



# Analysis and implementation of the Material Point Method (MPM) on fluid-structure interaction (FSI) using the explicit Velocity-Verlet temporal scheme in C++/Python environment

by Karthik Vigneshwaran Muthukumar

type of report:  
Master Thesis

<b>Report - Preprint</b>	<b>No. 187</b>
--------------------------	----------------

20. April 2021

Editor: Prof. Dr.-Ing. habil. Jörg Schröder

# **Analysis and implementation of the Material Point Method (MPM) on fluid-structure interaction (FSI) using the explicit Velocity-Verlet temporal scheme in C++/Python environment**

Karthik Vigneshwaran Muthukumar  
Matr.-Nr.: 3087014

## **Supervisor:**

Prof. Dr.-Ing. habil. Jörg Schröder  
Sascha Maassen M.Sc.

Institut für Mechanik  
Fakultät für Ingenieurwissenschaften, Abteilung Bauwissenschaften  
Universität Duisburg-Essen, Campus Essen

Universitätsstr. 15  
45141 Essen  
Germany

*Master Thesis*

April 20, 2021

In die gebundenen Exemplare der Abschlussarbeit einzubinden und nach Fertigstellung der gebundenen Exemplare von der / dem Studierenden zu unterschreiben:

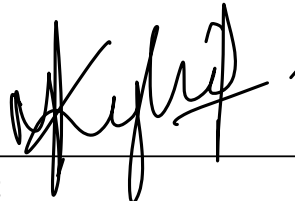
## Versicherung an Eides Statt

Ich versichere an Eides statt durch meine untenstehende Unterschrift,

- dass ich die vorliegende Arbeit - mit Ausnahme der Anleitung durch die Betreuer - selbstständig ohne fremde Hilfe angefertigt habe und
- dass ich alle Stellen, die wörtlich oder annähernd wörtlich aus fremden Quellen entnommen sind, entsprechend als Zitate gekennzeichnet habe und
- dass ich ausschließlich die angegebenen Quellen (Literatur, Internetseiten, sonstige Hilfsmittel) verwendet habe und
- dass ich alle entsprechenden Angaben nach bestem Wissen und Gewissen vorgenommen habe, dass sie der Wahrheit entsprechen und dass ich nichts verschwiegen habe.

Mir ist bekannt, dass eine falsche Versicherung an Eides Statt nach § 156 und nach § 163 Abs. 1 des Strafgesetzbuches mit Freiheitsstrafe oder Geldstrafe bestraft wird.

ESSEN, 20.04.2021  
Ort, Datum

  
Unterschrift



# Contents

<b>1</b>	<b>Abstract</b>	<b>1</b>
<b>2</b>	<b>Zusammenfassung</b>	<b>2</b>
<b>3</b>	<b>Introduction</b>	<b>3</b>
<b>4</b>	<b>Continuum Mechanics</b>	<b>6</b>
4.1	Kinematics . . . . .	6
4.1.1	Motion, deformation and deformation velocities . . . . .	6
4.1.2	Transport theorems . . . . .	8
4.1.3	Strain tensors . . . . .	9
4.2	Concept of stresses . . . . .	10
4.3	Conservation laws . . . . .	11
4.3.1	Balance of mass . . . . .	11
4.3.2	Balance of momentum . . . . .	13
4.3.3	Balance of moment of momentum . . . . .	14
<b>5</b>	<b>Constitutive models</b>	<b>17</b>
5.1	Solid mechanics . . . . .	17
5.2	Fluid mechanics . . . . .	19
<b>6</b>	<b>Galerkin weak form of balance of momentum</b>	<b>20</b>
<b>7</b>	<b>The Material Point Method</b>	<b>21</b>
7.1	General overview of the method . . . . .	21
7.2	Structured 2D grid . . . . .	21
7.3	Discretisation of the weak form balance of momentum via MPM . . . . .	23
7.4	Implementation and solution . . . . .	24
7.4.1	USL algorithm and workflow . . . . .	27
7.5	UML diagram . . . . .	29
7.6	C++ implementation of the MPM . . . . .	30
7.6.1	Containers . . . . .	30
7.6.2	MPM_Body . . . . .	31
7.6.3	MPM_Grid . . . . .	33

---

7.6.4	MPM_Solver . . . . .	34
<b>8</b>	<b>Numerical examples</b>	<b>35</b>
8.1	1D Spring verification . . . . .	35
8.2	2D Truss bench-marking . . . . .	37
8.2.1	Effect of St. Venant-Kirchoff material law . . . . .	37
8.2.2	Convergence study . . . . .	37
8.2.3	Observation and inference . . . . .	39
8.3	Two disks bench-marking . . . . .	41
8.3.1	Contact and locus of the disks . . . . .	41
8.3.2	Energies of the disks . . . . .	42
8.3.3	Momentum of the disks . . . . .	44
8.3.4	Von-Misses stresses . . . . .	45
8.4	Dam with barrier bench-marking . . . . .	47
8.4.1	Non-penetration boundary condition for fluids . . . . .	47
8.4.2	Effect of viscosity . . . . .	48
8.5	Impact of fluids bench-marking . . . . .	51
8.6	Gravity currents bench-marking . . . . .	56
8.7	Float - a FSI example . . . . .	58
<b>9</b>	<b>Conclusion</b>	<b>61</b>

## **1 Abstract**

The Material Point Method is a numerical technique, which uses both the Lagrangian and Eulerian framework to analyse and study the deformation and stresses in solids and fluids. The primary advantage is that it is free from mesh distortion and contacts are natural without any additional computational cost. The explicit Velocity-Verlet temporal scheme is used in this thesis to study problems involving large and rapid deformations. This thesis aims at exploring the tools' abilities to handle large deformations, solid-solid, fluid-fluid contact and fluid-structure interaction. The Update Stress Last (USL) computation algorithm is implemented in a C++ environment, and an Application Programming Interface (API) is developed in Python. Python is preferred because of its user-friendly environment for post-processing results. The code's implementation is briefed, describing the available functions and storage containers. Along with the Unified Modelling Language (UML) diagram, the code's architecture is also explained. The implementation is verified by running numerical examples referred from published literature and bench-marking the results. A self-developed problem representing fluid-structure interaction (FSI) is studied. Conclusively, the advantages and disadvantages are discussed, and further suggestions to improvise the method are presented.

## 2 Zusammenfassung

Die Material Point Method ist ein numerisches Verfahren, das zur Analyse und Untersuchung der Verformung und Spannungen in Festkörpern und Flüssigkeiten die Vorteile von Lagranger und Eulerscher Betrachtung vereint einsetzt. Der Hauptvorteil ist, dass sie frei von Netzverzerrungen ist und Kontakt ohne zusätzliche Rechenkosten darstellt. Ein explizites temporales Velocity-Verlet Schema wird in dieser Arbeit verwendet, um Probleme mit großen und schlagartigen Verformungen zu untersuchen. Ziel der Arbeit ist es, die Fähigkeiten der Methode zur Behandlung von großen Verformungen, Festkörper-Festkörper-Kontakt, Fluid-Fluid-Kontakt und Fluid-Festkörper-Interaktion zu untersuchen. Der „Update Stress Last“(USL) Algorithmus wird in einer C++-Umgebung implementiert, und eine Anwendungsprogrammierschnittstelle wird in Python entwickelt. Python wird wegen seiner benutzerfreundlichen Umgebung für die Nachbearbeitung der Ergebnisse gewählt. Die Implementierung des Codes wird kurz umrissen, wobei die verfügbaren Funktionen und Speichercontainer beschrieben werden. Zusammen mit dem „Unified Modelling Language“(UML) Diagramm wird auch die Architektur des Codes erläutert. Die Implementierung wird verifiziert, indem numerische Beispiele aus der veröffentlichten Literatur ausgeführt und die Ergebnisse verglichen werden. Ein selbst entwickeltes Problem, das die Fluid-Struktur Interaktion darstellt, wird ebenfalls untersucht. Abschließend werden die Vor- und Nachteile diskutiert und weitere Vorschläge zur Verbesserung der Methode vorgestellt.



### 3 Introduction

Computational engineering plays a vital role and has become significant in many application. Therefore, computational simulations are used in many sectors, including engineering, medicine and other noble disciplines. It helps in obtaining complex and problem-specific analytical solutions. Simulations can be run prior to manufacturing, thus reducing the risk of failure. It also replaces extreme and complicated real-time experiments like explosions, multi-phase simulations, penetrations, and so. The majority of the presently available simulation techniques are based on Finite Element Methods, Finite Difference Methods and Finite Volume Methods. The difference lies in the type of spatial discretisation. Discretising using geometries, finite differences, or volume defines Finite Element Methods, Finite Difference Methods and Finite Volume Methods.

Generally, the available spatial discretisation methods can be broadly classified into Lagrangian, Eulerian or hybrid method based on deformation and motion description. In the Lagrangian framework, the computational grid is appended to the body, which results in mesh deformation whenever the body is deformed. There is no convection of material parameters like mass, momentum and energy between the mesh and the body, reducing the computational cost. On the contrary, the mesh is fixed and independent of the body's deformation, and the material moves through in the Eulerian framework. The material parameters convect between the grid and the body hence introducing a convection term in equations. There are also hybrid methods available that incorporate both the Lagrangian and Eulerian frameworks, and one such method is Material Point Method.

Material Point Method herein referred to as MPM is a mesh-based particle method that implements Eulerian and Lagrangian description to define a physical domain. The body of interest is discretised into Lagrangian elements called material points and is bounded by an Eulerian frame called the computational grid. Before particle methods, Lagrangian and Eulerian descriptions were used individually. The disadvantage of such a description is that it suffers from mesh distortion and has issues handling problems with contacts. The well-known Finite Element Method (FEM) is an example that uses a Lagrangian description. Over the past years, MPM has developed rapidly, and currently, several MPM variations are available.

The idea of particle methods originated in the 1950s, and one of the first developed technique was Particle in Cell method (PIC) by Harlow [1964]. In PIC, the material points hold only the position and the mass of the body. Because of which it experienced high numerical dissipation. As a variation of PIC, Harlow and Welch [1965] developed Marker and cell method (MAC), see Mckee et al. [2008], Johnson [1996] to treat incompressible and free surface flows. Brackbill and Ruppel [1986] overcome the numerical dissipation in PIC with the introduction of the Fluid Implicit Particle method (FLIP). Unlike PIC, FLIP held all physical parameters of the body such as mass, volume, stress, density, momentum, and energy. But both PIC and FLIP were developed in the interests of Fluid mechanics. It was Sulsky in Sulsky et al. [1994] extended FLIP to handle solid mechanics and developed MPM.

In MPM, all material parameters and internal state variables required for the respective constitutive model are associated with material points. As the mass of the body is

associated with each material point, conservation of mass is satisfied. The mass of the material points is always constant, but the density and associated volume of the material points change accordingly, thus conserving the total mass. Also, the material points use a single-valued velocity in the algorithm, i.e. no-slip and no-penetration contact is integrated into the method and is essential in Sulsky et al. [1994]. Ideally, there is no direct particle-particle interaction in MPM, but an interaction between the material points and the grid exists. The weak form of balance of momentum is solved on the grid, and the state of each material point is updated. The advantage of an Eulerian grid is that it is free from mesh distortions because the grid is reset at every step. The mesh can also be reset at any  $n^{th}$  time step or never reset at all, see Guilkey et al. [2006].

The generic MPM algorithm can be summarised into four steps: mapping material points onto the grid, solving on the grid, remapping the grid to material points and resetting the grid. Firstly, physical parameters of each material point, such as mass, volume, velocity, force, is mapped onto the respective grid nodes. Then the parameters are transformed into nodal values using shape functions, also called grid-basis functions. The transformation subsequently assembles the local nodal values into a global vector and yields the global mass and stiffness matrix. The momentum equations are solved using the global matrices for the next time step using a suitable time integration scheme. The obtained nodal values are then remapped to the material points with which the position of the material points is updated, alongside its material state.

MPM can be broadly classified into a few variants based on the defined shape functions, type of the grid and the temporal scheme. The standard or conventional MPM suffers from cell crossing instability or cell crossing noise that causes material points to escape the computational grid. The reason for the instability is because the conventional MPM implements linear hat functions as grid-basis functions and are only  $C^0$  continuous. As an improvisation, in Generalised Interpolation Material Point (GIMP) in Bardebhagen and Kober [2004], uses  $C^1$  smooth functions resembling splines, see Steffen et al. [2008]. MPM also suffers from numerical fracture, which is a non-physical separation of the body. The distance between the material points becomes so large that the interaction between them seizes. This distance depends on the used grid-basis function. Instead of points as in conventional MPM, finite particle domains are used for element description in GIMP. However, there is still a gap between particle domains. Hence it also experiences numerical fracture and shows lower accuracy in arbitrary loading cases. This behaviour is overcome in Convected Particle Domain Interpolation (CPDI) as in Sadeghirad et al. [2011] and Sadeghirad et al. [2013]. CPDI uses e.g. quadrilateral elements in 2D and e.g. tetrahedral elements in 3D as particle domains, and the shape functions are constructed by interpolation over the nodes of the particle domain. This approach makes CPDI more suitable for problems involving large deformations and tensile loading. But the downside of CPDI is it encounters mesh distortion because it uses quadrilateral and tetrahedral elements to generate mesh as in FEM, which goes against the significant advantage of MPM. There is also an MPM technique that implements B-splines as grid-basis functions called BSMPM, see Stomakhin et al. [2014], Tielen et al. [2017] and Gan et al. [2018].

The temporal schemes can be explicit or implicit. Explicit schemes are of lower computational cost and less stable. The stability depends highly on the time increment and is more stable with smaller time steps. Commonly used explicit schemes are the explicit

Euler and the Velocity-Verlet time integration schemes. Few notable explicit temporal schemes in MPM are Lian et al. [2012], Zhang et al. [2016] and Bardebhagen and Kober [2004]. In contrast, implicit schemes are very stable, and this stability comes with a computational cost. In general, implicit schemes require finding the inverse of matrices and pre-treatment of global matrices to find the solution. Methods such as the implicit Euler scheme and the Crank-Nicolson scheme discretises time implicitly. Examples of implicit schemes in MPM are Cummins and Brackbill [2002], Sulsky and Kaul [2004] and Wang et al. [2016].

The defined computational domain can be a Cartesian or unstructured grid. Interaction of a material point with the corresponding grid is easy and cheap in structured grids because of the simplicity of locating a material point in the computational domain. Hence, structured grids are primarily used. But the advantage of unstructured over the structured grid is its ability to apply complex boundary conditions. Consequently, unstructured grids are more expensive. Unstructured grids are commonly used in Geo-Engineering, see Wiećkowski [2004], Wiećkowski et al. [1999], Beuth et al. [2011] and Jassim et al. [2013].

Because MPM is free from mesh distortions, its application is used to analyse machining processes like extrusion, cold forming, drawing, and so. It is also used in problems that involve large deformation, large strains, penetration and crack propagation. With no-slip and no-penetration contact being an integral part of MPM, it is also used in the problems involving contact and impact. The application of MPM is also extended to fluid mechanics, so to solve problems like fluid-structure interaction. The fluid solver for MPM is approached in two ways, a weakly compressible MPM and an incompressible MPM. The former implements explicit time schemes and uses e.g. an artificial equation of state to solve for the pressure state. This method was first introduced by York et al. [1999]. The same method is used to solve fluid flow problems in Li et al. [2014] and Mast et al. [2012] and gas dynamics in York et al. [2000], Tran et al. [2010] and Ma et al. [2009]. Whereas, the incompressible method is presented in Stomakhin et al. [2014].

## 4 Continuum Mechanics

Continuum mechanics is the study of mechanics of solids and fluids modelled as a continuum mass. Continuum mechanics assumes the body to be homogenous on a macroscopic scale despite its discontinuities in the atomic and molecular levels to define the behaviour of solids and fluids as smooth functions of spatial variables. The microscopic study of bodies is helpful in other aspects, but the macroscopic scale is sufficient for engineering problems. It primarily focuses on kinematics, concept of stresses and laws of conservation. For a clear cut of continuum mechanics, please refer to Holzapfel [2000] and Haupt [2013], and Bluhm [2015] for a detailed derivation of the appearing tensor quantities in this thesis and balance laws.

### 4.1 Kinematics

Kinematics is the study of motion and deformation of a body, and it does not consider the external forces that cause the deformation or motion.

**4.1.1 Motion, deformation and deformation velocities** A continuum body  $\mathcal{B}$  comprises a set of an infinite number of material points or particles associated with each other by material law and bounded by the surface of the body  $\partial\mathcal{B}$ , see figure 1. The setup renders the standard setup as in Bluhm [2015], Holzapfel [2000] and Haupt [2013]. Any arbitrary body can be positioned in the Euclidean space  $\mathbb{E}^3$  by a set of coordinates with respect to the spatial origin.

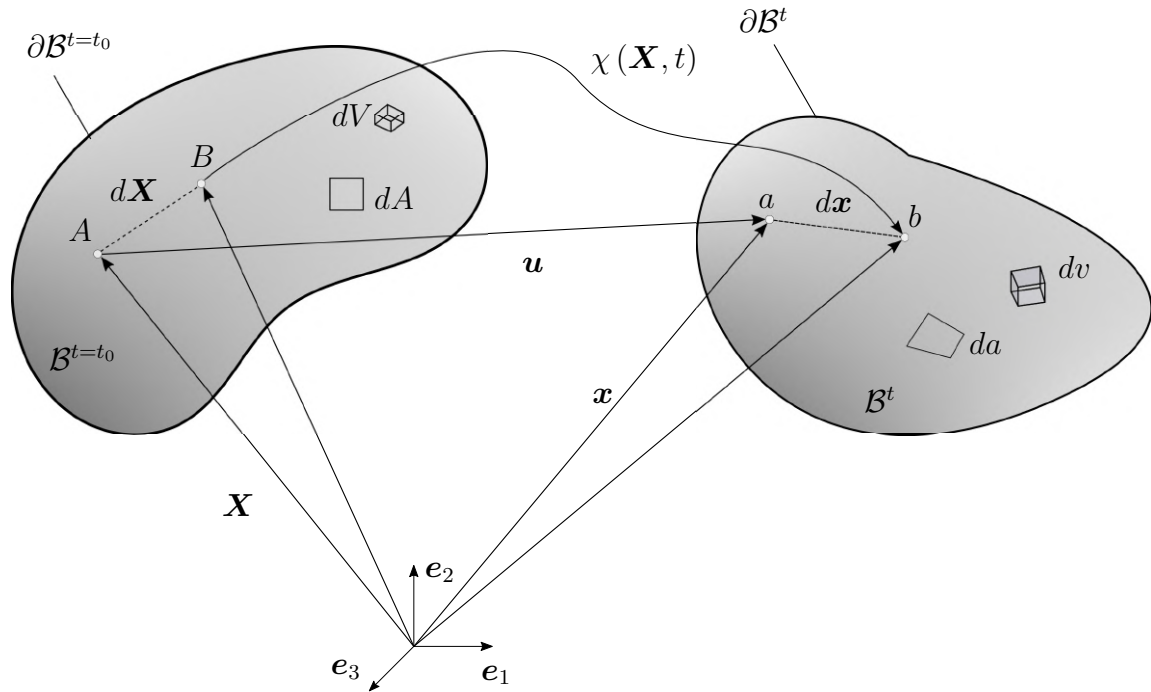


Figure 1: Configurations of Bodies

The time variable defines the configuration of a body. The initial or reference configuration of a body is at time  $t = 0$ , and the current configuration is at an arbitrary time  $t$ .  $\mathbf{X}$  represents the position vector of an arbitrary material point in the reference configuration and is the material or Lagrangian coordinates. The below function does the transformation from Lagrangian coordinate to spatial or Eulerian coordinate

$$\mathbf{x} = \chi(\mathbf{X}, t), \quad (1)$$

where  $\mathbf{x}$  is the position vector of the displaced material point in the current configuration. This Lagrangian motion description is a one-to-one function and is also continuous and continuously differentiable. The displacement of the material point  $\mathbf{x}$  at any time instant can be found using the function. From which the velocity  $\dot{\mathbf{x}}$  and acceleration  $\ddot{\mathbf{x}}$  are defined

$$\begin{aligned} \dot{\mathbf{x}} &= \frac{\partial \chi(\mathbf{X}, t)}{\partial t} = \frac{\partial \mathbf{x}}{\partial t}, \\ \ddot{\mathbf{x}} &= \frac{\partial^2 \chi(\mathbf{X}, t)}{\partial t^2} = \frac{\partial^2 \mathbf{x}}{\partial t^2}. \end{aligned} \quad (2)$$

Also, the Eulerian motion description can be expressed as the inverse mapping of the Lagrangian motion function,

$$\mathbf{X} = \chi^{-1}(\mathbf{x}, t). \quad (3)$$

The invertibility of the Lagrangian function is essential for the existence of a non-singular determinant  $J$  of the material deformation gradient  $\mathbf{F}$ . The deformation gradient gives the displacement of a material point in the current configuration with reference to its initial configuration.  $\mathbf{F}$  is defined as the gradient of the Lagrangian function with respect to the undeformed vector.

$$\begin{aligned} \mathbf{F} &= \frac{d\chi(\mathbf{X}, t)}{d\mathbf{X}} \\ &= \frac{d\mathbf{x}}{d\mathbf{X}} \\ &= \text{Grad } \mathbf{x}. \end{aligned} \quad (4)$$

The mapping function can be of any order. By introducing a displacement vector  $\mathbf{u}$ , the function can be approximated linearly. Hence, for an arbitrary material point  $A$ , in figure 1, it is written as

$$\mathbf{x} = \mathbf{X} + \mathbf{u}. \quad (5)$$

Applying equation (5) on the deformation gradient  $\mathbf{F}$ ,

$$\begin{aligned}\mathbf{F} &= \frac{d(\mathbf{X} + \mathbf{u})}{d\mathbf{X}} \\ &= \mathbf{I} + \frac{d\mathbf{u}}{d\mathbf{X}} \\ &= \mathbf{I} + \text{Grad } \mathbf{u} .\end{aligned}\tag{6}$$

The spatial deformation gradient  $\mathbf{F}^{-1}$  is then given as

$$\begin{aligned}\mathbf{F}^{-1} &= \frac{d[\chi^{-1}(\mathbf{x}, t)]}{d\mathbf{x}} \\ &= \frac{d\mathbf{X}}{d\mathbf{x}} \\ &= \text{grad } \mathbf{X} .\end{aligned}\tag{7}$$

Further, the material velocity gradient  $\dot{\mathbf{F}}$  and spatial velocity gradient  $\mathbf{L}$  are

$$\begin{aligned}\dot{\mathbf{F}} &= \frac{d\dot{\mathbf{x}}}{d\mathbf{X}} = \text{Grad } \dot{\mathbf{x}} , \\ \mathbf{L} &= \frac{d\dot{\mathbf{x}}}{d\mathbf{x}} = \text{grad } \dot{\mathbf{x}} .\end{aligned}\tag{8}$$

With additive decomposition  $\mathbf{L}$  is split into its symmetric part  $\mathbf{d}$  and skew symmetric part  $\mathbf{w}$

$$\mathbf{L} = \mathbf{d} + \mathbf{w} ,\tag{9}$$

with

$$\begin{aligned}\mathbf{d} &= \frac{1}{2} (\mathbf{L} + \mathbf{L}^T) = \mathbf{d}^T , \\ \mathbf{w} &= \frac{1}{2} (\mathbf{L} - \mathbf{L}^T) = -\mathbf{w}^T .\end{aligned}\tag{10}$$

