PROCEDURAL SURFACE WEATHERING OF CULTURAL STONE THROUGH PHYSICALLY BASED MESH DEFORMATIONS

A Thesis

by

SCHAEFER ARON MITCHELL

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Ann McNamara
Co-Chair of Committee,	Philip Galanter
Committee Member,	Charles W. Smith
Head of Department,	Tim McLaughlin

August 2017

Major Subject: Visualization

ABSTRACT

Weathering in computer graphics is crucial to creating believable objects. Common methods to add wear to an object include the application of 2D texture maps or direct model deformations through sculpting. These methods require an artist’s time and expertise, often at a cost to quality or iteration time for the asset. This creates a need for a method that is quick to iterate upon and easy to manipulate.

This thesis outlines a new procedural method that combines principles of 2D texture maps and 3D sculpting to achieve large-scale mesh deformations in stone surfaces. The method involves the use of procedural 3D noise to define a stone’s composition combined with particle systems to simulate rainfall on an object, deforming it over time. A custom user interface with preset stone options is implemented to reduce the level of expertise needed to use the system.

CONTRIBUTORS AND FUNDING SOURCES

This work was supervised by a thesis committee consisting of Philip Galanter and Ann McNamara of the Department of Visualization and Charles Wayne Smith of the Department of Archeology. All work for the thesis was completed by the student, under the advisement of Philip Galanter of the Department of Visualization.

There are no outside funding contributions to acknowledge related to the research and compilation of this document.

NOMENCLATURE

CG	Computer Graphics
3D	Three-Dimensional
2D	Two-Dimensional
UV	Two-Dimensional coordinate system used to place image maps
VDB	Volumetric, dynamic grid
UI	User Interface

TABLE OF CONTENTS

Page

Page

LIST OF FIGURES

Page

Page

LIST OF TABLES

Page

1. INTRODUCTION

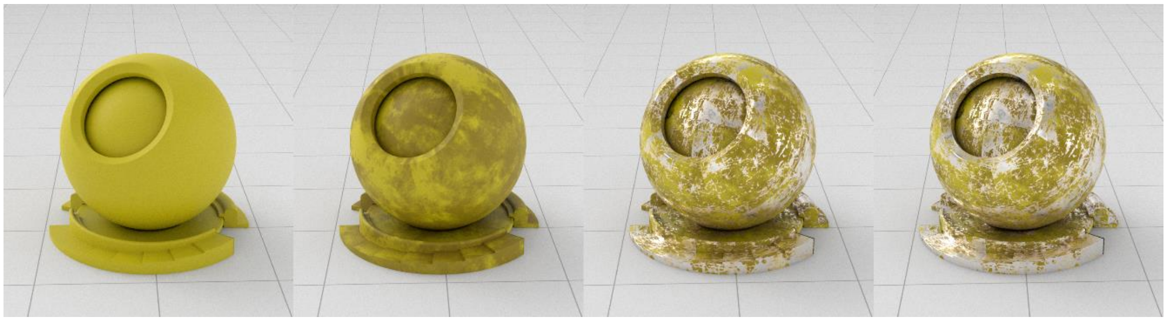
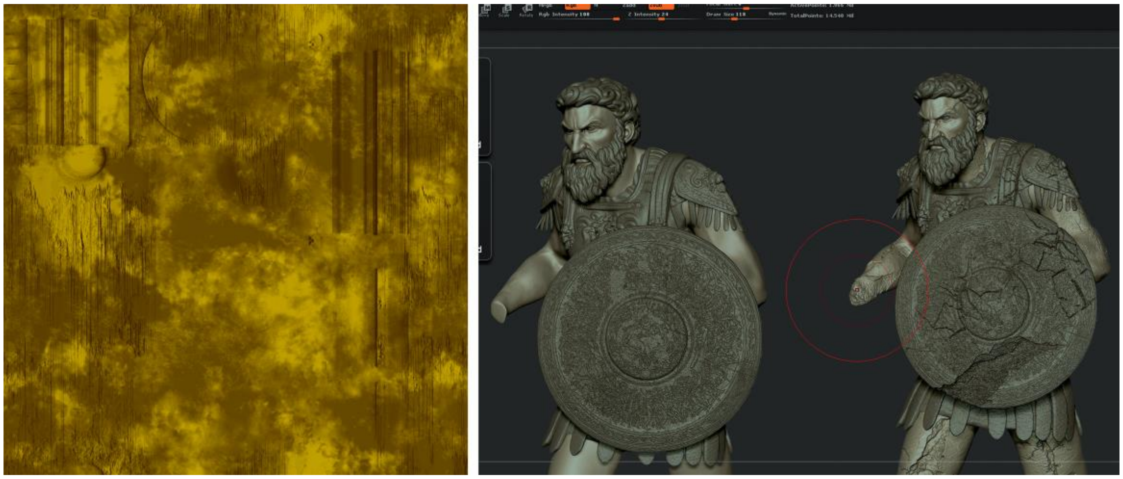


Figure 1.1: Effect of adding weathering through dirt, scratches, and edge wear to achieve a believable surface.

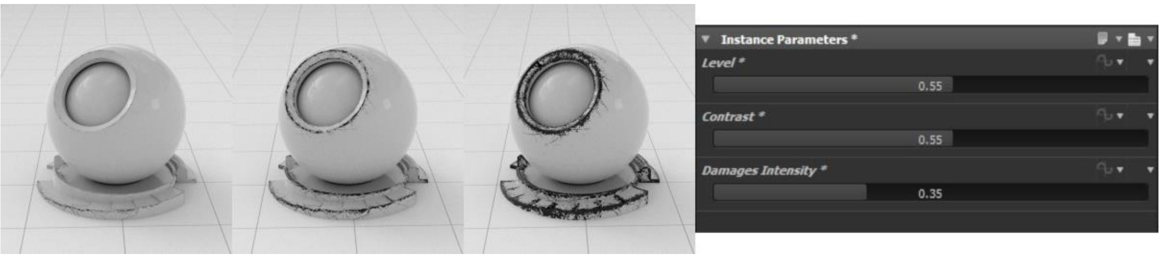
Weathering in Computer Graphics (CG) is crucial to creating believable objects. The absence of wear and tear on a surface results in objects appearing perfect and clean, reducing believability. Figure 1.1 displays the progression of adding wear and tear to a painted metal object. The addition of small detail to simulate weathering on CG objects is a compelling way to add realism to an object and increase viewer immersion in the environment as a whole. In the real world, all objects are weathered through natural forces (wind, rain, snow, or sun), or through human forces (use, denting, scratching, or pollution). A current solution for adding smaller scale weathering is to add the detail using a texture map. A texture map is a 2D image that is applied to a 3D object. Figure 1.2a, the diffuse texture map for the object in Figure 1.1, is applied to the surface using UV coordinates much like how a globe is represented in a map. Texture maps are good for simulating subtle weathering effects such as dust, small scratches, dirt, or other color variations through use. Another solution is to model, or sculpt in, the weathering patterns, much like how physical clay sculpting works (Figure 1.2b). Both of these solutions require an artist’s time and expertise, which could cost quality in other departments or assets and may reduce the number of iterations possible.



(a) *(b) Carrie Mok [30]*

Figure 1.2: (a) 2D texture map of dirt. (b) 3D sculpted stone weathering patterns.

One remedy to this is to work procedurally. By providing a program with values and expressions automated results are generated. While a procedural solution does require more time in front-end development; the outcome of a procedural solution can be replicated many times using different variables to create many different results very quickly. One program that is currently using a procedural weathering system for texture mapping is Allgorethmic's Substance Designer [1]. In Allgorethmic's system, the user defines an object and then connects preset nodes together to create a texture map that is weathered based on the 3D object and the user's set parameters. An example is shown in Figure 1.3 [1]. Although their solution is very powerful in generation of 2D textures, the program cannot generate deformations in the mesh. This limits its application to smaller scale weathering patterns. This thesis develops a solution that employs procedural methods, such as those used by Substance Designer, with the mesh deformations achieved through sculpting. This allows for a new approach to facilitate easier deformation and weathering of stone objects. In addition this approach will be fast, repeatable and accessible to novice users.



