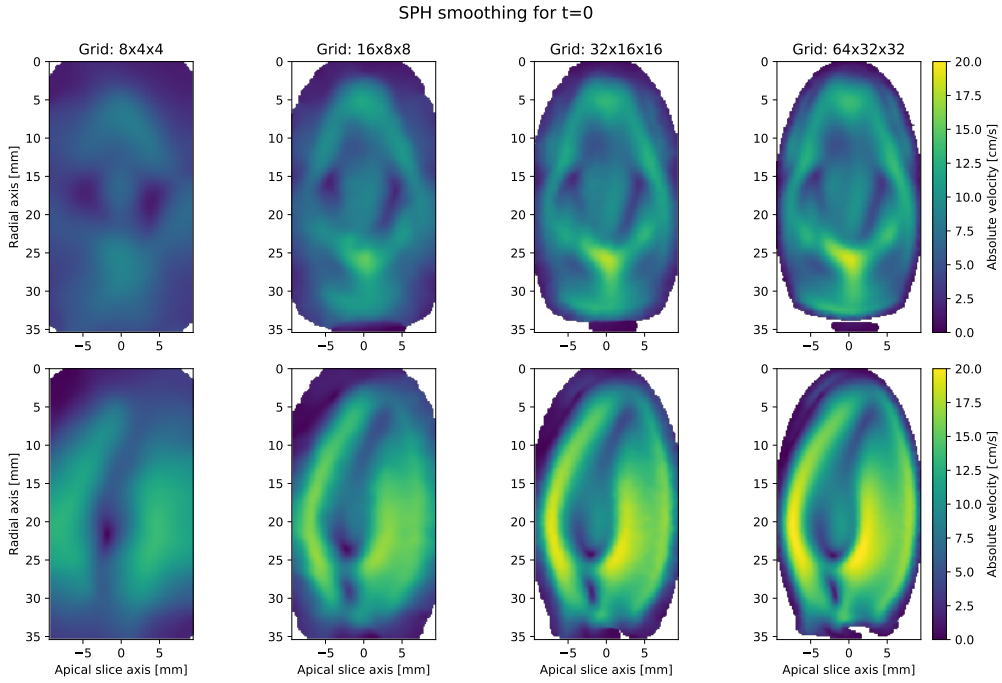


**Figure 4.9:** Effect of increasing the number of cells in the SPH domain. The frame in view is taken at $t = 0$.

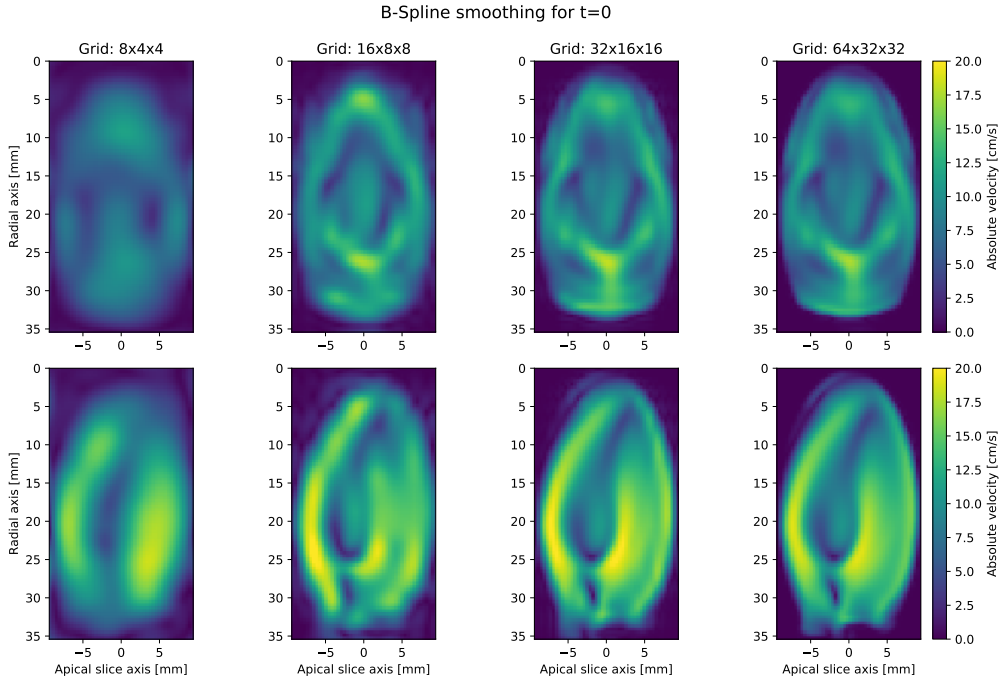


**Figure 4.10:** Effect of increasing the number of nodes in the spline grid. The frame in view is taken at $t = 0$

## 4.5 Prediction validation

In figures 4.12 and 4.13, the model's ability to simulate left ventricular flow is evaluated. The GT is applied at every frame before a prediction over an interval and the final flow field compared to the expected solution as given by the CFD phantom. To imitate the conditions at the valves in the CFD model, pulsating gauge pressure sources was added inside the fluid domain to simulate opening and closing of the valves, as illustrated in figure 4.11. This helps the model anticipate opening or closing of the valves during the prediction period where this could lead to issues.

Figure 4.12 does not take into account the dynamic pressure at the valves, as described in section, and the effect of introducing these can be seen in 4.13. The fluid system parameters used in this simulation is described in table 4.1