**4.1.2 Transport theorems** The deformation gradient and its determinant are used to transport elements  $d\mathbf{x}$ ,  $d\mathbf{a}$  and  $dv$  from the reference  $d\mathbf{X}$ ,  $d\mathbf{A}$  and  $dV$  to the actual configuration and vice versa at any arbitrary time  $t$ . Where,  $d\mathbf{x}$ ,  $d\mathbf{X}$  are the infinitesimal length elements,  $d\mathbf{a}$ ,  $d\mathbf{A}$  are infinitesimal area elements and  $dv$ ,  $dV$  represents infinitesimal volume elements, refer figure 1.

$$\begin{aligned}
d\mathbf{x} &= \mathbf{F} \cdot d\mathbf{X} , \\
d\mathbf{a} &= d\mathbf{x}_1 \times d\mathbf{x}_2 \\
&= \mathbf{F} \cdot d\mathbf{X}_1 \times \mathbf{F} \cdot d\mathbf{X}_2 \\
&= (\det \mathbf{F}) \mathbf{F}^{-T} (d\mathbf{X}_1 \times d\mathbf{X}_2) = (\det \mathbf{F}) \mathbf{F}^{-T} \cdot d\mathbf{A} , \\
dv &= d\mathbf{x}_1 \cdot (d\mathbf{x}_2 \times d\mathbf{x}_3) \\
&= \mathbf{F} \cdot d\mathbf{X}_1 \cdot (\mathbf{F} \cdot d\mathbf{X}_2 \times \mathbf{F} \cdot d\mathbf{X}_3) \\
&= \det \mathbf{F} [d\mathbf{X}_1 \cdot (d\mathbf{X}_2 \times d\mathbf{X}_3)] = J dV .
\end{aligned} \tag{11}$$

**4.1.3 Strain tensors** The deformation gradient provides only the displacement value but not the ratio of magnitudes of displacement between two configurations. The strain tensor gives this value. Strains are of interest in engineering problems because of the application and relation with stresses. The strain tensor is introduced by the difference of the square of infinitesimal length elements  $d\mathbf{x}$  and  $d\mathbf{X}$ . Refer to figure 1,  $d\mathbf{x}$  and  $d\mathbf{X}$  are the length between two arbitrary material points in reference and current configuration. Also, using transport equation (11), the differences can be expressed in the reference configuration and current configuration. A similar illustration can be seen in Bluhm [2015].

$$\begin{aligned}
d\mathbf{x}^2 - d\mathbf{X}^2 &= d\mathbf{x} \cdot d\mathbf{x} - d\mathbf{X} \cdot d\mathbf{X} \\
&= (\mathbf{F} \cdot d\mathbf{X}) \cdot (\mathbf{F} \cdot d\mathbf{X}) - d\mathbf{X} \cdot d\mathbf{X} \\
&= d\mathbf{X} \cdot \mathbf{F}^T \cdot \mathbf{F} \cdot d\mathbf{X} - d\mathbf{X} \cdot \mathbf{I} \cdot d\mathbf{X} \\
&= 2 d\mathbf{X} \cdot \frac{1}{2} (\mathbf{F}^T \cdot \mathbf{F} - \mathbf{I}) \cdot d\mathbf{X} ,
\end{aligned} \tag{12}$$

$$\begin{aligned}
d\mathbf{x}^2 - d\mathbf{X}^2 &= d\mathbf{x} \cdot d\mathbf{x} - d\mathbf{X} \cdot d\mathbf{X} \\
&= d\mathbf{x} \cdot d\mathbf{x} - (\mathbf{F}^{-1} \cdot d\mathbf{x}) \cdot (\mathbf{F}^{-1} \cdot d\mathbf{x}) \\
&= d\mathbf{x} \cdot \mathbf{I} \cdot d\mathbf{x} - d\mathbf{x} \cdot \mathbf{F}^{-T} \cdot \mathbf{F}^{-1} \cdot d\mathbf{x} \\
&= 2 d\mathbf{x} \cdot \frac{1}{2} (\mathbf{I} - \mathbf{F}^{-T} \cdot \mathbf{F}^{-1}) \cdot d\mathbf{x} .
\end{aligned} \tag{13}$$

From equations (12), (13) we get the Cauchy-Green Strain tensor  $\mathbf{E}$  and Almansi strain tensor  $\mathbf{A}$ , along with the right and left Cauchy-Green deformation tensor  $\mathbf{C}$  and  $\mathbf{B}$  respectively.

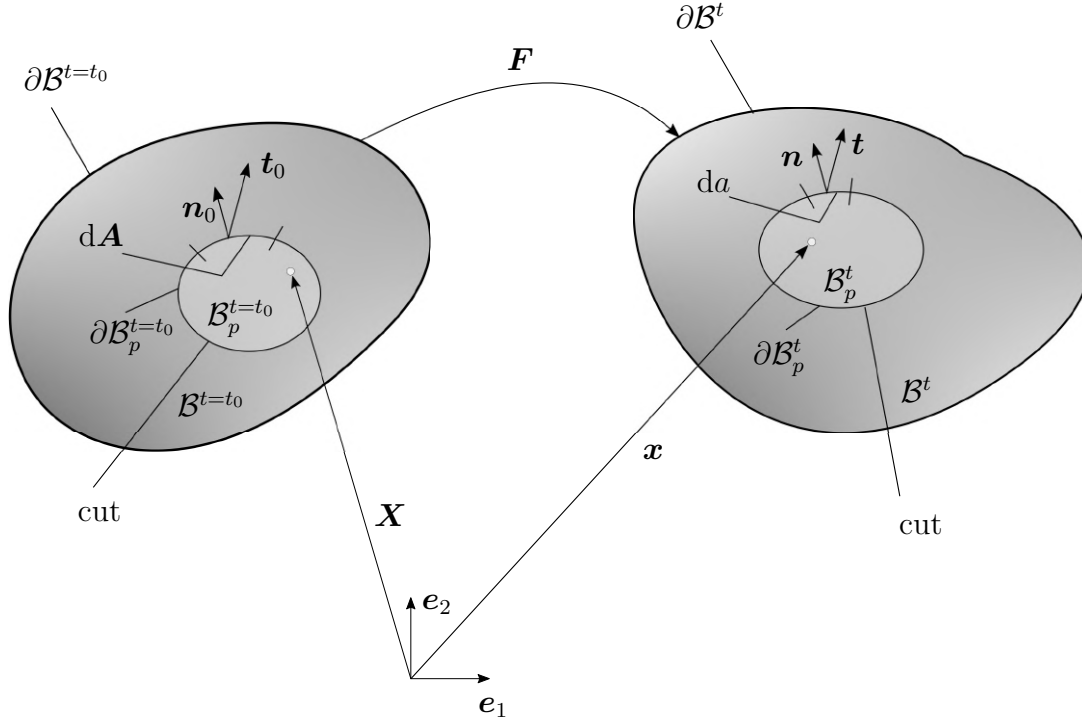
$$\begin{aligned}
\mathbf{E} &= \frac{1}{2} (\mathbf{F}^T \cdot \mathbf{F} - \mathbf{I}) & \mathbf{A} &= \frac{1}{2} (\mathbf{I} - \mathbf{F}^{-T} \cdot \mathbf{F}^{-1}) \\
&= \frac{1}{2} (\mathbf{C} - \mathbf{I}) , & &= \frac{1}{2} (\mathbf{I} - \mathbf{B}^{-1}) .
\end{aligned} \tag{14}$$

It is also noted that the deformation tensors are symmetrical

$$\begin{aligned} \Rightarrow \quad \mathbf{C} &= \mathbf{F}^T \cdot \mathbf{F} = \mathbf{C}^T, \\ \mathbf{B} &= \mathbf{F} \cdot \mathbf{F}^T = \mathbf{B}^T. \end{aligned} \quad (15)$$

## 4.2 Concept of stresses

Stresses are the internal reaction forces per unit area against the external forces. Stresses are generated whenever a body is deformed and are responsible for keeping the body intact. Hence, it is important to understand the behaviour and effect of stresses. The nature of stress can be compressive or tensile, depending on the force direction. Here stresses are introduced with the help of traction vector  $\mathbf{t}$ . Traction vector is the force per unit area on a cross-sectional surface. Consider a continuum body  $\mathcal{B}$  with surface  $\partial\mathcal{B}$  at time  $t_0$  in its reference configuration, see figure 2. The body is cut at a plane surface  $\partial\mathcal{B}_p$  passing through a point  $\mathbf{X}$  which introduces a traction force  $\mathbf{t}_0$ . At an arbitrary time  $t$  in its current configuration the body is deformed with a deformation gradient  $\mathbf{F}$ . The traction vector at this instance is  $\mathbf{t}$  with a surface normal vector  $\mathbf{n}$ . The traction vectors  $\mathbf{t}_0$ ,  $\mathbf{t}$  have different lengths but are in the same direction, and the normal vectors  $\mathbf{n}_0$ ,  $\mathbf{n}$  are of unit length perpendicular to the surfaces.



**Figure 2:** Concept of stresses



**Cauchy stresses** : The Cauchy stress tensor  $\boldsymbol{\sigma}$  is a symmetrical second order tensor and provides the stress value in the current configuration. It is generally referred to as true stresses that shows the actual stress state in a deformed body. The Cauchy's stress theorem relates  $\boldsymbol{\sigma}$  and the Cauchy traction vector  $\mathbf{t}$ ,

$$\mathbf{t} = \boldsymbol{\sigma} \cdot \mathbf{n} . \quad (16)$$

**First Piola-Kirchoff stress tensor** : Unlike Cauchy stresses the first Piola-Kirchoff stress tensor  $\mathbf{P}$  gives the stress state in the reference configuration, and is named as engineering or nominal stresses.  $\mathbf{P}$  is asymmetric and is a two-field tensor. Using Cauchy's stress theorem, the traction vector in reference configuration  $\mathbf{t}_0$  can be expressed in terms of  $\mathbf{P}$  and  $\mathbf{n}_0$

$$\mathbf{t}_0 = \mathbf{P} \cdot \mathbf{n}_0 . \quad (17)$$

Further the relation between the Cauchy stress and first Piola-Kirchoff stress is

$$\mathbf{P} = J \boldsymbol{\sigma} \cdot \mathbf{F}^{-T} . \quad (18)$$

**Second Piola-Kirchoff stress tensor** : The second Piola-Kirchoff stress tensor  $\mathbf{S}$  has no physical interpretation, but because of its symmetry, it is used in the formulation of constitutive material laws. The calculated stress fields are completely in the reference configuration. Its relation with other stress tensors are given as below

$$\mathbf{S} = J \mathbf{F}^{-1} \cdot \boldsymbol{\sigma} \cdot \mathbf{F}^{-T} = \mathbf{F}^{-1} \cdot \mathbf{P} . \quad (19)$$

### 4.3 Conservation laws

Conservation laws or balance laws for a continuum body describes the physics of conservation of state variables. Quantities like mass, linear and angular momentum and energy are of interest in continuum mechanics. In this thesis the balance equations are solved in the current configuration and only the balance of mass and momentum are presented.

**4.3.1 Balance of mass** Mass is a physical scalar quantity of a system that defines the amount of matter in a body. Mass does not change with time in a non-relativistic frame of reference, and hence it is always conserved in classical physics.

$$\implies \dot{m} = 0 . \quad (20)$$

Expressing mass in terms of density  $\rho$  and integrating over the continuum body  $\mathcal{B}$

$$\begin{aligned} \dot{m} &= \left( \int_{\mathcal{B}} \rho \, dv \right) \cdot \\ \Rightarrow &= \left( \int_{\mathcal{B}_0} \rho J \, dV \right) \cdot = \int_{\mathcal{B}_0} (\dot{\rho} J + \rho \dot{J}) \, dV, \end{aligned} \quad (21)$$

with the relation  $\dot{J} = J \operatorname{div} \dot{\mathbf{x}}$  we get

$$\begin{aligned} \dot{m} &= \int_{\mathcal{B}_0} (\dot{\rho} J + \rho J \operatorname{div} \dot{\mathbf{x}}) \, dV \\ &= \int_{\mathcal{B}_0} (\dot{\rho} + \rho \operatorname{div} \dot{\mathbf{x}}) J \, dV. \end{aligned} \quad (22)$$

Using transport theorem from equation (11),

$$\int_{\mathcal{B}} (\dot{\rho} + \rho \operatorname{div} \dot{\mathbf{x}}) \, dv = 0, \quad (23)$$

which yields the local statement of balance of mass

$$\dot{\rho} + \rho \operatorname{div} \dot{\mathbf{x}} = 0. \quad (24)$$

The local statement can be reformulated further using  $\operatorname{div} \dot{\mathbf{x}} = \frac{\dot{J}}{J}$  as

$$\begin{aligned} \dot{\rho} + \rho \left( \frac{\dot{J}}{J} \right) &= 0, \\ \Rightarrow \frac{1}{\rho} \dot{\rho} &= -\frac{1}{J} \dot{J}. \end{aligned} \quad (25)$$

In the above equation  $\dot{\rho}$  and  $\dot{J}$  are time dependent variables. Hence, it can be integrated over a time interval

$$\begin{aligned} \int_{t_0}^t \frac{1}{\rho} \dot{\rho} \, dt &= \int_{t_0}^t \frac{1}{J} \dot{J} \, dt \\ \ln \rho \Big|_{t_0}^t &= - \ln J \Big|_{t_0}^t \\ \ln \rho(t) - \ln \rho(t_0) &= - [\ln J(t) - \ln J(t_0)] \\ \ln \rho - \ln \rho_0 &= \ln J_0 - \ln J. \end{aligned} \quad (26)$$

At an initial time  $t_0$  the body is undeformed and does not have any volume changes. Hence, the deformation gradient will result in an identity matrix  $\mathbf{I}$  and the Jacobian  $J(t_0) = 1$

$$\begin{aligned} \implies \ln \rho - \ln \rho_0 &= \ln 1 - \ln J \\ \implies \ln \left( \frac{\rho}{\rho_0} \right) &= \ln \left( \frac{1}{J} \right), \end{aligned} \quad (27)$$

resulting in the integral form of balance of mass

$$J = \frac{\rho_0}{\rho}. \quad (28)$$

**4.3.2 Balance of momentum** Balance of momentum refers to the conservation of linear momentum only. It states that the rate of change of momentum equals the body force and external forces on a body, which is the principle of Newton's second law of motion.

$$\left( \int_{\mathcal{B}} \rho \dot{\mathbf{x}} \, dv \right)^{\cdot} = \int_{\mathcal{B}} \rho \mathbf{b} \, dv + \int_{\partial \mathcal{B}} \mathbf{t} \, da. \quad (29)$$

The left-hand side represents the time derivative of the linear momentum. The right-hand side represents the sum of force due to density over the volume and the system's external forces.

$$\begin{aligned} \left( \int_{\mathcal{B}_0} \rho \dot{\mathbf{x}} \, dv \right)^{\cdot} &= \left( \int_{\mathcal{B}} \rho \dot{\mathbf{x}} J \, dV \right)^{\cdot} \\ &= \left( \int_{\mathcal{B}_0} \rho \dot{\mathbf{x}} J \right)^{\cdot} dV \\ &= \int_{\mathcal{B}_0} (\dot{\rho} \dot{\mathbf{x}} J + \rho \ddot{\mathbf{x}} J + \rho \dot{\mathbf{x}} \dot{J}) \, dV. \end{aligned} \quad (30)$$

Using  $\dot{J} = J \operatorname{div} \dot{\mathbf{x}}$  in the above equation

$$\begin{aligned} \left( \int_{\mathcal{B}_0} \rho \dot{\mathbf{x}} \, dv \right)^{\cdot} &= \int_{\mathcal{B}} (\dot{\rho} \dot{\mathbf{x}} J + \rho \ddot{\mathbf{x}} J + \rho \dot{\mathbf{x}} J \operatorname{div} \dot{\mathbf{x}}) \, dV \\ &= \int_{\mathcal{B}_0} (\dot{\rho} \dot{\mathbf{x}} + \rho \ddot{\mathbf{x}} + \rho \dot{\mathbf{x}} \operatorname{div} \dot{\mathbf{x}}) J \, dV. \end{aligned} \quad (31)$$

Applying transport equation (11)

$$\begin{aligned}
 \left( \int_{\mathcal{B}} \rho \dot{\mathbf{x}} \, dv \right)^\cdot &= \int_{\mathcal{B}} (\dot{\rho} \mathbf{x} + \rho \ddot{\mathbf{x}} + \rho \dot{\mathbf{x}} \operatorname{div} \dot{\mathbf{x}}) \, dv \\
 &= \int_{\mathcal{B}} [\dot{\mathbf{x}} \underbrace{(\dot{\rho} + \rho \operatorname{div} \dot{\mathbf{x}})}_{=0} + \rho \ddot{\mathbf{x}}] \, dv \\
 &= \int_{\mathcal{B}} \rho \ddot{\mathbf{x}} \, dv .
 \end{aligned} \tag{32}$$

The term cancels to zero because of the local statement of balance of mass, equation (24).

The traction vector  $\mathbf{t}$  can be reformulated into Cauchy stress  $\boldsymbol{\sigma}$  using Cauchy theorem. Further applying divergence theorem on the external force term yields

$$\int_{\partial \mathcal{B}} \mathbf{t} \, da = \int_{\partial \mathcal{B}} (\boldsymbol{\sigma} \cdot \mathbf{n}) \, da = \int_{\mathcal{B}} \operatorname{div} \boldsymbol{\sigma} \, dv . \tag{33}$$

Assembling the terms from equations (32) and (33) into equation (29)

$$\begin{aligned}
 \int_{\mathcal{B}} \rho \ddot{\mathbf{x}} \, dv &= \int_{\mathcal{B}} \rho \mathbf{b} \, dv + \int_{\mathcal{B}} \operatorname{div} \boldsymbol{\sigma} \, dv \\
 \implies \int_{\mathcal{B}} (\operatorname{div} \boldsymbol{\sigma} + \rho(\mathbf{b} - \ddot{\mathbf{x}})) \, dv &= 0 ,
 \end{aligned} \tag{34}$$

which gives the local statement of balance of momentum

$$\operatorname{div} \boldsymbol{\sigma} + \rho(\mathbf{b} - \ddot{\mathbf{x}}) = 0 . \tag{35}$$

**4.3.3 Balance of moment of momentum** Balance of moment of momentum is the conservation of angular or rotational momentum. The law reads that the rate of change of angular momentum over a fixed point is equal to the sum of angular momentum due to the body force and external forces over the same point.

$$\left( \int_{\mathcal{B}} \mathbf{x} \times \rho \dot{\mathbf{x}} \, dv \right)^\cdot = \int_{\mathcal{B}} \mathbf{x} \times \rho \mathbf{b} \, dv + \int_{\partial \mathcal{B}} \mathbf{x} \times \mathbf{t} \, da . \tag{36}$$

Similar to the balance of momentum, the left-hand side represents the time derivative of the angular momentum, and the right-hand side represents the sum of momentum due to the body force and external forces on the surface  $\partial \mathcal{B}$ .

Simplifying the terms as below

$$\begin{aligned}
 \left( \int_{\mathcal{B}} \mathbf{x} \times \rho \dot{\mathbf{x}} \, dv \right)^\cdot &= \left( \int_{\mathcal{B}_0} \mathbf{x} \times \rho \dot{\mathbf{x}} J \, dV \right)^\cdot \\
 &= \int_{\mathcal{B}_0} (\mathbf{x} \times \rho \dot{\mathbf{x}} J)^\cdot \, dV \\
 &= \int_{\mathcal{B}_0} (\dot{\mathbf{x}} \times \rho \dot{\mathbf{x}} J + \mathbf{x} \times \dot{\rho} \dot{\mathbf{x}} J + \mathbf{x} \times \rho \ddot{\mathbf{x}} J + \mathbf{x} \times \rho \dot{\mathbf{x}} \dot{J}) \, dV .
 \end{aligned} \tag{37}$$

Applying the relation  $\dot{J} = J \operatorname{div} \dot{\mathbf{x}}$  the equation becomes

$$\begin{aligned}
 \left( \int_{\mathcal{B}} \mathbf{x} \times \rho \dot{\mathbf{x}} \, dv \right)^\cdot &= \int_{\mathcal{B}_0} (\dot{\mathbf{x}} \times \rho \dot{\mathbf{x}} J + \mathbf{x} \times \dot{\rho} \dot{\mathbf{x}} J + \mathbf{x} \times \rho \ddot{\mathbf{x}} J + \mathbf{x} \times \rho \dot{\mathbf{x}} J \operatorname{div} \dot{\mathbf{x}}) \, dV \\
 &= \int_{\mathcal{B}_0} (\dot{\mathbf{x}} \times \rho \dot{\mathbf{x}} + \mathbf{x} \times \dot{\rho} \dot{\mathbf{x}} + \mathbf{x} \times \rho \ddot{\mathbf{x}} + \mathbf{x} \times \rho \dot{\mathbf{x}} \operatorname{div} \dot{\mathbf{x}}) J \, dV .
 \end{aligned} \tag{38}$$

Rearranging the terms and using transport equation (11)

$$\begin{aligned}
 \left( \int_{\mathcal{B}} \mathbf{x} \times \rho \dot{\mathbf{x}} \, dv \right)^\cdot &= \int_{\mathcal{B}} [\underbrace{\dot{\mathbf{x}} \times \dot{\mathbf{x}}}_{=0} \rho + \mathbf{x} \times \rho \ddot{\mathbf{x}} + \mathbf{x} \times \dot{\mathbf{x}} \underbrace{(\dot{\rho} + \rho \operatorname{div} \dot{\mathbf{x}})}_{=0}] \, dv \\
 &= \int_{\mathcal{B}} (\mathbf{x} \times \rho \ddot{\mathbf{x}}) \, dv .
 \end{aligned} \tag{39}$$

The terms cancel out to zero because of the property of cross product of vectors (parallel vectors) and the local statement of balance of mass, equation (24).

Using Cauchy stress theorem on the traction vector  $\mathbf{t}$  and applying divergence theorem on the external force term gives

$$\int_{\partial \mathcal{B}} \mathbf{x} \times \mathbf{t} \, da = \int_{\partial \mathcal{B}} [\mathbf{x} \times (\boldsymbol{\sigma} \cdot \mathbf{n})] \, da = \int_{\mathcal{B}} [(\mathbf{x} \times \operatorname{div} \boldsymbol{\sigma}) + (\mathbf{I} \times \boldsymbol{\sigma})] \, dv . \tag{40}$$

Inserting equations (39) and (40) into equation (36)

$$\begin{aligned}
\int_{\mathcal{B}} (\mathbf{x} \times \rho \ddot{\mathbf{x}}) \, dv &= \int_{\mathcal{B}} \mathbf{x} \times \rho \mathbf{b} \, dv + \int_{\mathcal{B}} [(\mathbf{x} \times \operatorname{div} \boldsymbol{\sigma}) + (\mathbf{I} \times \boldsymbol{\sigma})] \, dv \\
\int_{\mathcal{B}} (\mathbf{x} \times \operatorname{div} \boldsymbol{\sigma} + \mathbf{x} \times \rho \mathbf{b} - \mathbf{x} \times \rho \ddot{\mathbf{x}} + \mathbf{I} \times \boldsymbol{\sigma}) \, dv &= 0 \\
\int_{\mathcal{B}} (\mathbf{x} \times \underbrace{[\operatorname{div} \boldsymbol{\sigma} + \rho(\mathbf{b} - \ddot{\mathbf{x}})]}_{=0} + \mathbf{I} \times \boldsymbol{\sigma}) \, dv &= 0 \\
\implies \int_{\mathcal{B}} (\mathbf{I} \times \boldsymbol{\sigma}) \, dv &= 0 .
\end{aligned} \tag{41}$$

The terms cancel out to zero because of the local statement of balance of momentum, equation (35) and results in the local statement of balance of moment of momentum

$$\mathbf{I} \times \boldsymbol{\sigma} = 0 . \tag{42}$$

The local statement of balance of momentum is satisfied if and only if the Cauchy stress tensor  $\boldsymbol{\sigma}$  is symmetric

$$\implies \boldsymbol{\sigma} = \boldsymbol{\sigma}^T . \tag{43}$$

## 5 Constitutive models

The kinematic and stress equations quantify a state variable for a body at an arbitrary state and are not material specific. Hence, constitutive equations are used to solve the stress state of a material at an arbitrary time and is usually associated with balance laws and other equations of state. A constitutive model is the mathematical description of the physical behaviour of material to external disturbance or loading. The constitutive equations generally have a functional relationship with thermodynamic equations and are a function of field quantities. The complexity involved in the physical approximation is the non-linearity of materials since no material is linear in practice. Therefore, the accuracy of constitutive models does not always meet the experimental results. For this thesis, St. Venant-Kirchoff law and Neo-Hookean type elasticity are implemented for solid mechanics, followed by a fluid model.