Instance Parameters *

Level * 0.55

Contrast * 0.55

Damages Intensity * 0.35

Figure 1.3: Edge wear mask using the smart weathering nodes in Substance Designer.

2. SIGNIFICANCE

An efficient production environment enforces strict time constraints and requires a mechanism to enable rapid feedback, quick edits and fast turnaround times. The method presented in this thesis has an iterative nature which makes it ideal for use by a 3D artist in such a pressurized environment. Using procedural noise and particle systems allow for easy pattern manipulation. Results are unique for each simulation. This new approach is valuable for the efficient creation of many weathered objects for use in a 3D environment. This method for creating weathered objects offers advantages relative to current modeling methods. Improvements include increased flexibility through procedural workflows, the ability to test multiple outcomes from the method's iterative nature, reducing the user's software expertise through an easy to navigate UI, and reducing the research time by implementing preset stone parameters.

3. BACKGROUND

3.1. Lithology of Stone

This section explores the lithology of a stone. Lithology refers to the properties of stone visible by the naked eye [3]. For this thesis, the sections of lithology that will be used include:

- Stone type
- Stone texture
- Grain size and shape

To provide a quick and iterative workflow, lithological properties are used to define stone rather than microscopic detail. Using microscopic detail to define the stone would exponentially increase the simulation time without a great visual change.

3.1.1. Stone Type

Rocks can be categorized broadly by the in which the rock formed. The formation processes result in several changes in the rock such as: the types of minerals in the composition, grain shape and size, compression of the grains, and how the grains correlate with each other. The three rock types are:

- Sedimentary rock
- Igneous rock

- Metamorphic rock

3.1.2. Stone Texture

Stone texture refers to the shapes of grains in the composition, how the grains are arranged, and the size of the grains in relation to each other [32]. The texture is formed by either the physical transport processes for most sedimentary stones or by chemical or biochemical processes for most igneous and metamorphic stones. For individual grains, the texture includes:

- Grain size
- Grain shape

The textural properties describing interrelationships of the grains are:

- Hardness
- Porosity
- Permeability

The texture and composition of a stone define its main classification. For example, granite and pegmatitic granite are similar in composition but differ in their textures. Granite and pegmatitic granite are composed of the same minerals but pegmatitic granite contains larger crystals, giving it a pegmatitic texture. Due to the individual independence of materials and textures, this thesis separates the two elements for more control and variation in results [32].

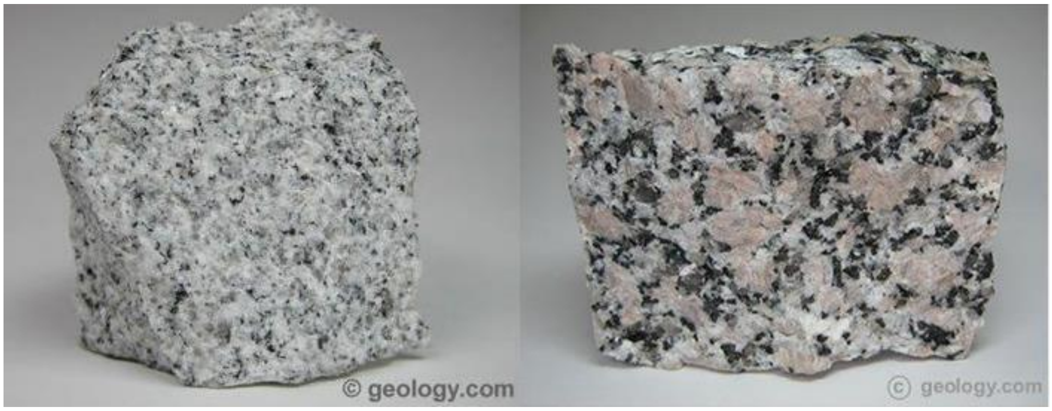


Figure 3.1: (a) A sample of granite. (b) A sample of pegmatitic granite [12].

3.1.2.a. Grain Size

Grain size refers to the size of the particles in a stone sample. Moving from left to right in Figure 3.2, the size of grains can range from large particles to small, clay-sized particles. Larger grains have more resistance to weathering due to the amount of energy required to transport larger particles.

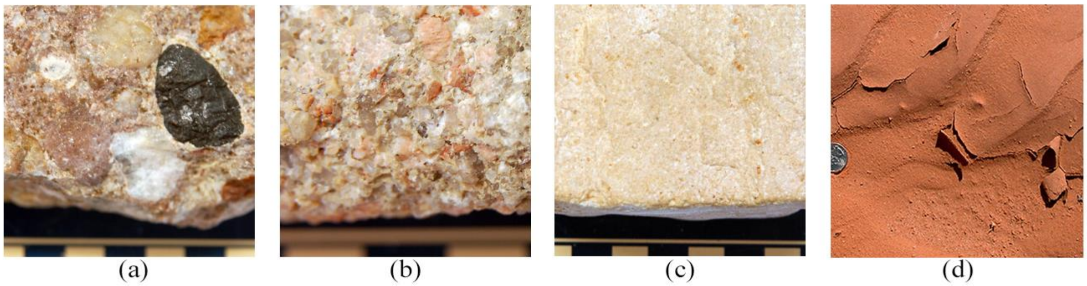


Figure 3.2: Grain size in physical stone samples [18].

The measure of the range of grain sizes in a composition and the amount of variation between them is referred to as sorting, shown in Figure 3.2. Grains that are well sorted conform to mostly the same grain size while poorly sorted grain have much more fluctuation in their sizes. Sorting has a direct relationship with porosity where better sorted grains have a higher porosity value while poorly sorted grains have a lower porosity value. Porosity determines the percentage of the rock's volume occupied by pore space. A high porosity value allows increased amounts of water to flow through its grains, thus increasing the rock's rate of weathering. [3].

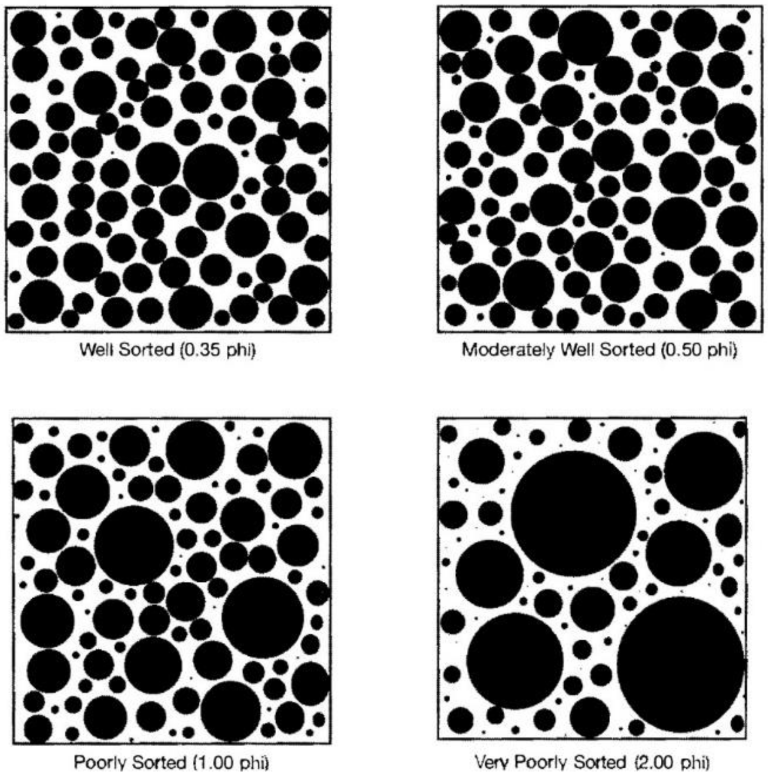

Well Sorted (0.35 phi)
Moderately Well Sorted (0.50 phi)
Poorly Sorted (1.00 phi)
Very Poorly Sorted (2.00 phi)

Figure 3.3: Particle sorting diagram [3].