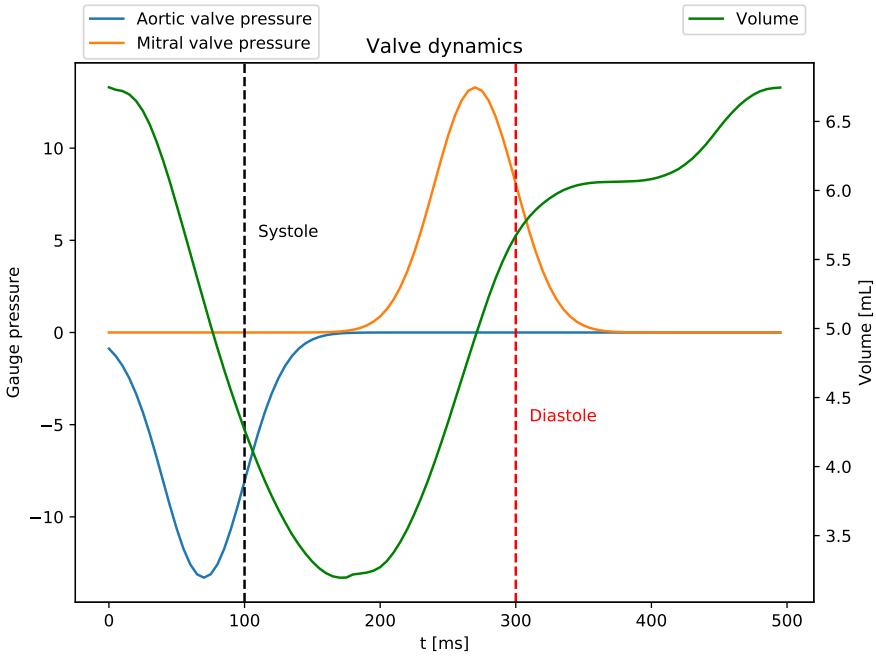


**Figure 4.11:** Dynamic pressure at the valves in the SPH model. Domain volume is drawn in green.

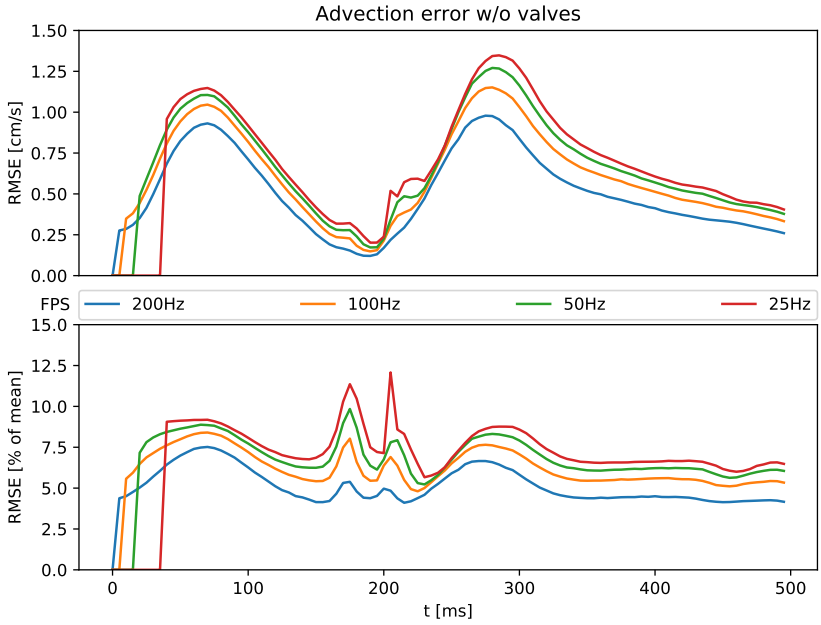


**Figure 4.12:** Frame error for advections over prediction intervals at different measurement rates. The simulations are performed without valve modeling.

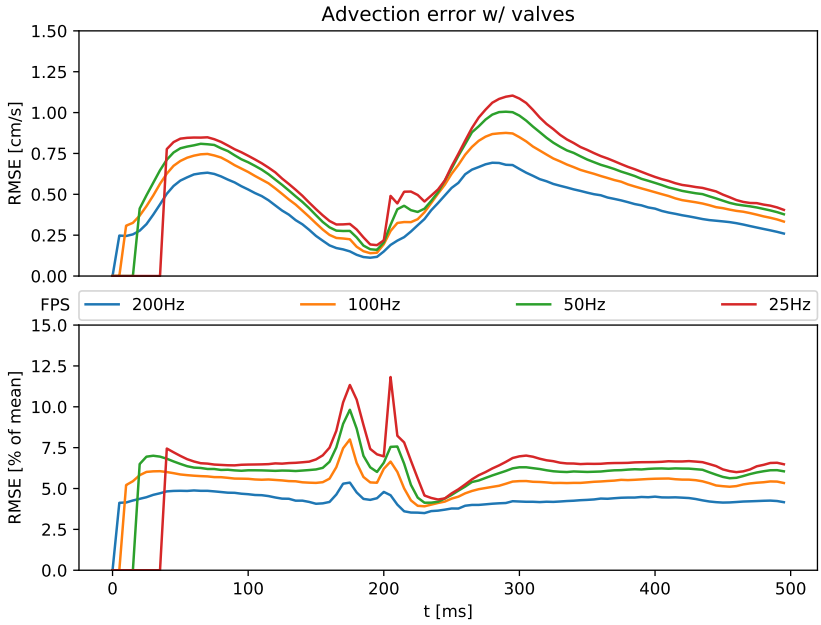


**Figure 4.13:** Frame error for advections over prediction intervals at different measurement rates. The simulations are performed with valve modeling.

**Table 4.1:** The parameters used in the SPH fluid simulation

| SPH fluid parameters | | B-Spline parameters | |
|---|---|---|---|
| Parameter | Value | Parameter | Value |
| Grid size | [40,20,20] | Grid | [32,16,16] |
| Cell size | 32 | $L_2$ damping | 0.5 |
| Rest density $\rho_0$ | 1 | Divergence damping $\lambda$ | 500 |
| Speed of sound $c_0$ | 5 | Wall regularisation $\kappa$ | 200 |
| Background pressure $p_0$ | 5 | Weighting power | 2 |
| Polytropic constant $\gamma$ | 3 | Temporal blur | 0 |
| XSPH parameter $\epsilon$ | 0.2 | | |
| Artificial viscosity $\alpha$ | 0.5 | | |
| Frame subsampling | 100 | | |

## 4.6 Regularising power

Figure 4.14 shows how the SPH model compares to the B-spline regularisation for reducing noise from estimator data. The setup for each method is described in table 4.1.

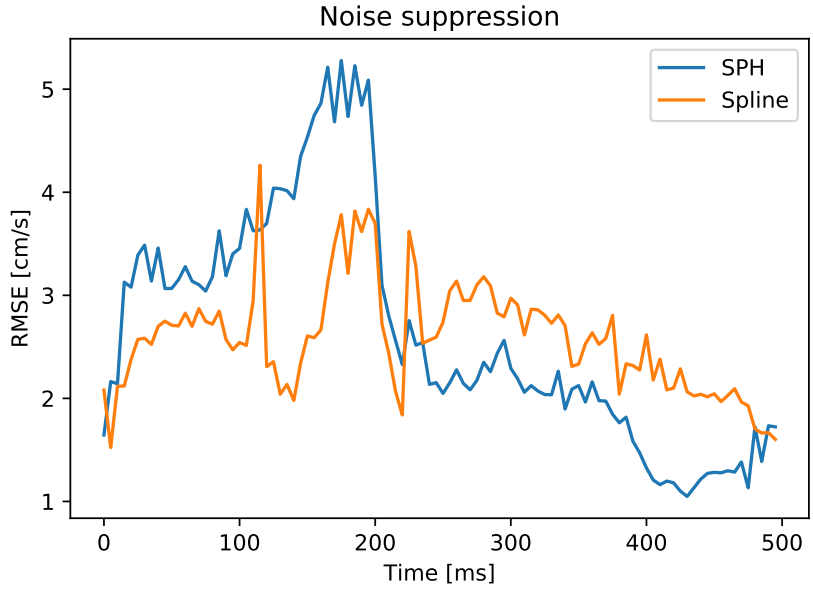


**Figure 4.14:** Comparison of the quality of the smoothing effect caused by both B-spline and SPH models. The setup for the fluid model and the spline grid is described in table 4.1.

Figure 4.15 show how the error is distributed at $t = 50$ ms for the SPH method. Note how most of the error is loacated near the outlet.

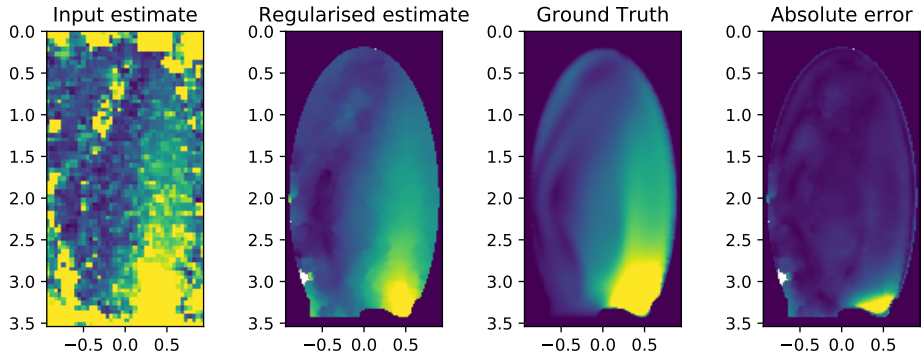


**Figure 4.15:** Visualisation of the origin of regularisation errors

## 4.7 Processed sequences

Figures 4.16 and 4.17 displays snapshots of the sequence using SPH and the spline smoother respectively. The same setups were used as described in table 4.1