### 5.1 Solid mechanics

**St. Venant-Kirchoff law :** St. Venant-Kirchoff material law is a simple hyper-elastic non-linear material model. It is a modification of linear elasticity by expressing stresses as a function of non-linear strains. The material model is compressible and is suitable for deformation with low strain values. The general form is given as

$$\mathbf{S} = \mathbb{C} : \mathbf{E} . \quad (44)$$

The fourth order stiffness matrix  $\mathbb{C}$  can be expanded as

$$\mathbb{C} = \lambda \overset{4}{\mathbf{I}} + 2\mu \overset{4}{\overline{\mathbf{I}}} , \quad (45)$$

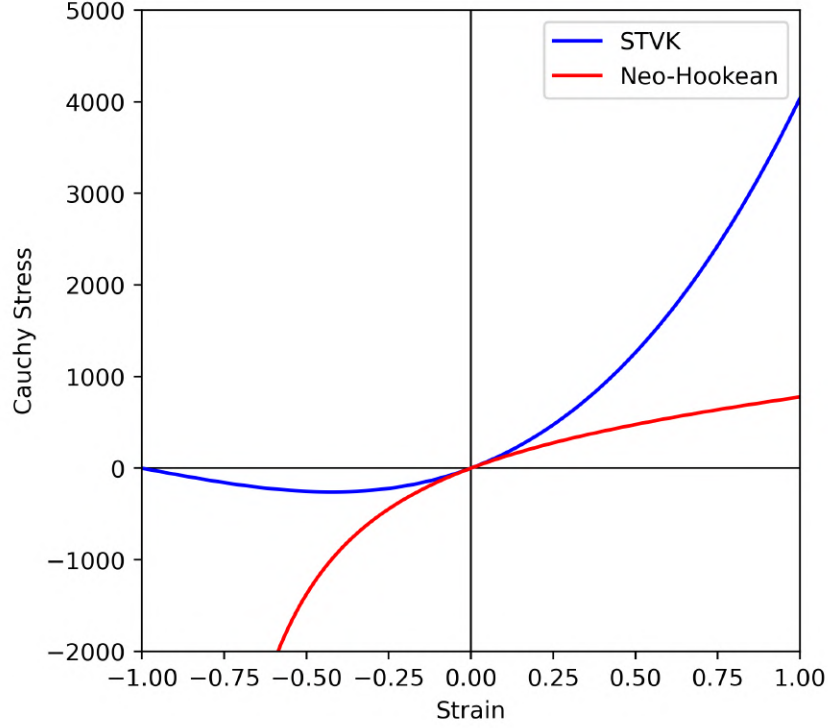
where  $\lambda$  and  $\mu$  are the 1<sup>st</sup> and 2<sup>nd</sup> Lamé's constants,  $\overset{4}{\mathbf{I}} = \mathbf{I} \otimes \mathbf{I}$  and  $\overset{4}{\overline{\mathbf{I}}} = (\mathbf{I} \otimes \mathbf{I})^{\frac{23}{T}}$  are the respective fourth order identity tensors. Also, as in Korelc and Wriggers [2016] the isotropic form of the material law is

$$\mathbf{S} = \lambda \text{tr}(\mathbf{E}) \mathbf{I} + 2\mu \mathbf{E} . \quad (46)$$

**Neo-Hookean law :** Neo-Hookean law is a modification of the linear Hooke law. Neo-Hookean materials are hyper-elastic non-linear and are good in handling compressive stresses. At higher compression, the strain energies are high, and the volume change is almost negligible. It models materials with large deformation and large strains. From Korelc and Wriggers [2016] the relation between stress and strain is given as

$$\boldsymbol{\sigma} = \frac{\mu}{J} (\mathbf{F} \cdot \mathbf{F}^T - \mathbf{I}) + \frac{\lambda}{J} (\ln J) \mathbf{I} . \quad (47)$$

**Comparison study :** A comparison study is made on the material laws mentioned above. The apparent difference is the compressive nature of the materials. Thereby, the generated stresses also vary. The compressive stresses offered by St. Venant-Kirchoff materials are too low and are not rigid to compression. In contrast, Neo-Hookean material is resilient to infinite compression. A stress-strain curve is plotted to picture the difference.



**Figure 3:** Comparison of St. Venant-Kirchoff stresses and Neo-Hookean stresses. The blue curve depicts stresses generated by St. Venant-Kirchoff and the red curve depicts the Neo-Hookean stresses. The stress values are plot for a 1D case against the strain  $\epsilon$  in first direction. It can be seen that as the compressive strain increases the stresses generated by Neo-Hookean model increases, preventing the material to compress to a singularity. It is also noted that the stress tends to infinity as  $\epsilon$  approaches  $-1$ , and no stresses are generated when strains are zero.

In the graph, stresses in the first direction are plotted against strains in the first direction. The blue and red curve represents St. Venant-Kirchoff and Neo-Hookean material laws, respectively. St. Venant-Kirchoff material shows a typical behaviour of compressible materials. For  $\epsilon < 0$ , the stresses are very low, and the stresses are non-linear for  $\epsilon \neq 0$ . Neo-Hookean materials offer greater resilience to compression, which is seen from the generated compressive stresses. As the strains  $\epsilon$  approach  $-1$ , stresses tend to infinity. It is also interesting that there are no stresses when strains are zero, hence both models satisfy a stress-free undeformed reference state.



## 5.2 Fluid mechanics

In the scope of this thesis, a compressible Newtonian fluid is taken for simulations. The fluid is also assumed to satisfy Stokes condition

$$\implies \lambda_f = \frac{2\mu_f}{3}, \quad (48)$$

the parameters  $\lambda_f$  and  $\mu_f$  are the bulk and shear viscosity of the fluid respectively, see de Vaucorbeil et al. [2019]. The constitutive model uses the field quantities in the current configuration. The stress state for the fluid is then given as in Cueto-Ferguson et al. [2004] and de Vaucorbeil et al. [2019]

$$\boldsymbol{\sigma} = 2\mu \left[ \boldsymbol{d} - \frac{1}{3} \text{tr}(\boldsymbol{d}) \boldsymbol{I} \right] - p \boldsymbol{I}, \quad (49)$$

and the pressure of the fluid  $p$  is updated by the equation of state,

$$p = \kappa \left[ \left( \frac{\rho}{\rho_0} \right)^\gamma - 1 \right], \quad (50)$$

where  $\kappa$  represents the bulk modulus of the fluid, and  $\gamma$  is a material parameter. The value of  $\gamma$  is taken as 7 for water and 1.4 for air. The compressibility of the fluid increases with higher bulk modulus values, hence the exactness of the simulation is relatable to a physical behaviour for high  $\kappa$  values. Therefore, the model can be used to simulate e.g. water provided a very high bulk modulus to achieve nearly incompressible behaviour.

## 6 Galerkin weak form of balance of momentum

The current stresses and forces can be obtained by solving the balance of momentum. However, it is very expensive or impossible to solve the strong form of an equation. Hence, in this chapter, the variational or weak form of the balance of momentum is deduced from its strong formulation. The deduction is approached by the Galerkin method. First, the strong form is multiplied with a test function and then it is integrated over the volume of the continuum body to get the weak form. The test functions are chosen as defined in Bathe [2006]. After discretisation, the weak form yields a linear set of equations that are solved on the computational grid for the unknown degrees of freedom. The strong form of balance of momentum is; refer equation (35)

$$G = \text{div } \boldsymbol{\sigma} + \rho (\mathbf{b} - \ddot{\mathbf{x}}) = 0. \quad (51)$$

Multiplying it with the test function  $\delta \mathbf{d}$  and integrating over the body volume  $dv$ ,

$$G = \int_{\mathcal{B}} -\text{div } \boldsymbol{\sigma} \cdot \delta \mathbf{d} \, dv - \int_{\mathcal{B}} \rho (\mathbf{b} - \ddot{\mathbf{x}}) \cdot \delta \mathbf{d} \, dv = 0. \quad (52)$$

Using tensor calculus for relation of  $\text{div } (\boldsymbol{\sigma} \cdot \delta \mathbf{d})$

$$\implies -\text{div } (\boldsymbol{\sigma} \cdot \delta \mathbf{d}) = \boldsymbol{\sigma} : \text{grad } \delta \mathbf{d} - \text{div } (\boldsymbol{\sigma} \cdot \delta \mathbf{d}). \quad (53)$$

Applying on equation (52), yields

$$G = \int_{\mathcal{B}} \boldsymbol{\sigma} : \text{grad } \delta \mathbf{d} \, dv - \int_{\mathcal{B}} \text{div } (\boldsymbol{\sigma} \cdot \delta \mathbf{d}) \, dv - \int_{\mathcal{B}} \rho (\mathbf{b} - \ddot{\mathbf{x}}) \cdot \delta \mathbf{d} \, dv = 0. \quad (54)$$

With the help of divergence theorem the equation can be reformulated as

$$G = \int_{\mathcal{B}} \boldsymbol{\sigma} : \text{grad } \delta \mathbf{d} \, dv - \int_{\partial \mathcal{B}} \boldsymbol{\sigma} \cdot \mathbf{n} \cdot \delta \mathbf{d} \, da - \int_{\mathcal{B}} \rho (\mathbf{b} - \ddot{\mathbf{x}}) \cdot \delta \mathbf{d} \, dv = 0. \quad (55)$$

Further applying Cauchy's stress theorem results in the weak form

$$G = \int_{\mathcal{B}} \boldsymbol{\sigma} : \text{grad } \delta \mathbf{d} \, dv - \int_{\partial \mathcal{B}} \mathbf{t} \cdot \delta \mathbf{d} \, da - \int_{\mathcal{B}} \rho \mathbf{b} \cdot \delta \mathbf{d} \, dv + \int_{\mathcal{B}} \rho \ddot{\mathbf{x}} \cdot \delta \mathbf{d} \, dv = 0. \quad (56)$$

Since no external forces are considered in this thesis, the boundary term representing the external forces is not taken further. In Galerkin method, the test functions  $\delta \mathbf{d}$ ,  $\text{grad } \delta \mathbf{d}$  can be discretised using shape functions  $N^I$ , with  $I$  being the local nodal index.

$$\delta \mathbf{d}(\mathbf{x}) \approx \sum_I N^I(\mathbf{x}) \delta \mathbf{d}^I, \quad \text{grad } \delta \mathbf{d}(\mathbf{x}) \approx \sum_I \text{grad } N^I(\mathbf{x}) \delta \mathbf{d}^I. \quad (57)$$

## 7 The Material Point Method

### 7.1 General overview of the method

Material Point Method is a continuum based approach to analyse physical problems involving large strains. The method was first introduced in Sulsky et al. [1994]. Later many variants of MPM like Generalised Interpolation Material Point (GIMP, see Bardebhagen and Kober [2004]), Convected Particle Domain Interpolation (CPDI, see Sadeghirad et al. [2011], Sadeghirad et al. [2013]) were developed and introduced to overcome the sufferings of previous versions. In the scope of this thesis, only the standard or conventional MPM is discussed and implemented.

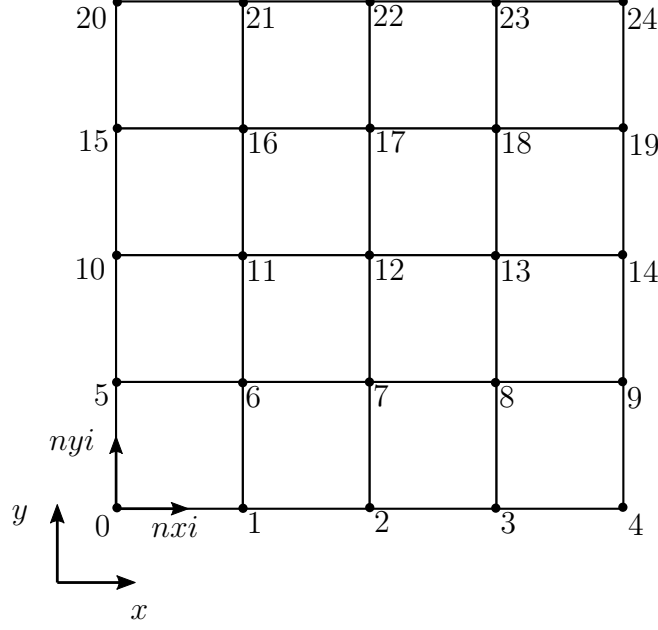
Since MPM is based on continuum mechanics, the algorithmic principle is similar to other analytical methods like FEM and allows MPM to implement the existing constitutive laws and material models. Alternatively, MPM discretises the physical domain into Lagrangian and Eulerian elements, and the state variables of the body are distributed to the material points. The weak form of balance of momentum is solved on the Computational Background Grid (CBG) in the current configuration using the nodal values. The nodal values are obtained by mapping the state variables to the nodes using a suitable ansatz function. The Eulerian mesh gives an advantage over other methods by eliminating mesh distortion because the CBG can be reset at any time step. Consequently, MPM finds application in the analysis of large deformations. For solution, explicit or implicit time schemes are used. However, because of its ability to handle problems with large deformation rates, generally explicit schemes are preferred. Nevertheless, in cases where the strain rate is minimal implicit schemes are useful because of the privilege to choose larger time steps. MPM encounters numerical instability, extension instability and non-physical material separation. These can be overcome by taking higher-order and higher-order continuous ansatz functions and choosing a good number of material points per cell. Comparatively, MPM is also less accurate and has a low convergency rate. Despite the shortcomings, MPM is used in many applications like contact problems, fluid-structure interaction, land sliding, computer graphics and a few machining processes.

### 7.2 Structured 2D grid

In this thesis, a structured Cartesian 2D grid is used as CBG. A regular two-dimensional grid can be defined by the number of cells  $nx$  and  $ny$ , size of the cells  $dx$  and  $dy$ , and the grid's length  $lx$  and  $ly$  in the X and Y direction. An example of the Cartesian grid is shown in figure 4.

$$dx = lx/nx, \quad dy = ly/ny. \quad (58)$$

**Grid index :** Grid index is the number labelled to the nodes of a cell. A grid index can be a local or global variable. Global indexing starts from the origin  $(X_0, Y_0)$  of the grid, whereas any local indexing starts from the cell's left corner. The indices increase in the X-direction first and then increments in the Y direction. The construction of a regular grid is based on a nodal index parametrisation  $nx_i$  and  $ny_i$ .

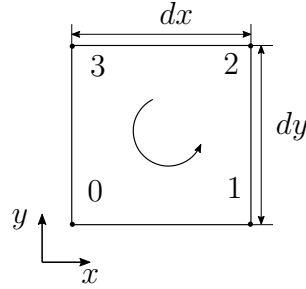


**Figure 4:** A regular 2D Cartesian grid

A regular grid is advantageous in locating the cell enclosing a material point, as in

$$\begin{aligned} nx_i &\in [0, \dots, n_x] & nx_i &= \text{floor}(x_p - X_0 / dx) , \\ ny_i &\in [0, \dots, n_y] & ny_i &= \text{floor}(y_p - Y_0 / dy) . \end{aligned} \quad (59)$$

where  $x_p, y_p$  are the coordinates of an arbitrary material point. The local node numbering runs anti-clockwise starting from bottom left of the cell and as shown in the figure 5.



**Figure 5:** Local indexing

The local node index  $I_0, I_1, I_2, I_3$  corresponding to the nodes 0, 1, 2, 3 are calculated as

$$\begin{aligned} I_1 &= ny_i (n_x + 1) + nx_i , \\ I_2 &= ny_i (n_x + 1) + (nx_i + 1) , \\ I_3 &= (ny_i + 1) (n_x + 1) + (nx_i + 1) , \\ I_4 &= (ny_i + 1) (n_x + 1) + nx_i . \end{aligned} \quad (60)$$

**Local Coordinates** : Global nodal coordinates are the vertices of local coordinates of each cell assembled as a vector in the order of global nodes. Local coordinates of a cell is updated at every time step as the material point moves in the grid. The local coordinate  $(x_n, y_n)$  corresponding to the  $I^{th}$  node of a cell is calculated as

$$\begin{aligned}(x_0, y_0) &= (nx_i dx + X_0, ny_i dy + Y_0), \\(x_1, y_1) &= (x_0 + dx, y_0), \\(x_2, y_2) &= (x_0 + dx, y_0 + dy), \\(x_3, y_3) &= (x_0, y_0 + dy).\end{aligned}\tag{61}$$

### 7.3 Discretisation of the weak form balance of momentum via MPM

In MPM, the physical domain is discretised with the material points and the nodes on the computational grid. The material points are discrete and are associated with the volume. They hold all state variables like mass, momentum, forces, and stresses. Further, the summation of mass and volume over the material points should account for the body's total volume and mass. Using Reimann sum, any discrete quantity of the body is discretised in terms of material points as

$$\int_{\mathcal{B}} (\bullet) dv \approx \sum_{MP} (\bullet)_{MP} v_{MP} . \tag{62}$$

The transition between the material points and the corresponding local nodes is achieved by mapping. The shape functions map the values from material points to grid cells and vice versa. Linear hat function is implemented as ansatz function. Similar shape functions are described in Hughes [2012] and de Vaucorbeil et al. [2019].

$$\begin{aligned}N^0(\mathbf{x}) &= a^0(x) b^0(y), & N^1(\mathbf{x}) &= a^1(x) b^0(y), \\N^2(\mathbf{x}) &= a^1(x) b^1(y), & N^3(\mathbf{x}) &= a^0(x) b^1(y).\end{aligned}\tag{63}$$

$N^I$  corresponds to the shape functions of  $I^{th}$  node of cell and is a function of the coordinate of the material point  $\mathbf{x}$ . The predictors  $a^n$  and  $b^n$  can be expressed in terms of local coordinates  $(x_n, y_n)$  as

$$\begin{aligned}a^0(x) &= \frac{x_2 - x}{dx}, & a^1(x) &= \frac{x - x_0}{dx}, \\b^0(y) &= \frac{y_2 - y}{dy}, & b^1(y) &= \frac{y - y_0}{dy}.\end{aligned}\tag{64}$$

The discretisation is analogous to the Finite Elements Method, wherein the quantities are expressed in terms of nodal points in the Lagrangian mesh, and the material points in MPM are equivalent to the Gauss-Legendre points in FEM. The mass  $m$ , momentum  $\mathbf{mv}$  and force  $\mathbf{f}$  of body are discretised in terms of nodal quantities as

$$\begin{aligned}
m^I &\approx \sum_{MP} N^I(\mathbf{x}_{MP}) m_{MP}, \\
mv_i^I &\approx \sum_{MP} N^I(\mathbf{x}_{MP}) m_{MP} v_{iMP}, \\
f_i^I &\approx \sum_{MP} -N^I_{,j} \sigma_{ij} v_{MP} + b_i m_{MP} N^I(\mathbf{x}_{MP}).
\end{aligned} \tag{65}$$

The nodal quantities  $m^I$ ,  $mv_i^I$ ,  $f_i^I$  represents the mapped nodal mass, momentum and force respectively at the local node  $I$ . The subscripts  $i$  and  $j$  represent the first and second direction, i.e. the X and Y direction respectively, for any tensor quantity. The described convention is followed throughout this thesis. Further,  $N^I_{,j}$  is the spatial derivative of the shape function. The gradient of shape function is expressed in terms of predictors as

$$\begin{aligned}
\frac{dN^I(\mathbf{x})}{dx} &= \frac{d(a^I(x) b^I(y))}{dx} = b^I(y) \frac{d(a^I(x))}{dx}, \\
\frac{dN^I(\mathbf{x})}{dy} &= \frac{d(a^I(x) b^I(y))}{dy} = a^I(x) \frac{d(b^I(y))}{dy}.
\end{aligned} \tag{66}$$

And the derivatives of predictors are

$$\begin{aligned}
\frac{da^0(x)}{dx} &= -\frac{1}{dx}, & \frac{da^1(x)}{dx} &= \frac{1}{dx}, \\
\frac{db^0(y)}{dy} &= -\frac{1}{dy}, & \frac{db^1(y)}{dy} &= \frac{1}{dy}.
\end{aligned} \tag{67}$$

## 7.4 Implementation and solution

The discretisation is implemented into the weak form of balance of momentum and the equation is solved for the acceleration. Using equation (57) in equation (56), the implementation of stress term in the weak form results in

$$\begin{aligned}
\int_B \boldsymbol{\sigma} : \text{grad } \delta \mathbf{d} \, dv &\approx \sum_{MP} (\boldsymbol{\sigma} : \text{grad } \delta \mathbf{d})_{MP} v_{MP} \\
&= \sum_{MP} \boldsymbol{\sigma}_{MP} : \text{grad } \delta \mathbf{d}(\mathbf{x}_{MP}) v_{MP} \\
&= \sum_{MP} \boldsymbol{\sigma}_{MP} : \sum_I \text{grad } N^I(\mathbf{x}_{MP}) \delta \mathbf{d}^I v_{MP} \\
&= \sum_{MP} \sum_I \boldsymbol{\sigma}_{MP} : \text{grad } N^I(\mathbf{x}_{MP}) \delta \mathbf{d}^I v_{MP}.
\end{aligned} \tag{68}$$

Similarly the implementation of body force and mass term gives

$$\begin{aligned}
 \int_{\mathcal{B}} \rho \mathbf{b} \cdot \delta \mathbf{d} \, dv &\approx \sum_{MP} \rho_{MP} \mathbf{b}_{MP} \delta \mathbf{d} \, v_{MP} \\
 &= \sum_{MP} \sum_I \rho_{MP} \mathbf{b}_{MP} N^I(\mathbf{x}_{MP}) \delta \mathbf{d}^I \, v_{MP} \\
 &= \sum_{MP} \sum_I m_{MP} \mathbf{b}_{MP} N^I(\mathbf{x}_{MP}) \delta \mathbf{d}^I .
 \end{aligned} \tag{69}$$

$$\begin{aligned}
 \int_{\mathcal{B}} \rho \ddot{\mathbf{x}} \cdot \delta \mathbf{d} \, dv &\approx \sum_{MP} \rho_{MP} \ddot{\mathbf{x}}_{MP} \delta \mathbf{d} \, v_{MP} \\
 &= \sum_{MP} \sum_I \rho_{MP} \ddot{\mathbf{x}}_{MP} N^I(\mathbf{x}_{MP}) \delta \mathbf{d}^I \, v_{MP} \\
 &= \sum_{MP} \sum_I m_{MP} \ddot{\mathbf{x}}_{MP} N^I(\mathbf{x}_{MP}) \delta \mathbf{d}^I .
 \end{aligned} \tag{70}$$

The global mass matrix  $\mathbf{M}$  and force vector  $\mathbf{F}$  are then obtained from the equations (68), (69) and (70), using appropriate assembly operations.

