AN ABSTRACT OF THE THESIS OF

Alexander Clucas for the degree of Master of Science in Computer Science presented on June 14, 2016.

Title: Rendering and Simulation for Tires Rolling on Snow

Abstract approved: _____

Yue Zhang

Modeling tire-snow interaction is important in designing effective snow tires, which directly affects road safety during wintry weather. Unfortunately, tires have complex tread designs and the physical properties of snow have not been characterized. We employ the Material Point Method(MPM) for simulating a material that mimics the fracturing and plasticity of snow using established physical models. This realistic simulation allows us to assess the interactions between a breakable material like snow and a continuum structure like rubber tires, and determine the effect of tread patterns on tire traction. We also design a novel snow rendering technique for particle simulation using a combination of surface reconstruction, surface instanced geometry and sub-surface scattering.

Rendering and Simulation for Tires Rolling on Snow

by

Alexander Clucas

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented June 14, 2016
Commencement June 2017

Master of Science thesis of Alexander Clucas presented on June 14, 2016.

APPROVED:

Major Professor, representing Computer Science

Director of the School of Electrical Engineering and Computer Science

---

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Alexander Clucas, Author

ACKNOWLEDGEMENTS

Foremost, I would especially like to thank my advisor Dr. Yue Zhang for all of the guidance and help throughout this project. I would like to thank the rest of my thesis committee, Dr. Eugene Zhang, Dr. Mike Bailey and Dr. Kyle Niemeyer. I would also like to thank the other students who helped me during this process, Prathik Sannecy and Paris Kalathas. I would like to thank Dr. John Nairn for his help with the simulation software, and Patrick Neill who has helped me meet my advisor, get an internship, and a fulltime job. Finally, I would like to thank my aunt Jean who introduced me to the world of computer graphics.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF APPENDIX FIGURES

To my fiancée Jordan, for helping me and supporting me every step of the way, and to my parents for their love.

Chapter 1: Introduction

Snow tires are important for creating safe driving during wintry weather. Snow and ice on roads can cause accidents, injury and death. On average, snow and ice kills over 1500 people in the United States each year. The Federal Highway Administration produced Table 1.1 to illustrate the dangers of driving during wintry weather. Driving with snow tires help maintain driver control and significantly reduce the number of accidents. Before snow tires are created, the tread pattern needs to be designed to maintain traction on snow. To design the tread pattern, the designer needs to be able to know what type of patterns work best. There are three options: go with the established design, create a new design and physically build and test a new tire with it, or simulate it.

Road Condition	**Statistic**
Snow\ Sleet	210,341 crashes
	55,942 persons injured
	739 persons killed
Icy Pavement	151,944 crashes
	38,770 persons injured
	559 persons killed
Snow\ Slushy Pavement	174,446 crashes
	41,597 persons injured
	538 persons killed

Table 1.1: Annual Average Weather-Related Crash Statistics from the Federal Highway Administration [1]. (2005-2014)

The problem with simulating a tire interacting with snow is that characterizing snow properties is difficult. Snow is a complicated natural material that is challenging to simulate correctly. The composition of snow is a microstructure of loosely connected ice crystals. The microstructure is not a uniform lattice, but a consistently varying structure based on atmospheric conditions during its formation and lifetime. Current temperature and past temperatures also greatly affect snow structure and strength. The crystals may start to melt together and reduce the internal air gaps, and then refreeze to make bigger

Figure 1.1: Typical Snow Tire, from www.alamjaadlabs.com/en/laboratories/view/9.

ice crystals. Ice crystals also increase in hardness the colder they get. When it snows on a road, there may also be foreign materials such as gravel or sand that mix in with the snow. There are a lot of variables regarding accurate snow behavior.

While it is difficult to model snow, work has been done evaluating specific phenomena of snow, such as accumulation, drift, and avalanches. These simulations frequently use particle based physical simulation to mimic the fragile fracturing of the ice microstructure, but this fails to model the compression of snow accurately. The focus of this thesis is to explore and evaluate the use of the Material Point Method for simulating tire-snow interactions.

The ability to simulate snow accurately is important, but being able to convince people that a tire pattern is effective, when they do not know about physical simulation, requires a high quality visualization to go with it. Rendering snow accurately is also complicated for many of the same reasons it is hard to simulate. Not knowing the internal structure of the snow makes it hard to accurately render the internal light scattering.

Also, converting the results of a particle based simulation into a coherent material that appears like snow is a challenge because of the size of small crystals that are smaller than each simulation particle.

This thesis presents a model for physically convincing snow and a proof-of-concept system to test tire-snow interactions. We also present a novel technique for rendering particle based snow simulations.

Chapter 2: Background

This chapter covers background information that is important to understand the work that follows. While this research touches many deep research areas, I focus on covering some introductory information for snow theory, snow simulation history, the material point method and rendering methods that pertain to snow.

2.1 Physics and Snow

Before discussing how snow is simulated and rendered, some details on solid mechanics and snow is explained below.

The two types of solid materials we focus on are elastic materials and plastic materials. Elastic materials are defined as solid objects that will return to its initial shape when deformed due to force. Purely elastic materials are very rare in the real world, and most are defined with an elastic limit property that states the maximum stress that the material behaves elastically. Beyond this point, the material will return to a permanently deformed shape instead of the original shape. This irreversible deformation defines plasticity. Perfect plasticity is where the material does not experience any change in physical qualities. While near perfect plasticity may be common, some materials get harder to deform the more it deforms. This phenomenon is called strain hardening.

Physical qualities of materials are defined by a variety of different measurements of their behavior. Homogeneous isotropic materials have their elastic properties determined by a set of moduli. The moduli include the Bulk modulus(K), Young's modulus(E), the Shear modulus(G), and Poisson's ratio(v). The bulk modulus is the material's resistance to uniform compression, as given by $K = -V\frac{dP}{dV}$ where V is volume and P is pressure. The Young's modulus, also known as the elastic modulus, is the relationship between stress and strain in a material resulting in the stiffness of a solid material. This modulus is given by $E = \frac{FL_0}{A_0\Delta L}$, or the force, F, times the original length, L_0, divided by the area where the force is applied, A_0, times the amount the length changes, ΔL. The Shear modulus is similar to the Young's modulus, but it defines the ratio of shear stress and

shear strain, as given by $E = \frac{Fl}{A\Delta x}$ where F is the force, l is the initial length, A is the area where the force is applied, but now Δx is the transverse displacement, or the amount the shearing part of the material is displaced. Finally, Poisson's ratio, v, is a unitless ratio of percent expansion divided by percent compression during compression. When an object is pressed down on, the sides will usually expand. The percentage the sides expand divided by the percentage the object is compressed downward is this ratio. Knowing two of these moduli, the others can be determined using the conversion equation given in Table 2.1.

	$K =$	$E =$	$G =$	$v =$
(K, E)	K	E	$\frac{3KE}{9K-E}$	$\frac{3K-E}{6K}$
(K, G)	K	$\frac{9KG}{3K+G}$	G	$\frac{3K-2G}{2(3K+G)}$
(K, v)	K	$3K(1-2v)$	$\frac{3K(1-2v)}{2(1+v)}$	v
(E, G)	$\frac{EG}{3(3G-E)}$	E	G	$\frac{E}{2G} - 1$
(E, v)	$\frac{E}{3(1-2v)}$	E	$\frac{E}{2(1+v)}$	v
(G, v)	$\frac{2G(1+v)}{3(1-2v)}$	$2G(1+v)$	G	v

Table 2.1: Conversion equations between the four moduli. The columns are the value solving for, and the rows are the two known moduli.

Once a material reaches the elastic limit, plastic deformation occurs. In the real world, the true elastic limit may be very small, but the amount of plastic deformation at that point may be unnoticeable. In engineering, a modified elastic limit is defined, called the yield strength. The yield strength is the lowest stress at which permanent deformation can be measured. Most materials generally still behave according to the elastic material properties, but some materials, particularly metals, hardening and strengthening occur during plastic deformation. Metals are a well studied example. Metals are constructed of a lattice structure, which gives it its strength. The hardening occurs due to a failure in the lattice, where bonds between atoms break and rearrange in a different or tighter pattern, making it harder to deform.

Snow is comprised of a structure of very small ice crystals, similar to the lattice found in metals, but at a larger scale. This microstructure is called the ice matrix. The ice matrix is very complex, since the size and shape of the ice crystals vary widely in natural snowpacks. This changes both how the snow looks, and how it behaves. In Figure 2.1, a sample of snow was scanned and reconstructed on the computer, which

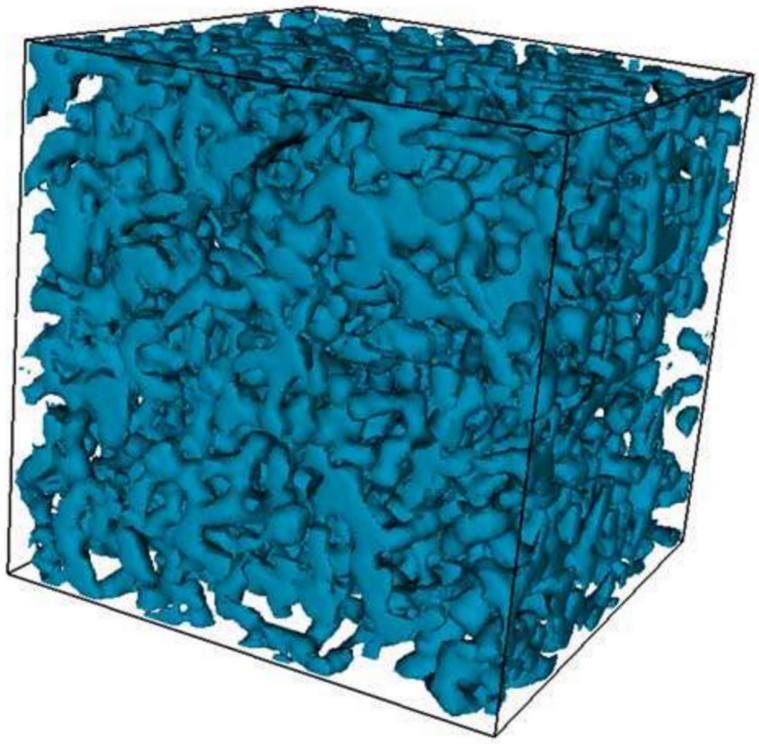

Figure 2.1: A reconstructed scan of snow microstructure [10].

clarifies its complexity. Lower density snow would be lighter colored because more light can travel through it without being absorbed. Lower density snow would also fracture more easily, because there is more air between the particles. According to the International Classification for Seasonal Snow on the Ground [5], "no standard method or parameter exists for characterizing snow microstructure." This is a problem because the microstructure determines the physical properties. While the structure looks very random, without the proper characterization, a simulated microstructure will not behave like real a microstructure.

The microstructure defines the internals of the snow, but on the surface, the appearance can be very different. Snow found on the ground is directly impacted by the external weather, and will be the first to melt, or the first to grow more crystals. In Figure 2.2, a closeup of snow surface crystals called surface hoar is shown. Hoar can

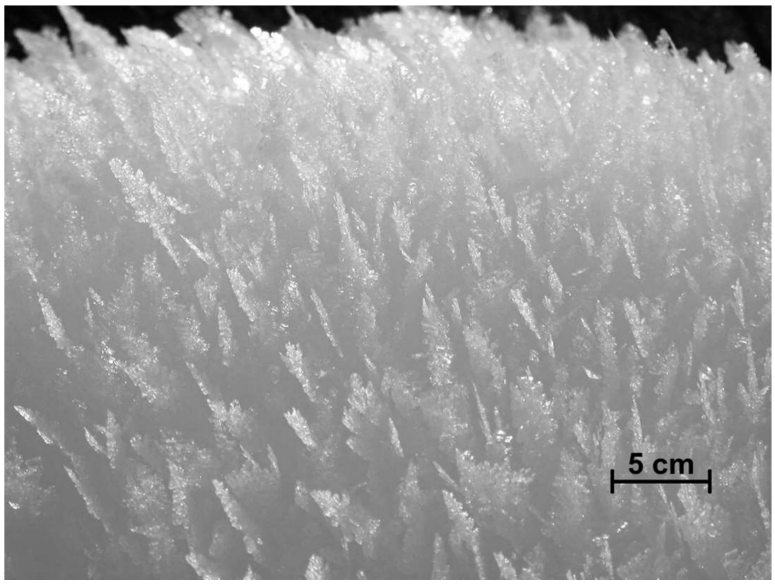

Figure 2.2: A closeup shot of snow surface crystallization [5].

grow internally as well, and is then called depth hoar. Another surface phenomena that may occur is called a crust. Freezing rain on snow creates a thin surface glaze of ice. A similar crust is formed when the sun melts surface snow and is then refrozen at the surface.

2.2 The Material Point Method

The Material Point Method (MPM), initially developed in 1995 [23], is a powerful method for simulating stiff materials that can undergo topological changes. It is a finite element based particle method that simulates particles moving through a fixed Eulerian grid. MPM relies on the continuum approximation to avoid modeling every particle in an object.