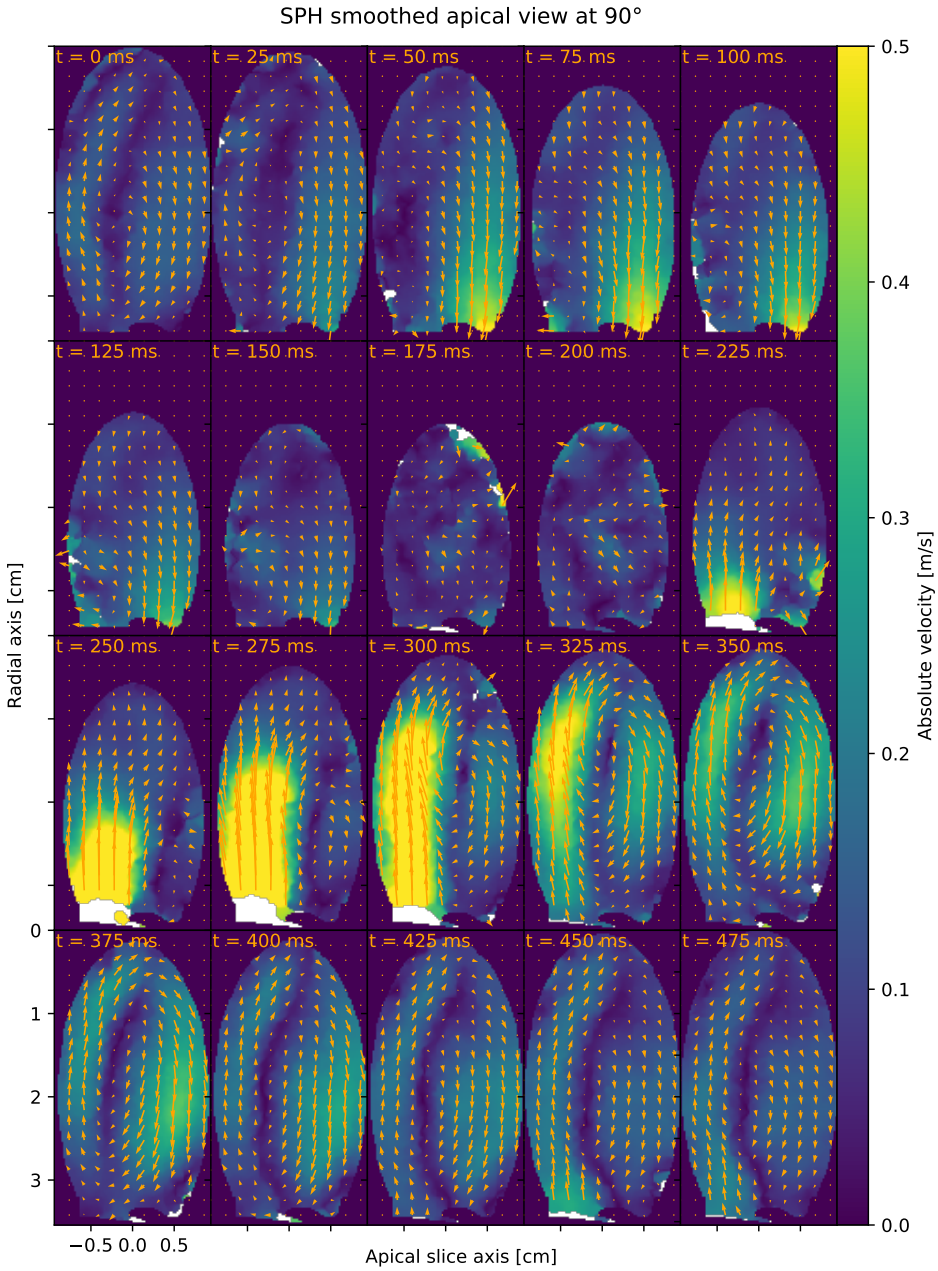


**Figure 4.16:** Sequence of SPH smoothed BST estimates using the parameters from table 4.1

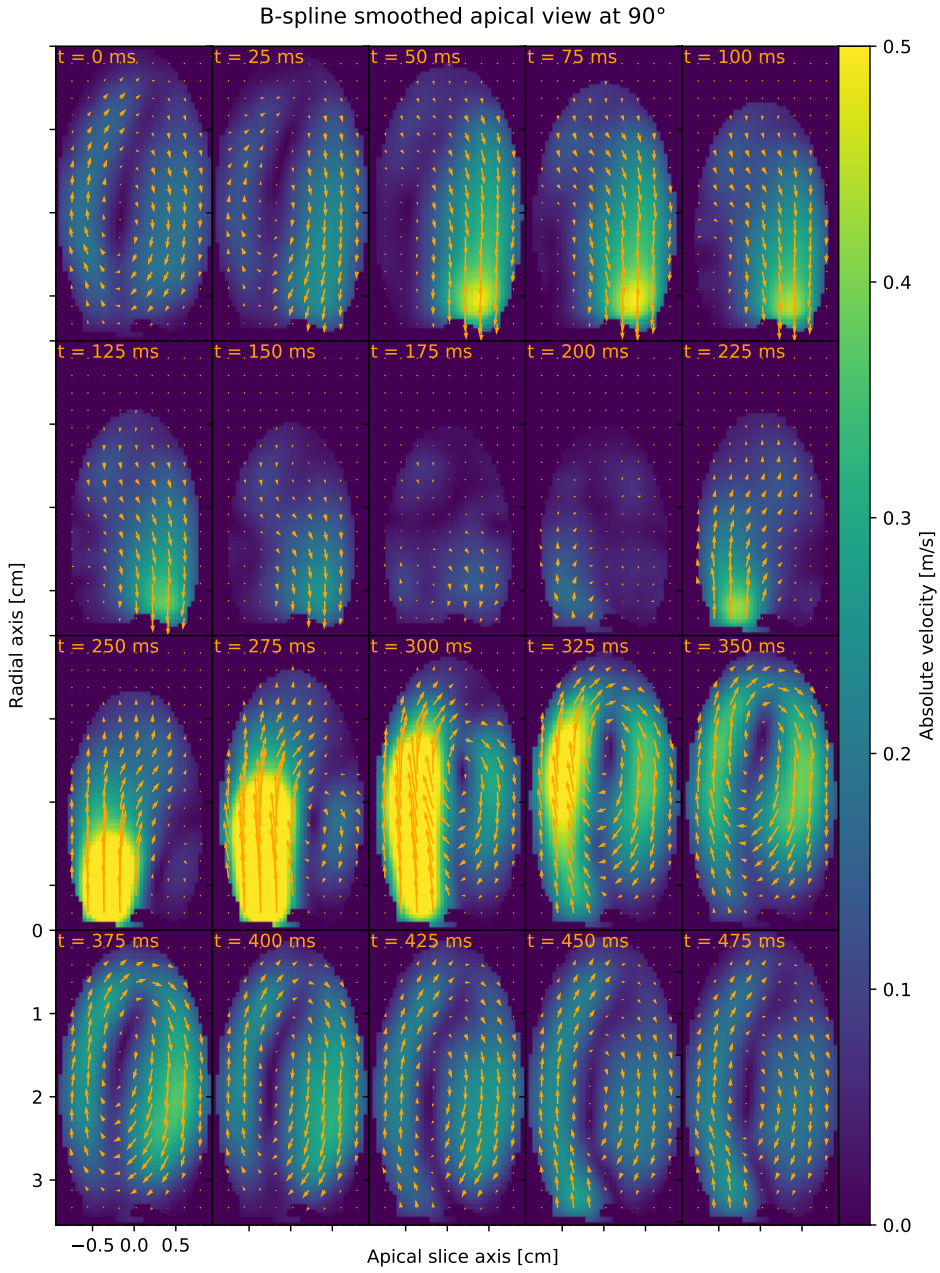


**Figure 4.17:** Sequence of B-spline smoothed BST estimates using the parameters from table 4.1

## 4.8 Performance

Figure 4.18 shows the performance of the advection kernel using different cell sizes. Larger cell sizes increase the number of interactions computed per particle.