The solution for the set of unknowns  $\mathbf{d}^I$  is generally given by the solution of Galerkin weak-form with nodal test functions  $\delta \mathbf{d}^I$  and  $\mathbf{d}^I$

$$G(\delta \mathbf{d}^I, \mathbf{d}^I) = 0 . \tag{71}$$

The unknowns  $\mathbf{d}^I$  are a set of nodal acceleration  $\mathbf{a}^I$  and nodal displacements  $\mathbf{x}^I$ , and the set relation is given as

$$\mathbf{d}^I = \mathbf{a}^I \cup \mathbf{x}^I , \quad \mathbf{a}^I = \mathbf{d}^I \setminus \mathbf{x}^I , \quad \mathbf{d}^I = \mathbf{d}^I \setminus \mathbf{x}^I . \tag{72}$$

Equation (71) can be represented as

$$\begin{aligned}
 G &= (\delta \mathbf{d}^I)^T \cdot \mathbf{K}^{eff} \cdot \mathbf{d}^I \\
 &= (\delta \mathbf{d}^I)^T \cdot (\mathbf{M} \cdot \mathbf{a}^I + \mathbf{K} \cdot \mathbf{x}^I) \\
 &= (\delta \mathbf{d}^I)^T \cdot (\mathbf{M} \cdot \mathbf{a}^I + \mathbf{F}) = 0 ,
 \end{aligned} \tag{73}$$

with  $\mathbf{K}$  being the stiffness matrix. Also, the global mass and stiffness matrices can be calculated from the Galerkin weak form as

$$\mathbf{K}^{eff} = \frac{d^2 G}{d\delta \mathbf{d}^I d\mathbf{d}^I} , \quad \mathbf{K} = \frac{d^2 G}{d\delta \mathbf{d}^I d\mathbf{x}^I} , \quad \mathbf{M} = \frac{d^2 G}{d\delta \mathbf{d}^I d\mathbf{a}^I} . \tag{74}$$

From equation (73), nodal acceleration for the next time step  $(\mathbf{a}^I)^{t+1}$  is found from which nodal velocities and displacements are computed

$$\begin{aligned} \implies \mathbf{M} \cdot \mathbf{a}^I + \mathbf{F} &= 0 \\ \implies (\mathbf{a}^I)^{t+1} &= -(\mathbf{M}^t)^{-1} \cdot \mathbf{F}^t. \end{aligned} \quad (75)$$

The superscript  $(\bullet)^{t+1}$  represents a variable in the next time-step and  $(\bullet)^t$  represents the current time-step. In order to reduce the computational effort in inverting the global mass matrix  $\mathbf{M}$ , lumped mass matrix is used.

The instability of explicit time schemes can be approached by introducing numerical damping. Numerical damping refers to the damping modelled using thermodynamic laws. Numerical damping does not mean the presence or implementation of any physical type of damping. A simple model is taken from Li et al. [2018], which introduces an artificial damping force  $\mathbf{f}_d^I$ , and the magnitude of damping is influenced by  $\alpha_d$ . The value of  $\alpha_d$  is to be chosen cautiously depending on the boundary value problem. The artificial damping force does not affect the accuracy of the solution but changes the mechanical behaviour.

$$\mathbf{f}_d^I = \alpha_d \mathbf{m} \mathbf{v}^I. \quad (76)$$

Introducing numerical damping into equation (75) yields

$$(\mathbf{a}^I)^{t+1} = -(\mathbf{M}^t)^{-1} \cdot (\mathbf{F}^t - \mathbf{F}_d), \quad (77)$$

which updates the nodal velocities at each grid node. Equation (77) can be expressed in terms of nodal vectors as

$$(\mathbf{a}^I)^{t+1} = - \left( \frac{1}{m^I} \right)^t (\mathbf{f}^t - \mathbf{f}_d). \quad (78)$$

From which the nodal velocities and displacements at each node is updated with the time incremental value  $\Delta t$ .

$$\begin{aligned} (\mathbf{v}^I)^{t+1} &= - \left( \mathbf{m} \mathbf{v}^I \frac{1}{m^I} \right)^t + (\mathbf{a}^I)^{t+1} \Delta t, \\ (\mathbf{x}^I)^{t+1} &= (\mathbf{v}^I)^{t+1} \Delta t. \end{aligned} \quad (79)$$

Using the computed nodal acceleration, velocity and displacement, the material points' variables are updated. Firstly, the nodal displacement  $\mathbf{x}_{MP}$  and nodal velocity  $\dot{\mathbf{x}}_{MP}$  for each material point for the new time-step  $(\bullet)^{t+1}$  is calculated as

$$\begin{aligned} (\mathbf{x}_{MP})^{t+1} &= (\mathbf{x}_{MP})^t + \sum_I N^I(\mathbf{x}_{MP}) (\mathbf{v}^I)^{t+1} \Delta t, \\ (\dot{\mathbf{x}}_{MP})^{t+1} &= (\dot{\mathbf{x}}_{MP})^t + \sum_I N^I(\mathbf{x}_{MP}) (\mathbf{a}^I)^{t+1} \Delta t. \end{aligned} \quad (80)$$



Secondly the deformation gradient for each material point is updated by

$$(\mathbf{F}_{MP})^{t+1} = (\mathbf{I} + \sum_I \text{grad } N^I(\mathbf{x}_{MP})^{t+1} (\mathbf{v}^I)^{t+1} \Delta t) + (\mathbf{F}_{MP})^t. \quad (81)$$

The continuum mechanics part to update deformation gradient is explained below.

$$\begin{aligned} (\mathbf{F})^{t+1} &= (\Delta \mathbf{F})^{t+1} \cdot (\mathbf{F})^t \\ &= (\mathbf{I} + \mathbf{L} \Delta t)^{t+1} \cdot (\mathbf{F})^t \\ &= \left( \mathbf{I} + \frac{\partial \mathbf{v}^{t+1}}{\partial \mathbf{x}^t} \Delta t \right) \cdot \frac{\partial \mathbf{x}^t}{\partial \mathbf{X}}, \end{aligned} \quad (82)$$

where  $\mathbf{L}$  is the velocity gradient of the material point and is discretised with shape functions in equation (81). In equation (82), the multiplication of velocity  $\mathbf{v}$  and time increment  $\Delta t$  yields deformation  $\mathbf{u}$ . Hence, the equation can be further simplified as

$$\begin{aligned} (\mathbf{F})^{t+1} &= \left( \mathbf{I} + \frac{\partial \mathbf{u}^{t+1}}{\partial \mathbf{x}^t} \right) \cdot \frac{\partial \mathbf{x}^t}{\partial \mathbf{X}} \\ \iff & \\ &= \left( \frac{\partial \mathbf{x}^t}{\partial \mathbf{x}^t} + \frac{\partial \mathbf{u}^{t+1}}{\partial \mathbf{x}^t} \right) \cdot \frac{\partial \mathbf{x}^t}{\partial \mathbf{X}}. \end{aligned} \quad (83)$$

With  $\mathbf{x}^{t+1} = \mathbf{x}^t + \mathbf{u}^{t+1}$ , we get

$$\iff (\mathbf{F})^{t+1} = \frac{\partial \mathbf{x}^{t+1}}{\partial \mathbf{x}^t} \cdot \frac{\partial \mathbf{x}^t}{\partial \mathbf{X}}. \quad (84)$$

Thirdly, the volume of each material point  $v_{MP}$  is computed using the initial volume of the material point and the Jacobian  $J$  of the updated deformation gradient

$$(v_{MP})^{t+1} = (\det \mathbf{F})^{t+1} (v_{MP})^{t=0}. \quad (85)$$

Lastly, the Cauchy stress tensor values of each material point are updated corresponding to the material law, using the above computed  $(\bullet)^{t+1}$  variables.

$$(\boldsymbol{\sigma})^{t+1} = \hat{\boldsymbol{\sigma}}(\mathbf{F}^{t+1}). \quad (86)$$

**7.4.1 USL algorithm and workflow** The Update Stress Last (USL) explicit method is used in this thesis to solve for the body's current position and velocity. The stress values of the body are updated at the end of every time step hence the name. The algorithmic flow is shown in figure 6, and the work flow is enlisted below.

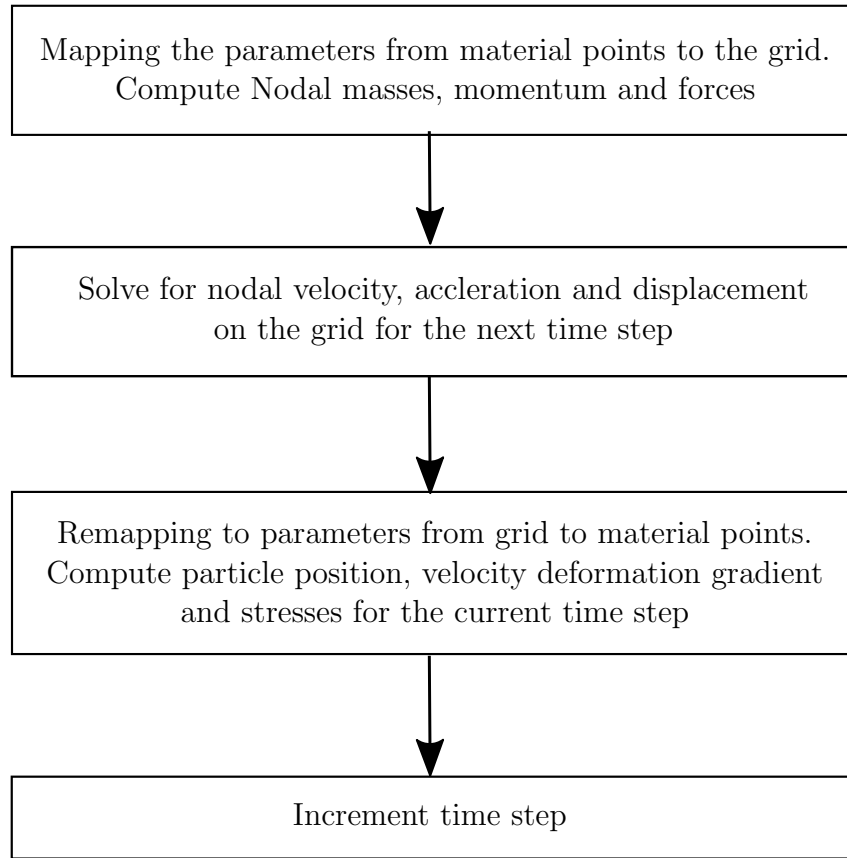


Figure 6: USL Algorithm

1. The grid is reset at the start of every time step to ensure zero mesh distortions
2. The local nodal indices of each material point are computed and assembled into the global matrix. Similarly, each material point's shape function and derived shape functions are calculated locally and assembled into the global matrix
3. The material point parameters are mapped to the grid as its nodal values
4. The nodal acceleration, velocity and displacement is calculated for the next time step by solving the weak form of balance of momentum
5. The required boundary conditions are enforced onto the grid
6. The nodal values are again remapped to the material points and the state variables are updated
7. Increment the time step and repeat the cycle

## 7.5 UML diagram

The Unified Modeling Language (UML) diagram is an overview of the code design from which one can obtain the used data types, variables and available functions. It pictures the architecture and implementation of a code and also provides information about the class hierarchy and relation between classes. The below chart, see figure 7, shows the UML diagram for the code used in this thesis. The code uses three parent classes, namely MPM\_Body, MPM\_Grid, MPM\_Solver. Further, the MPM\_Body class has three children classes.

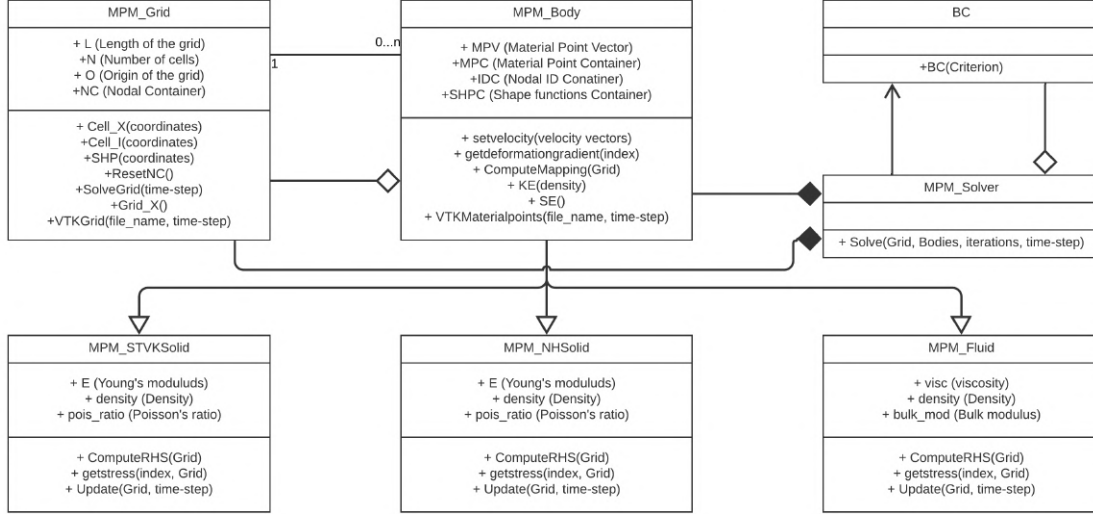


Figure 7: UML Diagram

**MPM\_Body** : The body class defines the material or fluid properties, and the sub-classes MPM.STVKSolid, MPM.NHSolid, and MPM.Fluid correspond to St. Venant-Kirchoff solid material, Neo-Hookean solid material and the fluid model, respectively. The associated physical parameters of the discretised material points are stored in the Material Point Container (MPC). The container holds the coordinates, velocity, volume, stresses, deformation gradient and body forces of each material point. The Shape functions Container (SHPC) stores the shape functions and the derived shape functions at the local nodes of the corresponding material point. The local nodal IDs are saved in the nodal container IDC. The body class maps the material points to the computational grid nodes and computes the global mass matrix and force vector. Further, with the updated nodal displacement, the current position of the material point is found, and the physical parameters are remapped to the material points.

**MPM\_Grid** : The grid class holds the Nodal Container (NC), which stores the nodal mass, momentum, force, acceleration, velocity and displacement computed by the body class. The weak form-of balance of momentum is solved for nodal acceleration using USL explicit time scheme, and subsequently, the nodal velocity and displacement is updated.

**MPM\_Solver** : The algorithmic work-flow for the USL scheme is defined in the solver class. The new time step starts with resetting the grid, and updates the material points'

position at the end of the time step. The solver class also defines and enforces the boundary conditions onto the grid.

## 7.6 C++ implementation of the MPM

The MPM 2D solver is written in C++ and uses the Velocity-Verlet explicit scheme for time integration. Two solid models, namely St. Venant-Kirchoff and Neo-Hookean material laws, are used for solid mechanics, and one fluid model is used for fluid mechanics. The general layout of the architecture is shown in the UML diagram. A brief of the functions, containers and the implementation is given below. Access to the full code is available in GitLab under the link : [https://git.uni-due.de/snkamuth/mpm\\_2d\\_api](https://git.uni-due.de/snkamuth/mpm_2d_api)

**7.6.1 Containers** The containers used in the code are Material Point Vector (MPV), Material Point Container (MPC), local nodal index container (IDC), Shape functions Container (SHPC) and Nodal Container (NC). These containers are global and store the values of all material points and nodes.

**MPV** The Material Point Vector is an array of coordinates of material points and their volume. It is just a storage container and is not used in any operations. The MPV holds only the initial coordinates and the volume of material points, which is later passed to the Material Point Container MPC.

X-Coordinate  $x_{MP}$  | Y-Coordinate  $y_{MP}$  | Volume per material point  $v_{MP}$

Table 1: Material Point Vector (MPV)

**MPC** MPC is the array which holds all the required current parameters of a material point. At start, it augments the data from MPV. The values stored in the Material Point Container are

X-Coordinate $x_{MP}$	Y-Coordinate $y_{MP}$	
Velocity $v_x$	Velocity $v_x$	Volume $v_{MP}$
Cauchy Stress $\sigma_{xx}$	Cauchy Stress $\sigma_{xy}$	Cauchy Stress $\sigma_{yy}$
Cauchy Stress $\sigma_{zz}$	Deformation Gradient $F_{xx}$	Deformation Gradient $F_{xy}$
Deformation Gradient $F_{yx}$	Deformation Gradient $F_{yy}$	Velocity Gradient $L_{xx}$
Velocity Gradient $L_{xy}$	Velocity Gradient $L_{yx}$	Velocity Gradient along $L_{yy}$
Gravity $b_x$	Gravity $b_y$	

Table 2: Material Point Container (MPC)

**IDC** The local nodal indices of the cell for each material point is stored in IDC.

Node 1 | Node 2 | Node 3 | Node 4

Table 3: Local nodal index container (IDC)

**SHPC** The shape functions and derived shape functions for each material point at each node is stored in the following order in SHPC.

Shape function $N^1$	Gradient $N^1_{,x}$	Gradient $N^1_{,y}$
Shape function $N^2$	Gradient $N^2_{,x}$	Gradient $N^2_{,y}$
Shape function $N^3$	Gradient $N^3_{,x}$	Gradient $N^3_{,y}$
Shape function $N^4$	Gradient $N^4_{,x}$	Gradient $N^4_{,y}$

Table 4: Shape function Container (SHPC)

**NC** For each material point, the global mass matrix and global force vector values are stored alongside the nodal momentum, acceleration, velocity and displacement. The columns are arranged in below order for each grid node.

Nodal masses	Nodal momentum X	Nodal momentum Y	Nodal forces X
Nodal forces Y	Nodal acceleration X	Nodal acceleration Y	Nodal velocity X
Nodal velocity Y	Nodal displacement X	Nodal displacement Y	—

Table 5: Nodal container (NC)

**7.6.2 MPM\_Body** The body class is the parent class for the solid and fluid classes. The MPV container initialises the body class, and in turn, the class initialises MPC, SHPC and IDC containers. The children classes are initialised by their respective solid or fluid properties. The right-hand side of the weak form of balance of momentum is computed within the body class. Below is a brief of the available functions in the body class.

`setvelocity(vector<double> &vx, vector<double> &vy)` Initiates the velocity of the body in X and Y direction.

`getdeformationgradient(size_t p)` Returns the deformation gradient of the material point p.

`ComputeMapping(MPM_Grid &Grid)` Maps each material point to its local cell and returns IDC and SHPC.

`ComputeRHS(MPM_Grid &Grid)` Computes the local nodal values and assembles it to the Nodal Container. Equation (65) is then implemented as

```

1 //Computing and assembling Nodal Masses
2 m_l_mp[j] = M[i] * SHP_mp[j];
3 Grid.NC[ID_mp[j]][0] += m_l_mp[j];
4
5 //Computing and assembling nodal momentum vector
6 mvx_l_mp[j] = mvx[i] * SHP_mp[j];
7 mvy_l_mp[j] = mvy[i] * SHP_mp[j];
8 Grid.NC[ID_mp[j]][1] += mvx_l_mp[j];
9 Grid.NC[ID_mp[j]][2] += mvy_l_mp[j];
10
11 //Computing and assembling nodal force vector
12 F = getdeformationgradient(i);
13 J = (F[0][0] * F[1][1]) - (F[1][0] * F[0][1]);
14
15 fx_l_mp[j] = MPC[i][17]*M[i]*SHP_mp[j]
16             - dSHPx_mp[j]*Sig_mp[0]*J*MPC[i][4]
17             - dSHPy_mp[j]*Sig_mp[1]*J*MPC[i][4];
18
19 fy_l_mp[j] = MPC[i][18]*M[i]*SHP_mp[j]
20             - dSHPx_mp[j]*Sig_mp[1]*J*MPC[i][4]
21             - dSHPy_mp[j]*Sig_mp[3]*J*MPC[i][4];
22 Grid.NC[ID_mp[j]][3] += fx_l_mp[j];
23 Grid.NC[ID_mp[j]][4] += fy_l_mp[j];

```

Listing 1: Implementation of the discretisation of weak form of balance of momentum in C++.

The above listing shows the discretisation of the weak form and assembling it into the global form. **m\_l\_mp** is a vector which contains the local nodal mass at all four nodes associated to a material point. Similar to **m\_l\_mp**, **mvx\_l\_mp** and **mvy\_l\_mp** are the nodal momentum along X and Y direction respectively, and **fx\_l\_mp** and **fy\_l\_mp** are the nodal forces along X and Y direction respectively.

`getstress(size_t i, MPM_Grid &Grid)` Computes the Cauchy stresses depending on the material model.

`Update(MPM_Grid &Grid, double dt)` Remaps the current  $(\bullet)^{t+1}$  nodal values to the corresponding material points and updates the current displacement, velocity, deformation gradient and Cauchy stresses of the material point.

`VTKMaterialpoints(string output_file_name, size_t step)` Returns the .vtk files of the material points for post-processing.

**7.6.3 MPM\_Grid** The grid class is defined and initialised by the grid's origin, dimensions and the number of cells. The weak form is solved and updated at every time-step. It also initialises and holds the Nodal Container. An overview of the available functions in the grid class is given below.

`Cell_X(array<double,2> X)` Returns the nodal coordinates for a material point.

`Cell_I(array<double,2> X)` Returns the local nodal indices for a material point.

`SHP(array<double,2> X)` Returns the shape functions and derived shape functions for a material point.

`ResetNC()` Resets the Nodal Container at any time instant.

`SolveGrid(double dt, double alpha, double mass_cutoff=  $1^{-10}$ )` Solves equation (77) and updates the nodal acceleration, velocity and displacement. Inverting the lumped mass matrix  $\mathbf{M}$  is expensive. Hence, the equation is solved locally at each cell. The parameter alpha is the the damping factor  $\alpha_d$  as in equation (76). The mass\_cutoff value is the threshold nodal mass value to preserve stability. The implementation into the code is as follows.

```

1  for (size_t i = 0; i < (NoNodes) ; i++)
2      {
3          if (NC[i][0] >= mass_cutoff)
4              {
5                  m_inv = 1/NC[i][0];
6                  //Compute Nodal acceleration along X and Y
7                  NC[i][5] = (NC[i][3] - (alpha * NC[i][1])) * m_inv;
8                  NC[i][6] = (NC[i][4] - (alpha * NC[i][2])) * m_inv;
9
10                 //Compute Nodal velocity along X and Y
11                 NC[i][7] = (NC[i][1] * m_inv) + NC[i][5] * dt;
12                 NC[i][8] = (NC[i][2] * m_inv) + NC[i][6] * dt;
13
14                 //Compute Nodal velocity along X and Y
15                 NC[i][9] = NC[i][7] * dt;
16                 NC[i][10] = NC[i][8] * dt;
17             }
18     }

```

Listing 2: Implementation of the solution of weak form of balance of momentum in C++.

`Grid_X()` Returns the the coordinates of every cell in the computational background grid.

`VTKGrid(string output_file_name, size_t step)` Returns .vtk files of the grid for post-processing.