At the beginning of the simulation, the following steps must be taken. We first need to set up the initial simulation grid which sets the bounds of the simulation. This gives the governing equations boundary conditions so that they are solvable. Next we initialize the starting material points (particles). The simulated objects are translated into material points inside the cells of the grid that the object initially exists in. The material points

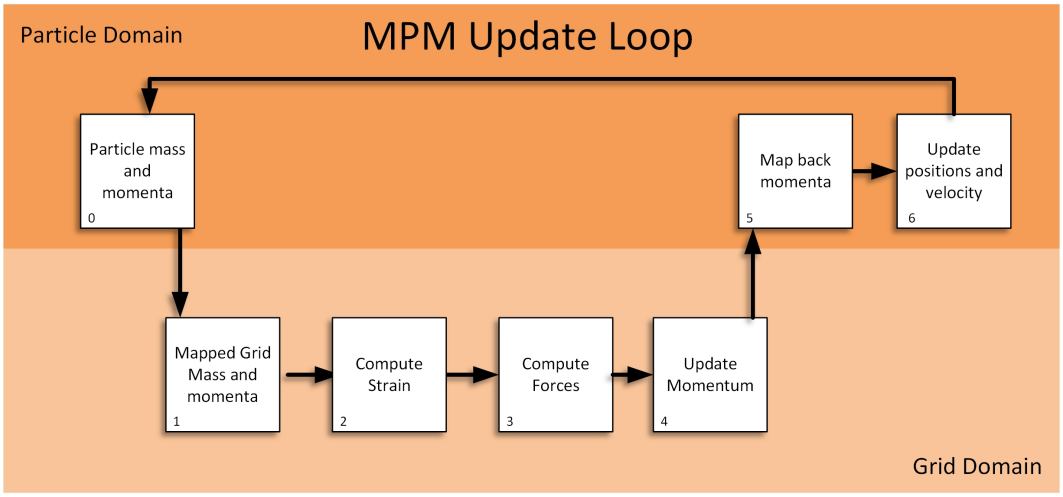

Figure 2.3: An overview of the material point method(MPM) implemented in our simulation software. The top row represents particle based operations, while the bottom row represents grid based operations.

are also given a material, the materials average density, an initial velocity. After that, we set up the shape function. The shape function is used to map the particles to the grid cell and back. It aggregates the values of the particles inside a cell to just the cell itself, and in reverse it interpolates between the values of each cell and assigns it to each particle.

Just before we get into the simulation loop, we first need to compute particle volumes and densities by mapping the particles to the grid using the shape function. We use an estimate of the grid cell's density based on the initial mass of the material and the volume of the cell. We then use the grid cell's density to map back to each particle mapped to that grid cell. Now that we have an estimate for each particle's density, we can estimate each particle's volume by dividing the mass of the particle by the estimated volume. See algorithm 1 for an example algorithm for the initialization of MPM.

Now we are ready to enter the simulation loop. The simulation loop is the same set of tasks that the simulation runs to move the simulation forward in time. There are six core tasks that MPM does every time step:

1. Extrapolate particle mass and momenta to the grid

Algorithm 1 MPM Initialize

Map the mass from the particles to the grid, then compute the density of each cell, then map back to compute volume
for Each grid cell n **do**
  for Each particle i **do**
    $m_i^n = \sum_p m_p w_{ip}^n$
  end for
end for
for Each grid cell n **do**
  $\rho^n = m_i^n / h^3$ // h is a grid length
end for
for Each grid cell n **do**
  for Each particle i **do**
    $\rho_p^n = \sum_i \rho^n w_{ip}^n$
  end for
end for

2. Compute strain tensors
3. Compute grid forces
4. Update grid momenta
5. Map particle momenta
6. Update particle positions and velocity

Tasks 1 and 5 are done using the same mapping process shown in Algorithm 1, but task 5 is doing the process in reverse, dividing the velocity of the grid cell among the neighboring particles. The next task is to compute the strain tensors. This step uses the momentum from the particles to solve for grid cell's velocities. It then uses the velocity to update the stress and strain tensors based off the material's constitutive law. Next, we apply grid based forces, such as the stress we just updated, and external forces like gravity. After we have calculated the total forces on the grid node, we update the momentum using $p_i = p_{i-1} + F_{i-1} * dt$, where dt is the time step in the simulation, p_{i-1} is the known momentum and F_{i-1} is the calculated forces. Here there is an optional task that depends on the situation. If there is contact between objects taking place at the grid point, the momentum must be adjusted for the interaction of the other material.

Now that we have a new momentum calculated for the next timestep on the grid node, we need to extrapolate back to the particles. Once that is complete, we use the new momentum to update each particle's position and velocity. Then the simulation time increases and the loop starts over. This is done for every node on the grid for every step of the simulation. This can become very computationally expensive with complex material models and thousands of particles.

2.3 Rendering

Realistic rendering has historically been accomplished using ray tracing. Ray tracing is a method of simulating the way rays of light bounce in an environment, in order to determine the color of a pixel in an image. Instead of going the direction that light bounces normally, ray tracing generally goes the reverse direction, starting at the viewpoint of the image and bouncing in the scene towards the light, and other directions. Classical papers have addressed using multiple rays to simulate softer lights and shadows[3], internal light scatter of objects, and microscopic detail on surfaces. Raytracing is a light simulation approximation algorithm which makes knowing the physics of light and optics important to designing visually accurate scenes.

When light hits ice, part of the light is reflected, and part of the light gets transmitted into the material. The ice matrix inside snow makes this computation of hundreds of light scattering equations very difficult to characterize. In most light scattering equations, they are based around homogeneous medium. The randomness of snow makes these equations close but inaccurate. In many papers, a function called the Henyey-Greenstein function is used to approximate these complex functions [2] [22] [11]. This function is not based on any physical theory, but its shape can be easily modified to approach realism. One simple form of the function is shown in Equation 2.1, where θ is the scattering angle, and g controls the shape.

$$p_{hg}(\theta) = \frac{1 - g^2}{4\pi(1 + g^2 - 2g(\theta))^{\frac{3}{2}}} \tag{2.1}$$

Many simulation methods use particles as their unit of simulation. The problem is that fluids and other continuous materials do not appear as particles in real life. The solution to this is called surface reconstruction. With surface reconstruction, a

scalar field is generated around the simulation's data points and a surface is made a specific value of the scalar field. The scalar field is generated by equations that add up multiple particles' contribution to the surface. Another name for this is an isosurface or an implicit surface. One of the simplest equations for generating an implicit surface is called metaball. The equation given in Equation 2.2 is the typical function. This equation is summated together for all of the particles at each a point sampled in the scalar field. Due to the the small effect far away particles will have on the sampling location, we can store all nearby particles and just add the effect of those.

$$f(x, y, z) = \frac{1}{(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2} \tag{2.2}$$

Where (x_0, y_0, z_0) is the particle location, and the x,y,z sent in is the sampling location. The surface is created by sampling this theoretical scalar field. The established method for this is called Marching Cubes [17]. The idea for marching cubes is to section off all of the data into a set of cubes, making a 3d grid. For every cube in the grid, the same steps are followed. At every corner of the cube, the scalar field is sampled. Then we check if the surface value, the value we decide to create the surface at, lies between two connected vertexes of the cube. If it does, part of the surface is inside this grid. There are approximately 256 different combinations of edge intersections, which can be boiled down to 15 unique triangulated cubes, rotated in different ways [17]. These 15 cases are shown in Figure 2.4. Once all of the cubes in the grid are finished, they combine their triangles to create a surface around the material. From there, the surface can be smoothed and enhanced.

There were some problems with the original implementation of Marching Cubes. As it turned out, there were some mesh ambiguities that could not be resolved by just sampling the corners of the cube. This caused the surface to not be topologically accurate and would generate holes. Some improvements change to sampling the scalar field with a mesh of tetrahedra. This method is called marching tetrahedron. The problem is that the amount of sampling done with marching tetrahedron is much higher than with marching cubes. Other improvements have been made to remove the ambiguities from Marching Cubes. We utilize one of these improvements developed by Thomas Lewiner [14]. They used 32 unique mesh designs, and solved ambiguities by sampling in the middle of edges of the cube and determining the relations of each sampling point in the cube to the other

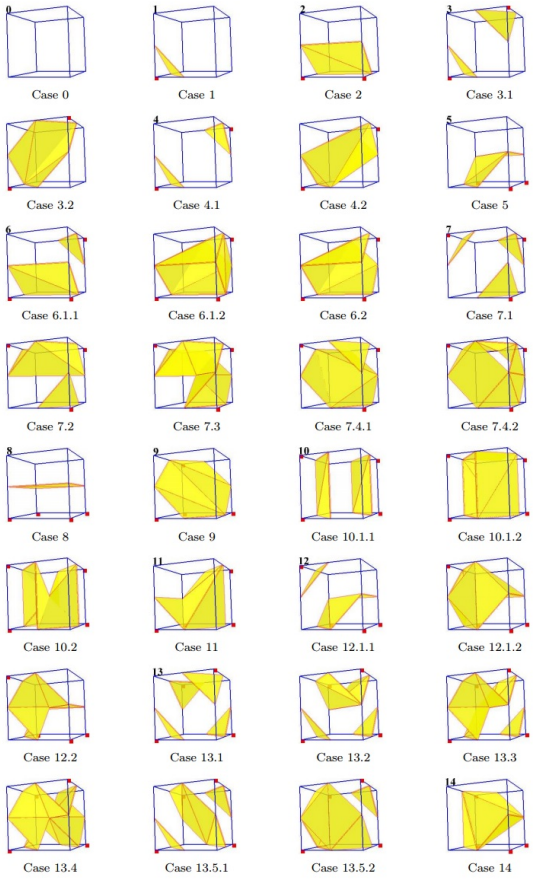

Figure 2.4: The 15 unique cases of the original Marching Cubes.

points in the cube.

Implicit surfaces are generally used to create fluids from simulation particles. While snow shares some behaviors of a fluid, the surface of snow is very different from that of a fluid. Implicit surface improvements have all been working towards more smooth and consistent surfaces, but for snow we would want the surfaces to be consistent, but we do not want them to be smooth. Snow has a very grainy appearance and naturally has a very rough and uneven exterior.

Chapter 3: Literature Review

Tire-snow interaction has been approached in a variety of ways. Past research focused on such subjects as: gathering measurements of friction and heat generated on snowy or wintry roads, studying what happens when cars slip on snow, and simulating properties of snow in various conditions.

3.1 Simulation

To fully understand snow and how it may affect tire performance, it is important to look at analysis and experiment based research. We found the experiments fan by Fujimoto et al. [6] important for understanding the many considerations and factors that go into making driving on snow safe. They studied the heat generated as a tire passes over snow and the changes this causes to the structure of the snow. Tires running over snow can cause some of the snow to melt, adding water to the snow. In turn this water can freeze and add ice to the snow. These are all safety hazards that can cause a tire to lose traction.

Walus and Oszeweski [24] discovered the friction coefficients of tire-road interaction in various winter conditions. Their results are significant because they helped explain a wide range safety hazards in winter. They examined how tires responded under different road conditions, including on cold and dry asphalt, as well as tires on ice, black ice, snow with ice, sand and gravel, fresh snow, compacted snow, and deep snow both for with chains and without chains. Their research makes it clear why it is important to examine all the road hazards in winter. Our current focus is on tire-snow interaction under two different conditions: when the road is covered with icy and hard snow and when it is covered with soft and fluffy snow.

While these studies make it clear that it is important to study tire-snow interaction, the continuing challenge is in finding the best approach for modeling snow because of its complex characteristics.

The bulk of work on snow simulation is focused on accumulation and avalanches.

Our focus is on snow that has already accumulated, and the interactions it has with objects like tires and other relatively small clumps of snow.

There is a diverse set of material models that are used to represent snow. The Drucker-Prager model, which was initially created in 1951 [4], has been used in papers by Meschke [18] and Lee[13]. This model has been improved and customized, and the version used in Lee's paper[13] was first developed by R. Haehnel [8]. This model was referenced in Stomakhin's paper [22] as being more accurate than was needed for creating animation in movies. In the future, we intend to study the Drucker-Prager models and implement them to achieve better accuracy in modeling snow.

Snow simulation has been performed in a variety of ways. Finite element analysis (FEA) has been the most popular way to study snow-tire interactions. In 2011, Lee used FEA to study contact stress for different types of tire slips [13]. There are three different styles of slippage that he studied: longitudinal slippage, a forward slip, which comes from braking; lateral slips, or sideways slides, which comes from turning; and a combination of both longitudinal slippage and lateral slips. While Lee's research was instrumental in studying what happens after a tire slips, our focus is on how well tire treads can keep a tire free of snow.

Another approach to simulating the different properties of snow is to assume the system is purely stochastic. Li et at. [15] proposed that the snow on any roads would demonstrate random properties throughout. He decided to take an established deterministic model of snow and make depth and density uncertain parameters. With this new stochastic model, he studied the probability of wheel slippage using polynomial chaos. We conducted our study with the recognition that snow on roads will not be homogenous throughout. Snow can be denser, wetter, or deeper depending on where it is. This is why there are no perfect variables to describe all snow. We strive for approximate material parameters for snow, and accepted it as a general case to test tire treads.

Both Lee [12] and Stomakin [22] have shown the application of MPM on modeling snow. Lee [12] started out by testing tires running on snow using a finite element software. He then started to research more into snow simulation, using MPM for the microstructure of snow. He used a 3.6mm^3 grid in Uintah, another MPM software, to simulate the gaps and holes in the random internal structures of snow. He took recordings of actual collected samples and compared them to a stochastic reconstruction that he designed. The microstructure of snow is important for changing the governing

properties of the snow, therefore building the microstructure modeling to determine the snow's physical properties is the next phase of our research.