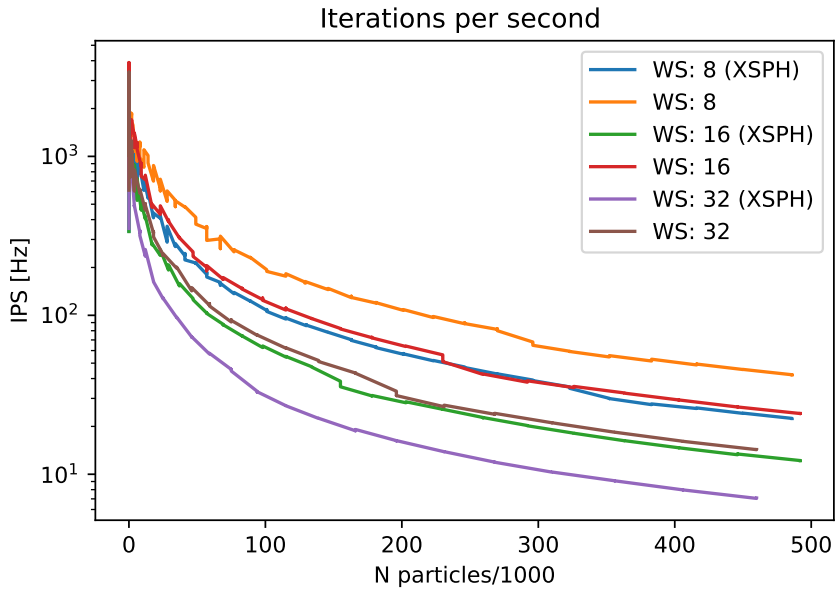


**Figure 4.18:** Performance analysis of the particle advection step for different warp subdivision sizes. The achieved number of iterations per second is shown for a range of particle numbers. Performance is evaluated both with and without XSPH-correction.

# 5 Discussion

## 5.1 Results

From figures 4.9 and 4.10 it is observed that, qualitatively, the SPH filter seems to provide more fluid texture in the resulting field similar to CFI estimates, whereas the B-spline smoother applies more of a uniform Gaussian blur to the field. This should come as no surprise, considering the mechanism behind the spline interpolation. However, the SPH textures may be explained by the same phenomenon Gingold and Monaghan encountered, namely that controlled disorder in the particles allow for a more random sampling that estimates the kernel in a way that is more consistent with the dynamics of the system.

The theory appears to be backed up quantitatively as seen in figures 4.7 and 4.8, where the B-spline saturates at a higher overall error than the SPH model. But this is a fluke; saturation in the B-spline smoother causes the interpolation to improve very little when increasing node density. The reason being that the grid spacing has shrunk below the characteristic scale of the flow features, removing the spatial low pass filtering effect of the smoother. The smoother should in principle fit the system exactly at this point, but we suspect that the hidden source of error is loose stopping tolerances in the LSQR solver required for stiffer systems.

This points to a difference between the two implemented methods that cannot be captured by grid-spacings alone, but the two models appear to provide approximately the same accuracy, but the B-spline results has a "softer" expression, as seen in the sequences 4.16 and 4.17.

This is related to the problem of overfitting vs. underfitting; grid size restricts the minimum extent of any flow structure that can be represented by the coefficient tensor, but fine grids tend to fit noise in the measurements. Ideally, we would want the SPH model not to suffer under this trade off, i.e. to dampen noise the solution should be to evolve the system further, not reduce the number of particles and lose the ability to resolve fine scale details.

An important consideration is that the B-spline framework could not handle the full resolution of the CFD model, and a decimation of the flow grid had to be done. This problem is "invisible" in SPH, since any grid can be applied and interpolated onto the particles. The required memory at a given resolution is dramatically less in SPH.

From the visual validation of the method and boundary condition 4.6, we observe a natural flow pattern, suggesting that the specific choice of computation model and boundary handling is a viable choice for performing smoothed particle hydrodynamics. The performance found in 4.18 should be comparable to other SPH codes using alternative sorting procedures and neighbourhood searches, which in most cases is the limiting factor on realisable particle throughput.

Experiments showed that in order to maintain stability in the weakly compressible regime, the timestep had to be sub-milliseconds, which means that the particle number for feasible realtime performance is quite limited in the current implementation, particularly in 3D. This can possibly be solved by some of the targets for optimisation outlined below or by improving the stability of the code, allowing for larger timesteps. If compressibility is allowed to some chosen degree reasonable results may be obtained without severely degrading the prediction quality; the trade off between performance and accuracy is clear.

The smoothing from interpolations slightly overshoots the expected integrand error, this points to an error in the computation of grid positions in the state imposition or fetching procedure, possibly related to the half-pixel offsets. This could be fixed by using an asymmetric form of the kernel in the state interaction, i.e. fully scattering or gathering mode instead of $\frac{h_1+h_2}{2}$. This allows more control over the effective smoothing in the interpolation procedure, and, importantly, there are no constitutive equations to be violated by asymmetry at this stage of the filtering.

When attempting to mimic CFD based on velocities alone the model has reduced prediction accuracy in eject and filling phases unless some form of kinematic pressure from valvular dynamics is included, as seen in figure 4.12. This is certainly as expected, seeing as the CFD phantom model explicitly drives the flow field by pulsing pressure curves at the valves to close the boundary value problem, in short; we cannot hope have comparative long term dynamics without including these. This is of course undesirable since it assumes some *a priori* information about the physiology. If not included, however, another assumption is implicitly enforced, namely static conditions at the mitral and aortal valves during the prediction period. This assumption turns more severe as the frame rate goes down, and it is clear that in order to avoid relying on a model, a reasonably high measurement framerate has to be achieved in the acquisition scheme. In situations where the

framerate is low, the method has to rely on the model to a greater degree. We argue that it is beneficial to include pressures from normal data in this case; to obtain a realistic estimate for the flow it is therefore required that the model is consistent with the true dynamics of the system.

Recent work on this suggests feasibility for in vivo setups using a multibeat setup where multiple cardiac cycles are recorded and stitched to obtain upwards to 100Hz, well within the prediction range of the filter, as seen in figure 4.13

When primitive valves are included, it is observed that the SPH model can emulate the CFD phantom with reasonable accuracy even for medium-sized prediction intervals, suggesting a good fit for a model based filtering technique such as the Kalman filter. It is observed a dominant contribution of inertial forces which contest the incompressibility condition, this is symptomatic of the early stage nature of the implementation, and that further tuning is required to correctly model hemodynamics. Although a major constituent of blood is near-incompressible plasma, the stiffness is somewhat loosened by the presence of hematocytes which increases the compressibility in a non-newtonian manner. This rheology is, however, not sufficiently manifested in intracardiac conditions at the scales current numerical methods are able to resolve

High pressure clustering is found in the regions experiencing strong compressive forces such as during the diastole. This causes an effective resolution that is less than expected by the interpolating grid in these areas. The consequence is missing support inside the domain and velocities cannot be extracted at these points, as seen in figure 5.1. Alternative kernels or the tensile correction proposed by Monaghan(2.12) should be investigated as means to correct this. From the addition of pressure sources and sinks the overall prediction quality is improved, it shows problems related to inadequate particle flux at domain interfaces. In particular the lack of dynamic particle reseeding causes the same effect as the particle clustering. The fast moving particles near the inlet 5.1 Use reservoir!