**7.6.4 MPM\_Solver** The algorithm to solve the weak form of balance of momentum is implemented into the solver class. It calls the defined functions from the body and grid classes in the algorithmic order. The member function in the Solver class is

`Solve(MPM_Grid* Grid, vector<MPM.Body* > Bodies, size_t NoS, double dt, double alpha)` Pointers are used to call the functions and execute the work flow instructions. The below lambda function enforces the boundary conditions.

```

1 class BC
2 {
3     public:
4     BC(const function<bool(array<double,2>>) &Criterion_) : Criterion(
        Criterion_), active_vx(false), active_vy(false) {}
5
6     const function<bool(array<double,2>>) &Criterion;
7     bool active_vx;      double value_vx;
8     bool active_vy;      double value_vy;
9 };

```

Listing 3: Function to enforce boundary conditions.

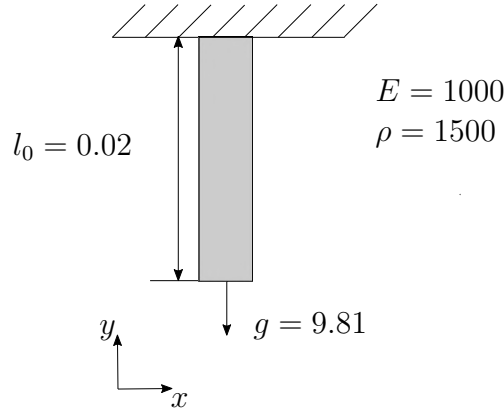


## 8 Numerical examples

A few simulations from solid and fluid mechanics are carried out, and the implementation of the code is verified. The obtained results are compared and validated with other literature. The numerical examples include large deformations, contact problems, fluid impact, fluid-fluid interaction and fluid-structure interaction. Units are not indicated in these examples. Hence, they are to be chosen consistently from any unit system.

### 8.1 1D Spring verification

A 1D spring of length  $l_0 = 0.2$  and breadth  $b = 0.02$  is fixed at one end and is subjected to loading under its self-weight, see figure 8. The material parameters are Young's modulus  $E = 10^5$  and density  $\rho = 1500$ . The acceleration due to gravity is  $g = 9.81$ .



**Figure 8:** 1D Spring

A computational grid of size  $1 \times 0.02$  is chosen and is discretised into 21 cells. The spring is discretised into 10 particles. The simulation is run for a period of  $T = 2$  with a time increment of  $\Delta t = 10^{-4}$ . The numerical damping factor  $\alpha_d$  is set to 20.

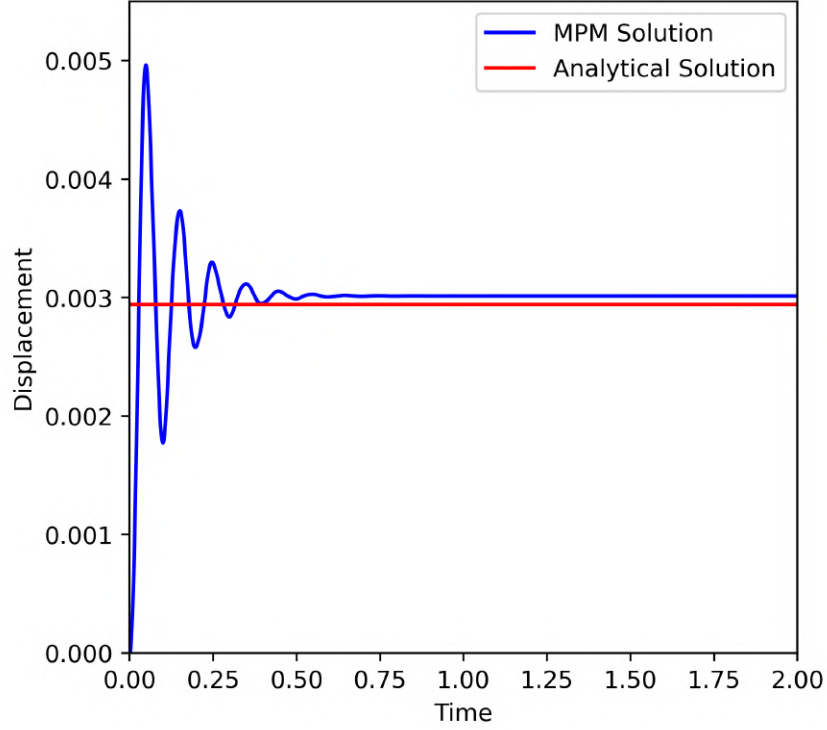
### Comparison of results

**MPM solution :** The final displacement of the material point at the lowest end of the spring is taken for MPM deformation. Because of numerical damping, the spring's oscillation dampens to  $\mathbf{u}_{\text{MPM}} = 3.005 \times 10^{-3}$ . The value  $\mathbf{u}_{\text{MPM}}$  is considered as the total deformation of the spring.

**Analytical solution :** The analytical solution of the total deformation of the spring  $\mathbf{u}$  is also calculated considering linear mechanics, which yields

$$\begin{aligned} \mathbf{u} &= \frac{g \rho l_0^2}{2 E} \\ &= 2.943 \times 10^{-3} . \end{aligned} \tag{87}$$

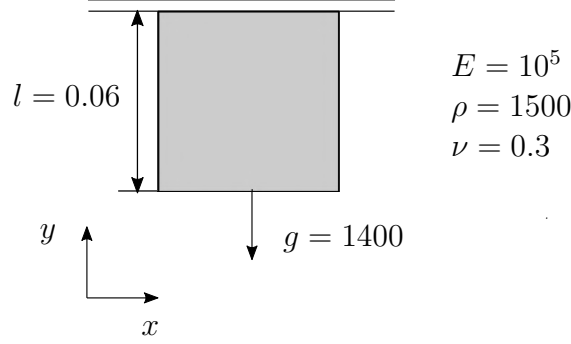
**Observation and inference :** The results obtained analytically and numerically are plotted as a graph with displacements along  $X$ -axis and time along  $Y$ -axis. The below graph, see figure 9 shows a comparison between the two solutions. The red line depicts the analytical solution  $\mathbf{u}$ , and blue curve represents the numerically damped MPM solution  $\mathbf{u}_{\text{MPM}}$ . From the plot, it is evident that the converged MPM solution is close to the analytical solution and is of the same order, verifying the implementation of the code.



**Figure 9:** Time vs. Displacement plot of the spring. The red line represents the analytical solution  $\mathbf{u}$  and the blue curve represents the numerically damped MPM solution  $\mathbf{u}_{\text{MPM}}$ .

## 8.2 2D Truss bench-marking

A 2D square truss of size  $l = 0.06$  is fixed vertically at one end, and is subjected to loading under its self weight, see figure 10. The material parameters are Young's modulus  $E = 10^5$ , density  $\rho = 1500$  and Poisson's ratio  $\nu = 0.3$ . Take the gravitational force  $g = 1400$ .



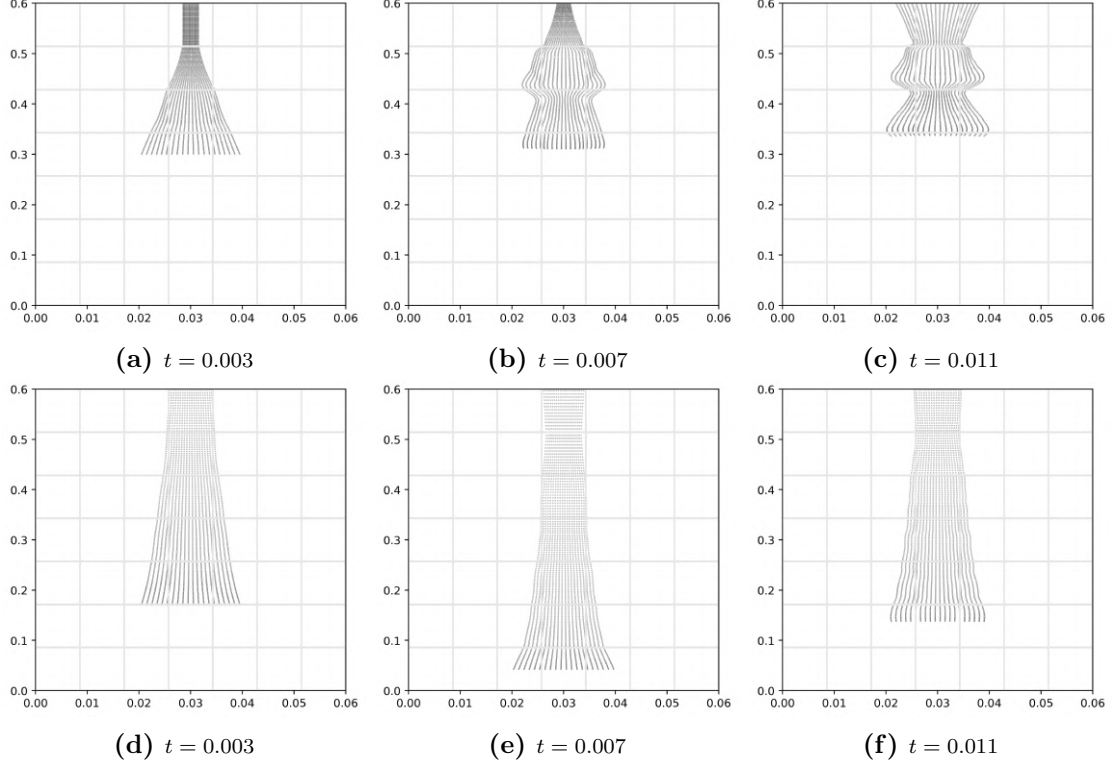
**Figure 10:** 2D Truss

The problem is defined to analyse the effect of Poisson's ratio in a 2D truss. Hence, a significant body force is chosen such that the longitudinal deformation is approximately 100%. Since the problem involves a large deformation, Neo-Hookean material law is used, and the consequence of St. Venant-Kirchoff law is also discussed. Finding an analytical solution for the defined problem is very particular. Deriving at an analytical solution and comparing it with the MPM results is not the motive of this task. Hence, the convergence of the results is observed for the number of material points and cells. Numerical dampening factor  $\alpha_d$  is taken as 25.

**8.2.1 Effect of St. Venant-Kirchoff material law** A small case study on a modified truss with dimensions  $0.02 \times 0.2$  and  $g = 500$  is done to visualise the effect of St. Venant-Kirchoff material law on large deformation problems. The truss is discretised with  $20 \times 150$  material points with a  $7 \times 7$  computational grid for this problem.

In figure 11, the upper row shows the elongation of the truss using St. Venant-Kirchoff material law and the lower shows that of Neo-Hookean material law. The St. Venant-Kirchoff law, because of its weakness to deal compressive deformations, compresses a lot at the top, which also causes the truss to deform, contradicting physical behaviour. From the images, it is seen that Neo-Hookean material law is more appropriate for problems with large strains due to its nature of handling compressive stresses and the exactness of physical behaviour.

**8.2.2 Convergence study** Four grids namely with  $3 \times 3$ ,  $4 \times 4$ ,  $7 \times 7$  and  $8 \times 8$  cells are chosen. The discretisation of the grid is fixed, and a convergence study is made on the longitudinal and lateral displacement by increasing the number of material points in the order of  $5 \times 5$ ,  $20 \times 20$ ,  $100 \times 100$  and  $500 \times 500$ . It is also noted that the number of material points laterally affects longitudinal displacement and vice-versa. Each simulation is run for a period of  $T = 2$ , with a time increment of  $\Delta t = 10^{-5}$ .



**Figure 11:** The series of images shows the elongation of truss. The material points are scaled to a larger size for the purpose of visualisation. The upper row shows the effect of St. Venant-Kirchhoff material law, and the below row shows for Neo-Hookean material law. Because of its compressible nature, St. Venant-Kirchhoff material law compresses too much at the top and results in a non-physical behaviour.

In the tables below,  $l_y$  represents the distance between the first and last material point of the centre column of material points, and  $l_x$  represents the distance between the first and last material point of the top row of material points at the end of the simulation.

Material points	Longitudinal displacement $l_y$	Lateral displacement $l_x$
$5 \times 5$	0.1135	0.0230
$20 \times 20$	0.1167	0.0318
$100 \times 100$	0.1190	0.0345
$500 \times 500$	0.1197	0.0349

Table 6: Convergence study for a grid with  $3 \times 3$  cells

Material points	Longitudinal displacement $l_y$	Lateral displacement $l_x$
$5 \times 5$	0.1365	0.0247
$20 \times 20$	0.1202	0.0308
$100 \times 100$	0.1206	0.0326
$500 \times 500$	0.1214	0.0329

Table 7: Convergence study for a grid with  $4 \times 4$  cells

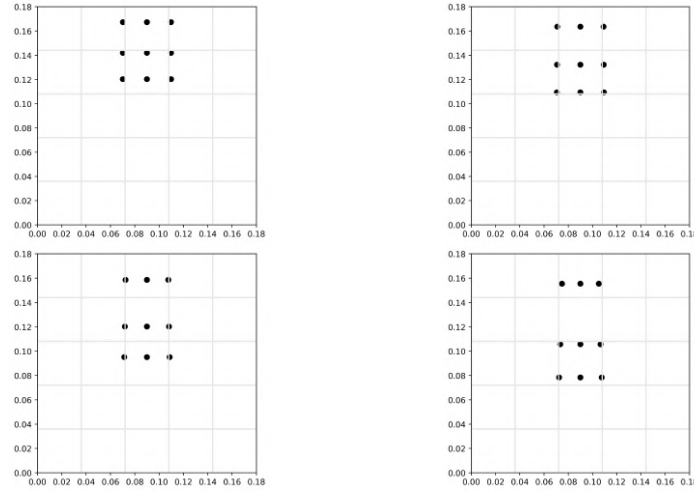
Material points	Longitudinal displacement $l_y$	Lateral displacement $l_x$
$5 \times 5$	Extension Instability	Extension Instability
$20 \times 20$	Extension Instability	Extension Instability
$100 \times 100$	0.1202	0.0314
$500 \times 500$	0.1199	0.0322

Table 8: Convergence study for a grid with  $7 \times 7$  cells

Material points	Longitudinal displacement $l_y$	Lateral displacement $l_x$
$5 \times 5$	Extension Instability	Extension Instability
$20 \times 20$	Extension Instability	Extension Instability
$100 \times 100$	0.1213	0.0312
$500 \times 500$	0.1208	0.0321

Table 9: Convergence study for a grid with  $8 \times 8$  cells

**Extension instability** : When a material point moves out of a cell, it loses connection with that cell creating spurious material discontinuities and possibly a cell gap between two or more material points resulting in extension instability, see figure 12. Consequently, the material points escape the grid, and the simulation fails. Extension instability can be overcome by applying proper boundary conditions, choosing denser discretisation of material points and maintaining a healthy number of material points per cell.

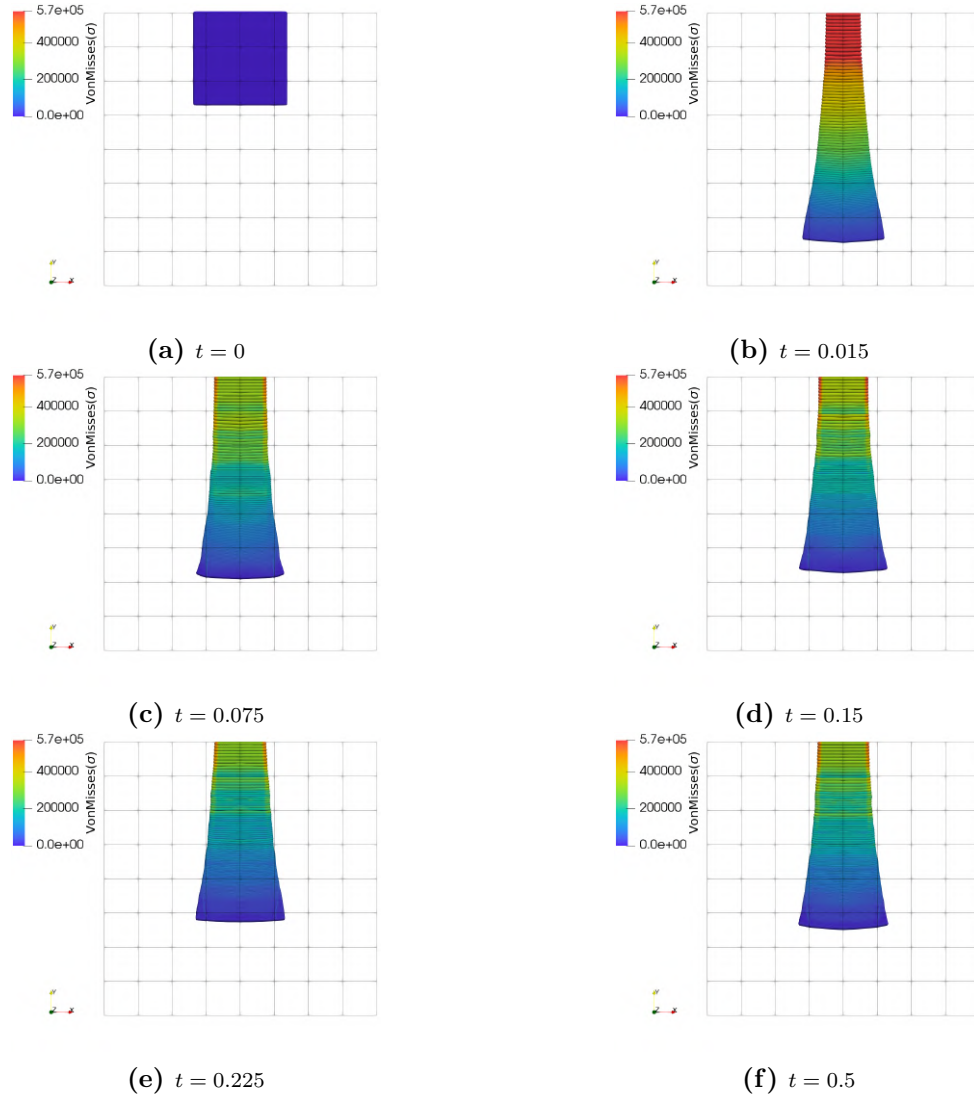


**Figure 12:** Series of plots showing the movement of material points across the cell leading to extension instability. The material points are scaled to a larger size for visualisation.

**8.2.3 Observation and inference** The increase in material points improves the results because of a better approximation of the truss dimension and apparently, comes with a computational cost. Besides, the grid's refinement demands a finer discretisation of the truss, which can be witnessed from tables 8 and 9, with more cells and fewer material points results in extension instability. From the values displayed in the above table, coarser discretisation results might give a false impression of exactness. Hence, it

is advisable to post-process the data and observe the deformation. The exactness of deformation is closer to a physical deformation only in fine discretisation. In MPM increase in the number of cells or material points alone does not yield a better solution. Choosing an appropriate number of cells with a healthy number of material points per cell helps achieve good results. Notably, for explicit time schemes, the time increment  $\Delta t$  should be decreased appropriately to avoid numerical instability.

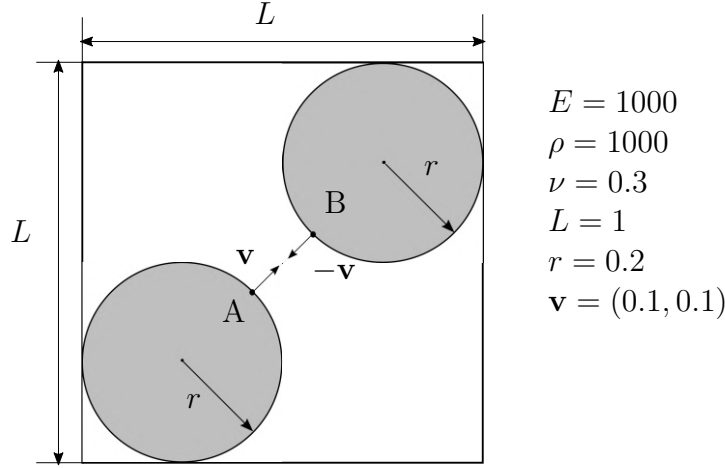
**Elongation and stresses in the truss :** The below series of images pictures the deformation and generated Von-Misses stresses in the truss. The material points are scaled by a factor of 8000 for visualisation. The scenes are taken from the simulation of the  $8 \times 8$  grid with  $100 \times 100$  material points. The effect of the Poisson's ratio can be seen along the lateral deformation. Oscillation is induced by self-weight, and the numerical damping dampens the oscillation at approximately  $t = 0.5$ .



**Figure 13:** The material points of the truss are scaled by a factor of 8000 for visualisation. Due to its self weight, the truss elongates and oscillates. With the effect of Poisson's ratio, the truss also deforms laterally. Because of the introduced numerical damping, the oscillation dampens approximately at  $t = 0.5$

### 8.3 Two disks bench-marking

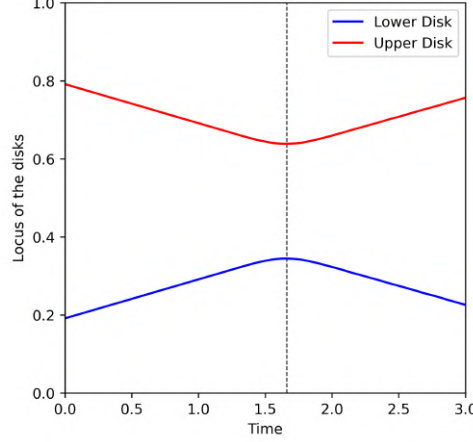
Two elastic disks of radius  $r = 0.2$  are moving towards each other with velocity  $\mathbf{v}$ , see figure 14. The material parameters are Young's modulus  $E = 1000$ , density  $\rho = 1000$  and Poisson's ratio  $\nu = 0.3$ . The initial velocity for upper disk is  $-\mathbf{v} = (-0.1, -0.1)$  and lower disk is  $\mathbf{v} = (0.1, 0.1)$ . No boundary conditions are considered for the problem because the simulation is stopped before the disks move out of the computational domain.



**Figure 14:** Two Disks

A computational grid of size  $1 \times 1$  is chosen and discretised into  $20 \times 20$  cells. Each disk is discretized into 316 material points. The simulation is run for a period of  $T = 2$  with a time increment of  $\Delta t = 10^{-4}$  and without damping factor  $\alpha_d = 0$ . The energies of the system are compared for both St. Venant-Kirchoff and Neo-Hookean material models. Further, the contact in MPM framework is also studied. The problem is referred from Sulsky et al. [1994] and de Vaucorbeil et al. [2019].

**8.3.1 Contact and locus of the disks** The material point approximately at centre of the disks are tracked to estimate the time of collision. From the below locus plot, see figure 15 the points at which the gradient changes its sign is taken to be the point of maximum deformation, approximately at  $t = 1.65$ . From the energy plot, see figure 16 the point where the kinetic energy starts to fall is theoretically the initial point of contact, approximately at  $t = 1.21$ . Whereas, the actual point of contact should be at  $t = AB/2\sqrt{2}\mathbf{v} = 1.5858$ , according to de Vaucorbeil et al. [2019] where  $\mathbf{v} = 0.1$  is the velocity and  $AB$  is the distance between the points  $A$  and  $B$ , see figure 14.



**Figure 15:** Locus of the centre of the disks is shown in the above plot. The blue line shows the locus of the centre of the lower disk and the red line is of the upper's. The point of inflection of the curves is where the disks attain maximum deformation (approx,  $t = 1.65$ ), indicated by black dashed line.