The problem with the majority of the previous research is that they do not model the complex dynamics of clumped and fracturing snow. Stomakhin [22] demonstrated that MPM is a powerful tool for simulating the complex dynamics of snow, and that its uses have not been fully researched. We were inspired by this paper to investigate the uses of MPM for snow simulation to study vehicle safety. Stomakhin et al. used standard equations but reformulated some of them or used approximate equations to speed up rendering and give artists more control over the snow. In Stomakhin's paper [22], MPM is used to describe the macroscopic features of the snow, whereas Lee [12] used MPM for the microstructure of the snow. In this paper, we used MPM for snow simulation and snow-tire interactions at a macroscopic level.

We are simulating a tire interacting with snow, so it is also important to accurately simulate the rubber of the tire as well. In contrast to snow, tire rubber has been studied a lot more thoroughly. Rubber is practically a totally elastic material, so using a simulation method like the finite element method works exceptionally well, because it is a continuous material throughout most simulations. As far as we could tell, MPM has not been used to simulate tire rubber at this scale before. There has not been the need to simulate whole tires breaking apart accurately, and the computational costs of accurately modeling a detailed tread pattern are large. Because of the geometric complexity of tire treads, and the uniform mesh that MPM uses, accurately simulating the tread would require a very fine mesh. This fine mesh would slow down the simulation immensely. We use established equations and parameters for tires but we use the MPM method to simulate them, which has never been done before.

3.2 Snow Rendering

Research into realistic rendering of snow has not been a focus recently. Most research being done is targeting real time approximations of snow's visual appearance. In Stomakhin et. al [22], they used a volumetric path tracer which treats the particles of the simulation as particles in a cloud. This gives the snow a very blurry appearance without much surface detail, though it approximates the internal scattering very realistically.

Most modern research into snow rendering has targeted real-time performance, focus-

ing on speed rather than visual accuracy. While Krošlák's thesis [11] and Liu's 2010 [16] paper targeted real-time, the researchers had a solid overview on snow rendering and physics, and were very informative.

In the past, snow has generally been rendered as either snow falling or snow accumulation. There are not a lot of rendering methods that can do both stationary snow and active snow. Metaballs have been used to render snow accumulation [21], as well as using Monte-Carlo simulation for randomizing the snow crystals covering any model [2].

In Stomakhin et al. [22], they focus on the simulation of realistic behavior, and skim over creating the most realistic snow visually. They used a volume path tracer and put in measured values for snow, which gives them convincing colors for the snow, but it makes the appearance seem cloudy or blurry. The lack of details made the snow look less realistic, and it fails to capture the granular and sparkling property of snow. An example of their snow can be seen in Figure 3.1.

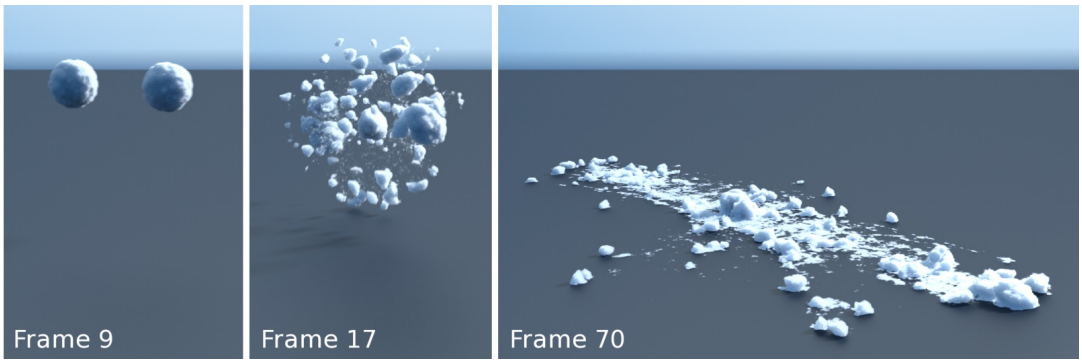

Figure 3.1: An example of snow from Disney's paper [22]. Notice the cloudy appearance of the surface.

Chapter 4: Experimental Design

4.1 Simulation

There are two main materials I simulated: snow and tires. To perform my experiments, I knew that I wanted to use an established MPM package that was well respected. We discovered an open source tool called NairnMPM [20] that was created at Oregon State University. We found that this tool was able to satisfy our constraints.

4.1.1 Snow Modeling

While researching snow simulation, we found that the main focus of previous papers were on avalanches, snow drift, accumulation or on the microstructure. There was not a lot of research into the physics of snow at a scale between avalanches and microstructure. We needed to model snow as a continuous body made of particles, instead of modeling the microstructure, and avalanches were too large and avoid the fracturing and compression detail that we needed for tire-snow interactions. We needed to obtain the pertinent mechanical properties through trial and error. Achieving correct snow-snow interactions required setting the values for the snow parameters, running the simulation, judging the visual output of the simulation, and making changes to the parameters for the next iteration. The only other MPM simulation of snow at this size was done by Stomakhin et al. [22]. Stomakhin et al. gave the equations and parameters that they used for their implementation, which gave us a starting point. The examples for different parameters given by Stomakhin is shown in Figure 4.1.

We did not accept the variables as correct for our situation for a few reasons. The first reason is that we were using a different program. Stomakhin et. al. created their own simulation code from scratch, which could generate very different results than the established code that we used. We also did not have the exact implementation of the constitutive equation they used for their snow material. They created extra parameters to help artists out, and skimmed over such parameters as the yield stress, which forced

Parameter	**Notation**	**Value**
Critical compression	θ_c	2.5 x 10^-2
Critical stretch	θ_s	7.5 x 10^-3
Hardening coefficient	ξ	10
Initial density (kg/m^3)	ρ_0	4.0 x 10^2
Initial Young's modulus (Pa)	E_0	1.4 x 10^5
Poisson's ratio	v	0.2

Table 4.1: Example parameters from Stomakhin et al. [22]

us to not be able to use the same variables. The last reason is that we had a different use case than Stomakhin had. While they were targeting movies, and allowing artists ease of controlling the snow to do what they wanted. Our goal is to realistically simulate a few general snow models, and be able to test snow tires and their different tread patterns on them. We specifically did not want to use the same equations and variable values because we wanted to be as accurate as possible. They defined a critical compression θ_c and stretch θ_s that numerically limit the force required to compress and stretch before plastic deformation or fracture. This is very unrealistic for physical accuracy, but because that was not their goal, it was adequate for them. Another point they made was that they defined their plastic yield by using principal stretch rather than principal stress, which is the established method. They stated: "While principal-stress-based plasticity is more appropriate for physical accuracy, principal-stretch-based yield gives the user more control over the visual behavior of the simulation." As we did not want control over the visual behavior, but wanted physical accuracy, we knew we needed to use established material models.

Stomakhin's model used a neo-Hookean, a type of hyperelastic material, elastic-plastic material, with artificial breaking points instead of a hardening law. The material model we used for snow was an isotropic, hyper elastic-plastic material with a nonlinear hardening law governing its plastic deformation. We chose this to imitate the actions of snow. A fully elastic material regains its shape after any deformation, whereas an elastic-plastic material has a certain threshold of deformation where it loses the ability to fully regain its shape. This plasticity threshold is modeled by this nonlinear hardening law, which means that the object nonlinearly gets harder to deform the more it deforms. This is what happens with snow when a snowball is formed. It no longer retains its

original shape and gets harder to pack any further. An elastic-plastic material has a stored energy based on its elastic and plastic internal energies. Its elastic energy is given by the expressions in Equations 4.1- 4.4.

$$W = U(J) + \frac{G_1}{2}(\bar{I}_1 - 3) + \frac{G_2}{2}(\bar{I}_2 - 3) \tag{4.1}$$

$$U(J) = \frac{k}{2}(\frac{1}{2}(J^2 - 1) - ln(J)) \tag{4.2}$$

$$\bar{I}_1 = \frac{B_{xx} + B_{yy} + B_{zz}}{J^{\frac{2}{3}}} \tag{4.3}$$

$$\bar{I}_2 = \frac{1}{2}(\bar{I}_1^2 - \frac{B_{xx}^2 + B_{yy}^2 + B_{zz}^2 + 2B_{xy}^2 + 2B_{xz}^2 + 2B_{yz}^2}{J^{\frac{2}{3}}}) \tag{4.4}$$

Where U(J) is the volumetric energy term, and I1 and I2 are the strain invariants. J is the relative volume change, G1 and G2 are shear properties of the material, B is the left Cauchy-Green strain tensor and k is the bulk modulus. The plastic stored energy, the nonlinear hardening law, is based on the yield stress of the material, is shown in Equation 4.5, where σ is the initial yield stress, α is the cumulative equivalent plastic strain, and K and n are dimensionless coefficients.

$$\sigma_y = \sigma_{y0}(1 + K\alpha)^n \tag{4.5}$$

To compare this established material model, we used a similar material model to the model introduced in Stomakhin's paper. In that paper, they used a neo-Hookean elastic plastic material, but added in set, user controllable fracture points and compression points. These points would be the exact part of the stress curve where the material would fracture, or plastically deform. In the paper they were called the critical stretch and critical compression values. The elastic neo-Hookean potential energy equation for this model is shown in Equation 4.6.

$$W = \Phi(F_E, G(J_P), \lambda(J_P)) = G(J_P)\sum_k(\lambda_k - 1)^2 + \frac{\lambda(J_P)}{2}(J_E - 1)^2 \tag{4.6}$$

This equation depends on the current elastic deformation gradient, F_E, and the shear

and Lamé moduli G and λ. The proposed law has not been proven to be an accurate representation of the physical world. We compare this model with our own to determine the effectiveness and accuracy of the proven model compared to this proposed one.

4.1.2 Tire Modeling

Tires have a much more thorough research history than snow. Tires are a solid elastic material with relatively uniform mechanical properties throughout. This makes tires much easier to simulate than snow. Rubber is also a man made material and manufacturing benefits greatly from correct simulation.

Tire materials are often modeled by Mooney-Rivilin stress-strain relationships due to the hyperelastic properties under loading. The equation for this is almost the same as Equation 4.1, because unlike snow, rubber is not considered to have plastic properties. Equation 4.1 was modified so that there is no change in internal energy during isothermal loading. This modification is generally used to model ideal rubbers. The physical properties of tires and tire treads have been thoroughly researched, so we used established values for our parameters. We set Poisson's ratio to 0.49 as the rubber material is nearly incompressible. We also set Young's Modulus to 1x10^7 Pascal and the density to 0.93 g/cm^3 [25].

To describe the motion of the tire on a snowy surface, we first had to research the forces acting upon it. Primarily, we had to determine how friction was affecting the tire. The formula for finding the frictional force acting on an object is given by Equation 4.7.

$$F_{friction} = \mu * F_N \tag{4.7}$$

Where μ, the coefficient of friction, is some constant and F_N is the normal force on the object. However, μ is not really a constant; it changes with velocity for some objects. An example of this is a comparison between a car traveling at 5mph and a car traveling 100mph. Assuming the normal force on each car is about the same, since they have the same mass, and that they are both traveling on the same surface. When a sudden, hard brake is applied to the wheels, we notice a big difference. The slower car would immediately come to a stop, but the faster car would lose traction and lose control as it tried to slow down. This is due to the fact that tires travelling at higher speeds have a

lower coefficient of friction. This implies that faster objects experience a smaller fiction force, so the fast car would have less force helping it slow down, resulting in skidding.

Previous experiments show that although the coefficient seems to decrease with speed, there is actually a very small range in which the coefficient is actually increasing. As the speed increases from standing still, the coefficient of friction briefly increases before decreasing. To deal with this issue and to determine the ranges in which the coefficient is increasing and decreasing, we had to develop a friction model for our tire to follow.

To develop a friction law, we used the equation developed in Peter Wriggers and Jana Reinelt's paper [26]. After performing a series of experiments, they conjectured that the coefficient of friction between rubber and a rigid surface can be approximated by Equation 4.8.

$$\mu(v, p_n) = (\frac{2vv_{max}}{v^2 + v_{max}^2})^c * \mu_{max} \tag{4.8}$$

Here, the coefficient is a function of velocity. μ_{max} is the maximum value of the coefficient and v_{max} is the velocity where that maximum coefficient of friction occurs. The constant c determines the steepness of the curve. Based on their experiments, we assumed that a rubber material surface experiences the highest frictional force when the rubber material is moving about 110 mm/s across a surface. Also, to get the general steepness of the curve, we assumed .17 to be the constant c. We further assumed that the static coefficient of friction, i.e. the initial value when the rubber is not moved, is a given quantity. Thus, we rearranged the equation and solved for μ_{max}, and after graphing, decided that the coefficient of friction in this model reaches the static friction coefficient when the speed is approximately 1mm/s. Substituting all the values in, we get Equation 4.9.

$$\mu(v) = \mu_0(\frac{12101v}{v^2 + 12100})^{\cdot 17} \tag{4.9}$$

The static friction coefficient between a tire and the road covered in snow can be near .25 (it may vary based on the condition of the snow). We chose this value based on the range of values given in Walus and Olszewski's paper [24], which states that the range of coefficients under snow and ice conditions is between .12 and .39. Our coefficient of friction curve is plotted in Figure 4.1.2.