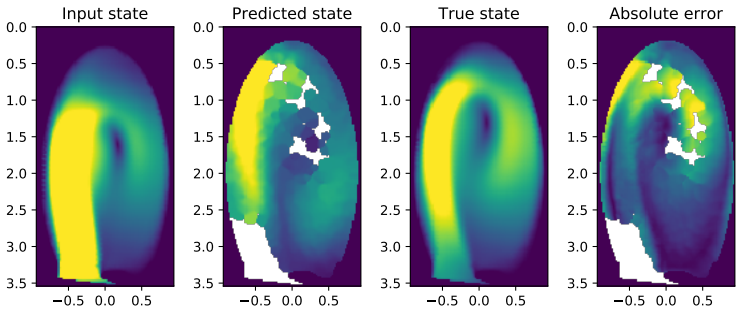


**Figure 5.1:** Extreme clustering effect during filling due to low Bulk modulus. Note the grossly degraded resolution in the compressed region. Also, lack of dynamic reseeding near inlet creates particle vacuum.

Although a particle can be placed in a region in the reference potential level, it might experience boundary force induced by artifacts in the finite difference gradients from nearby walls. This creates an excited initial state, impacting the overall quality of the prediction. relaxation needed to avoid initialisation artifacts impacting the state prediction

A potential problem that needs to be addressed is the mesh computation provided by the `qhull`-library. It is observed that in cases where the domain boundary is flat over an extended region a significant undersampling of wall points for the B-spline regularisation scheme occurs. This causing the wall description to skip the discretisation distance of the grid nodes and exert no wall pressure in the flat region. The behaviour is as expected, as coplanar points are removed due to numerical singularities in the triangulation, and are redundant in the mesh description. A solution to this would be an automatic mesh refinement when the control points of the mesh grow beyond a certain limit w.r.t to the B-spline grid spacing. A different issue in the current implementation is due to incompleteness of the kernel

## 5.2 Further work

The first step onwards from here is to employ the simulator as a system dynamics model in a Kalman filter context. The nonlinear effects of the model requires

an Extended Kalman Filter (EKF) or an Unscented Kalman Filter (UKF). Promising results were found with sensor fusion of BST and Doppler estimates in an UKF(HWN+16), exploring this direction could indeed be a reasonable option following this. Additionally, the model should be evaluated towards experimental data in a standard SPH benchmarking case for free surface flow compiled by Issa and Violeau(IR06).

As noted in chapter 2, there has not been a focus on the rendering aspects of SPH in this work. However, in order to have visual verification of the methods, primitive scatter renderings have been provided by CPU-based visualisation libraries. This visualisation method introduces a significant amount of overhead through a device to host transfer for each displayed frame. In order to improve the realtime visualisation capabilities, we may utilise OpenGL interop supported by CUDA, allowing a direct write into a pixel buffer object(PBO) residing on the GPU which in turn can be rendered *in situ* by OpenGL shaders without requiring a costly device to host memory transfer.

The nature of the type of fluid warrants some deliberation. From the velocity estimators and the inherent variance in these it is clear that the fluid state at the time of imposition does not represent a smooth solution to the incompressible $\gamma = 7$ polytrope Navier-Stokes equation (2.1). The pressures involved in order to satisfy the initial state appear unphysical and some damping from WCSP proves beneficial. It is unknown to the author whether incompressible SPH schemes such as PCISPH or Implicit Incompressible Smoothed Particle Hydrodynamics (IISPH) can readily handle this situation. It is, however, shown that the weakly compressible formulation deals with this by softening the differential equations and the degree of compressibility can be tuned by the polytropic constant $\gamma$. This poses the possibility of an approach where $\gamma$ is allowed to vary throughout the prediction period; i.e. $\gamma$ is ramped up to enforce incompressibility at a higher degree as noise dissipates the flow turns placid under the physical constraints placed upon the particles. The process may be viewed as a continuous transformation of a pseudo-gas cluttered with estimator variance into an incompressible fluid through relaxation. The effect has to be evaluated in combination with Kalman filtering which was shown by Høgenes to greatly suppress noise(HWN+16) in a simplified advection model. Moving forward, maybe $\gamma$ has to be adjusted according to some estimated noise variance metric of the current state.

Another concern towards the nature of the fluid is the incompressibility condition. By construction, WCSPH is a penalty-based method only creating counteracting pressures after a density fluctuation has already occurred. This may result in bouncy fluid behaviour, where Some argue that SPH can never fully model incompressible fluids even though proposed schemes such as IISPH and PCISPH aims

to amend this.

We hypothesise that assigning the correlation measure of the velocity estimates would improve the quality of the filtering. Particularly in the XSPH update, individual weighting of the neighbours would improve the neighbourhood correction. It is desirable to implement variable particle mass, and following (2.19), the correlation could be encoded in the mass.

The boundary model can be extended to include virtual particles at the domain boundaries based on the local potential. This way, completeness of the kernel in these regions is restored. This allows the use of the Summation Density formulations which generally is more stable for long running simulations.

A general trend in which particle based visualisations(ASN+14)(SPK+12) are favoured by clinicians proposes the use of the simulator to generate particle trace visualisations virtually for free by marking certain particles as tracer particles

Cell overflow is a potential issue in the computational model studied here. The effect can be visible in cases where the weakly compressible method cannot counteract extreme stress, causing an accumulation of particles and eventually overflow. This is a far more critical concern in the summation density method than the continuity equation, because the spilling of particles equates to a significant loss of mass and failure to build up a neutralising pressure. In (2.19), density still builds in the cell, eventually counteracting the flow of particles, however, effective resolution is lost in the process due to particle elimination.

An effect that is apparent in the continuity equation by design is the lack of signalling from vacuums, meaning a block of particles with a shared velocity will continue to move collectively forever. This is good for representing free surface flows, but in closed domains, this creates unrealistic particle vacuums behind the valves.

However, we believe that the (2.19) formulation will harmonise well with a dynamic reseeding process adapting the resolution to keep the number of interacting particles constant. Using summation density, special care has to be taken by intelligent splitting and merging in order to have consistent mass conservation. Each particle tends to form an exclusion zone around itself to normalise its density which may be damaged by continuously changed self-contribution(WBTW95). Equation(2.19) facilitates dynamic removal and insertion of particles by interpolating continuum properties from surrounding particles according to the kernel approximation(2.7).

The dynamic reseeding and elimination can be performed in the rebinning phase,

where exact information about particle spills are known.

An interesting approach would be to feed the particles directly into B-spline smoother. The particles can be thought of as point measurements on an unstructured grid in the unit domain, which is the expected input for the B-spline smoother. Maybe a sophisticated temporal filtering would work well with a spatial filter.

The previous implementation of the B-spline framework was limited to 2D regularisation and suffered under the ill-posed condition of divergence free flow. The blood motion could indeed have out of plane components that were closed under the divergence free assumption. In 3D the concept of out of plane motion is void, and the fluid vector velocity is fully described, so the assumption can safely be imposed on the system.

An interesting possibility for both frameworks is the addition of viscoelastic physics for tissue mechanics. In SPH, a myocardial model using a tensorial solid mechanics description can be explored to perform similar regularisation on strain rate measurements in elastography. Further, if both models undergo sufficient validation, a fluid-structure interaction (SDSD$^+$12) may be considered for a full model of the heart capable of a simultaneous regularising of flow and strain rate measurements. Splines are already widely used in strain rate regularisation, suggesting an elegant coupling to the flow splines through some suitable penalty term. The combined model driven by measurements from both flow estimator could possibly perform better than the individual components separately.