The contact in MPM happens earlier than the actual physical contact because it shares the same velocity field. The sharing of velocity fields is advantageous because it makes no-slip and non-penetrative contact natural in MPM. The disadvantage is premature contact. When the material points are separated by a cell or two, they share the same nodal points, detecting the material points; therefore, the body's are in contact. Sometimes the bodies stick and become inseparable, which accounts for a second drawback. In this example, the disks drift apart without sticking because of their high momentum.

**8.3.2 Energies of the disks** The kinetic and strain energy of the whole system is calculated to verify energy conservation. The kinetic energy  $K$  and strain energy  $U$  are found by

$$\begin{aligned}
 K &= \frac{1}{2} \sum_p \mathbf{v}_p \cdot \mathbf{v}_p m_p, \\
 U &= \sum_p \frac{1}{2} \sigma_{p,ij} \varepsilon_{p,ij} v_p,
 \end{aligned} \tag{88}$$

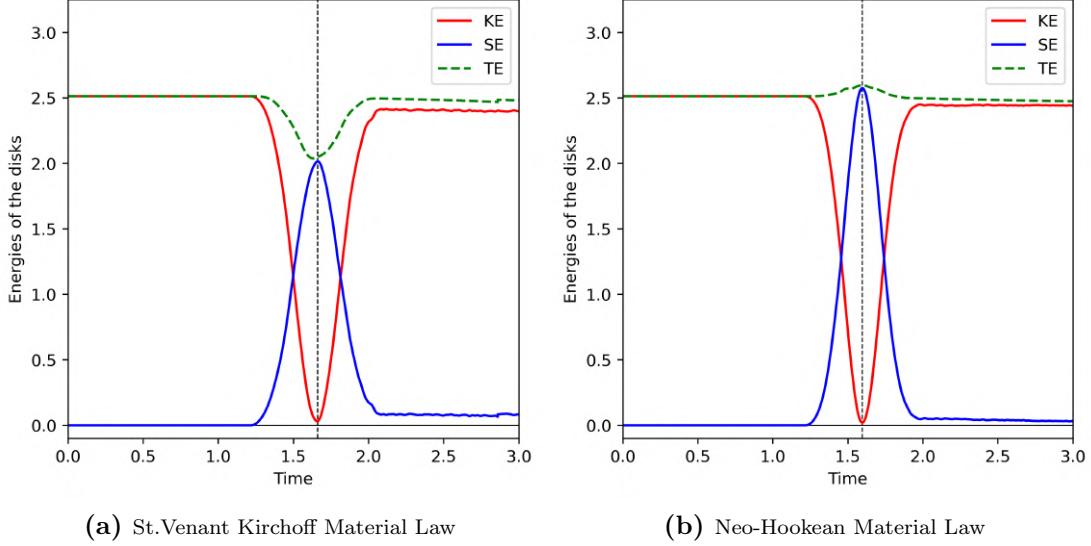
where,  $p$  is the index of a material point and  $\mathbf{v}_p$ ,  $m_p$ ,  $v_p$  are the velocity, mass and volume of the  $p^{th}$  material point.  $\sigma_p$ ,  $\varepsilon_p$  are the stress and strain of the  $p^{th}$  material point. The total kinetic and strain energy are calculated by summation over all material points, from which the total energy of the system is calculated

$$Total\ Energy = K + U. \tag{89}$$

The maximum kinetic energy is found to be  $K = 2.513$  and is exactly the same as in de Vaucorbeil et al. [2019]. Also, the energy plot curves matches that of Sulsky et al. [1994], Buzzi et al. [2008] and de Vaucorbeil et al. [2019], which verifies the implementation.



**Comparison of material laws :** The energy of the system is computed by using St. Venant Kirchhoff and Neo-Hookean material laws, and are compared in the plot below, see figure 16. The left image shows energy plot using St. Venant Kirchhoff law and right shows the plot using Neo-Hookean law.



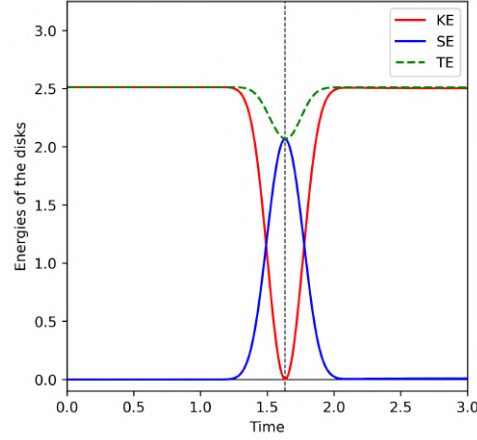
**Figure 16:** A comparison of the energies of the system by two material laws. The total energy of the system is shown by the green curve, whereas the kinetic and strain energy are shown by red and blue curves, respectively. The dashed black line is where the disks undergoes maximum deformation.

It can be read from the plot that the minimum kinetic energy and the maximum strain energy occurs when the disks experience maximum deformation. When the disks move towards each other, kinetic energy remains maximum, and when the contact is made, the kinetic starts converting into strain energy. The decrease in the total energy is associated with the free vibration of the disk and numerical dissipation. Because of less existence of higher modes, perfect energy conservation is generally not expected. This behaviour is the same in both material laws. However, the actual difference rises with *Total Energy*. St. Venant Kirchhoff law shows a drop in the *Total Energy* of the system, and Neo-Hookean law shows a rise at the maximum deformed state. The difference is because of the maximum strain energy obtained from the two different laws.

Strain energy is dependent on the stresses and resilience of a body. The generated compressive stresses in St. Venant Kirchhoff material model are less, leading to smaller strain energy. In contrast, Neo-Hookean is used for large strain problems. The stress equation contains an exponential of the Jacobian term, see equation (47). Even a linear rise in the strain could cause the stresses to rise exponentially, leading to significant strain energy. This variance in stresses could potentially explain the reason for the behavioural difference in the energy plots.

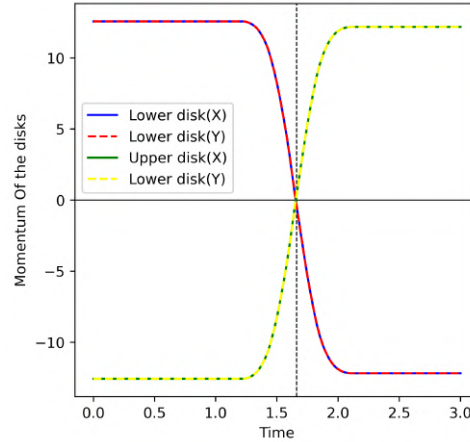
**Energy plot for a finer discretisation :** The disks are discretised approximately 100 times denser with 3016 material points each. With an increase in material points, the availability of higher modes is also increased. Hence, a better energy conservation is

observed, see figure 17. The energy plot is using St. Venant-Kirchoff material law.



**Figure 17:** Energy plot for a finer discretisation using St.Venant-Kirchoff material law.

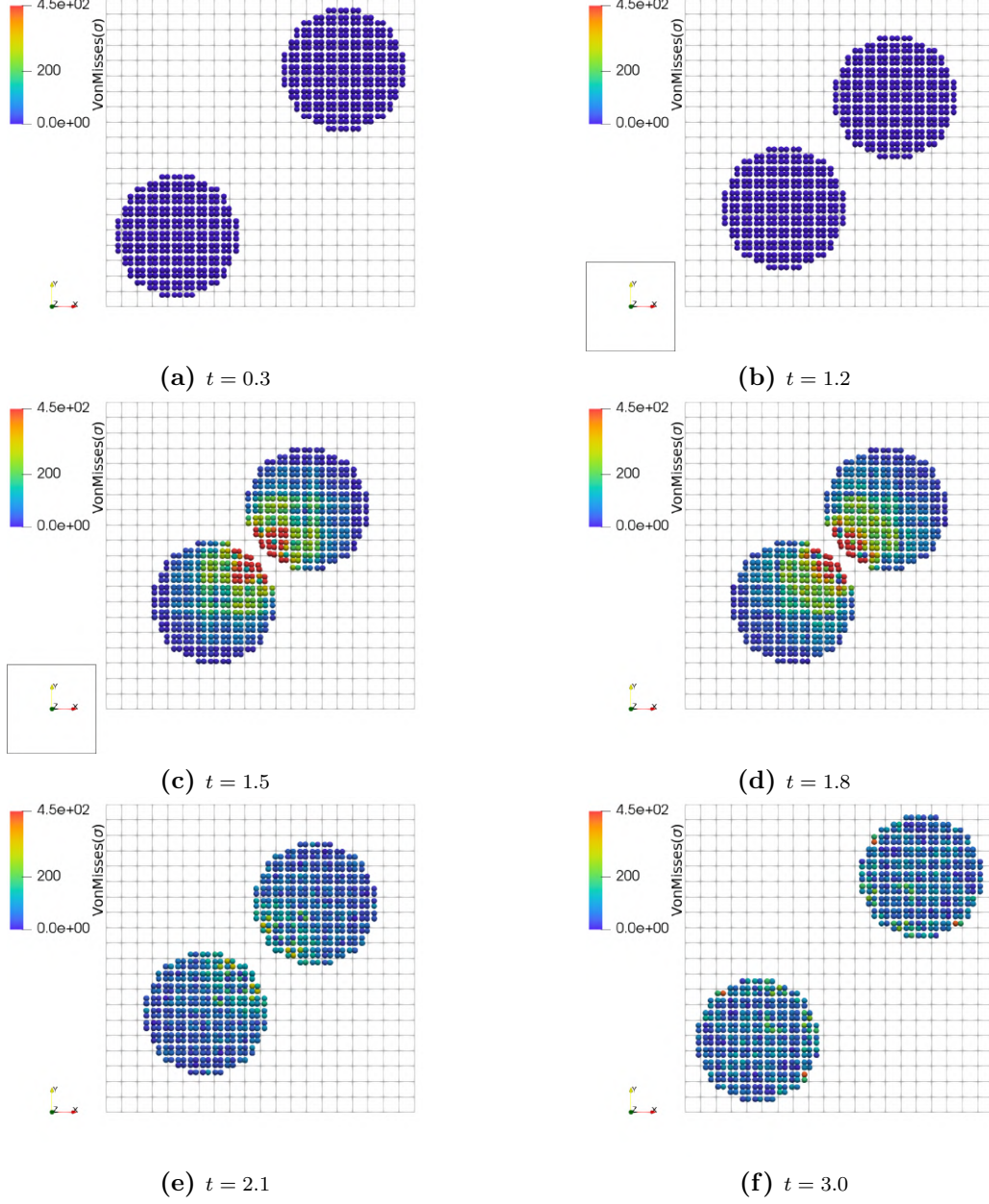
**8.3.3 Momentum of the disks** The disks' momentum are plotted, and its conservation is verified. The simulation is run with 316 material points and uses St. Venant-Kirchoff material law. In the below graph, see figure 18, the blue and green line depicts the momentum along X-axis, and the red and yellow line depicts the momentum along Y-axis. It can be seen that the magnitude of the momentum are the same in both directions, and they overlap. The momentum is maximum before contact and starts decreasing when it comes in contact. The momentum is completely zero at maximum strain energy as the disks come to rest. Subsequently, they start moving in the opposite direction; hence the momentum inverses its direction. A similar plot is also reported in Sulsky et al. [1994].



**Figure 18:** The above graph plots the momentum of the lower and upper disk. The blue and red line represents the lower disk's momentum along X and Y direction respectively, and the green and yellow line represents the upper disk's momentum along X and Y direction respectively. The curves overlap because the values along their X and Y direction are the same. As the disks approach each other the momentum drops and becomes zero at its lowest kinetic energy. Thereafter the direction is reversed. The results are taken from a simulation using St. Venant-Kirchoff material law for 316 material points.

### 8.3.4 Von-Misses stresses

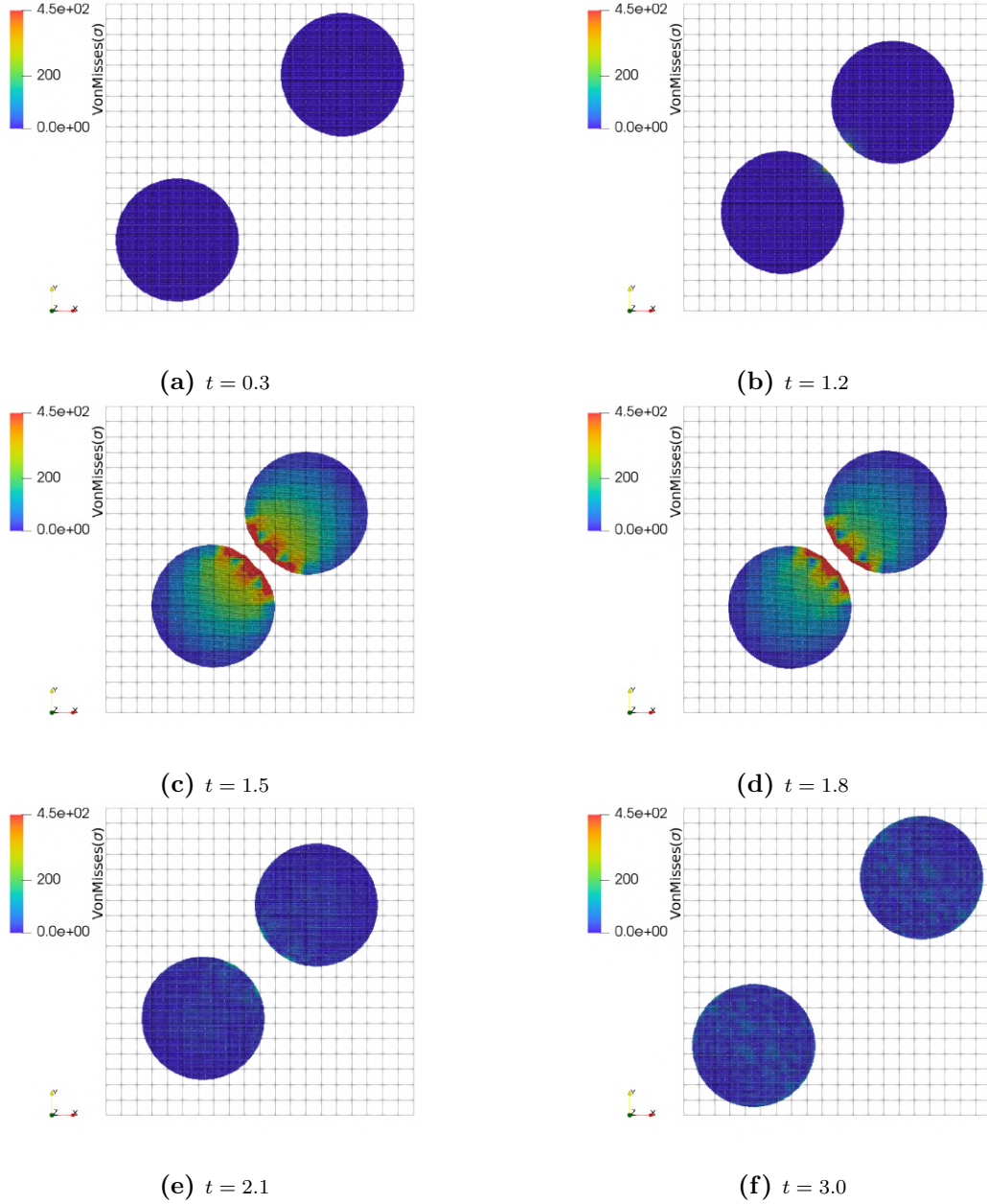
**Coarse discretisation :** The series of images shows the Von-Misses stresses developed in the disks for a coarser discretisation of 316 particles each disk, using St.Venant-Kirchoff material law. The material points are scaled to a larger size for visualisation.



**Figure 19:** The material points of the disks are scaled by a factor of 50 for visualisation. The images above show the generated Von-Misses stresses in both disks, for St.Venant-Kirchoff material law. Maximum stress values occur approximately at  $t = 1.65$ . The above images also showcases the symmetry of the simulation.

### Fine discretisation :

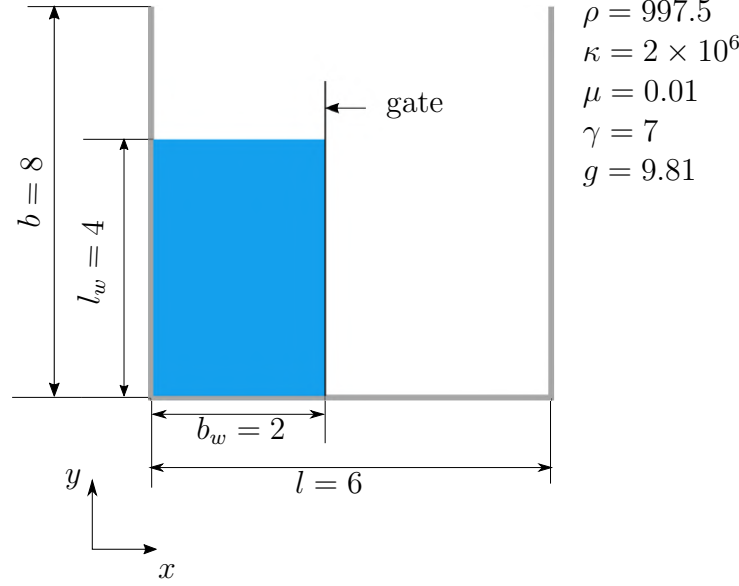
The series of images shows the Von-Misses stresses developed in the disks for a finer discretisation of 3016 particles each disk, using St.Venant-Kirchoff material law. In the below images, the stress distribution is not smooth and follows a pattern, which is similar to the mesh shape. This distribution indicates a mesh dependency. The finer discretisation of material points makes the mesh dependency more clearly visible. The material points are scaled to a larger size for visualisation.



**Figure 20:** The material points of the disks are scaled by a factor of 650 for visualisation. The images above show the generated Von-Misses stresses in both disks, for St.Venant-Kirchoff material law. Notably, the stress distribution is not smooth and follows a similar pattern of the mesh shape, which shows the mesh dependency. Maximum stress values occur approximately at  $t = 1.65$ . The above images also showcases the symmetry of the simulation.

#### 8.4 Dam with barrier bench-marking

A liquid column of length  $l_w = 4$  and breadth  $b_w = 2$  is held by a gate. The gate is removed instantaneously and the liquid flows under the influence of gravity alone with  $g = 9.81$ . A barrier is placed at a distance of  $l = 6$ , see figure 21. The fluid properties are density  $\rho = 997.5$ , bulk modulus  $\kappa = 2 \times 10^6$  and viscosity  $\mu = 0.01$ . Consider all surfaces to be smooth. The problem is referred from Mast et al. [2012].



**Figure 21:** Dam with barrier

A computational grid of size  $6 \times 8$  is chosen and discretised into  $18 \times 24$  cells. The liquid column is discretised with 19200 material points. The simulation is run till the fluid flows out of the container. A time increment of  $\Delta t = 10^{-5}$  is chosen, and non-penetration boundary conditions are applied. The damping factor  $\alpha_d$  is zero. As a parameter study, the effect of viscosity is observed by keeping other fluid properties as constant. The studied viscosity values are  $\mu = 0.01, 50, 220$ .

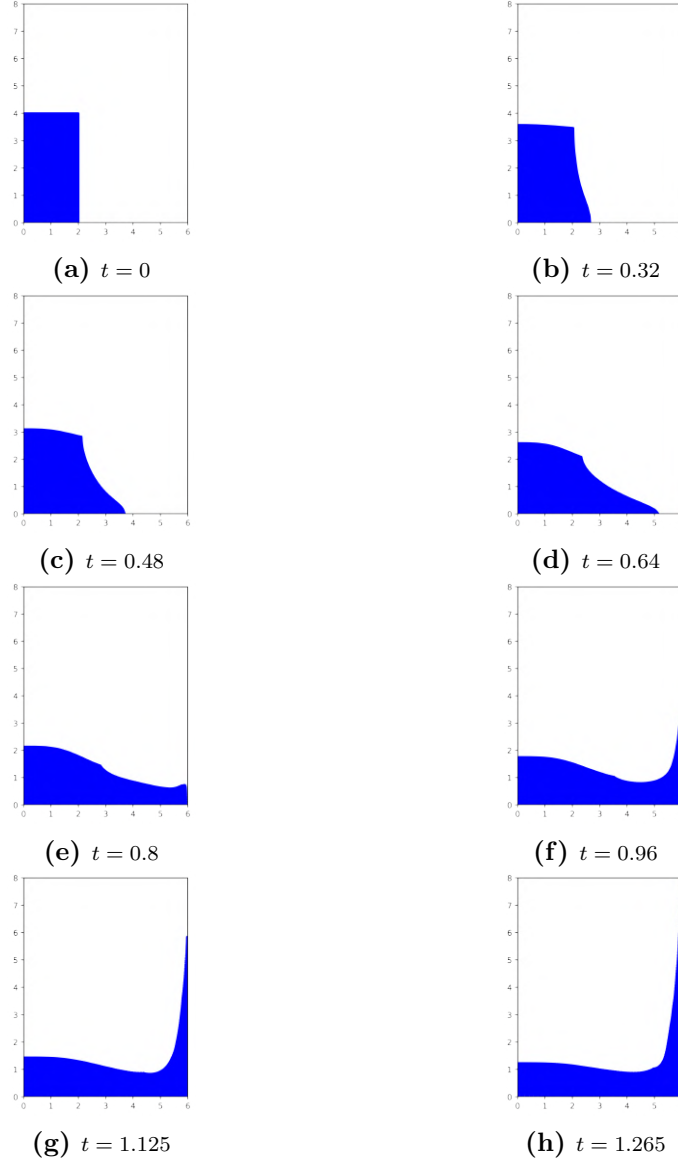
**8.4.1 Non-penetration boundary condition for fluids** The material points should flow smoothly on the surfaces but should not penetrate the surface, i.e. the velocities perpendicular to the surface are zero. Also, the material points should not be fixed to any surface. The non-penetration boundary condition serves the purpose.

$$\mathbf{v} \cdot \mathbf{n} = 0, \quad (90)$$

i.e. for a surface along the  $X$ -axis, the velocity of material points along its  $Y$  direction at the boundary is zero and vice versa. Non-penetrative boundary conditions are used for all discussed fluid mechanics problems.

**8.4.2 Effect of viscosity** Viscosity is the friction or resistance offered by the fluid to its flow. A highly viscous fluid experiences more internal friction, causing the kinetic energy to dissipate into heat and damp the fluid's motion. Thus, an increase in viscosity restrains the motion, and the same behaviour can be seen in the below three cases.