3.1.2.b. Grain Shape

The rock fragments have a wide range of shapes depending on how the rock was formed. Grain shape is classified by three parameters:

- Form
- Roundness
- Surface texture

Form (Figure 3.3a) is the overall shape of the particle and is classified by its outline. The form of a particle is defined by the particle’s sphericity, or if that particle’s axes are nearing the same length. Krumbein [20] numerically describes form so that the outcome nears 1 for a perfect sphere and nears 0 for more variable forms. In the weathering process, form affects the transportability of particles. The higher the sphericity, the further that particle will be transported [3].

Particle Roundness (Figure 3.3b) differs from particle form in that roundness focuses on the shape of the particle’s edges and corners rather than its overall shape. Measuring the particle’s roundness is similar to measuring the particles form; when the formula yields values closer to one, the particle has perfectly rounded edges and corners. Particle roundness is subject to change through the weathering process with an inverse relationship with the grain’s hardness; the harder, more resistant a grain is, the less susceptible it is to becoming rounded [3].

Surface texture (Figure 3.3c) refers to the smaller dents, scratches, and markings that come from sediment transport and deposition processes. Due to the small size and shallow depth of the markings, they are susceptible to change through the weathering process. The cycle of creating and polishing away the surface texture makes limits its use when studying the origins of the grain [3].

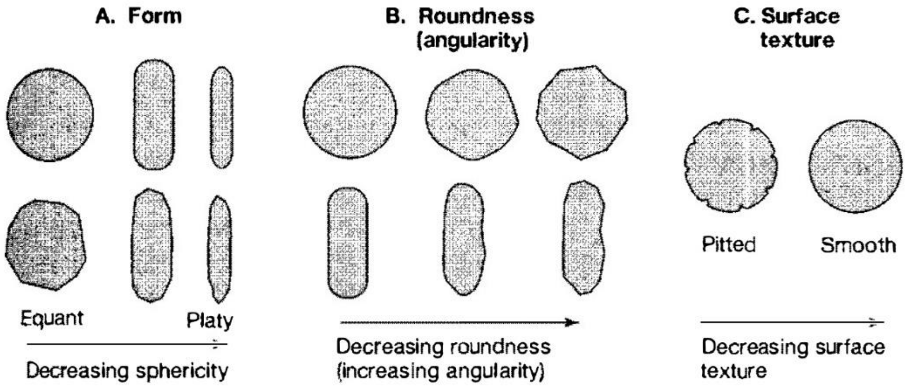

A. Form
B. Roundness (angularity)
C. Surface texture
Equant
Platy
Decreasing sphericity
Decreasing roundness (increasing angularity)
Pitted
Smooth
Decreasing surface texture

Figure 3.4: Diagrams of (a) particle form, (b) particle roundness, (c) and particle surface texture [3].

3.1.3. Hardness

Hardness is the resistance of solid minerals to the displacement of their particles, the magnitude of which constitutes their degree of hardness [39]. The most common way of quantifying hardness is by using the Mohs hardness scale. The scale was created by German mineralogist Friedrich Mohs [43]. The Mohs test (Figure 3.4a) is carried out by scratching a known mineral with a test mineral; if the test mineral creates a mark, the user can conclude that the test mineral has a higher hardness value than the control mineral. To create this scale, Mohs chose 10 different control minerals with values ranging from 1 to 10. Keeping the test as accessible as possible, Mohs chose minerals that were common and also gave alternatives: a finger nail was given a rating of 2.5 and a copper penny a hardness rating of 3.5.

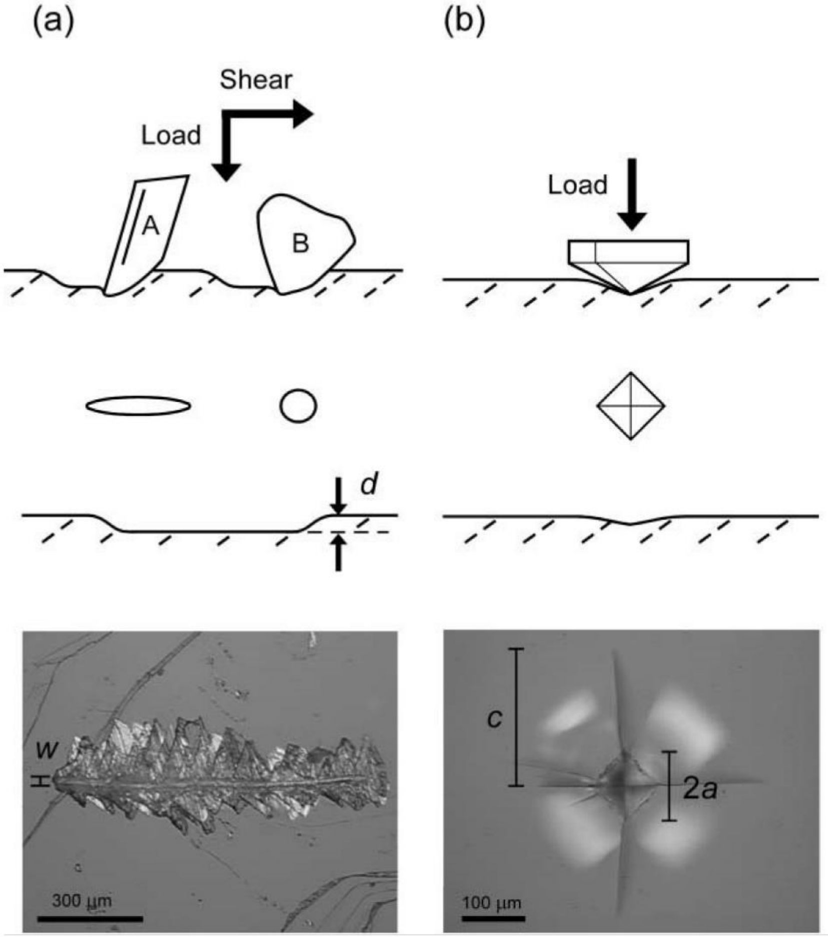


Figure 3.5: (a) Diagram of the method used for the Mohs hardness scale. (b) Diagram of the method used for the Vickers hardness scale [4].

The nature of the Mohs scale skews the hardness values used in the scale because the values are relative each other. The scale portrays an inaccurate representation of hardness for the minerals in it due to this relativity. By only including easily acquired minerals in Mohs scale, the scale resulted in a comparative index rather than a linear scale. In Figure 3.5, Broz graphs the minerals in the Mohs hardness scale with an accurate micro-hardness scale to visualize the non-linear shape of the Mohs scale.

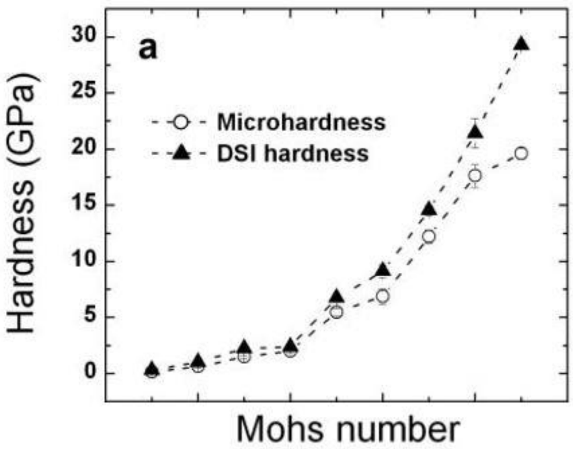
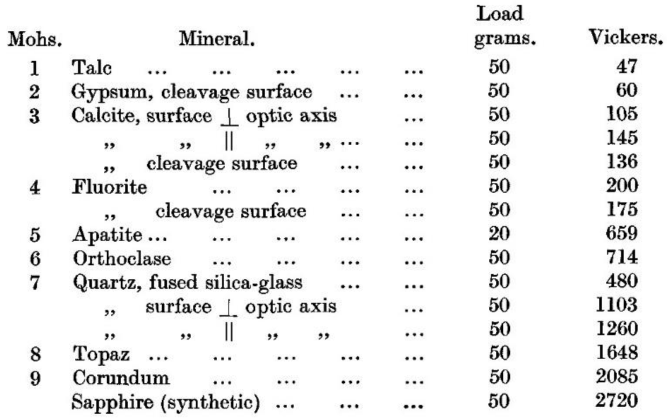


Figure 3.6: Graph plotting the correlation between the accurate microhardness scale with the minerals used in the Mohs hardness scale [4].