Incorporating this model into our simulation, the frictional force is determined based

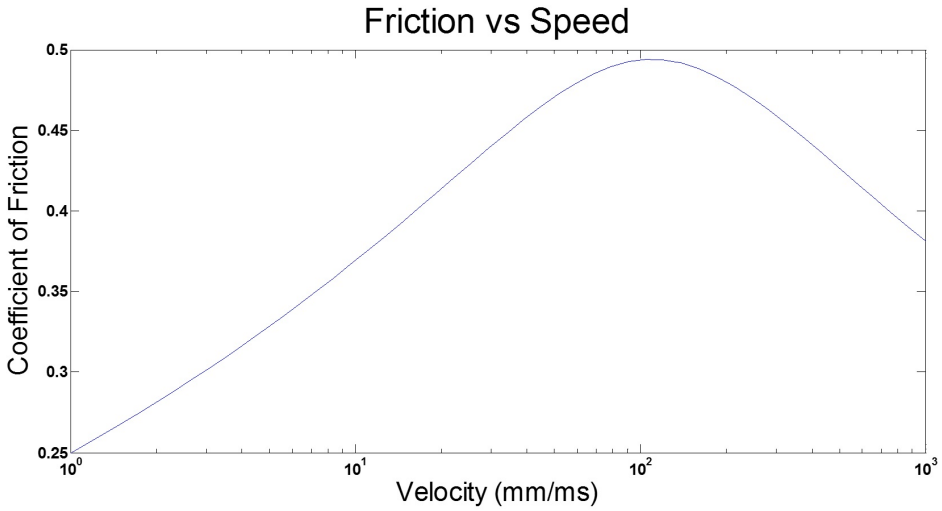

Figure 4.1: The relationship between velocity and the coefficient of friction.

on the speed of the tire. At each time-step, the velocity of the tire is analyzed and the friction coefficient is adjusted. Using this, we can thus model how the frictional force on the tire, proportional to the friction coefficient, varies with time. We then determined how its average velocity changes. We measured frictional force on the tire versus time with the tire's initial velocity at 0.5 m/s, 1.4m/s, and 4.47 m/s. The results are shown in figure 4.1.2.

All the friction force values are negative, which is expected since the friction force acts to oppose velocity (which is positive). We also found that at higher speeds, the magnitude of the friction force decreases, just as we expected. We also noticed that the force values started to oscillate after a long period of running time.

4.2 Rendering

To visualize our simulation data, we had to decide how to render snow and tires correctly. The particle data we were given from the MPM simulation had to be converted into a form that looked realistic. When we first started our simulations, the simplest visualization method we used rendered a sphere at every material point, using an open source renderer called POV-Ray. This was quick, but it did not look very realistic.

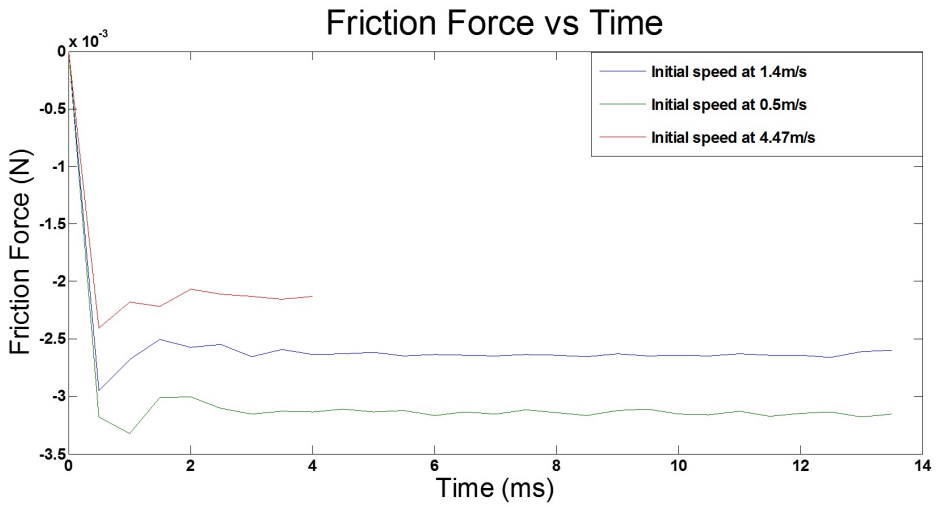

Figure 4.2: Simulation of the friction force versus time for different initial speeds. We note that the force values start to stabilize after 3 ms.

Our motivation for rendering snow realistically is to be able to communicate with field experts, such as road engineers from the Department of Transportation, who are not familiar with MPM but have years of experience working with snow covered roads. In addition, we aim for visual realism to enable us to iterate on the snow material model after viewing the rendered visualization.

MPM records its data in terms of grid cells and material points, or particles. Visualizing this data can be done in a few different ways. With just particle data from MPM, we had the choice of point-based rendering or surface reconstruction. Point-based rendering is rendering each data point as a sphere, and surface reconstruction is creating a continuous surface around a set of points. We first utilized point-based rendering, due to the ease of implementation. This allowed us to focus on the material properties and particle descriptors. For the results in this paper, we utilized a surface reconstruction formula, to achieve a more realistic appearance of both the tire and the snow.

All the modern approaches to surface reconstruction for fluid and fluid-like simulations are either based on the color field equation [19] to define its scalar field, or on a signed distance equation [27]. We implemented the signed distance equation for our scalar field because it provides more consistent surfaces. The equations for the signed

distance formula is shown in Equations 4.10- 4.13.

$$\varphi(x) = |x - \bar{x}| - \bar{r} \tag{4.10}$$

$$\bar{x} = \sum_i w_i x_i \tag{4.11}$$

$$\bar{r} = \sum_i w_i r_i \tag{4.12}$$

$$w_i = \frac{k(|x - x_i|/R)}{\sum_j k(|x - x_j|/R)} \tag{4.13}$$

Where $\varphi(x)$ is the scalar field function for each particle i, with its own position, radius and weight, given by x_i, r_i, and w_i respectively. k is a kernel function that smoothly drops to zero, and R is the radius we consider around x_i. The kernel function we used was supplied in the paper [27] and is shown in 4.14.

$$k(s) = max(0, (1 - s^2)^3) \tag{4.14}$$

After the surface equation was defined for the data points, we used the efficient marching cubes algorithm [14] to construct the mesh. The code for this was supplied on their project website. We imported the simulation data for a timestep, and created a grid for sampling the scalar field. We solved Equation 4.10 and then used the efficient marching cubes code to generate a .ply file that we could modify. We noticed that because the simulation runs on a grid, the surfaces at the beginning of the simulation display grid-like patterns. We adjusted this by smoothing the resulting surface. Then to create more detail, we added noise to each vertex on the surface.

Using the signed distance equation and surface reconstruction creates a much smoother appearance than snow cover usually appears. We wanted to recreate a surface that appeared like that in Figure 2.2. To achieve this effect, we needed to add the surface hoar crystals to the smoothed surface. These crystals would effect the physics of the snow, but the simulation would need to be run at a much higher grid density to be able to cover these crystals, thus they were not simulated. As mentioned previously, MPM is not designed for complex geometry, but more for bulk and continuous materials. Snow

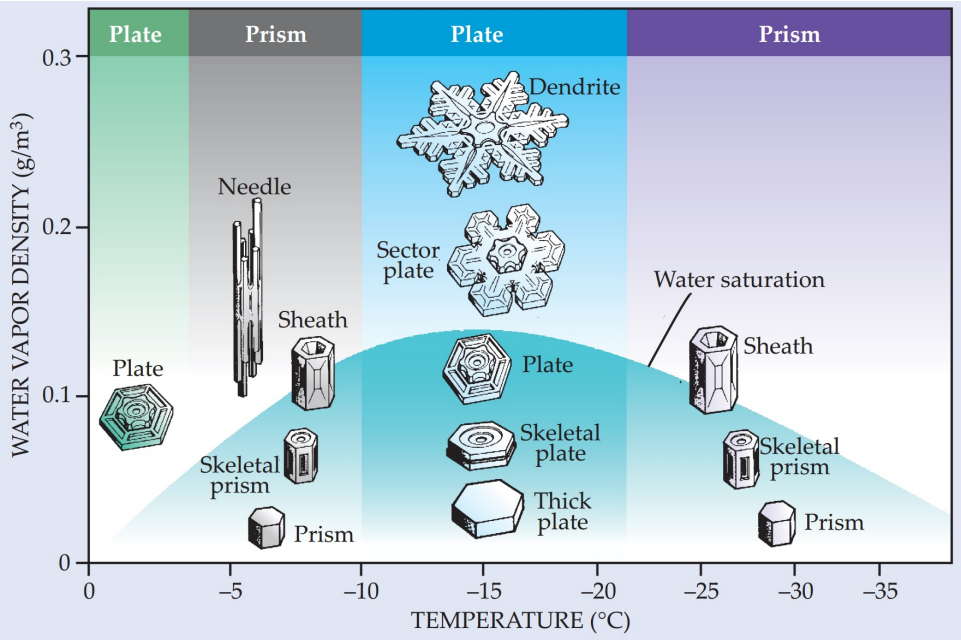

Figure 4.3: The shape of snow crystals depends on temperature and humidity. This is called a Nakaya diagram, which shows the density of water vapor with respect to ice vs temperature. The curved line shows saturation with respect to liquid water. [7]

crystals ideally grow as hexagonal prisms. The different shapes that snow crystals form can be seen in Figure 4.3. We decided to model ice crystals and place them on the surface, to form the appearance of snow crystals.

We modeled the crystals and added them to our modified snow surface. We placed crystals in the centers of faces and at vertices. The mesh was created semi-randomly with marching cubes, so there was no pattern to the crystals and they appeared random. Then we rendered them in Mitsuba [9], an open source rendering tool, and more advanced than POV-Ray. We first rendered the snow as a flat white material, so that we could focus on the surface details. When we were happy with the surface, we applied internal scattering using the Henyey-Greenstein phase function and values of $g = 0.5$ and a slightly blue scattering albedo of [0.9,0.95,1.0] as defined in Stomakhin's paper [22].

Compared to snow, tires are generally easier to render realistically. Rubber does not allow light to travel through it, at least not to the degree that snow and ice do. Creating a tire out of a collection of points in space requires a much different technique

than creating snow. Tires have very rigid and hard edges on their treads, and creating a smooth surface around a set of points did not look like a real tire, but more like a balloon. Our focus was to work on creating realistic snow, so we only rendered tires with our initial sphere method in POV-Ray.

4.3 Test Setup

4.3.1 Snow-Snow Interaction

Before exploring snow-tire interactions, we designed a set of scenes to test our snow parameters. Due to the complexity of snow, we needed to test the material's response to different types of external forces. We created three scenarios, where two focus on stress and compression forces, and a third scenario that focused on strain and stretch forces.

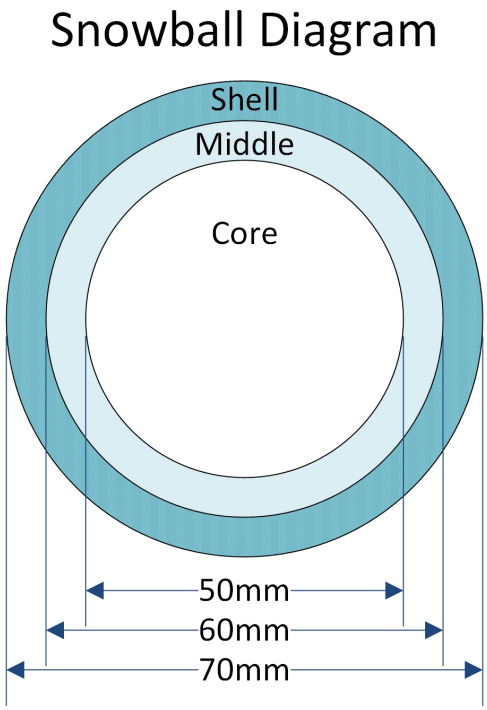

Figure 4.4: This diagram shows the structure of our snowballs in our simulation.

The first scenario we created was two snowballs having a head on collision. This

scenario was created to test our snow material interacting with itself, and explore the reaction to strong compression forces. Our motivation for using snow-snow interaction as our starting point was to watch how snow balls break apart. It takes very little force to pull apart a snow ball, and so we set up our models so the snow behaves in a realistic way, while keeping the modeling well-posed. We were looking for snow that would crack apart but have some small amount of cohesion, so that nearby pieces of snow would stick together. After some deliberation, we decided on layering the snowball with co-centric spheres. When a snowball is formed, pressure and heat is applied to the outside of the snow by the crafter's hands, causing the snow on the surface to get denser and harder. Thus we created three layers, a core sphere, a transition layer, and the crust. The layout is shown in Figure 4.3.1. The two snowballs were slightly offset from each other to not create numerical mirroring that would not be present in the real world. The snowballs we created were 70mm in diameter and thrown at 6000mm/s. We decided that this was a realistic size a velocity for a snowball. A diagram of the scene can be seen in Figure 4.3.1.