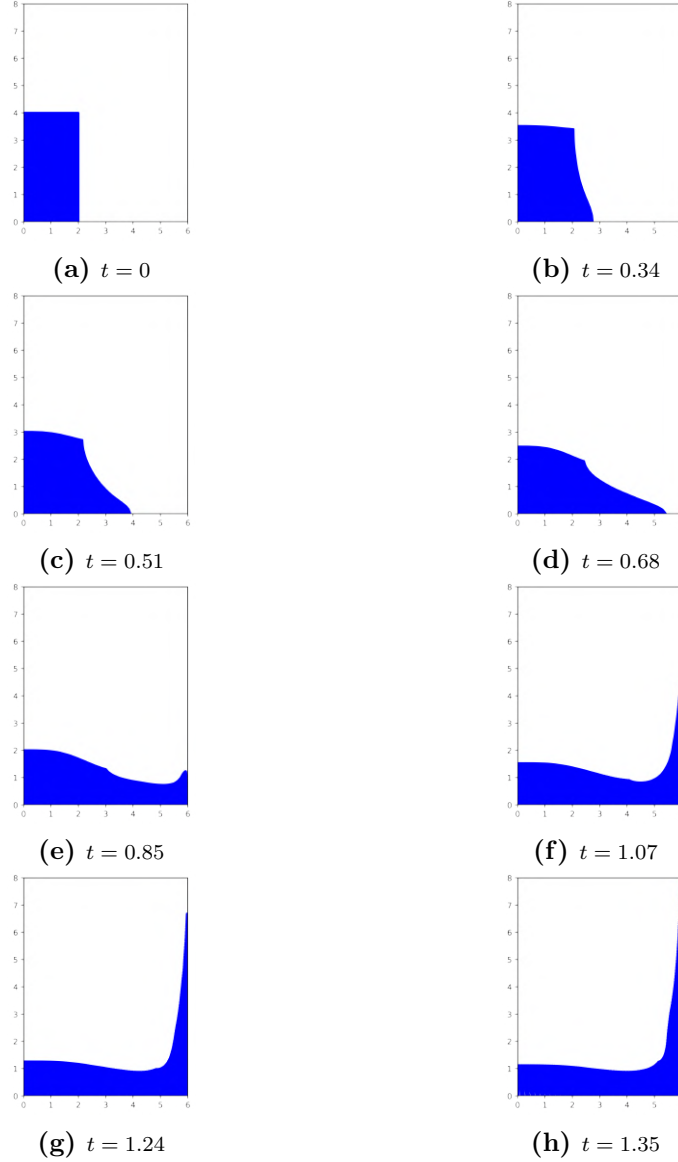
**Fluid flow for a viscosity  $\mu = 0.01$ :** The simulation runs only for a time period of  $T = 1.265$ , and after  $t = 1.265$  the fluid flows outside the computational domain. The below series of images shows the fluid flow when the barrier is lifted. The material points are scaled to a larger size for visualisation purpose.



**Figure 22:** When the gate is removed with the effect of gravity the fluid starts to flow. Because of the implemented non-penetration boundary conditions, the fluid creeps above the barrier, and escapes the grid at  $t = 1.265$ . The material points are scaled to a larger size for visualisation.

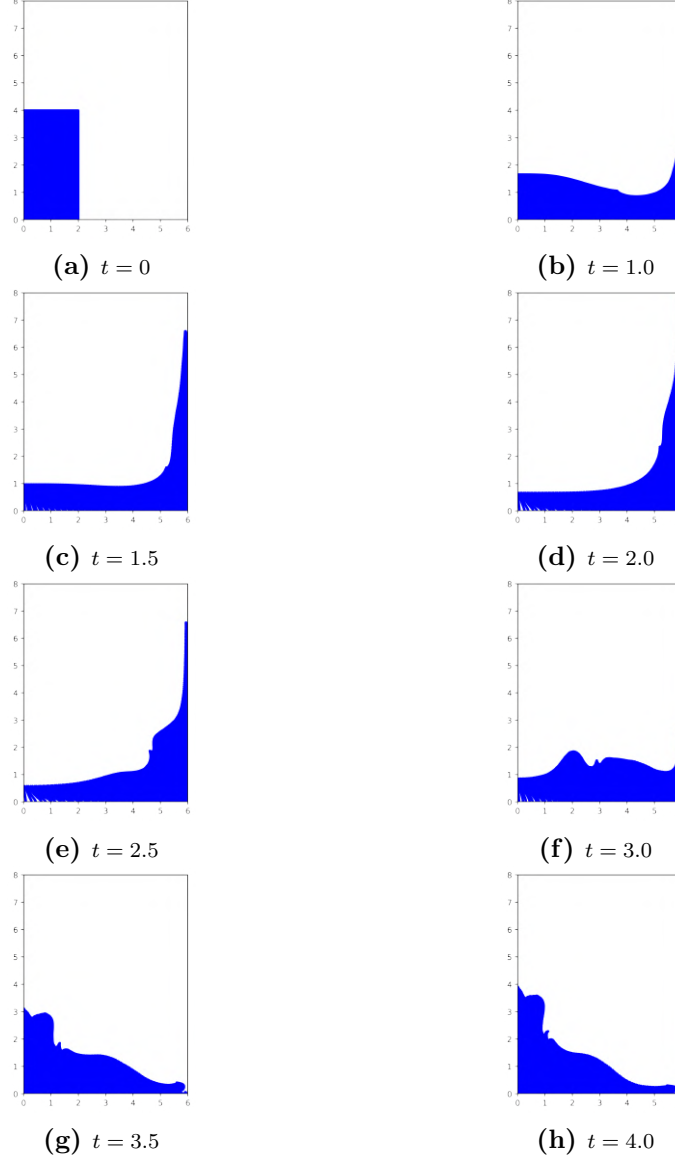


**Fluid flow for a viscosity  $\mu = 50$ :** The simulation runs only for a time period of  $T = 1.35$ , and after  $t = 1.35$  the fluid flows outside the computational domain. The below series of images shows the fluid flow when the barrier is lifted. The material points are scaled to a larger size for visualisation purpose.



**Figure 23:** When the gate is removed with the effect of gravity the fluid starts to flow. Because of the implemented non-penetration boundary conditions, the fluid creeps above the barrier and escapes the grid at  $t = 1.35$ . The material points are scaled to a larger size for visualisation.

**Fluid flow for a viscosity  $\mu = 220$ :** Because of the high viscosity value, the fluid does not escape the grid. The simulation is let to run for a period of  $T = 4$ . The below series of images shows the fluid flow when the barrier is lifted. The material points are scaled to a larger size for visualisation purpose.



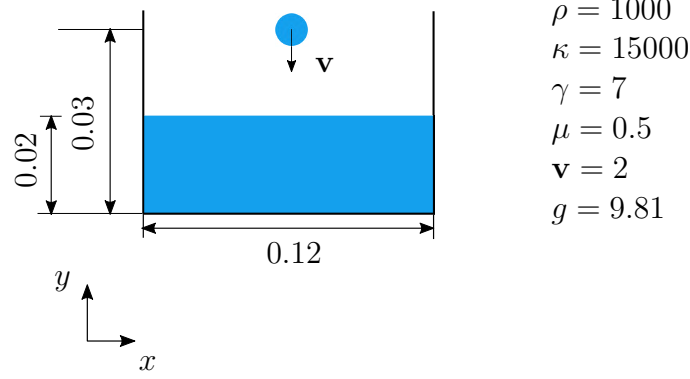
**Figure 24:** When the gate is removed with effect of gravity the fluid starts to flow. Since the fluid is highly viscous, the fluid rolls back and stays inside the domain. The material points are scaled to a larger size for visualisation.

**Observation and inference :** For  $\mu = 0.01$ , the fluid flow is smooth and with less damping. Hence the friction offered along the wall to the fluid flow is insufficient to dampen its motion completely and allowing the fluid to exit earlier at  $t = 1.27$ . Increasing the viscosity to  $\mu = 50$  increases the damping rate and holds the fluid for a slightly longer time till  $t = 1.35$ . With a higher viscosity of  $\mu = 220$ , the barrier's height entirely damps the fluid at approximate  $t = 2$ . Then with the influence of gravity, the fluid falls and oscillates till the kinetic energy dissipates.



### 8.5 Impact of fluids bench-marking

A liquid drop of radius 0.005 falls vertically with a velocity of  $\mathbf{v} = 2$  from a height of 0.03 into a tank of breadth 0.12. The tank holds a liquid mass of height 0.02 and is initially at rest, see figure 25. The drop and the liquid mass share the same fluid properties with density  $\rho = 1000$ , bulk modulus  $\kappa = 15000$ , viscosity  $\mu = 0.5$ . Consider all surfaces to be smooth and gravitational force  $g = 9.81$ . The problem is referred from Cueto-Fergueroso et al. [2004].

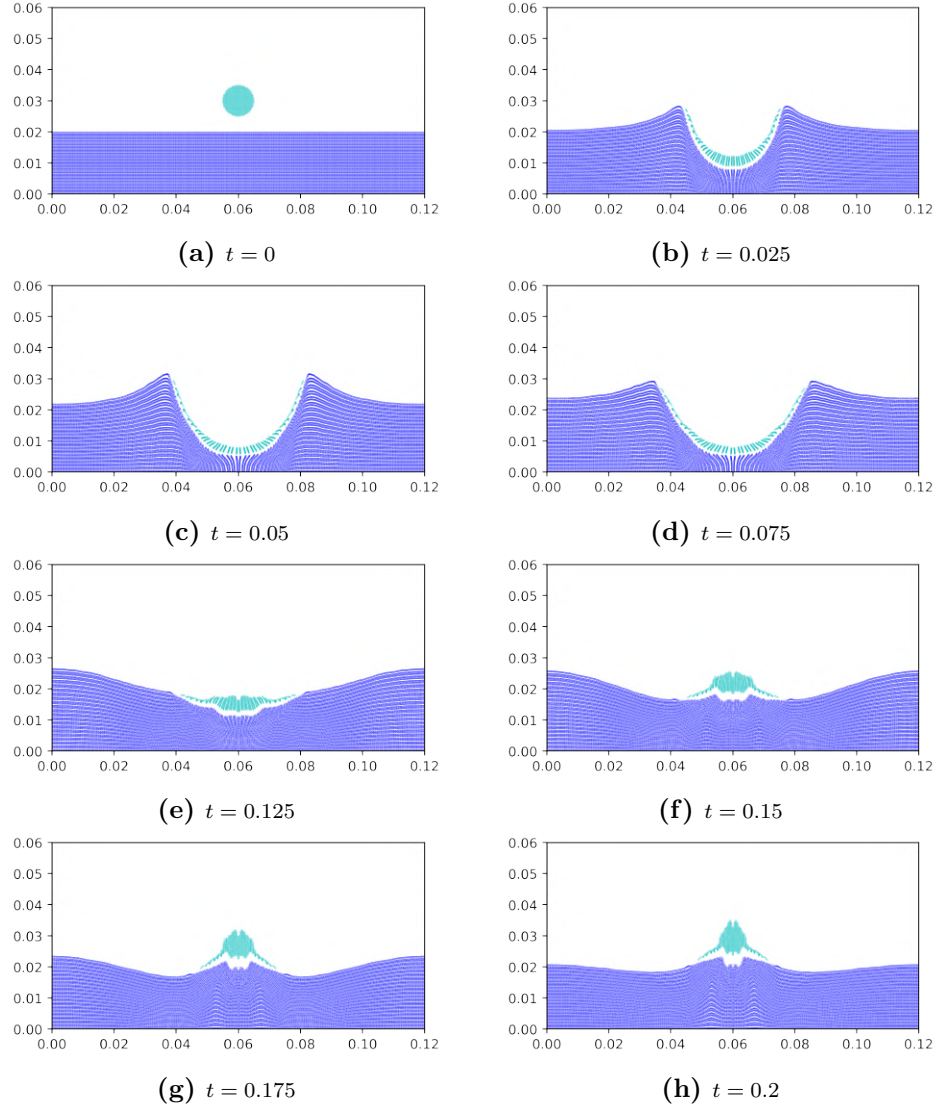


**Figure 25:** Impact of fluids

The computational grid is of size  $0.12 \times 0.06$  and is discretised into  $120 \times 10$  cells. The liquid drop is discretised with 1256 material points and the liquid mass in the tank with 19200 material points. The simulation is run for a period of  $T = 0.2$  with a time increment of  $\Delta t = 10^{-6}$ , and non-penetration boundary conditions enforced. The damping factor  $\alpha_d$  is zero. Furthermore, the effect of bulk modulus is studied by keeping other fluid parameters as constant. The chosen bulk modulus values are  $\kappa = 15000, 30000, 60000$ .

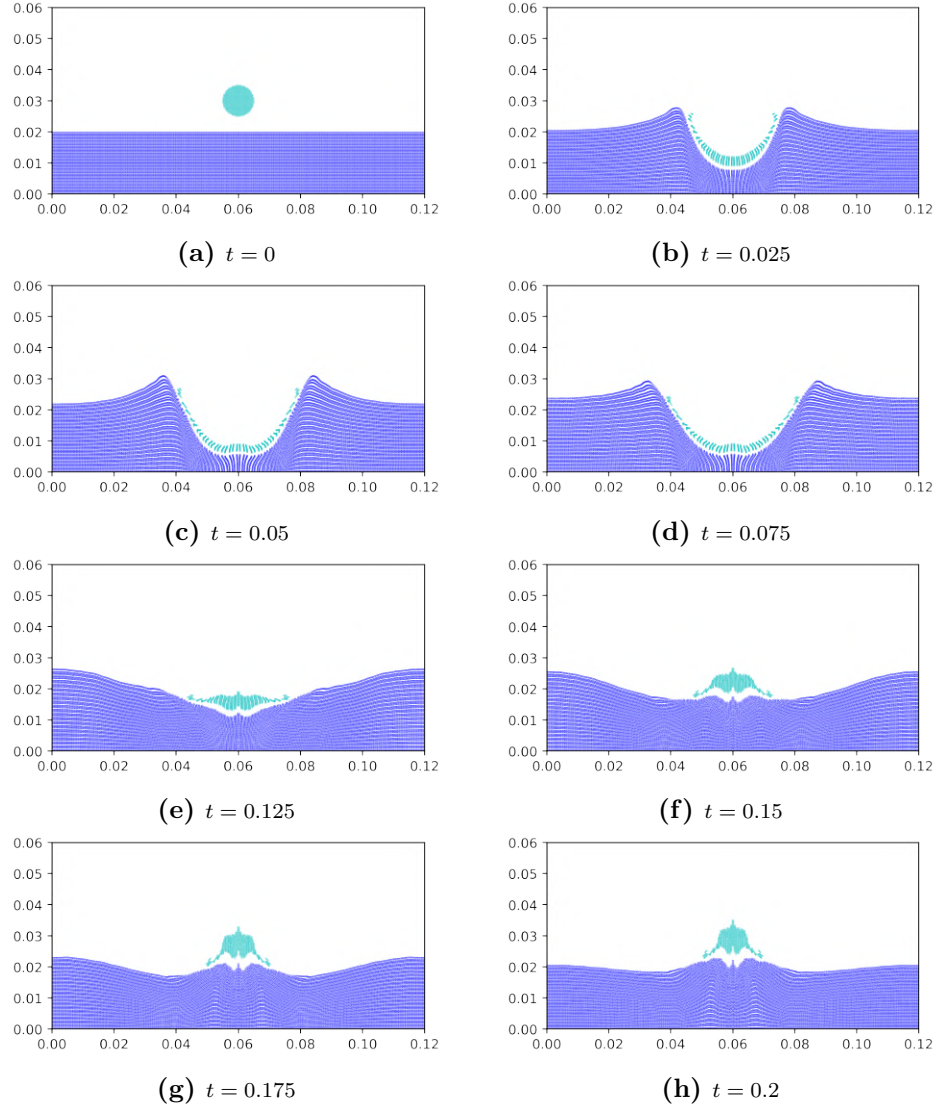
**Effect of bulk modulus :** Bulk modulus is the resistance offered by a body to compression. Hence, an increase in the bulk modulus makes the fluid more incompressible and resilient to deformation, which is evident in this case study.

**Deformation of fluids for  $\kappa = 15000$ :** The following series of images show the impact of fluids with material points scaled to a larger size for visualisation purpose. Shades of blue are used to differentiate the material points of the drop and liquid mass.



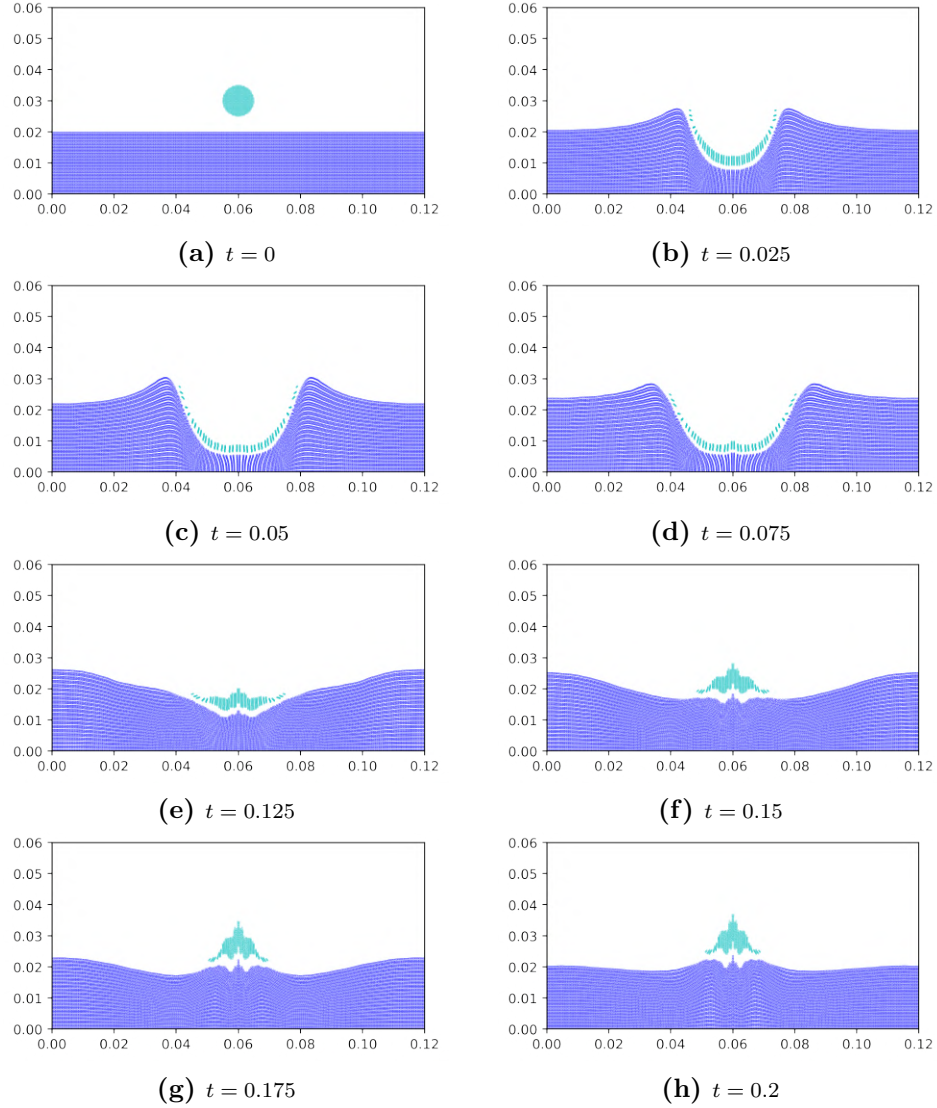
**Figure 26:** The series of images show the impact of fluids for  $\kappa = 15000$ . The drop pierces the liquid mass with high momentum, and in turn, the stresses generated by the liquid mass offers a reaction force and pushes the drop upwards. The material points are scaled to a larger size for visualisation.

**Deformation of fluids for  $\kappa = 30000$ :** The following series of images show the impact of fluids with material points scaled to a larger size for visualisation purpose. Shades of blue are used to differentiate the material points of the drop and liquid mass.



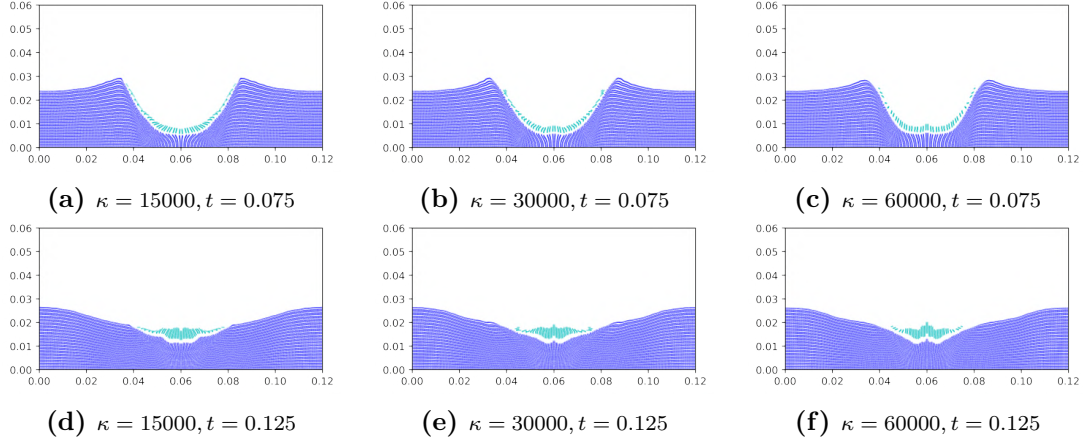
**Figure 27:** The series of images show the impact of fluids for  $\kappa = 30000$ . The drop pierces the liquid mass with high momentum, and in turn, the stresses generated by the liquid mass offers a reaction force and pushes the drop upwards. The material points are scaled to a larger size for visualisation.

**Deformation of fluids for  $\kappa = 60000$ :** The following series of images show the impact of fluids with material points scaled to a larger size for visualisation purpose. Shades of blue are used to differentiate the material points of the drop and liquid mass.



**Figure 28:** The series of images show the impact of fluids for  $\kappa = 60000$ . The drop pierces the liquid mass with high momentum, and in turn, the stresses generated by the liquid mass offers a reaction force and pushes the drop upwards. The material points are scaled to a larger size for visualisation.

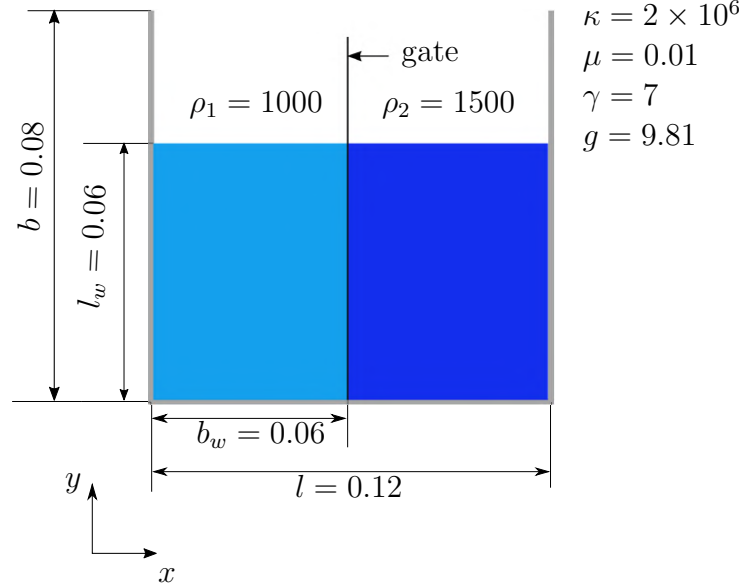
**Observation and inference :** For a better comparison of the results, images at time  $t = 0.075$  and  $t = 0.125$  are arranged in figure 29. The first row of images shows the impact at  $t = 0.075$ , and the second row shows the impact at  $t = 0.125$ , with increasing bulk modulus of the order 15000, 30000 and 60000. In the first row, the drop pierces the liquid mass, for which it offers a reaction force. With an increase in the fluids' bulk modulus, their resistance to compression increases significantly. Thus, the deformation decreases and correspondingly, the deformation shape also differs in each case. The same pattern is as well as observed in the second row of images. In the referred literature Cueto-Ferguson et al. [2004], the re-bounce of the water drop is not reported, but it has been included in this thesis.