For this thesis, we represent hardness using floating point values, so a numerically accurate scale is required. The scale chosen is Vickers hardness scale, where the values are found by measuring an impression made by a diamond indenter in the shape of a four-sided pyramid, shown in Figure 3.4b [4]. The test is performed with a constant load force to accurately compare the impression sizes left by the indenter. The numerical hardness value is then found by dividing the load force by the surface area of the indentation. The results of the hardness found through Vickers' method provide a consistent and accurate progression of hardness through the Mohs minerals, shown in Figure 3.6.



Mohs.	Mineral.	Load grams.	Vickers.
1	Talc	50	47
2	Gypsum, cleavage surface	50	60
3	Calcite, surface ⊥ optic axis	50	105
	,, ,, ‖ ,, ,,	50	145
	,, cleavage surface	50	136
4	Fluorite	50	200
	,, cleavage surface	50	175
5	Apatite	20	659
6	Orthoclase	50	714
7	Quartz, fused silica-glass	50	480
	,, surface ⊥ optic axis	50	1103
	,, ,, ‖ ,, ,,	50	1260
8	Topaz	50	1648
9	Corundum	50	2085
	Sapphire (synthetic)	50	2720

Figure 3.7: Table giving Vickers hardness values for the minerals in the Mohs hardness scale [4].

3.1.4. Sedimentary Rock

When the Earth’s surface is weathered or eroded, by water, wind, or ice, the resulting sediment is transported and deposited. The accumulation of deposits from the transportation process over time produces many layers of sediment, creating sedimentary rocks [2].

3.1.4.a. Mineralogy

The contents, or minerology, of a sedimentary rock vary greatly depending on where the sediment samples originated. Sedimentary rock may also contain sediments from igneous or metamorphic rock. Because of this wide variation, the minerology and composition of sedimentary rock has less importance in the classification process [2].

3.1.4.b. Texture

In sedimentary rock, the shape, form, and sorting of the grains may reveal the origins of the rock. The roundness of the grain often represents the amount of time that grain was in transport, longer transport times resulting in rounder grains. If the grain's form is nearly spherical, it often suggests the grain was deposited by wind. The sorting of a grain gives detail about the rock's maturity. If the sediment contains a wide range of grain, it is immature and the sediment has been deposited rapidly [2].

The textures found in sedimentary rock [18] that are used in this thesis are:

Table 3.1: Sedimentary rock textures [18].

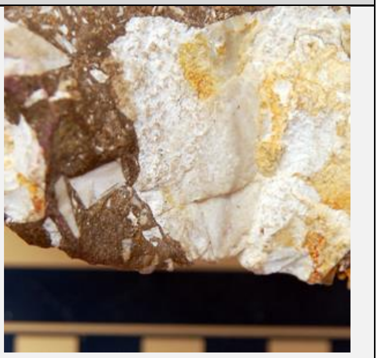
Texture	Description	Image
Large angular grain	Composition of large minerals and grains, may represent a shorter travel distance due to angularity of grain.	

Table 3.1 Continued

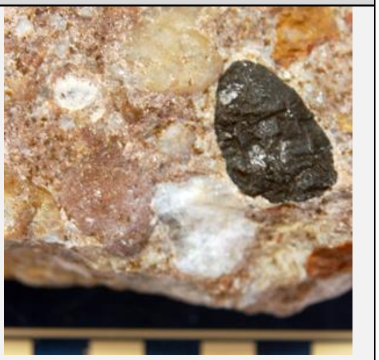
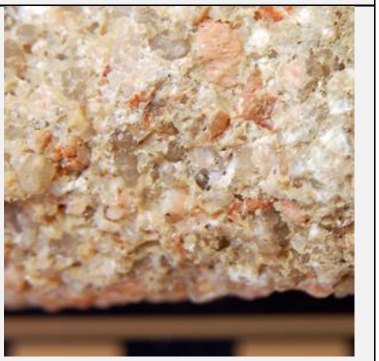
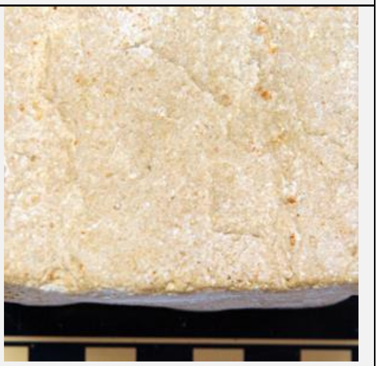
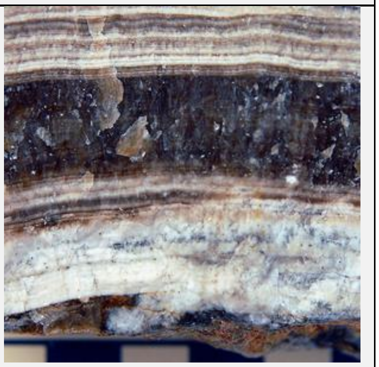
Texture	Description	Image
Large rounded grain	Composition of large minerals and grains, may represent a longer travel distance due to roundness of grain.	
Medium coarse-grained	Medium sized (greater than 2mm), angular grain.	
Medium-grained	Smaller sized grain, sand (0.06mm to 2mm).	

Table 3.1 Continued

Texture	Description	Image
Fine-grained	Small sized grain, silt (0.004mm to 0.06mm).	
Microcrystalline	Microscopic sized grain and crystals (less than 0.004mm).	
Layered microcrystalline	Layered pattern with microscopic sized grain.	

3.1.5. Igneous Rock

Igneous rocks are formed by the cooling and solidification of molten magma. The lava majorly consists of basalt with other minerals forming through differentiation. Differentiation is a process of crystallization that happens when the basalt magma reaches the upper mantle. The more time that the magma spends in this location, the greater the difference in composition between the magma and the crystals. When the liquid magma and crystals become separated, multiple rock types may result [2].

3.1.5.a. Mineralogy

The mineralogy and the proportions of minerals in an igneous rock define how the rock is classified. Igneous rock is composed mostly of silicates, which include: feldspars and feldspathoids, quartz, pyroxenes, amphiboles, micas, and olivine. The hardness of the minerals in an igneous rock range from 6 and harder on the Mohs scale with the exception being mica, rating at a 2.5 [2].

3.1.5.b. Texture

Textures in igneous rocks largely rely on the proportions and sizes of different minerals in the composition [2].

Table 3.2: Igneous rock textures [16].