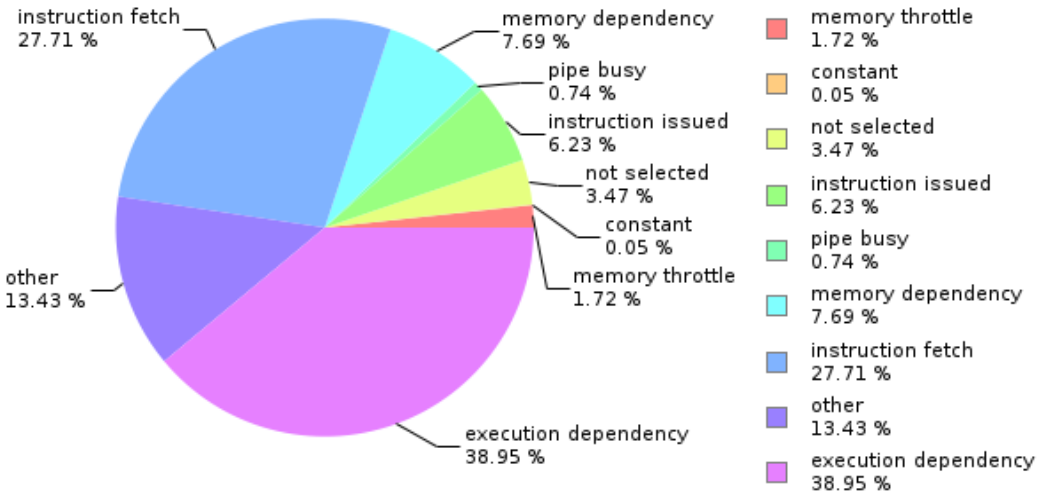


**Figure 5.2:** Instruction stall reasons in the force computation kernel as reported by the NVIDIA CUDA profiler.

As seen in the profiling for the physics computation kernel in figure 5.2, a major reason for kernel stalls is execution dependence. This may be resolved by increasing the instruction level parallelism(ILP) in the code so that the SM can fire multiple instructions back to back. We suspect that the primary source of this is the computation of the smoothing function $W$, where many small floating point operations depend on the previous instruction. A solution to this would be to identify common factors and operations to be pulled out of the loop body.

We also note that the other major factor is instruction fetching. The probable cause for this is a rather large loop body not able to fit in the instruction cache. It is not guaranteed that a loop compaction is possible in this computation model, but the effects of loop unrolling should be investigated.

Another performance concern is the frequent use of $\rho^{-1}$ operations, which is well known to have limited floating point throughput. An alternative scheme should store the inverse value, the volume $V$ of the fluid element to reduce the number of expensive division in the code, as suggested in (GSSP10).

Lower particle limit per cell increases the risk of cell overflow, so the cell capacity should be chosen as to saturate the accuracy of the model. Studies show that increasing the neighbourhood beyond 50 particles contributes little to accuracy of the method since the kernel sampling is sufficiently saturated. As indicated by the kernel profile in figure 5.2, the force kernel is bound by register usage, so

increasing the cell limit also imposes register pressure on the SM which forces spilling into the L1-cache, lowering overall performance.

Rustico *et al.*(RBG+12) discuss the migration of single-GPU SPH codes to multi-GPU and proposes an implementation by a domain partition along the major axis of the linear storage thereby enabling a fast contiguous memory transfer to each GPU. This multi-GPU method would be applicable in this implementation by splitting along the cell grid, allowing an overlap of one cell in the partition direction as illustrated in 2D in figure 5.3. This guarantees completeness in the reduction scheme for all cells except the fringe. A finalising pass over the stitched region collects the computed interactions from both GPUs.

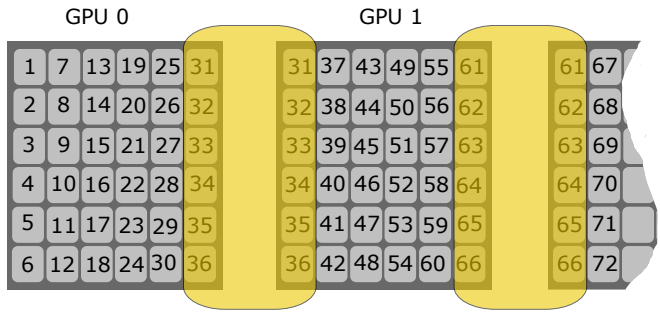


**Figure 5.3:** Proposed multi-GPU implementation of the computation model. The cell layout is split along the major axis and an overlap region is distributed to both GPUs.

A problem that arises frequently in multi-GPU solutions concerns how to do correct load balancing in order to avoid pipeline stalls where one or more GPUs are inactive due to dependency on other possibly slower or heavier loaded GPUs. It is reason to believe that the implementation suggested in this work suffers less from this type of events due to the way interactions are calculated; we always consider a fixed number of neighbours determined by the warp subdivision, regardless of whether the neighbour exists or not. The simplified computation model relies more on parallelism and less on sequential code to utilise the SIMD benefits of the GPU at the cost of redundant computations and increased memory usage. This makes it less prone to heavy warp divergence, a major factor in pipeline stalling. The load balancing can be done as simple as partitioning the cells according to the floating point performance of each . Other hardware factors will undeniably and unpredictably influence the exact performance of each , often by varying degree between passes, such as PCI interrupts and bus congestion.

To generalise the target platform, a very similar implementation can also be done using the low latency shared memory to perform the intrawarp communication.

This would allow a direct OpenCL translation of the code, enabling it to run on heterogeneous systems and avoid vendor lock-in. Profiling should follow this to ensure comparable performance.

Some of these optimisations are low hanging fruit, but they were discarded in favour of more explicit codes due to time constraints in order to avoid introducing inaccuracies from premature optimisation.

As seen in figure 4.18, XSPH incurs a significant extra step in the particle transport, halving the performance of the particle update. As a poor man's XSPH, we suggest a momentum correction following the lines of XSPH, but correcting with the neighbour average computed as the centroid momentum of each adjacent cell and weighted with the distance from the particle to the center of each aforesaid cell using the smoothing kernel. This will dramatically reduce the number of computations in the XSPH update, while still preserving the e. The XSPH method is somewhat ad-hoc, and for this reason we believe it is not of the full neighbour particle treatment if avoidable.

Another possible extension of this work is the coupling to a fast ultrasound simulator such as COLE, allowing realtime simulation of blood flow images. Fast COLE simulators using CUDA are available for this purpose.

In the end, the overall method planned for this project bears a strong resemblance to the FLIP method, and investigations should be done to see if the FLIP method is an even better fit.

Although 3D is an emerging technology in ultrasound, the low framerate and technological challenges surrounding volume acquisition still limits its current clinical use. 2D will still remain the *de facto* standard in ultrasound for , and 2D It is therefore beneficial to further look at how the filter can be downscaled to perform in the 2D+t modality where we hypothesise that the reduced degrees of freedom will allow a faster and more stable realtime filtering suitable for bedside evaluation on current scanner hardware.

# 6 | Conclusion

A fast SPH fluid system with grid interpolation methods was developed and evaluated. In a direct comparison with an accurate CFD model, the fluid system managed to forecast the flow phantom with reasonable accuracy.

Many of the model components were chosen as to readily model cardiac conditions from available estimators for this application.