**Figure 29:** The above images show a comparison of the deformation of fluids with different bulk modulus. The material points are scaled to a larger size for visualisation. The first row shows the deformation at  $t = 0.075$ , and the second row shows the deformation at  $t = 0.125$  with an increasing bulk modulus. The deformation of the bodies are varied and minimised with increasing  $\kappa$  values.

## 8.6 Gravity currents bench-marking

Two fluids of different densities,  $\rho_1 = 1000$  and  $\rho_2 = 1500$ , are held within a container of length  $l = 0.12$  and breadth  $b = 0.08$  and separated by a wall, see figure 30. The length and breadth of the fluid columns are  $l_w = 0.06$  and  $b_w = 0.06$ . The gate is removed instantaneously, and the fluids are allowed to flow into each other, leading to gravity currents. The fluids share the same bulk modulus  $\kappa = 2 \times 10^4$  and viscosity  $\mu = 0.01$ . The problem is referred from Cueto-Ferguson et al. [2004].

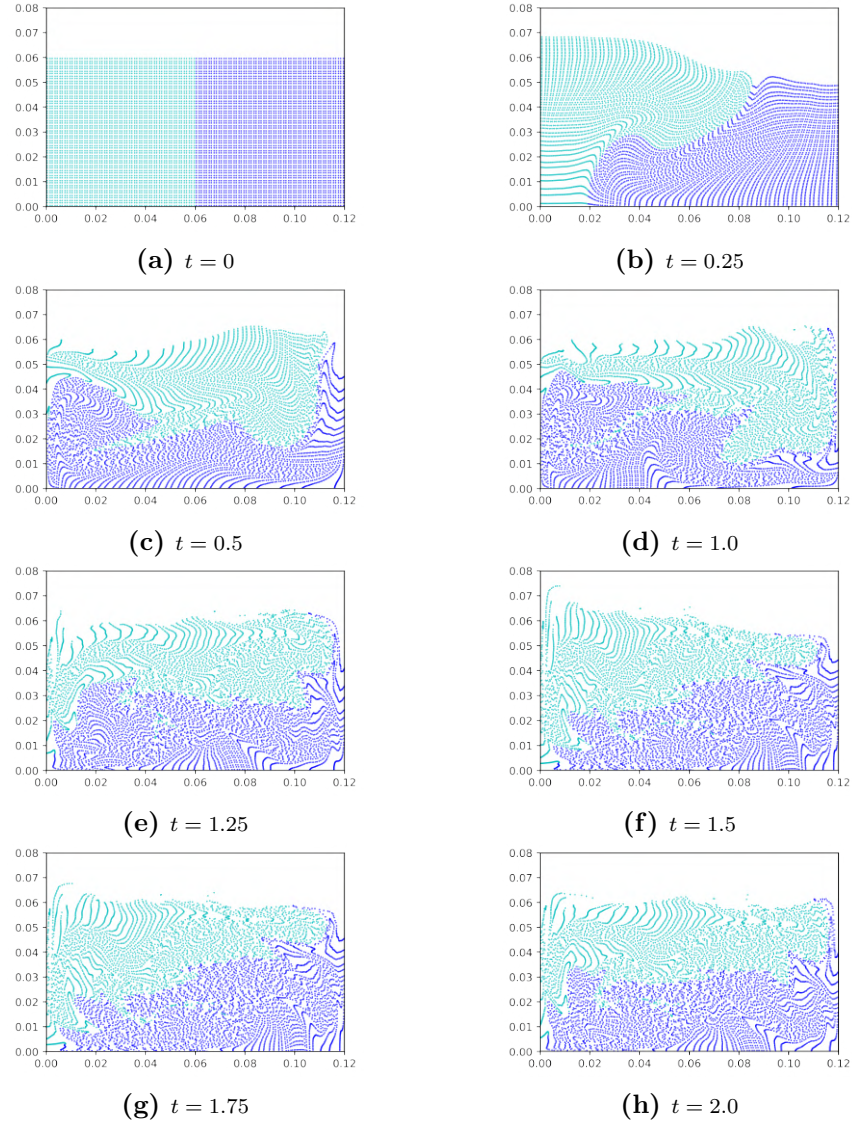


**Figure 30:** Gravity Currents

The computational domain is of length  $0.12 \times 0.08$  and discretised into  $24 \times 16$  cells. Each fluid column is discretised with 3600 material points. The simulation is run for a period of  $T = 2$  with a time increment  $\Delta t = 10^{-6}$ , and non-penetration boundary condition enforced. The damping factor  $\alpha_d$  is zero.

**Displacement of fluids :** The below images shows the flow of fluids into each other as soon as the gate is removed, see figure 31. The lighter blue shade represents the fluid with low density, and the darker blue shade represents the high-density fluid. Because of the density difference, less dense material points float on top of the high-density material points. The material points are scaled to a larger size for visualisation.





**Figure 31:** The above set of images show the displacement of fluids when the gate is removed instantaneously. Because of the difference in densities, the low-density fluid depicted by a lighter blue shade floats over the dark shaded high-density fluid. The material points are scaled to a larger size for visualisation.

**Observation and inference :** Density is the amount of matter contained in a body and is directly proportional to its mass. Hence a fluid with a high density stays lower to the surface, and a low-density fluid floats on its top. The same phenomenon is observed in the above example. The fluid with  $\rho_1 = 1000$  floats on top at the end of the simulation. The example also showcases a fluid-fluid contact. At the start, the material points closer to the gate share the same velocity fields. As the simulation proceeds, whenever there is a collision between two material points, the fluid-fluid contact is inevitable. A similar flow is also reported in Cueto-Fergueroso et al. [2004].

### 8.7 Float - a FSI example

In this section, the behaviour of a simple fluid-structure interaction (FSI) in MPM is discussed. The problem is self-developed to study the fluid-structure interaction in MPM and is not referred to from any literature. The density of the solid and fluid are the densities of ice and water. Hence, the problem is a recreation of ice floating on water. Lower values are chosen for other material parameters, particularly for the Young's modulus and bulk modulus, to avoid numerical instability in the simulation. Hence, the fluid behaves more compressible.

A solid body resembling an equilateral triangle of side 0.2 is rested on top of a fluid surface. The material properties of the solid are Young's modulus  $E = 2 \times 10^5$ , Density  $\rho_s = 917$  and Poisson's ratio  $\nu = 0.3$ . The fluid has a bulk modulus  $\kappa = 2 \times 10^3$ , density  $\rho_f = 997$  and viscosity  $\mu = 0.98$ . It is inside a container of dimensions,  $0.4 \times 0.4$ ; see figure 32. Consider gravity as  $g = 9.81$ .

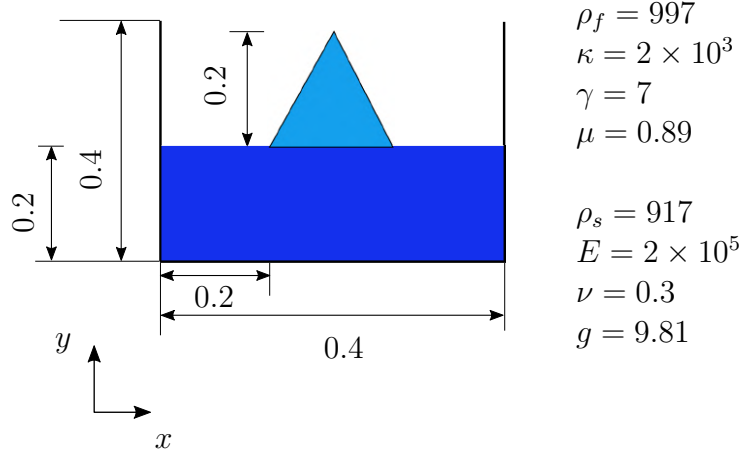
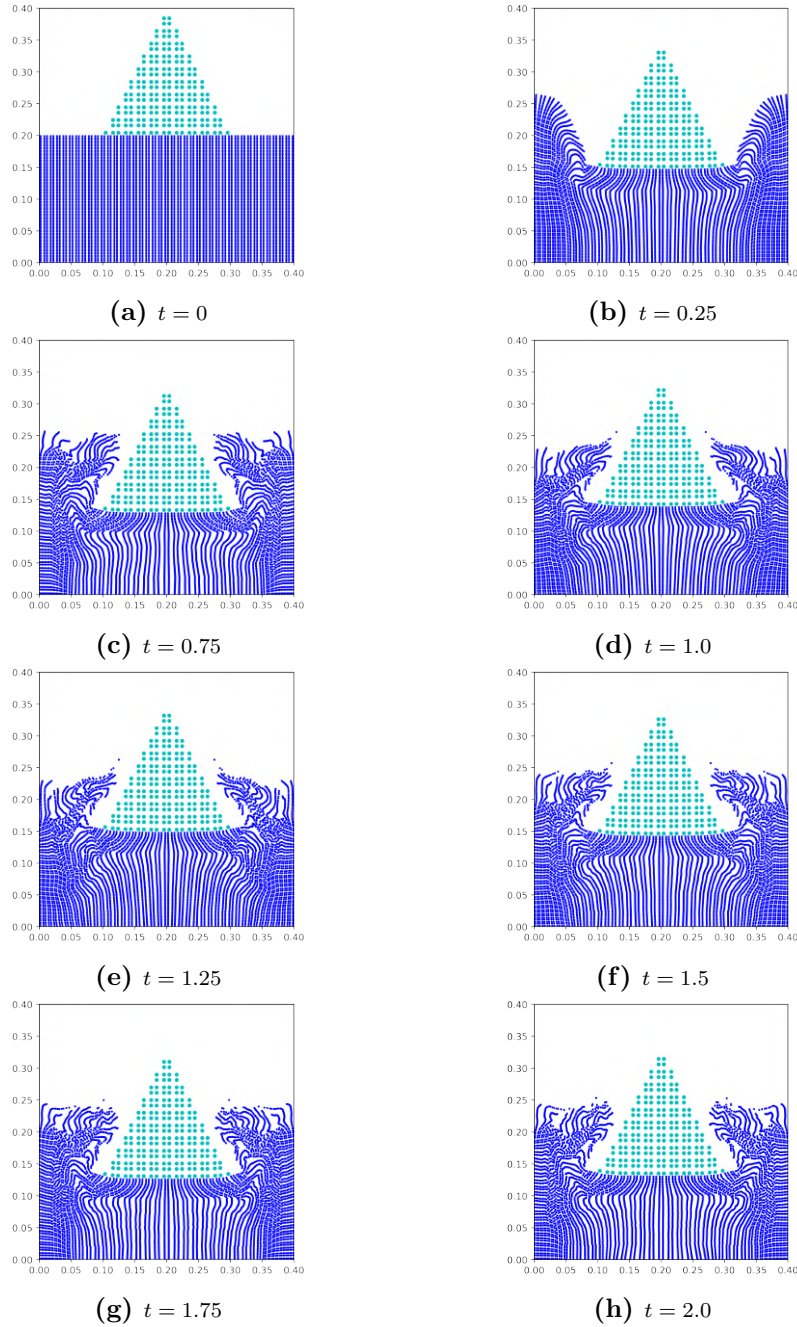


Figure 32: Float

The computational grid is of size  $0.4 \times 0.4$  with  $15 \times 15$  cells. The solid is discretised into 200 material points and the fluid into 6400 material points. The simulation is run for a period  $T = 2$  with a time increment  $\Delta t = 10^{-6}$ , and non-penetration boundary conditions enforced. The damping factor  $\alpha_d$  is zero.

**Displacement of the fluid :** The below images shows the displacement of fluid due to the mass of the solid block. The material points are scaled to a larger size for visualisation. The block floats on the surface because it is less dense than the fluid and displaces an equivalent volume of fluid.





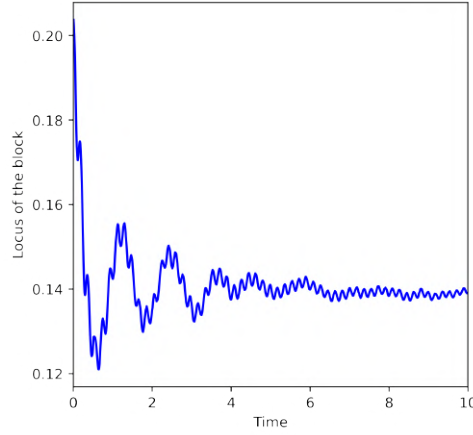
**Figure 33:** The above series of images shows the displacement of the fluid. The solid floats because of the difference in densities, displacing an equivalent volume of the fluid. The material points are scaled to a larger size for visualisation.

### Observation and inference :

The scenes plotted above does not precisely reproduce physical behaviour. In a real case, the triangular block would sink one-third of its height, raising the water level by a factor of its weight. Instead, in the simulation, the block displaces a volume of the fluid and water splashes around the block. For the fluid to rise and the block to sink, their relative velocities should be in the opposite direction, i.e. they should have two velocity fields.

The possibility of having different velocity fields can be achieved by modifying the MPM code, and because of its unavailability, the behaviour is partially off from physical. Also, the grid dependency can be noticed around the block as the fluid deforms relatedly to the deformed mesh.

Further, the simulation is extended till  $T = 10$ , and the locus of a material point at the bottom row is tracked, see figure 34. The locus potentially describes the motion of the block. The block compresses the fluid and rises because of the fluids reaction force, oscillating the block up and down. When the momentum is transferred completely between the bodies, the block comes to rest. Still, ripples of displacement are seen due to insufficient damping caused by coarse discretisation.

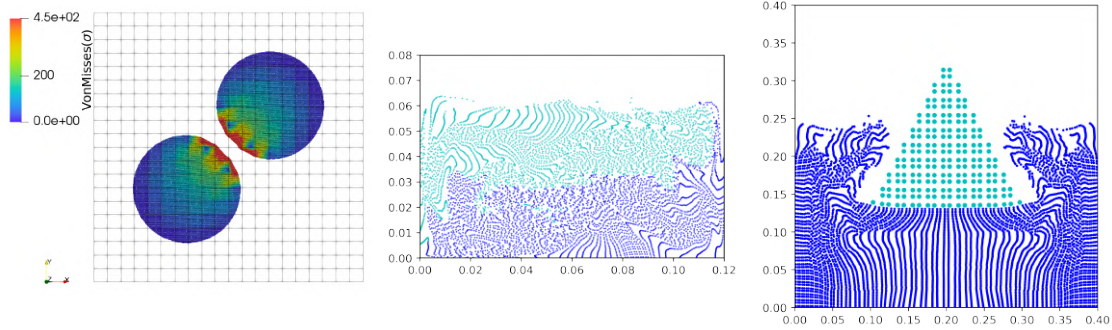


**Figure 34:** Time vs. Displacement plot of the block. The above plot tracks the motion of the block. The block oscillates until the momentum is transferred completely. The ripples in the locus is because of a coarse discretisation, causing an insufficient damping.

## 9 Conclusion

The MPM is a numerical method and is well suited for problems involving large deformation and contacts. It is also similar to FEM in terms of solving the Galerkin weak-form of the balance of momentum. The material points in the computational grid share the velocity field, making contact of bodies essential in MPM. Enforcing boundary conditions is relatively easy by controlling the nodal momentum values and arresting them to zero or presetting to any desired value. Since the body's total mass is distributed evenly to all material points, mass is self conserved. Conservation of momentum is not natural in MPM. However, the total momentum is conserved in the above-given examples, evident from the symmetry in obtained results. Unlike other numerical methods, one should always consider the density of discretisation in MPM because of the relation between the number of material points and cells.

MPM is a well-documented method. Nevertheless, this thesis gives an overview of conventional MPM and its abilities such as large deformation, solid-solid contact, fluid-fluid contact, and fluid-structure interaction. A few images of the results are shown in figure 35. In this thesis, an MPM solver has been developed in C++, and an API is built in Python.



**Figure 35:** The above images depicts the ability of MPM to handle solid-solid contact, fluid-fluid contact, and fluid-structure interaction, respectively.

Few pros of the implemented MPM are it does not experience mesh distortion, and it can handle both solids and fluids. Enforcing boundary condition is also comparatively simple. On the other hand, since Velocity-Verlet is an explicit-time method, it suffers from numerical instability. The computational cost for MPM is higher because of its additional step to map and remap material points when compared e.g. with FEM. An inappropriate density of material points per cells leads to unphysical behaviour like extension instability. Convergence study in MPM is also not well defined.

Further study could be made on the relation between the material points and cells, convergence, and no-slip condition, which would make this tool more complete and stable. Also, the code can be modified to store history for each material point, because the implemented code does not store history variables.

## References

- S. Bardebhagen and E.M. Kober. The generalized interpolation material point method. *Computer Modeling in Engineering and Sciences*, 5(6):477–495, 2004.
- K. J. Bathe. *Finite element procedures*. Prentice Hall, Pearson Education, Inc., 2006.
- L. Beuth, Z. Wieękowski, and P.A. Vermeer. Solution of quasi-static large-strain problems by the material point method. *International Journal for Numerical and Analytical Methods in Geomechanics*, 35(13):1451–1465, 2011.
- J. Bluhm. Continuum mechanics. *Institut für Mechanik Fakultät Ingenieurwissenschaften Abteilung Bauwissenschaften Universität Duisburg Essen*, 2015.
- J.U. Brackbill and H.M Ruppel. Flip: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational Physics*, 65(2):341–343, 1986.
- O. Buzzi, D.M. Pedroso, and A. Giacomini. Caveats on the implementation of the generalized material point method. *Computer Modeling in Engineering and Sciences*, 1(1):1–21, 2008.
- L. Cueto-Ferguson, I. Colominas, G. Mosqueira, F. Navarrina, and M. Casteleiro. On the galerkin formulation of the smoothed particle hydrodynamics method. *International Journal for Numerical Methods in Engineering*, 60(9):1475–1512, 2004.
- S. Cummins and J. Brackbill. An implicit particle-in-cell method for granular materials. *Journal of Computational Physics*, 180(2):506–548, 2002.
- A. de Vaucorbeil, V. P. Nguyen, S. Sinaie, and J. Y. Wu. Material point method after 25 years: theory, implementation and applications. *Submitted to Advances in Applied Mechanics*, 1, 2019.
- Y. Gan, Z. Sun, Z. Chen, X. Zhang, and Y. Liu. Enhancement of the material point method using b-spline basis functions. *International Journal for Numerical Methods in Engineering*, 113(3):411–431, 2018.
- J.E. Guilkey, J.B. Hoving, and J.A. Weiss. Computational modeling of multicellular constructs with the material point method. *Journal of Biomechanics*, 39(11):2074–2086, 2006.
- F.H. Harlow. The particle-in-cell computing method for fluid dynamics. *Methods for Computational Physics*, 3:319–343, 1964.
- F.H. Harlow and J.E. Welch. Numerical calculation of time dependent viscous incompressible flow of fluid with a free surface. *Physics of Fluids*, 8:2182–2189, 1965.
- P. Haupt. *Continuum mechanics and theory of materials*. Springer Science & Business Media, 2013.

- 
- G. A. Holzapfel. *Nonlinear Solid Mechanics: A Continuum Approach for Engineering*. Wiley & Sons, New York, 2000.
- T. J. Hughes. *The finite element method: linear static and dynamic finite element analysis*. Courier Corporation, 2012.
- I. Jassim, D. Stolle, and P.A. Vermeer. Two-phase dynamic analysis by material point method. *International Journal for Numerical and Analytical Methods in Geomechanics*, 37(15):2502–2522, 2013.
- N.L. Johnson. The legacy and future of cfd at los alamos. *1996 Canadian CFD conference*, 1996.
- J. Korelc and P. Wriggers. *Automation of Finite Element Methods*. Springer, 2016.
- J. Li, Y. Hamamoto, Y. Liu, and X. Zhang. Sloshing impact simulation with material point method and its experimental validations. *Computers and Fluids*, 103:86–99, 2014.
- T. Li, R.M. Horton, F. Bader, D.A. and Liu, Q. Sun, and P.L. Kinney. Long-term projections of temperature-related mortality risks for ischemic stroke, hemorrhagic stroke, and acute ischemic heart disease under changing climate in beijing, china. *Environment International*, 73:185–204, 2018.
- Y.P. Lian, X. Zhang, and Y. Liu. An adaptive finite element material point method and its application in extreme deformation problems. *Computer Methods in Applied Mechanics and Engineering*, 241:275–285, 2012.
- S. Ma, X. Zhang, Y. Lian, and X. Zhou. Simulation of high explosive explosion using adaptive material point method. *Computer Modeling in Engineering and Sciences (CMES)*, 39(2):101, 2009.
- C. Mast, P. Mackenzie-Helnwein, Arduino.P, G. Miller, and W. Shin. Mitigating kinematic locking in the material point method. *Journal of Computational Physics*, 231(16):5351–5373, 2012.
- S. Mckee, M.F. Tomé, V.G. Ferreira, J.A. Cuminato, A. Castelo, F.S. Sousa, and N. Mangiacchi. The mac method. *Computers and Fluids*, 37(8):907–930, 2008.
- A. Sadeghirad, R.M. Brannon, and J. Burghardt. A convected particle domain interpolation technique to extend applicability of the material point method for problems involving massive deformations. *International Journal for Numerical Methods in Engineering*, 86(12):1435–1456, 2011.
- A. Sadeghirad, R.M. Brannon, and J.E. Guilkey. Second-order convected particle domain interpolation (cpdi2) with enrichment for weak discontinuities at material interfaces. *International Journal for Numerical Methods in Engineering*, 95(11):928–952, 2013.
- M. Steffen, P. Wallstedt, R. Guilkey, R. Kirby, and M. Berzins. Examination and analysis of implementation choices within the material point method (mpm). *Computer Modeling in Engineering and Sciences*, 31(2):107–127, 2008.

- 
- A. Stomakhin, C. Schröder, C. Jiang, L. Chai, J. Teran, and A. Selle. Augmented mpm for phase-change and varied materials. *ACM Transactions on Graphics*, 33(4):1–11, 2014.
- D. Sulsky and A. Kaul. Implicit dynamics in the material-point method. *Computer Methods in Applied Mechanics and Engineering*, 193(12-14):1137–1170, 2004.
- D Sulsky, Z. Chen, and H.L. Schreyer. A particle method for history-dependent materials. *Computer Methods in Applied Mechanics and Engineering*, 118(1-2):179–196, 1994.
- R. Tielen, E. Wobbes, M. Moller, and L. Beuth. A high order material point method. *Procedia Engineering*, 175:265–272, 2017.
- L.T. Tran, J. Kim, and M. Berzins. Solving time-dependent pdes using the material point method, a case study from gas dynamics. *International Journal for Numerical Methods in Fluids*, 62(7):709–732, 2010.
- B. Wang, P.J. Vardon, M.A Hicks, and Z. Chen. Development of an implicit material point method for geotechnical applications. *Computers and Geotechnics*, 71:159–167, 2016.
- Z. Więkowski. The material point method in large strain engineering problems. *Computer Methods in Applied Mechanics and Engineering*, 193(39-41):4417–4438, 2004.
- Z. Więkowski, S.K. Youn, and J.H. Yeon. A particle-in-cell solution to the silo discharging problem. *International Journal For Numerical Methods in Engineering*, 45:1203–1225, 1999.
- A. York, D. Sulsky, and H. Schreyer. The material point method for simulation of thin membranes. *International Journal for Numerical Methods in Engineering*, 44(10):4129–1456, 1999.
- A. York, D. Sulsky, and H. Schreyer. Fluid-membrane interaction based on the material point method. *International Journal for Numerical Methods in Engineering*, 231(16): 901–924, 2000.
- X. Zhang, Z. Chen, and Y. Liu. The material point method: a continuum-based particle method for extreme loading cases. *Elsevier Science*, 2016.