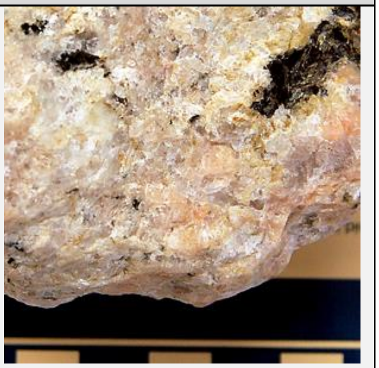
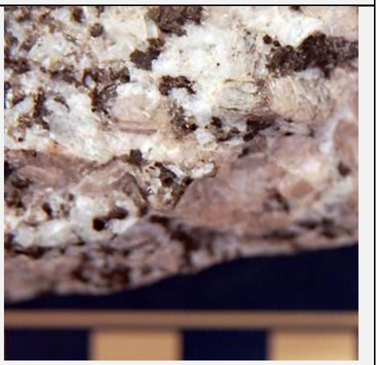
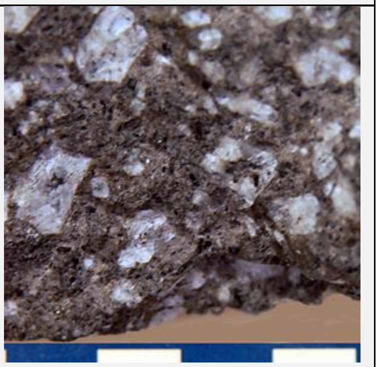
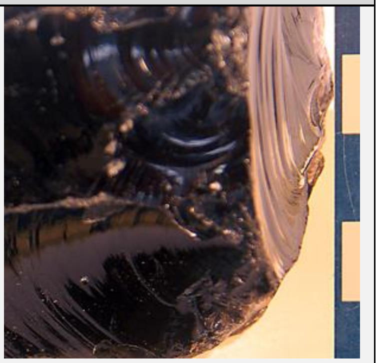
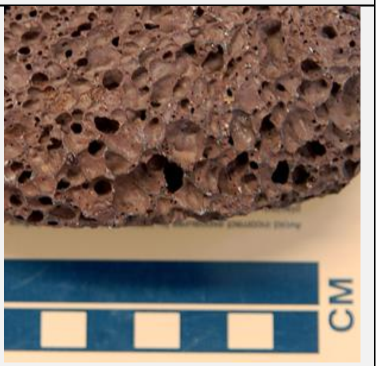
Texture	Description	Image
Pegmatitic	Similar sized grains.	
Phaneritic	Larger grains of one mineral enclose smaller grains of another.	
Porphyritic	Some large grains are set in a finer grained mix, classified based on its groundmass minerals.	

Table 3.2 Continued

Texture	Description	Image
Glassy	Crystals conform to a direction due to the flow of magma when formed.	
Vesicular	Magma forming rock cools with air bubbles trapped inside.	

The grain sizes for igneous rock have a similar scale to sedimentary rock:

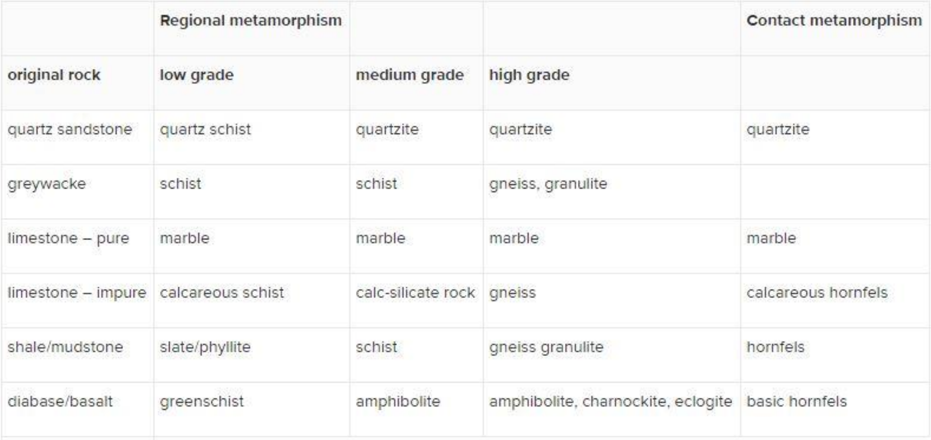
- Fine-grained: Less than about 0.1mm.
- Medium-grained: 0.1 to 2.0mm.
- Coarse-grained: 2mm up.
- Pegmatitic: Diameters of several cm.
- Glassy: Mostly consisting of glass [2].

3.1.6. Metamorphic Rock

Rock metamorphism refers to the geological process that changes an existing rock's mineralogical and chemical composition along with its structure. Metamorphic rock forms under high temperatures and large amounts of pressure, conditions usually found within the Earth's crust and mantle. The resulting rocks usually display wide ranges of texture and mineralogy due to of the range of temperatures and pressures during the metamorphic process as well as the many possible origin rocks. [5]

3.1.6.a. Mineralogy

All metamorphic rock comes from an origin rock, or protolith. Due to this, the resulting rock is determined by the protolith. The protolith mineral is changed through the amount of heat and pressure it undergoes during metamorphism. Through these changes, the mineral can be broken up into different stages, or grades, that increase during the metamorphic process. These grades and their resulting minerals are shown in Figure 3.9. [2]



	Regional metamorphism			Contact metamorphism
original rock	low grade	medium grade	high grade	
quartz sandstone	quartz schist	quartzite	quartzite	quartzite
greywacke	schist	schist	gneiss, granulite	
limestone – pure	marble	marble	marble	marble
limestone – impure	calcareous schist	calc-silicate rock	gneiss	calcareous hornfels
shale/mudstone	slate/phyllite	schist	gneiss granulite	hornfels
diabase/basalt	greenschist	amphibolite	amphibolite, charnockite, eclogite	basic hornfels

Figure 3.8: Table of resulting rock types through different metamorphic processes [2].

3.1.6.b. Texture

Heat and pressure define the process of creating a metamorphic rock and also determine the resulting texture. Much like the classification of textures in sedimentary and igneous rock, metamorphic rock can be described by grain size [2].

Table 3.3: Metamorphic rock textures [17].

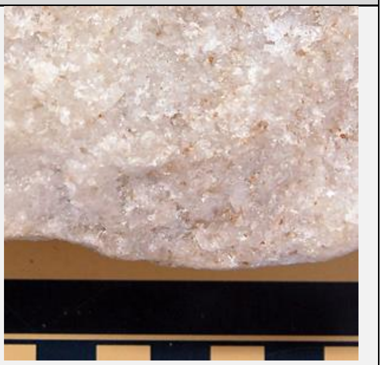
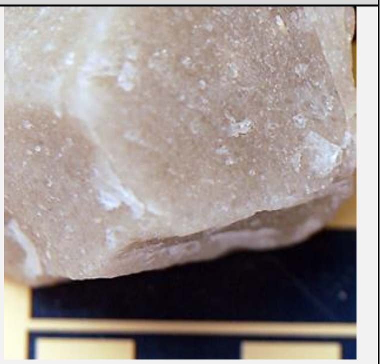
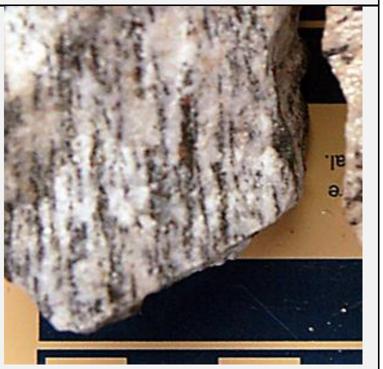
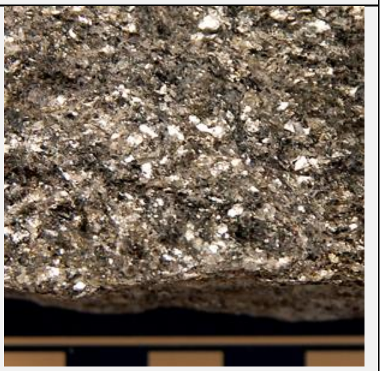
Texture	Description	Image
Macrocrystalline	Medium, similarly sized grains.	

Table 3.3 Continued

Texture	Description	Image
Microcrystalline	Smaller, similarly sized grains.	
Gneissic banding	Coarse grained, marked with an irregular layered structure.	
Schistotic	Large, scaly crystals.	

3.2. Weathering of Stone

Weathering of cultural stone, regardless of type, varies greatly depending on the environment it is in. The environment exposes the stone to varying levels of pollution, human interaction, and natural climate. Results of weathering over time physically and chemically alter the stone through discoloration, structural alteration, and surface recession. Corners become more rounded, inscriptions and detail become softer, and smooth surfaces become pitted and uneven. One of the highest contributors to accelerated rates of weathering is the amount of pollution in the stone's environment [37]. Pope found that in "pristine air" areas, such as Nevada or Hawaii, the rate of weathering on marble headstones was very low compared to cities and areas of higher human activity. One of the primary ways that pollution in the atmosphere affects the stone is through acid rain. Pope states that the pollution in the environment directly affects how acidic the rain becomes, further increasing the rate of weathering. Weathering can be broken up into three different types:

- Chemical: breakdown of a stone surface by water and pollutants [9].
- Mechanical: breakdown through physical forces such as abrasion, pressure, or the expansion and contraction of ice acting on the rock.
- Biological: breakdown from other organisms such as plants, animals, or microbes.