The SPH model showed capability of regularising noisy estimates in a way that retains the overall flow details and magnitudes to a greater degree than conventional methods. Further, the model demonstrated a predictive power that could be viable in a Kalman filtering context, provided that the absolute domain flux over the prediction period is limited. We hypothesise that this limitation can be overcome with improved inflow/outflow handling in our model either by a reseeding approach.

Although accurate 3D realtime application of the model is not feasible at the current stage, we believe that the current performance should handle realtime 2D simulations with relative ease given current scanner hardware.

Given the similarity in performance found when compared to an advanced grid based smoother, we hypothesise that the coupling of this model with a Kalman filter will cause it to outperform many of the existing methods available for blood flow regularisation in echocardiography.

# Bibliography


[ASN+14] Paolo Angelelli, Sten Roar Snare, Siri Ann Nyrnes, Stefan Bruckner, Helwig Hauser, and Lasse Løvstakken. Live ultrasound-based particle visualization of blood flow in the heart. In *Proceedings of the 30th Spring Conference on Computer Graphics*, pages 13–20. ACM, 2014.

[Bal95] Dinshaw S Balsara. Von neumann stability analysis of smoothed particle hydrodynamics—suggestions for optimal algorithms. *Journal of Computational Physics*, 121(2):357–372, 1995.

[Bat00] George Keith Batchelor. *An introduction to fluid dynamics*. Cambridge university press, 2000.

[BK02] Javier Bonet and Sivakumar Kulasegaram. A simplified approach to enhance the performance of smooth particle hydrodynamics methods. *Applied Mathematics and Computation*, 126(2):133–155, 2002.

[BKR87] JU Brackbill, DB Kothe, and HM Ruppel. Flip (fluid-implicit-particle): A low-dissipation, particle-in-cell method for fluid flow. Technical report, Los Alamos National Lab., NM (USA), 1987.

[BL99] J Bonet and T-SL Lok. Variational and momentum preservation aspects of smooth particle hydrodynamic formulations. *Computer Methods in applied mechanics and engineering*, 180(1):97–115, 1999.

[BT07] Markus Becker and Matthias Teschner. Weakly compressible sph for free surface flows. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 209–217. Eurographics Association, 2007.

[BTT09] Markus Becker, Hendrik Tessendorf, and Matthias Teschner. Direct forcing for lagrangian rigid-fluid coupling. *IEEE Transactions on Visualization and Computer Graphics*, 15(3):493–503, 2009.

[CALT09] Andrea Colagrossi, Matteo Antuono, and David Le Touzé. Theoretical considerations on the free-surface role in the smoothed-particle-hydrodynamics model. *Physical Review E*, 79(5):056701, 2009.

[CEL06] Fabrice Colin, Richard Egli, and Feng Ying Lin. Computing a null divergence velocity field using smoothed particle hydrodynamics. *Journal of Computational Physics*, 217(2):680–692, 2006.

[DB78] Carl De Boor. *A practical guide to splines*, volume 27. Springer-Verlag New York, 1978.

[DG86] Richard H Durisen and Robert A Gingold. Numerical simulations of fission. In *Origin of the Moon*, page 487, 1986.

[DGB+16] Edoardo Daly, Stefania Grimaldi, Ha Hong Bui, et al. Explicit incompressible sph algorithm for free-surface flow modelling: A comparison with weakly compressible schemes. *Advances in Water Resources*, 97:156–167, 2016.

[Eil03] Paul HC Eilers. A perfect smoother. *Analytical chemistry*, 75(14):3631–3636, 2003.

[Gar10] Damien Garcia. Robust smoothing of gridded data in one and higher dimensions with missing values. *Computational statistics & data analysis*, 54(4):1167–1178, 2010.

[GdVJ+15] Alberto Gomez, Adelaide de Vecchi, Martin Jantsch, Wenzhe Shi, Kuberan Pushparajah, John M Simpson, Nicolas P Smith, Daniel Rueckert, Tobias Schaeffter, and Graeme P Penney. 4d blood flow reconstruction over the entire ventricle from wall motion and blood velocity derived from ultrasound data. *IEEE transactions on medical imaging*, 34(11):2298–2308, 2015.

[GGRDC10] Moncho Gomez-Gesteira, Benedict D Rogers, Robert A Dalrymple, and Alex JC Crespo. State-of-the-art of classical sph for free-surface flows. *Journal of Hydraulic Research*, 48(S1):6–27, 2010.

[GM77] Robert A Gingold and Joseph J Monaghan. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Monthly notices of the royal astronomical society*, 181(3):375–389, 1977.

[GM78] Robert A Gingold and Joseph J Monaghan. Binary fission in damped rotating polytropes. *Monthly Notices of the Royal Astronomical Society*, 184(3):481–499, 1978.

[GM79] RA Gingold and JJ Monaghan. Binary fission in damped rotating polytropes–ii. *Monthly Notices of the Royal Astronomical Society*, 188(1):39–44, 1979.

[Gom13] Alberto Gomez. *Full 3D Blood Velocity Mapping and Flow Quantification from Doppler Echocardiographic Images*. PhD thesis, King's College London, 2013.

[GP11] Prashant Goswami and Renato Pajarola. Time adaptive approximate sph. 2011.

[GPS+13] Alberto Gomez, Kuberan Pushparajah, John M Simpson, Daniel Giese, Tobias Schaeffter, and Graeme Penney. A sensitivity analysis on 3d velocity reconstruction from multiple registered echo doppler views. *Medical image analysis*, 17(6):616–631, 2013.

[Gr6] Thomas Grønli. Applications of penalized b-spline grids in ultrasound blood flow imaging. Unpublished project thesis, NTNU, 2016.

[GSN+16] Thomas Grønli, Erik Smistad, Siri Ann Nyrnes, Alberto Gomez, and Lasse Lovstakken. Reconstruction of in vivo flow velocity fields based on a rapid ultrasound image segmentation and b-spline regularization framework. In *Ultrasonics Symposium (IUS), 2016 IEEE International*, pages 1–4. IEEE, 2016.

[GSSP10] Prashant Goswami, Philipp Schlegel, Barbara Solenthaler, and Renato Pajarola. Interactive sph simulation and rendering on the gpu. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 55–64. Eurographics Association, 2010.

[HCM06] Kyle Hegeman, Nathan A Carr, and Gavin SP Miller. Particle-based fluid simulation on the gpu. In *International Conference on Computational Science*, pages 228–235. Springer, 2006.

[HG10] Jason P Hughes and David I Graham. Comparison of incompressible and weakly-compressible sph models for free-surface water flows. *Journal of Hydraulic Research*, 48(S1):105–117, 2010.

[HKK07] Takahiro Harada, Seiichi Koshizuka, and Yoichiro Kawaguchi. Sliced data structure for particle-based simulations on gpus. In *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia*, pages 55–62. ACM, 2007.

[HL97] Xiaoyi He and Li-Shi Luo. Theory of the lattice boltzmann method: From the boltzmann equation to the lattice boltzmann equation. *Physical Review E*, 56(6):6811, 1997.

[HLW+03] Ernst Hairer, Christian Lubich, Gerhard Wanner, et al. Geometric numerical integration illustrated by the stormer-verlet method. *Acta numerica*, 12(12):399–450, 2003.

[HW65] Francis H Harlow and J Eddie Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *The physics of fluids*, 8(12):2182–2189, 1965.

[HWLC12] He Huang, Liqiang Wang, En-Jui Lee, and Po Chen. An mpi-cuda implementation and optimization for parallel sparse equations and least squares (lsqr). *Procedia Computer Science*, 9:76–85, 2012.