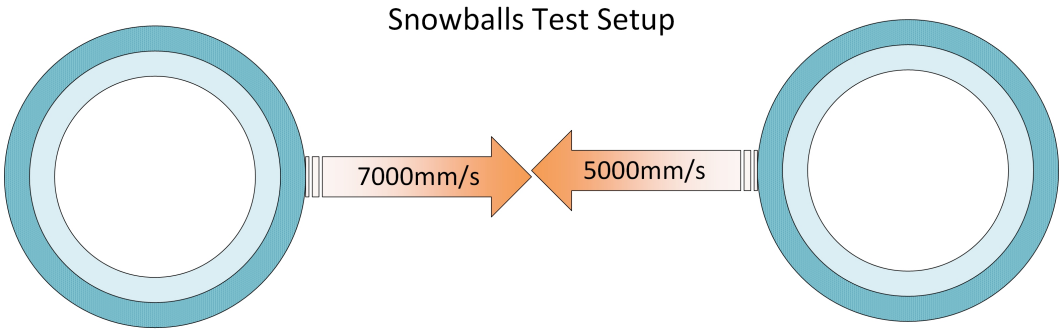

Figure 4.5: This diagram shows our two snowball collision scene.

The next scenario we created was testing one of our snowballs colliding with a rigid wall. Once we tested how the snow reacted to stress from other snow, we wanted to see how the snow would crack when all of the energy is returned back to the snow. We used the same layered snowball as the first snowball scene. Figure 4.3.1 shows the layout of this scene.

While the first two tests both displayed snow experiencing high amounts of pressure, our third test was a simple stretching test. When pulling apart a clump of snow, it

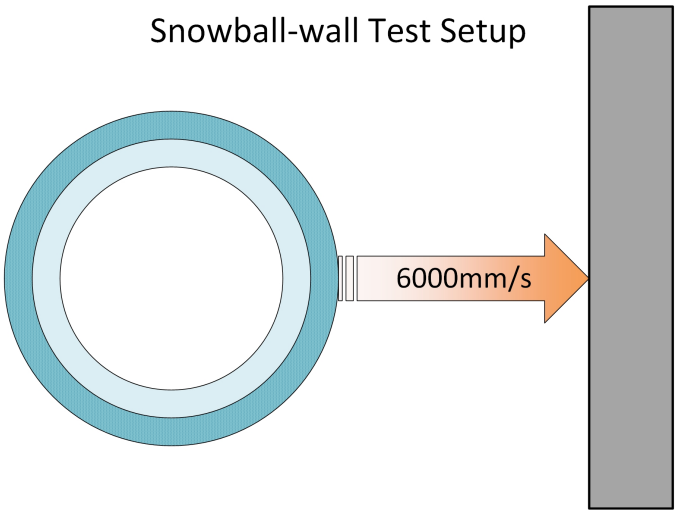

Figure 4.6: This diagram shows the test setup for our snowball colliding with a wall.

should stretch a very small amount before fracturing. Most materials display a stretch before fracture, and snow is no different. While is does not stretch like a rubber band, replicating the small fracture point is important for correctly modeling snow. Snow is a structure of very small ice crystals that will easily snap apart. To test this, we had to create a block of snow and pull it apart. This is achieved by fixing one end of the block and applying a load to the opposite end pulling it apart. Figure 4.3.1 illustrates how this is set up.

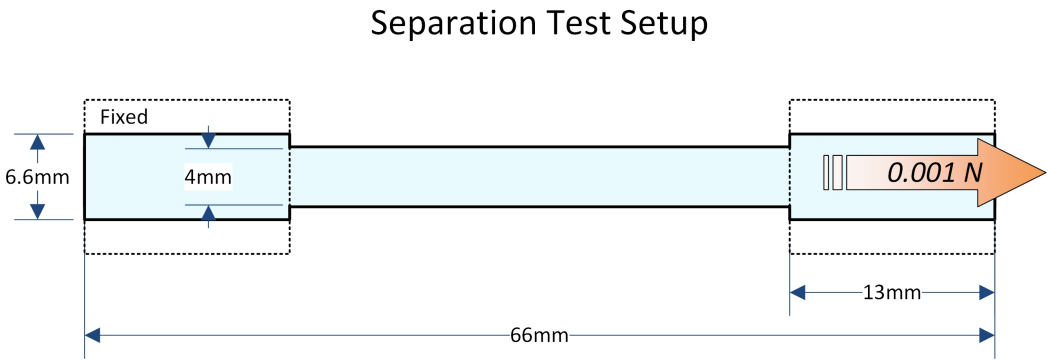

Figure 4.7: A diagram of our uniaxial stretch load test.

Variable	**Shell**	**Middle**	**Core**
Density (g/cm^3)	0.4	0.36	0.33
Bulk Modulus K(MPa)	0.078	0.051	0.0267
Shear Modulus G(MPa)	0.0583	0.0325	0.02
Hardening Coefficient	15	12	10
Nonlinear Hardening Coefficient	0.02	0.02	0.02
Initial Yield (MPa)	1x10^{-4}	1x10^{-4}	1x10^{-4}

Table 4.2: Material parameters for the snowball layers. The shell is the outermost layer, the core is the innermost layer, and the middle in between. Refer to Figure 4.3.1 fora diagram.

Snow's physical parameters are spatially varying, due to the process in which it forms. Instead of stochastically modeling the internal properties, we tested a few different homogenous material properties to act as the bulk properties of snow. The snowball was modeled with three layers of homogenous properties, with a harder crust, and a softer core. The snow stretch was tested with all three layer materials to make sure that the whole snowball acted realistically. Table 4.2 lists the parameters we used to model the three layers of snow.

We compared the simulation results we found with the Hyperelastic-Plastic Material to the results given by NairnMPM's implementation of the clamped hyperelastic material given by Stomakhin et al. Using the parameters given in their paper, and seen in Figure 4.1, we ran the three scenarios we created with the Clamped neo-Hookean model to compare visual results as a base reference.

4.3.2 Snow-Tire Interaction

Using MPM to model a continuum structure like a tire in a required level of detail is challenging. A realistic snow tire has tread patterns with fine grooves and sipes. In order to model one correctly, we would have to use a very fine simulation grid, which would slow the simulation down immensely. Consequently, we needed to design tire tread patterns that had large features but were approximately those used on real winter tires. We also wanted multiple distinct patterns that would help us understand what general designs are better than others for maintaining traction and for keeping snow out of the tire treads.

The three patterns we designed each had specific features that we considered important. See Figure 4.3.2 for a diagram of the patterns. The first pattern we called the angled footstep. The intent of this pattern is to dig into the snow, and to throw it off to the sides. This tread fits in the directional tire category. Another feature is that there is always a tread making contact with the ground. The second tread pattern, which we called the bowtie, is orthogonal to the first pattern in two aspects: the center of the tread and the continuous contact. In the center of the tire, the treads are flat, which would spread the weight of the tire over more of the snow, and not dig in as much. The third tread pattern, the six rib, is the closest to a commercially available winter or snow tire. It has a four inner angled blocks and two outer square blocks. It also has grooves in between the treads for snow or water to be shuffled out through. There are many more treads on this design, so there are always multiple treads touching the ground at the same time.

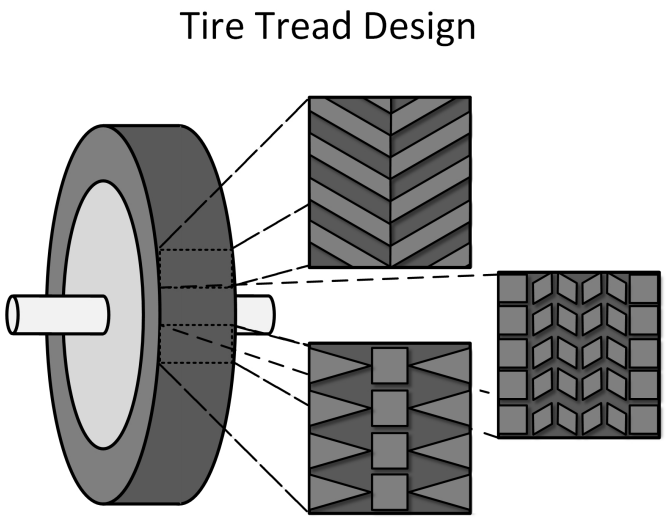

Figure 4.8: Diagrams of the angled footstep (top), bowtie (bottom) and six-rib (right) tread patterns.

To simulate these tires rolling on snow, we developed a one-wheel testing environment. Instead of simulating an entire car rolling over a snowy street, we focused on one tire for this stage of our research. We attached the tire and wheel to a fixed axle and pushed rigid ground covered in snow under it. This allowed us to keep track of how fast the tire

was spinning and how fast the ground was moving, so we could focus on a tire's ability to funnel out snow and not have it get stuck in its treads. Figure 4.3.2 illustrates the scene setup.

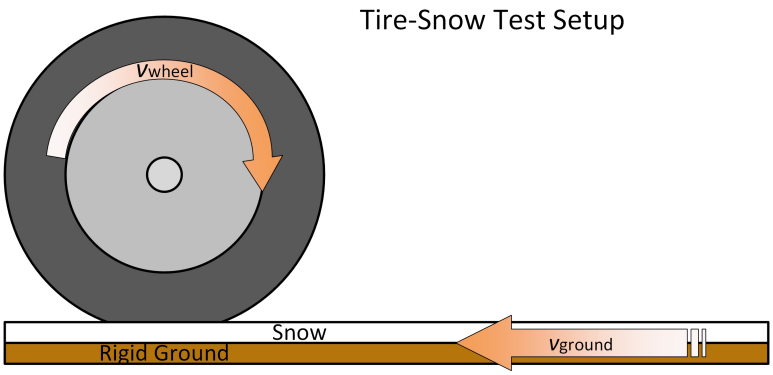

Figure 4.9: Tire-snow test setup

From this scene setup, we ran six different tests for each tire tread, for a total of 18 tests. We wanted to compare different types of snow on the different tread patterns. We also wanted to test different rotational velocities for the tires. For an accurate comparison we needed all possible combinations.

We tested the tires on two different types of snow: icy and hard snow, and soft and fluffy snow. The parameters for the two materials are listed in Table 4.3. By comparing different types of snow, we could infer the performance of the different tread patterns. Our focus is on gauging the amount of snow thrown from the tire, the amount of snow stuck on the tire, and the change due to slipping. We also compared the stresses on each of the tire treads. We expect that the icy and hard snow would stick less to the tire and break apart less, whereas the soft snow would stick to the tire more and break apart more. We also expect that if a tire was spinning faster than the ground, the amount of snow stuck on the tires would be less, and the amount of snow displaced would be more. We also compare the computational time to determine if there is a dependence on the interfacial contact of the tread pattern.

The friction between a tire and snow is very important in correctly modeling traction. As a tire rolls over snow, if it maintains traction the entire time, there is no slipping. If a tire loses traction, it will slip. To translate this to our one-wheel model, we are

Variable	**Hard Snow**	**Soft Snow**
Density (g/cm^3)	0.4	0.33
Bulk Modulus K(MPa)	0.078	0.0267
Shear Modulus G(MPa)	0.0583	0.02
Hardening Coefficient	15	10
Nonlinear Hardening Coefficient	0.02	0.02
Initial Yield (MPa)	1x10^{-4}	1x10^{-4}

Table 4.3: The snow material parameters used in the tire-snow interaction tests.

comparing the speed at which the tire is spinning to the speed at which the ground is moving. We set both the ground and tire to a constant speed, so if the tire is moving at a slower speed than the ground, it would be equivalent to the tires spinning slower than the car is moving. As a result, the difference between the speed of the ground and the speed that the tire is rotating is the amount of slipping that is occurring. We ran three different tests for each tire tread. In all cases we set the ground speed to 10mph. We then set the speed at which the tire was spinning to 5mph, 10mph and 20mph.

Chapter 5: Results and Analysis

5.1 Snow-Snow Interaction

Our initial tests were focused on developing our snow model. We knew that without correctly modelled snow, we could not apply the model to any other situation. Our first test was to hit two snowballs together. Figure 5.1 shows a frame from that simulation. We found that our model was harder to fracture than the clamped neo-Hookean model, and we determined that while our material has a hardening law, we do not have a corresponding softening law that dictates the fracture of the material.

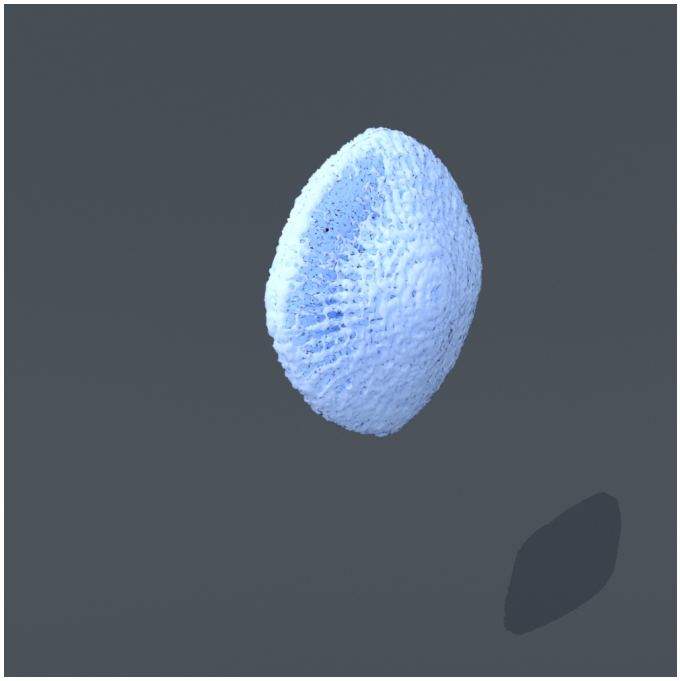

Figure 5.1: A snowball colliding with another snowball.