For this thesis, we will only be focusing on chemical weathering due to the number of varying factors in both mechanical and biological weathering. Limiting the process to only chemical weathering keeps the process focused on the flow of water over stone and how that water flow affects the stone over time. The limitation also reduces the need to include stone fracturing and the study behind the fractures as well as how different organisms break down stone over time.

During the chemical weathering process, shown in Figure 3.11, water and moisture along with the pollutants in the environment absorb into the material through the stone’s porous structure. The moisture then starts to dissolve and break down the minerals and salts that it comes into contact with; fine grains with a smaller surface area weathering quicker than larger grains. The breaking down of the minerals changes them chemically and makes them less durable, increasing their likelihood to break off and be transported by water. The final transport of the loosened material results in the recession of the stone’s surface as modeled in this thesis.

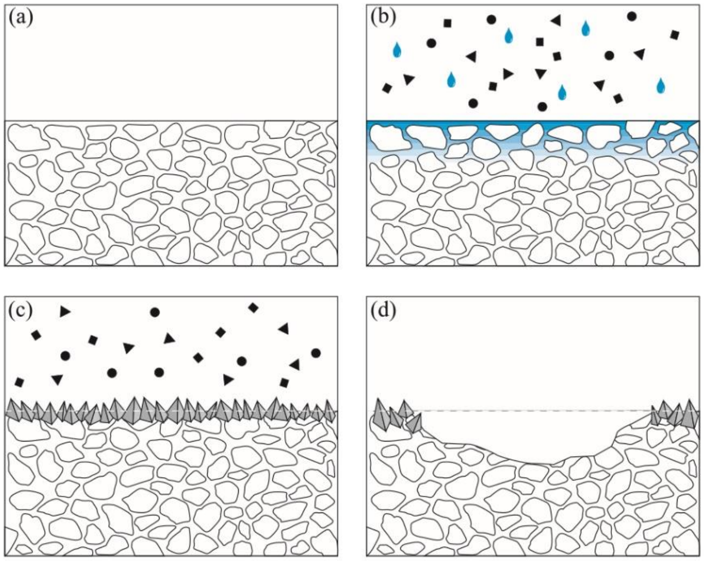


Figure 3.9: Diagram of the process of stone weathering on a stone surface [9].

3.3 Approaches to Modeling and Rendering Weathered Stone

Weathering is one of the most important aspects for realistic objects in CG, it adds character and believability to the scene or object. Due to the process' importance, there have been several methods used to approach the portrayal of weathering patterns. A few of the current approaches used in a CG production are: 3D sculpting wear on geometry, creating weathered 2D textures, or weathering a model procedurally; all with their own strengths and weaknesses.

3.3.1. 3D Sculpting Approach

The 3D sculpting approach is the most straightforward, allowing direct manipulations on an object. 3D sculpting gives the highest fidelity output, detail is added directly to the model so it can be as fine or as broad as the artist desires. The fidelity is where this method's strengths lie, allowing the manipulated objects to be viewed in great detail without a loss of quality. However, the approach's weaknesses lie in its replicability, mesh density, and its iteration time. The 3D sculpting approach is the most time intensive method, taking an artist quite a bit of time in either research gathering or in the sculpting of the model. The iteration time drastically increases for this method as well due to the difficulty of manipulating the sculpted detail without deforming the base object. The sculpted mesh's density also may have to be high enough so that the detail shows up in the model, making the model harder to work with. However, it is becoming common to create a normal map that translates the small details onto a model with a lower resolution. A lower resolution mesh is one of the keys to working quickly in a production environment due to the model's ability to be imported into a 3D scene and moved around effortlessly. Due to these factors, this approach is best used for objects that will be viewed close to the camera [36].

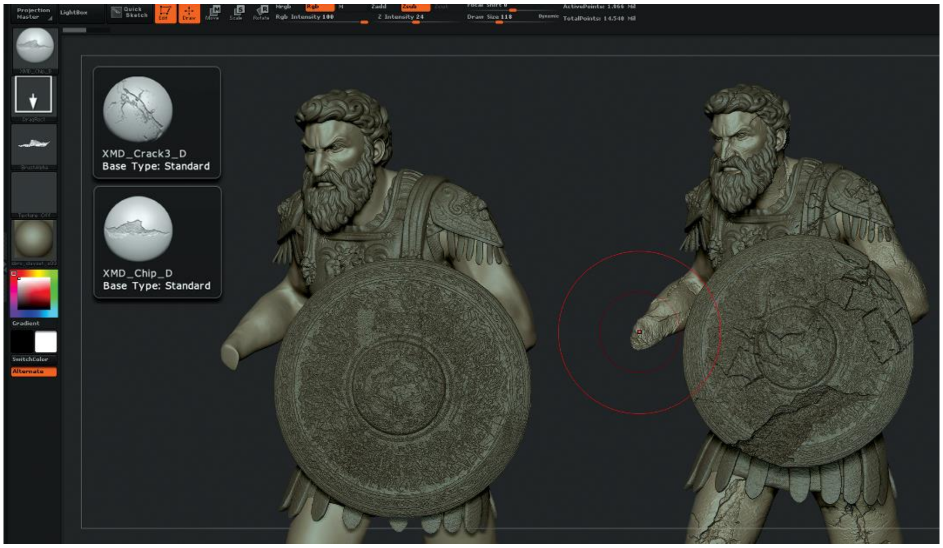


Figure 3.10: Example of producing wear patterns through sculpting [29].

3.3.2. Texture Based Approach

A texture based approach involves an artist working off of a base model and adding the weathered detail by either painting it directly on the model or by indirectly painting on a 2D image to later be applied to the 3D model. One of the strengths of this method is that it potentially takes very little space to store a flat image as opposed to a very dense sculpted model. A model using this method is usually much less dense than a sculpted model as well, making it much easier to work with in a scene. A texture based approach could have a quicker iteration time because the texture map is the only asset being changed. Using textures to display weathering effects is the most efficient method for smaller scale weathering effects such as scratches, color discoloration, bumps and dents. However, a texture based approach might not give the detail required; the model still retains its original form, so the edges of the surface will still be sharp and the deformations in the mesh are limited. [1]

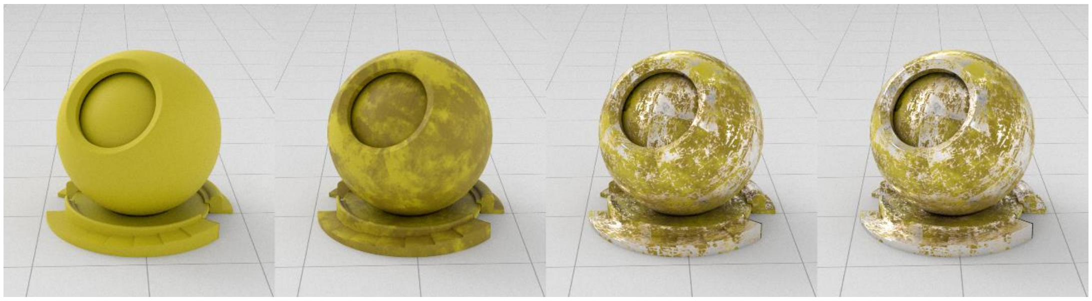


Figure 3.11: Example of producing wear patterns through texture maps.

3.3.3. Procedural Approach

A procedural approach is what this thesis follows. This approach uses computer defined functions and rules to produce a result. The user is able to change the rules or the variables in the system to control or vary the outcomes. Working with a procedural system requires more time in the front end of development but, if used efficiently, could save an artist much time. This thesis uses physically based modeling, 3D noise textures, and particle dynamics as procedural approaches to create a weathering system.