[HWN+16] Jakob Høgenes, Morten Wigen, Siri Ann Nyrnes, Patrick Segers, Abigail Swillens, and Lasse Lovstakken. Model-based estimation of intra-cardiac blood flow velocities using an unscented kalman filter. In *Ultrasonics Symposium (IUS), 2016 IEEE International*, pages 1–4. IEEE, 2016.

[ICS+14] Markus Ihmsen, Jens Cornelis, Barbara Solenthaler, Christopher Horvath, and Matthias Teschner. Implicit incompressible sph. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):426–435, 2014.

[IR06] Violeau D Issa R. 3d schematic dam break and evolution of the free surface, 2006. [Online; accessed 3-March-2017].

[JNAG16a] Jørgen Arendt Jensen, Svetoslav Ivanov Nikolov, CH Alfred, and Damien Garcia. Ultrasound vector flow imaging—part i: Sequential systems. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, 63(11):1704–1721, 2016.

[JNAG16b] Jørgen Arendt Jensen, Svetoslav Ivanov Nikolov, CH Alfred, and Damien Garcia. Ultrasound vector flow imaging—part ii: Parallel

systems. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, 63(11):1722–1732, 2016.

[KPL+12] Andreas Klöckner, Nicolas Pinto, Yunsup Lee, B. Catanzaro, Paul Ivanov, and Ahmed Fasih. PyCUDA and PyOpenCL: A Scripting-Based Approach to GPU Run-Time Code Generation. *Parallel Computing*, 38(3):157–174, 2012.

[LMX+08] E-S Lee, Charles Moulinec, Rui Xu, Damien Violeau, Dominique Laurence, and Peter Stansby. Comparisons of weakly compressible and truly incompressible algorithms for the sph mesh free particle method. *Journal of computational physics*, 227(18):8417–8436, 2008.

[LP96] Ben Leimkuhler and George W Patrick. A symplectic integrator for riemannian manifolds. *Journal of Nonlinear Science*, 6(4):367–384, 1996.

[Luc77] Leon B Lucy. A numerical approach to the testing of the fission hypothesis. *The astronomical journal*, 82:1013–1024, 1977.

[MNW+09] Francesco Maffessanti, Hans-Joachim Nesser, Lynn Weinert, Regina Steringer-Mascherbauer, Johannes Niel, Willem Gorissen, Lissa Sugeng, Roberto M Lang, and Victor Mor-Avi. Quantitative evaluation of regional left ventricular function using three-dimensional speckle tracking echocardiography in patients with and without heart disease. *The American journal of cardiology*, 104(12):1755–1762, 2009.

[Mon92] Joe J Monaghan. Smoothed particle hydrodynamics. *Annual review of astronomy and astrophysics*, 30(1):543–574, 1992.

[Mon94] Joe J Monaghan. Simulating free surface flows with sph. *Journal of computational physics*, 110(2):399–406, 1994.

[Mon00] Joseph J Monaghan. Sph without a tensile instability. *Journal of Computational Physics*, 159(2):290–311, 2000.

[Mon05] Joe J Monaghan. Smoothed particle hydrodynamics. *Reports on progress in physics*, 68(8):1703, 2005.

[MPN+11] Shanthi Mendis, Pekka Puska, Bo Norrving, et al. *Global atlas on cardiovascular disease prevention and control.* World Health Organization, 2011.

[MQR99] Robert I McLachlan, GRW Quispel, and Nicolas Robidoux. Geometric integration using discrete gradients. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 357(1754):1021–1045, 1999.

[OYG14] M Ozbulut, M Yildiz, and O Goren. A numerical investigation into the correction algorithms for sph method in modeling violent free surface flows. *International Journal of Mechanical Sciences*, 79:56–65, 2014.

[PLCAT14] Gianni Pedrizzetti, Giovanni La Canna, Ottavio Alfieri, and Giovanni Tonti. The vortex [mdash] an early predictor of cardiovascular outcome? *Nature Reviews Cardiology*, 11(9):545–553, 2014.

[PS82] Christopher C Paige and Michael A Saunders. Lsqr: An algorithm for sparse linear equations and sparse least squares. *ACM transactions on Mathematical Software*, 8(1):43–71, 1982.

[PS99] Michael K Pitt and Neil Shephard. Filtering via simulation: Auxiliary particle filters. *Journal of the American statistical association*, 94(446):590–599, 1999.

[RBG+12] Eugenio Rustico, Giuseppe Bilotta, G Gallo, Alexis Herault, C Del Negro, and Robert Anthony Dalrymple. A journey from single-gpu to optimized multi-gpu sph with cuda. In *7th SPHERIC Workshop*, 2012.

[SB12] Hagit Schechter and Robert Bridson. Ghost sph for animating water. *ACM Transactions on Graphics (TOG)*, 31(4):61, 2012.

[SDSD+12] Abigail Swillens, Gianluca De Santis, Joris Degroote, Lasse Lovstakken, Jan Vierendeels, and Patrick Segers. Accuracy of carotid strain estimates from ultrasonic wall tracking: a study based on multiphysics simulations and in vivo data. *IEEE transactions on medical imaging*, 31(1):131–139, 2012.

[SLK+09] Abigail Swillens, Lasse Løvstakken, Jan Kips, Hans Torp, and Patrick Segers. Ultrasound simulation of complex flow velocity fields based on computational fluid dynamics. *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, 56(3), 2009.

[SP09] Barbara Solenthaler and Renato Pajarola. Predictive-corrective incompressible sph. In *ACM transactions on graphics (TOG)*, volume 28, page 40. ACM, 2009.

[SPK+12] Partho P Sengupta, Gianni Pedrizzetti, Philip J Kilner, Arash Kheradvar, Tino Ebbers, Giovanni Tonti, Alan G Fraser, and Jagat Narula. Emerging trends in cv flow visualization. *JACC: Cardiovascular Imaging*, 5(3):305–316, 2012.

[TSGL99] A Tura, A Sarti, T Gaens, and C Lamberti. Regularization of blood motion fields by modified navier–stokes equations. *Medical engineering & physics*, 21(1):27–36, 1999.

[VBC08] G Viccione, V Bovolin, and E Pugliese Carratelli. Defining and optimizing algorithms for neighbouring particle identification in sph fluid simulations. *International Journal for Numerical Methods in Fluids*, 58(6):625–638, 2008.

[VC14] Joris Van Cauwenberge. Developing multidimensional ultrasonic blood flow visualisation techniques for newborns with cardiac pathologies, using multi-physics models. Master's thesis, Ghent University, 2014.

[WBTW95] Anthony Peter Whitworth, AS Bhattal, JA Turner, and SJ Watkins. Estimating density in smoothed particle hydrodynamics. *Astronomy and Astrophysics*, 301:929, 1995.

[Whi10] Frank M White. Fluid mechanics. edition, 2010.

[WL16] Morten Wigen and Lasse Løvstakken. In vivo three-dimensional intra-cardiac vector flow imaging using a 2d matrix array transducer. In *Ultrasonics Symposium (IUS), 2016 IEEE International*, pages 1–4. IEEE, 2016.

[WSL+07] Ulrik Wisløff, Asbjørn Støylen, Jan P Loennechen, Morten Bruvold, Øivind Rognmo, Per Magnus Haram, Arnt Erik Tjønna, Jan Helgerud, Stig A Slørdahl, Sang Jun Lee, et al. Superior cardiovascular effect of aerobic interval training versus moderate continuous training in heart failure patients. *Circulation*, 115(24):3086–3094, 2007.

[ZHWG08] Kun Zhou, Qiming Hou, Rui Wang, and Baining Guo. Real-time kd-tree construction on graphics hardware. *ACM Transactions on Graphics (TOG)*, 27(5):126, 2008.