In the second test, we threw a snowball into a rigid wall. Once again, we found that recreating the fracture of Stomakhin et al. proved difficult. When thrown at higher

speeds, the outer materials would shatter into a lot of pieces, but the core material would generally stay in tact, leaving an unnatural lump around the center. We believe that having exact spheres determine the layers created this unnatural phenomenon. Figure 5.2 shows the snow colliding with the wall.

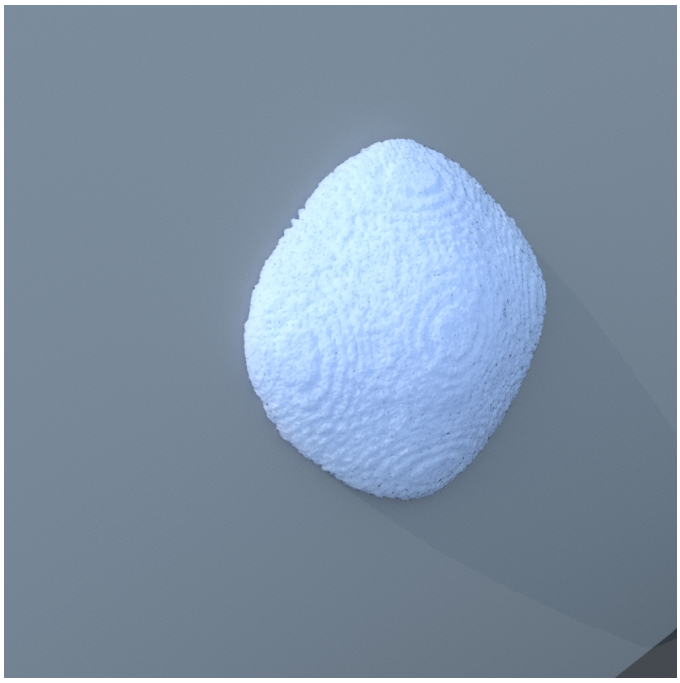

Figure 5.2: A snowball colliding with a wall.

Snow requires very little strain to fracture. With our third test, our goal was to recreate a uniaxial load test on a small bar of snow. We noted in our previous two tests that recreating fracture was difficult to do. This test was important to help determine how much force was required to pull the snow apart. When we applied a load of 0.01N, we can clearly see in Figure 5.3 that the snow stretches out thinly before fracturing. We also noticed that the strain does not occur near the middle of the material, but right on the edge of the load. We are not sure if the strain is being properly propagated throughout the material, or if this is what we should be seeing.

To compare our model, we used an implementation of the model proposed by Stomakhin [22]. This model, which is called clamped neo-Hookean in NairnMPM, adds fake fracture points that ignore the physics of plasticity. Figure 5.4 compares the same time

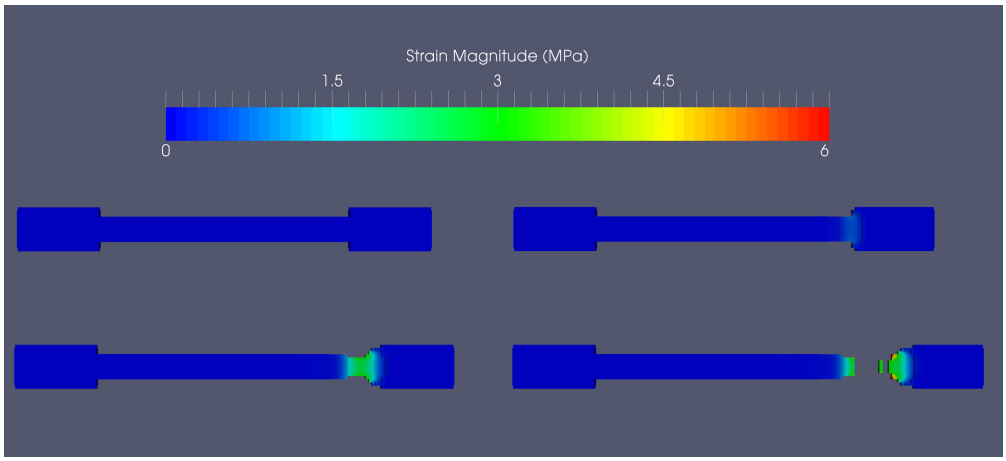

Figure 5.3: The uniaxial snow stretching strain test results.

step of the clamped neo-Hookean and our hyper elastic-plastic material, with the same setup other than the material model. Notice the complete fracturing of the clamped neo-Hookean, where nothing is bonded to anything else after collision.

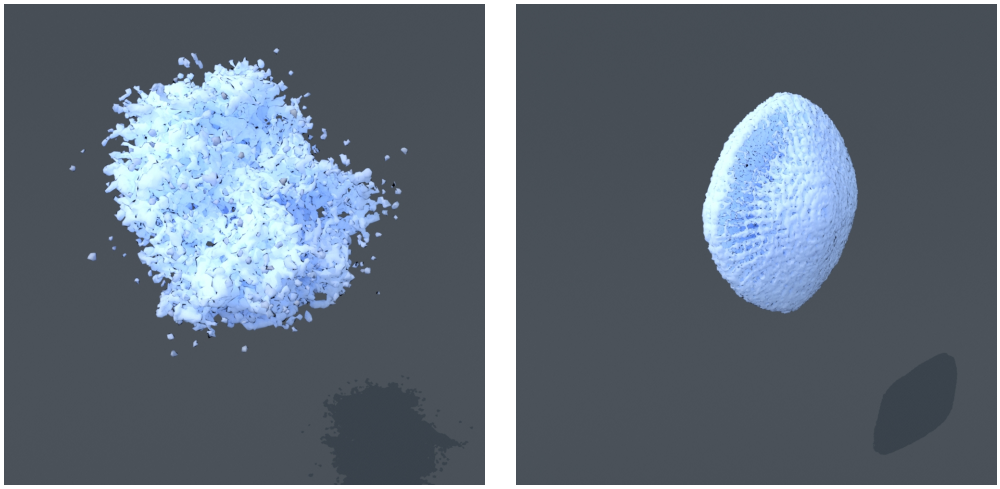

Figure 5.4: A comparison of clamped neo-hookean and our hyper-elastic plastic model.

Tread	Hard snow	Soft snow	Ratio
Angled Footstep	10938	14223	1.3
Bowtie	6870	11137	1.6
Six Rib	19655	15657	0.79

Table 5.1: Total amount of snow particles displaced after 20 times steps (10ms) when the ground and tires are moving at the same velocity.

5.2 Snow-Tire Interaction

After iterating our snow material, we applied it to the snow-tire test setup.

When we compared the amount of snow displaced by the tire treads, we counted up all the particles that had been gone a certain distance away from the main clump of snow after 20 time steps. The results of the summation are shown in Table 5.1.

For the angled footstep and bowtie patterns, we compared how much snow is displaced over a period of rolling. Our results are depicted in Figure 5.5. If we compare the speed of the tire to the speed of the ground, we can see that the faster tire, the black and the magenta curves, displace more snow than the slower tire, the blue and the red curves. This is what we expected. The faster the tire spins, the more snow that gets displaced. When comparing the angled footstep tread pattern to the bowtie tread pattern, the biggest difference is the spread between the fast tire and the slow tire. The bowtie tread pattern has a much tighter spread than that for the angled footstep, which means that it is less affected by how fast the tire is moving.

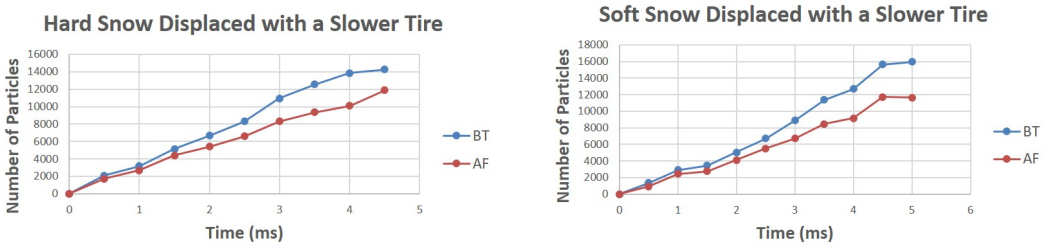

Figure 5.5: Displacement results over time for soft and hard snow.

We also wanted to see which tire tread accumulates the most snow on the treads. We measured this by counting up the particles that were inside the radius of the tire treads. The tires start in the snow, so the graphs all start at the same value. In Figure 5.6,

the plot of the snow particles accumulated on each tire tread is shown. If we look at the red curve on both of the graphs, the slow tire running over soft snow has the most snow stuck to it. We can also see that the red curve stays almost flat with the angled footstep pattern, which means the tread got rid of snow at the same rate that snow was introduced. Looking at the fast tire running over hard snow, the black curve, we see that it has the least snow stuck in its treads. This aligns with our expectations. A faster tire would throw off more snow, and harder snow would stick more to itself than a tire. The black curve consistently slopes downward, which means the tire is expelling snow faster than the snow introduced.

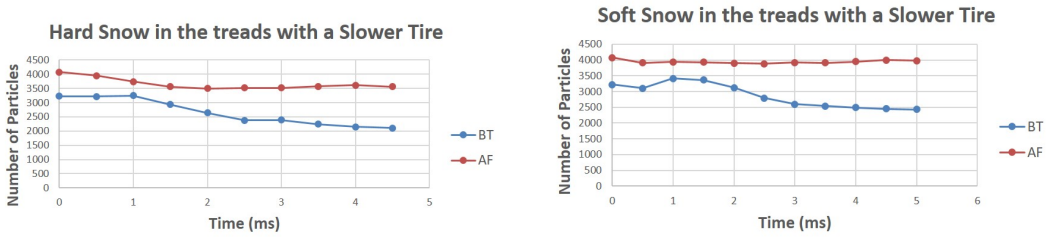

Figure 5.6: Accumulation results over time for hard snow and soft snow.

It is also important to see what effect the tread has on the tire itself. Our ground is a rigid surface, and the stress distribution over the footprints of each tire is shown in Figure 5.7. There are high stress concentration areas for each pattern. In the angled footstep tread pattern, the highest stress zones are on both the tread blocks as shown in the left image. For the bowtie tread pattern, the central block has higher stress than the two neighboring triangular shaped blocks. In the six rib design, the high stress areas are on the side ribs. The rigidity of the tire was kept the same for all three simulations. Without the capability of inputting tire internal air pressure in our current phase of research, our observations on stress distribution only provide a qualitative understanding of the effects of the different tread block designs.

5.3 Rendering Comparison

Our first rendering method was a point based method. The output of our simulation was the particles' locations, so the simplest way to visualize this is to render a sphere

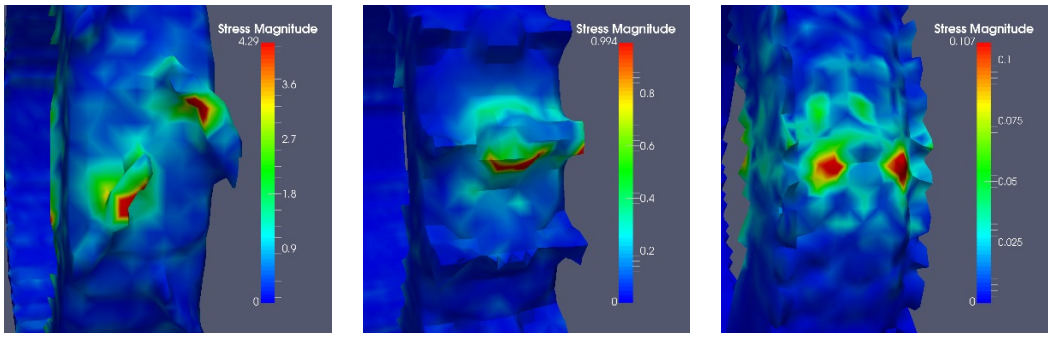

Figure 5.7: A close look at the stresses of the three tire treads.

at every point. This is a very fast method, and very easy to implement, but it does not look realistic. Our next step was to create a surface around the points using surface reconstruction. Surface reconstruction gave us a more realistic look at the simulation, but it looked more like ice cream than snow. In Figure 5.8, the point based method and basic surface reconstruction methods are compared.

Recall from equation 4.10 that the scalar field values are dependant on the radius of the particles and the radius of the neighborhood. Then the surface is created based on the value that we choose to define the surface and the sampling resolution. We generally sampled the scalar field at twice the resolution of the simulation grid, and then the radius of our neighborhood was approximately one sample grid cell away. Because we chose 27 material points in each cell of the grid, the radius of our particles was set at approximately $\frac{1}{27}^{th}$ the size of the simulation grid, so they would sum up to be one whole grid cell. We chose a small number between 0-1 for the surface, as we found this would not skip any parts of the surface.

The surface appearance was too smooth to be snow, so we knew we needed to enhance the surface. Notice in Figure 5.8 that the surface reconstructed snow balls have a line around where the simulation grid is. We knew that we needed to get rid of any mathematical phenomena like this so the first step we took was to lightly smooth the surface even more. We used a uniform mesh smoothing iterated three times. We felt the clean slate of the smoothed surface was a necessary starting point. Our first step away from the smoothed appearance was to add noise to the surface vertices. This created a more natural appearance, but it lacked the real crystalline appearance that defines snow.