3.3.3.a. Physically Based Modeling

Physically based modeling refers to a process that attempts to map a natural phenomenon to a computer simulation program. In these systems, the user interacts with the model by exerting virtual forces that the system responds to subject to the active constraints [19]. Physically based modeling can be split into geometry based modeling or into mechanics. The system uses geometry primarily for collision detection and to compute contact manifolds in geometry. The mechanics of a physically based modeling system include particle dynamics and rigid or non-rigid body dynamics, shown in Figure 3.14 [32, 33]. In this thesis, I will focus on collision detection with an object and particle dynamics.

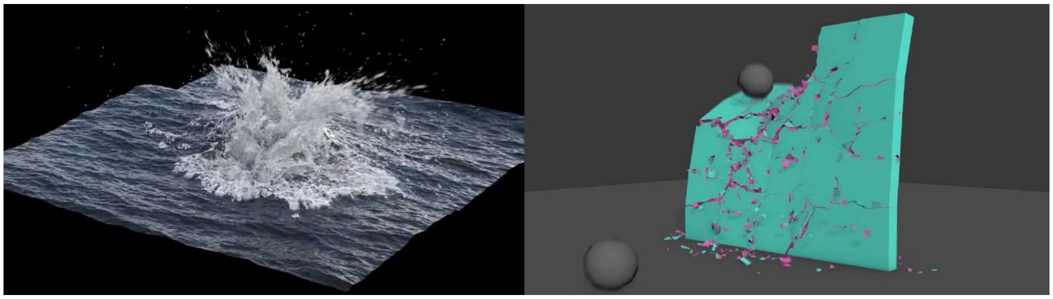


Figure 3.12: Examples of physically based modeling using particle dynamics and rigid body dynamics [33, 34].

3.3.3.b. Particle Dynamics

Particle systems refer to the generation of many points in a 3D software package. These points can be set up with parameters, such as lifespan, friction values, or weight. To emulate rainfall on an object, particle dynamics will be used. Particle dynamics are forces and directions given to the mass of particles as a whole, such as gravity, wind, or collision surfaces. Houdini will be used in conjunction with this simulation due to the software's ability to handle mass amounts of data easily.

3.3.3.c. 3D Generated Noise

Noise, in terms of this project, is the 'random number generator of computer graphics. It is a random and unstructured pattern, and is useful wherever there is a need for a source of extensive detail that is nevertheless lacking in evident structure' [22]. Due to stone's seemingly random structure, its physical makeup and defined texture can be replicated with manipulated and layered noises. Procedural noise is also a benefit to the quicker, iterative process in that it uses much less data than the more common texture maps. The noise is also continuous and has an infinite resolution, meaning that it can be scaled without having a noticeable, repeating pattern (Figure 3.15).

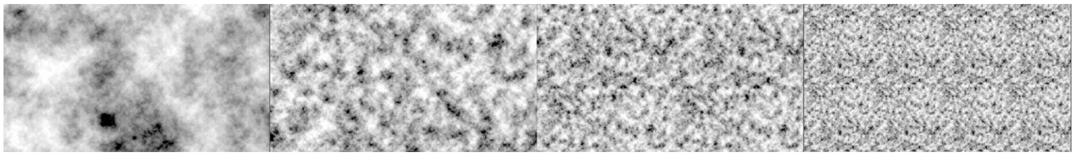


Figure 3.13: 3D generated noise pattern scaling down.

4. RELATED WORKS

4.1. Weathering of Materials in Computer Graphics

Dorsey et. al. present a new and realistic approach to modeling and rendering stone in different states of decay [9]. The approach consists of three major components: surface-aligned volumes to serve as the basis for simulations; a simulation to model the flow of water as well as the transportation and dissolution of minerals within the volume; and a subsurface scattering rendering technique [9].

The volumetric data structure is a surface-aligned volume around the boundary of the model. Basically, they took the original mesh, greatly reduced its resolution, and then took each polygon and extruded it in either direction to create a volume. These volumes were then filled with voxels which were assigned a value of either 1, 0, or a fraction between the two based on whether the voxel was inside of the surface or not. This voxel based data structure is referred to as a series of "slabs".

Dorsey et al’s treatment of the flow of water focuses on the absorption through the porous stone material. When water and external pollutants penetrate the surface and are evaporated, they form deposits of minerals and salts [9]. Under the surface after the absorption takes place, the stone breaks down and the areas that retain the water start to break off of the whole. Dorsey applies this to a slab data structure by allowing the water to diffuse under the surface and interact with the interior of the stone based on the stone's structure and makeup. In addition to this, they include wetness maps, which are computed by detecting the map's visibility to a directional rain source. The way she is handling the flow of water is the largest difference between this project and Dorsey's. This thesis’ system will focus on a particle system that will emulate the behavior of either rain or wind over an object and will weather the surface based on collisions.

Another difference between their system and the one reported on here is the time it takes to achieve a final result. In one example, the previous model's result took approximately 24 hours to compute the weathering effects and 31 minutes to render the image. The time it takes is understandable as the simulation has to deal with the model as a volume along with the data coming from its interior. They also created a rendering algorithm to achieve the subsurface scattering effect based on the stone's interior. Conversely, this thesis’ method aims to drastically reduce the amount of time to simulate the weathering effects by using 3D noise to simulate the stone’s interior rather than using a voxel based system.

Dorsey et. al.’s research outcome was a realistic system that took a model, weathered it over time, textured it based on various stone weathering effects, and gave it a subsurface scattering material with limited user input, Figure 4.1 [9]. This thesis develops a system that is *iterative*, meaning that an artist can produce results, judge the outcome, and make changes rapidly.

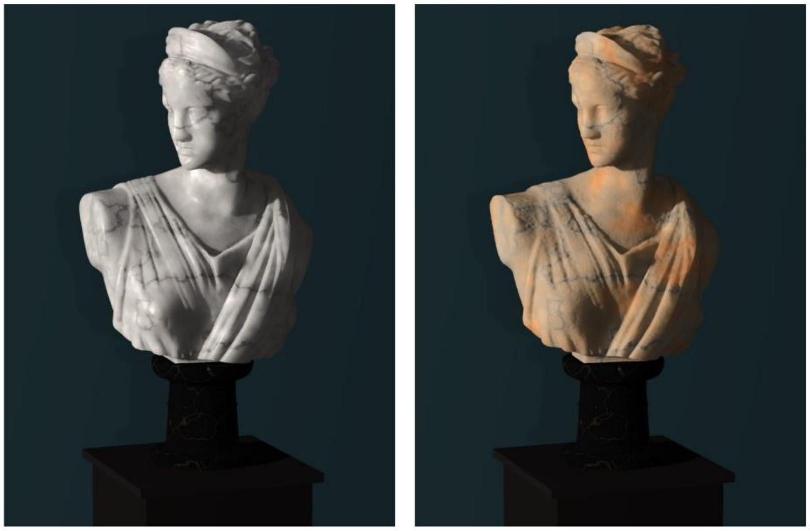


Figure 4.1: Results from the method proposed by Dorsey [9].