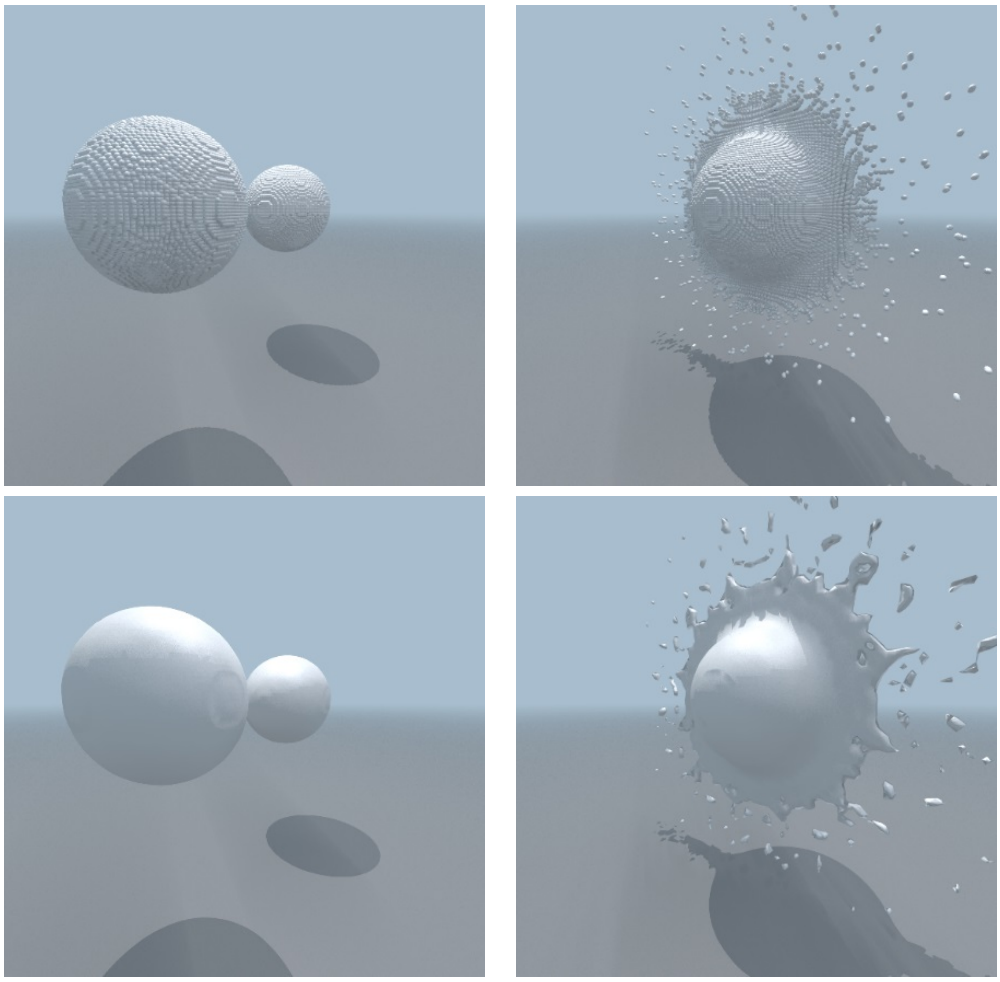

Figure 5.8: A comparison between simple sphere rendering, as seen on the top row, and surface reconstruction, shown on the bottom row. This rendering was done using POVRay.

Figure 5.9 presents a comparison between the base surface reconstruction, smoothing, noise and smoothing with noise.

To get the crystalline appearance we desired, we added the surface hoar crystals. We created hexagonal prisms on the surface of the mesh, which transforms the smooth surface into the crystalline surface we desired. Figure 5.10 shows a final result, without internal scattering. Mitsuba crashed whenever we attempted internal scattering with

(a) (b) (c)

Figure 5.9: A comparison between base surface reconstruction, smoothing and noise. This rendering was done using Mitsuba. (a) is the base surface, (b) is the smoothed base and (c) is the smoothed surface with added noise.

this model.

The biggest downside to adding noise and hoar crystals using this method was the lack of temporal coherence. Every time step of the simulation creates a whole new mesh, and we currently have no way to track where the noise and crystals should be placed in the same position as before. While these still frames fail to capture this, when an animation is run using these techniques, the surface changes every frame. This is an undesired trait.

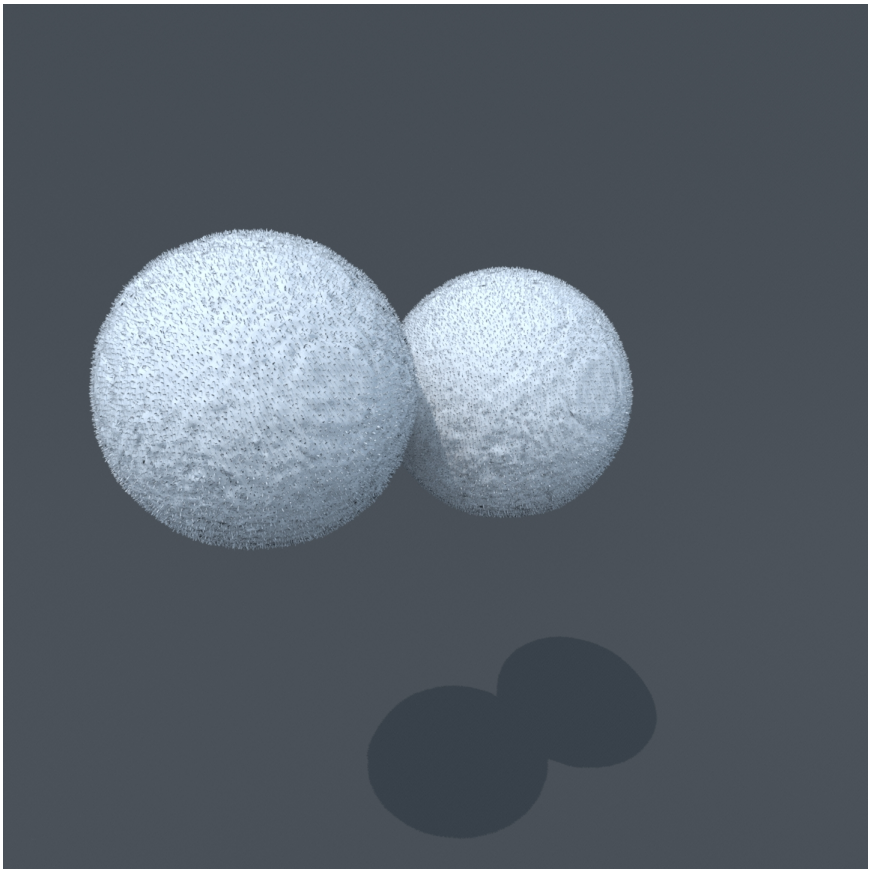

Figure 5.10: An example of our snow crystallized rendering. This rendering was done using Mitsuba.

Chapter 6: Conclusion

In this thesis, we have demonstrated a survey of the applicability of MPM to tire-snow interactions. In addition, we proposed a method of rendering snow from particle simulation data and developed a material model to simulate snow.

6.1 Simulation

Our results are an initial application of tire-snow interactions, but they are still indicative of effective tire snow tread patterns. As the model is improved, the usability of applicability of this method will greatly increase. However, currently there are areas the simulation is lacking. Snow modelling is very complex, and while our method shows promise, we need to further examine properties and techniques for improving it. Tires are not generally modelled using particles. Simulating tires with particles causes the tire to be imprecise and not well defined. Our six-rib tread pattern turned out to be more complicated than our simulation could handle. Images in the appendix show that the spaces in between the treads were mostly lost. The imprecise nature of pure material point method when modeling complex geometry may indicate creating a hybrid method would work better for tire-snow interactions in the future.

6.2 Rendering

Our theory on snow surface reconstruction worked generally well. We knew that snow surfaces are complex for the same reason that snow physics are complex. The ice matrix inside the snow complicates both the simulation and rendering aspects. We started from the established rendering method for particle simulations of fluids and enhanced the surface to appear more snow like. The problem we ran into is that while we know snow has a rough and varying surface, characterizing it was a challenge. The solution we provide is just one path we could have taken. Our results show a convincing method for recreating the crystalline surface of snow.

6.3 Future Work

Our simulation and rendering has generated compelling results, but the problem is far from solved. We hope to continue our work in the following ways.

While we were able to use an established simulation software to have more accurate equations, there were some limitations in the material models we had access to. In the future we want to compare the Drucker-Prager hardening model to the hardening model we use by implementing it as an additional hardening law in the MPM software. We would also benefit from some heterogenous material parameters, so the snow would clump together coherently.

Snow found in nature is rarely ever pure snow. Most of the time, there are other materials inside the clumps of snow, such as water, or solid blocks of ice. Another addition we could make to this research is to simulate water and snow mixed together. Beyond improving the snow, the road itself could be more complex. It is currently just a rigid material underneath the snow, but a real road would have dirt, salt and debris on it, and would have its own level of deformation under the weight of the car.

The car would be another extension of the simulation. Right now, we only simulate one tire rolling over the snow, which does not simulate the complexity of the car's axle or wheel socket. Nor does it really put into account the weight of the car pressing down. If we were able to simulate the wheel socket or the force of the car pressing down, the versatility of our simulation would be improved..

We could enhance our rendering results by finding and using established snow crystal distributions with some stochasticity to perturb the surface of the snow, as well as change the internal scattering of light.

Bibliography

[1] How Do Weather Events Impact Roads? - FHWA Road Weather Management, May 2016.

[2] Cameron Chrisman. Rendering Realistic Snow. *Unpublished Abstract.*

[3] Robert L. Cook, Thomas Porter, and Loren Carpenter. Distributed Ray Tracing. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '84, pages 137–145, New York, NY, USA, 1984. ACM.

[4] Daniel Charles Drucker and William Prager. *Soil Mechanics and Plastic Analysis Or Limit Design.* Division of Applied Mathematics, Brown University, 1951.

[5] C. Fierz, R.L. Armstrong, Y. Durand, P Etchevers, E. Greene, D.M. McClung, K. Nishimura, P.K. Satyawali, and S.A. Sokratov. The International Classification for Seasonal Snow on the Ground, 2009.

[6] Akihiro Fujimoto, Hiroshi Watanabe, and Teruyuki Fukuhara. Effects of Tire Frictional Heat on Snow Covered Road Surface. In *PROCEEDINGS OF THE 13TH SIRWEC CONFERENCE*, Torino, Italy, 2006.

[7] Yoshinori Furukawa and John S. Wettlaufer. Snow and ice crystals. *Physics Today*, 60(12):70–71, December 2007.

[8] Robert B. Haehnel and Sally A. Shoop. A macroscale model for low density snow subjected to rapid loading. *Cold Regions Science and Technology*, 40(3):193–211, December 2004.

[9] Wenzel Jakob. *Mitsuba renderer.* 2010. http://www.mitsuba-renderer.org.

[10] T. U. Kaempfer, M. A. Hopkins, and D. K. Perovich. A three-dimensional microstructure-based photon-tracking model of radiative transfer in snow. *Journal of Geophysical Research: Atmospheres*, 112(D24):D24113, December 2007.

[11] Martin Krolk. *Realistic Snow Rendering.* Master's Thesis, Czech Technical University, Prague, January 2014.

[12] Jonah Lee. Mechanical properties of snow as a random heterogeneous material using UINTAH, March 2008.

[13] Jonah H. Lee. Finite element modeling of interfacial forces and contact stresses of pneumatic tire on fresh snow for combined longitudinal and lateral slips. *Journal of Terramechanics*, 48(3):171–197, June 2011.

[14] Thomas Lewiner, Helio Lopes, Antonio Wilson Vieira, and Geovan Tavares. Efficient implementation of Marching Cubes' cases with topological guarantees. *Journal of Graphics Tools*, 8:2003, 2003.

[15] Lin Li, Corina Sandu, Jonah Lee, and Bradford Liu. Stochastic modeling of tires-now interaction using a polynomial chaos approach. *Journal of Terramechanics*, 46(4):165–188, August 2009.

[16] Fei Liu. *An Illumination Model for Realistic Rendering of Snow Surfaces.* Uppsala universitet, Institutionen fr informationsteknologi, November 2009.

[17] William E. Lorensen and Harvey E. Cline. Marching Cubes: A High Resolution 3d Surface Construction Algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, pages 163–169, New York, NY, USA, 1987. ACM.

[18] Gunther Meschke and Changhong Liu. Large strain finite-element analysis of snow. *Journal of Engineering Mechanics*, 122(7):591, July 1996.

[19] Matthias Mller, David Charypar, and Markus Gross. Particle-based Fluid Simulation for Interactive Applications. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 154–159, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[20] John Nairn. nairn-mpm-fea, May 2016.

[21] Tomoyuki Nishita, Hiroshi Iwasaki, Yoshinori Dobashi, and Eihachiro Nakamae. *A Modeling and Rendering Method for Snow by Using Metaballs.* 1997.

[22] Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. A Material Point Method for Snow Simulation. *ACM Trans. Graph.*, 32(4):102:1–102:10, July 2013.

[23] D. Sulsky, Sj Zhou, and Hl Schreyer. Application of a Particle-in-Cell Method to Solid Mechanics. *Computer Physics Communications*, 87(1-2):236–252, May 1995. WOS:A1995RB38900015.

[24] Konrad Walus and Zbigniew Olszewski. Analysis Of Tire-road Contact Under Winter Conditions. In *Proceedings of the World Congress on Engineering 2011*, volume 1, pages 1546–1550, January 2011.

[25] Lawrence A. Wood, Norman Bekkedahl, and Frank L. Roth. Measurement of densities of synthetic rubbers. *Journal of Research of the National Bureau of Standards*, 29(6):391, December 1942.

[26] Peter Wriggers and Jana Reinelt. Multi-scale approach for frictional contact of elastomers on rough rigid surfaces. *Computer Methods in Applied Mechanics and Engineering*, 198(2126):1996–2008, May 2009.

[27] Yongning Zhu and Robert Bridson. Animating Sand As a Fluid. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 965–972, New York, NY, USA, 2005. ACM.

APPENDICES

Appendix A: Results

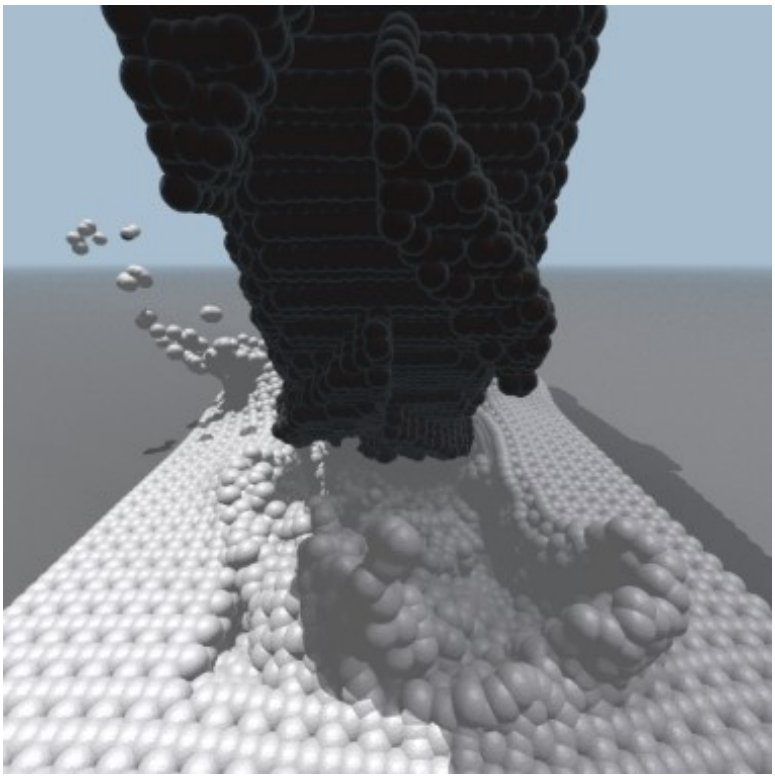

Figure A.1: The point based rendering of the angled footstep pattern in POVRay.

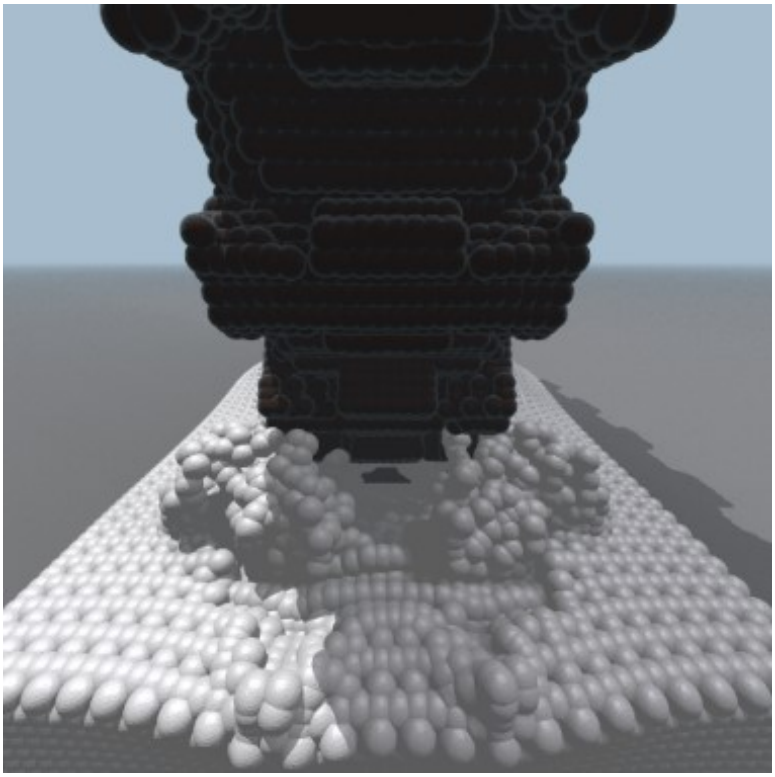

Figure A.2: The point based rendering of the bowtie pattern in POVRay.

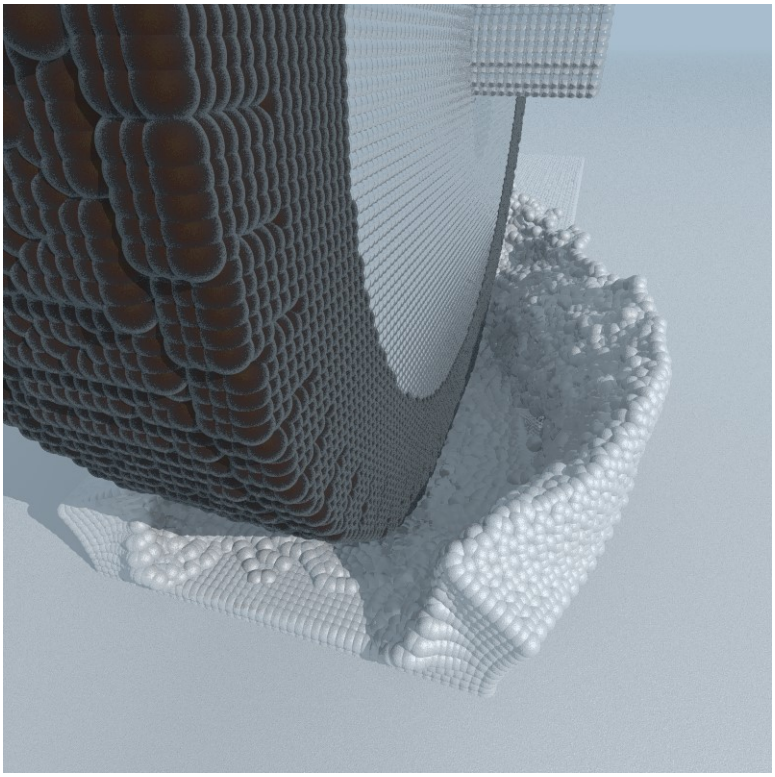

Figure A.3: The point based rendering of the six rib pattern in POVRay.

Figure A.4: The mitsuba rendering of snowballs before collision without the surface crystallization.

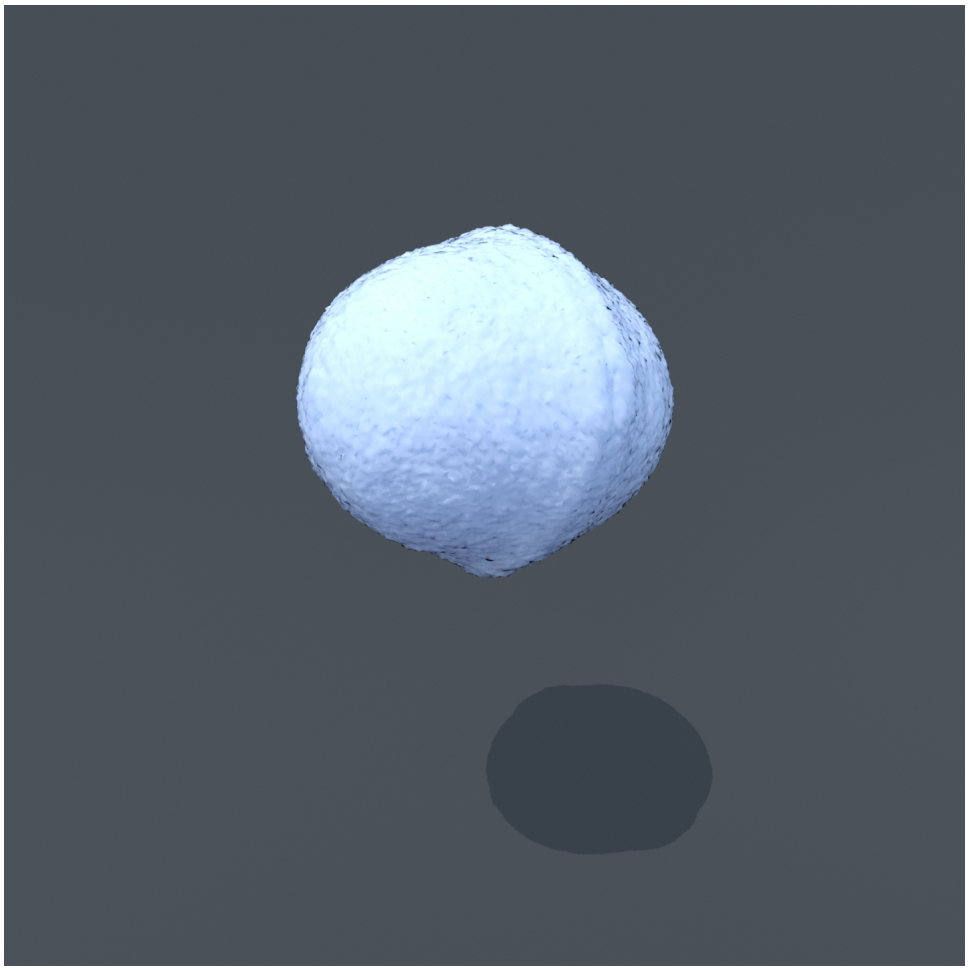

Figure A.5: The mitsuba rendering of snowballs mid collision without the surface crystallization.

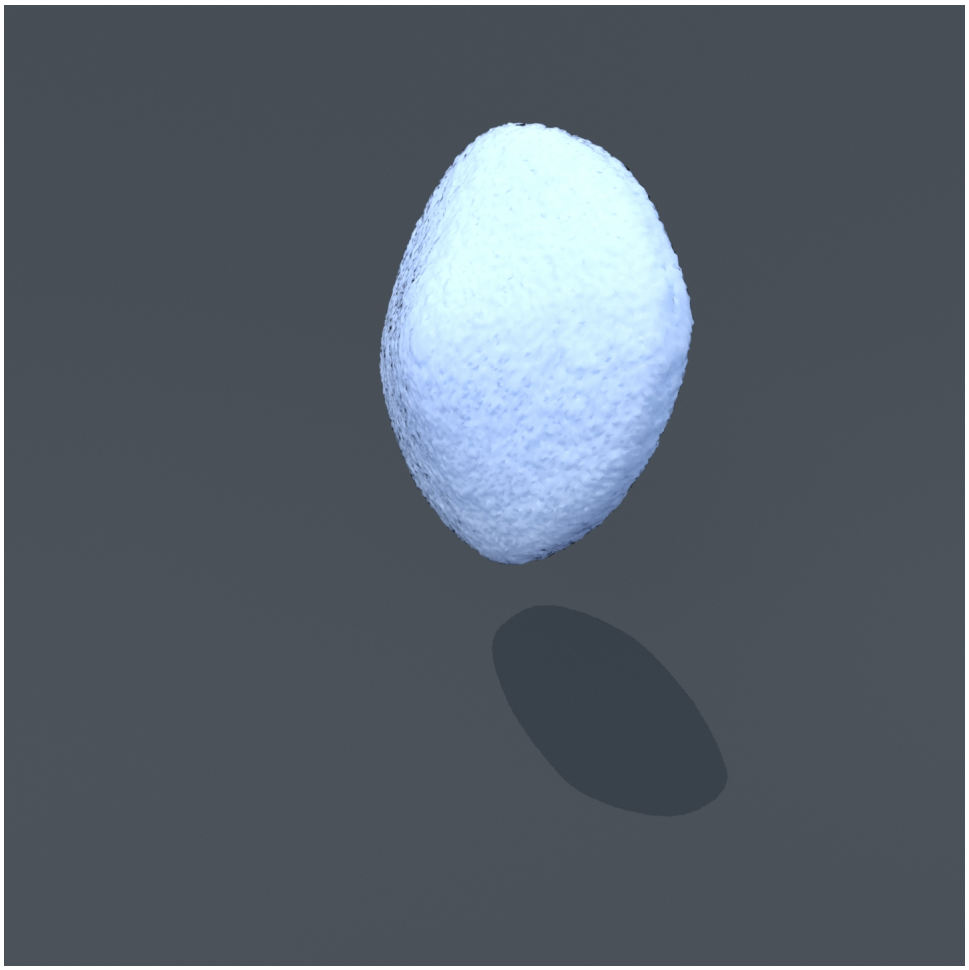

Figure A.6: The mitsuba rendering of snowballs mid collision without the surface crystallization.