4.2. Particle Staining: Physically Based Texture Generation

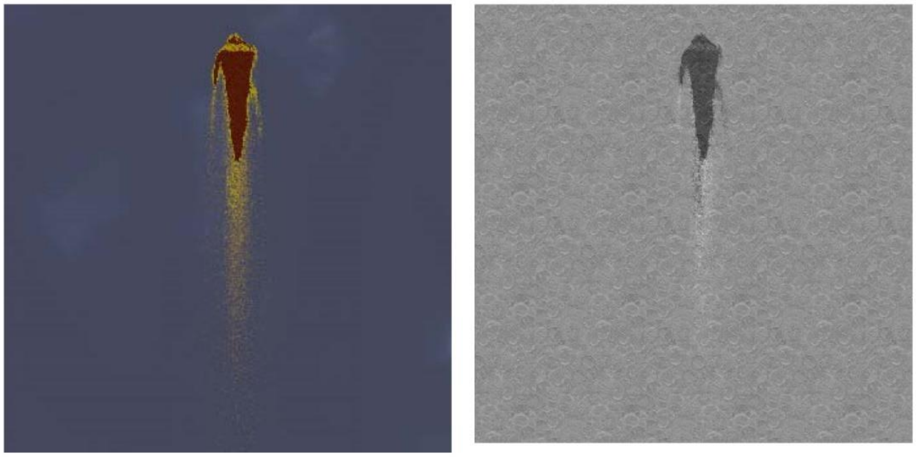
Mistrot [29] focuses on the dynamics of water flow across surfaces and how that flow affects the surface. Their goal for this tool was for it to be both imitative, achieving the look of accurate realism with regard to natural processes, and creative, allowing the user to manipulate the processes to achieve a desired result [29]. The goals here parallel Mistrot's by creating a system that achieves both natural accuracy while being interactive and iterative.

Mistrot divides his imitative design process into four steps: concept, observation, distillation, and modeling. By breaking up the process into steps, he makes the complex natural processes much more manageable and highlights only the most important details to imitate. He separates the water staining process into three areas: dynamic attributes, surface attributes, and fluid attributes.

Dynamic attributes cover the behavior and forces that act on the particles flowing over the mesh. These dynamic attributes include forces such as wind, gravity, wearing, sedimentation, adhesion, and dispersion. This distillation of a natural process is similar to my methods of approaching weathering and particle dynamics except for sedimentation. Sedimentation is a weathering process where sediment or smaller grain is picked up and deposited elsewhere. Since this thesis' focus is on only stone objects and not large-scale landscapes, I assume that the sediment eroded away off of the objects will settle off of the object being weathered. Ignoring the deposition of the sedimentation process will also make it much less computationally expensive and less time intensive in not having to implement it.

The surface attributes define the object that is being eroded. Mistrot distills these attributes as well to include color, hardness, and roughness. Surface hardness and roughness are two important attributes to weathering. The hardness of a surface dictates the resistance that material has to weathering [29]. Surface roughness describes the undulations across the surface. Mistrot observes that the rougher a surface is, the less the fluid will travel across it but the surface will weather faster, becoming smoother [29]. He also adds a layered map system to the surfaces, meaning that the user can set different maps to different layers of the object based on depth. The layers will create a more dynamic object that will change as it is being eroded. Fluid attributes describe the individual particles of the system. The discrete particles of his system pick up and transport the material of the object as well as the color of the object.

The large difference between my methods and Mistrot's are the output of the system. His output is a series of 2D texture maps: Color maps and height maps (Figure 4.2) [29]. The height maps are put back onto the object in a 3D package that displace the mesh of the object. However, using a height map restricts the accuracy of the weathering because the particles are only interacting with a static mesh rather than building on the weathered object. There were technical reasons for the restriction to only outputting 2D maps. Displacing a mesh requires it to be an extremely dense piece of geometry so that the particles have plenty of vertices to interact with. If the mesh was not dense enough, the result would not be visually appealing. My solution to the dense mesh problem is to use Houdini, a piece of software that works extremely well with large amounts of data.



(a) Color map. *(b) Height map.*

Figure 4.2: Results from Mistrot's method [29].

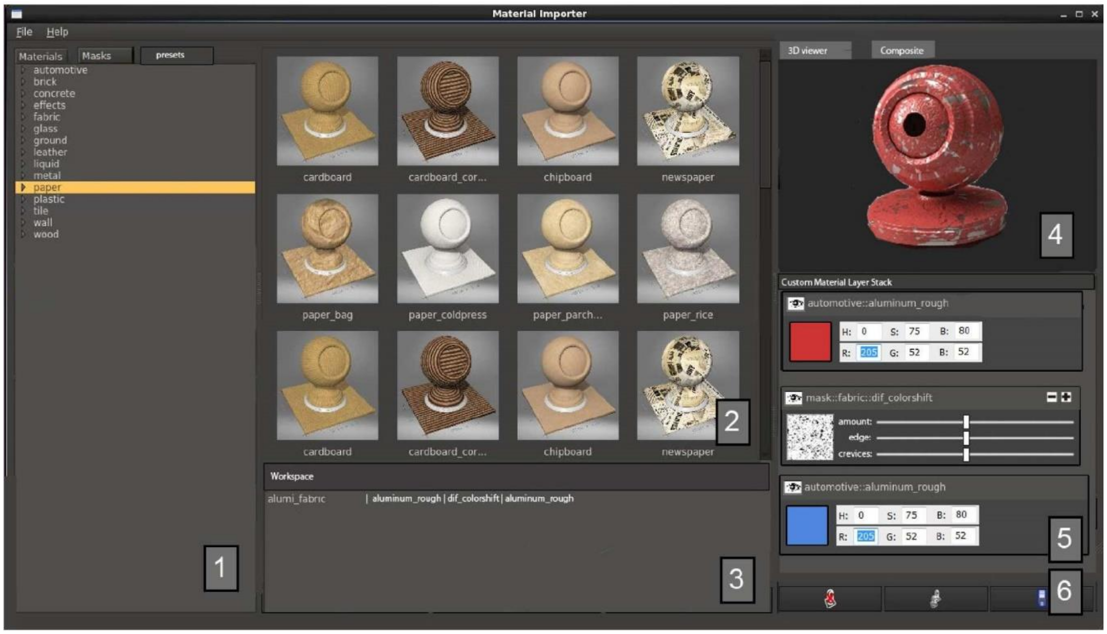
4.3. Realistic Aging of Materials in Computer Graphics

Walker developed a user directed, layer based shading system that quickly recreates weathering and aging patterns on objects [45]. To achieve this, she creates several base materials including woods, stones, metals, and plastics. She then studied the patterns of weathering over time on different objects. From the study of aging copper, she broke down the weathering process and re-contextualized it into shading parameters. The shader's color darkens and gets more saturation, values in the glossiness map decrease and become more varied, and scratches and dents form and gain more contrast in the bump map. A large part of research for my thesis will be studying stone, including the makeup of different stones and how their chemical compositions, the hardness and roughness of each stone, and how changes in the environment change the weathering patterns.

Changing the parameters of the shader, however, could only go so far. To counter-act this, she used a series of texture masks that blend between materials [45]. The masks were made to be geometry aware which means that the 3D geometry gets laid out in a 2d texture space using the geometry's UV coordinates. The geometry aware masks can detect the shape of the mesh, where the mesh's sharp edges are, and where it has any crevices which make it a fantastic way of detecting where the wear will be on an object.

Walker’s approach to the display of weathering stays strictly in the texture space, meaning the geometry will not change over time and the edge wearing, dents, and cracks will only be surface level using a bump map. My approach to it is to change the physical geometry over time, creating a more realistic method of displaying the weathering patterns.

The system was made accessible to artists and regular users through several iterations of the layout and UI design for the system. The UI, shown in Figure 4.3 aided in figuring out which parameters to expose and how to display the material and mask libraries through iterations [45]. Referencing her methods of laying out the interface will be useful because I will create an easily navigate-able interface in Houdini as well.



Workspace: alumi_fabric | aluminum_rough | dif_colorshift | aluminum_rough

automotive::aluminum_rough
H: 0 S: 75 B: 80
R: 205 G: 52 B: 52

mask::fabric::dif_colorshift
amount:
edge:
crevices:

automotive::aluminum_rough
H: 0 S: 75 B: 80
R: 205 G: 52 B: 52

Figure 4.3: User interface used in Walker's approach [45].

5. METHODOLOGY

5.1. Stone Composition

Determining the procedure to create the stone’s composition required several iterations to achieve a result that was accurate and quick to give results. The initial method was to use a texture map to define a pattern representing the composition of the stone. The next iteration was to use geometry to fill the object, simulating the stone’s interior minerals. 3D noise was decided on as a way to combine these two processes: texture mapping and the use of interior blockers.

5.1.1. Texture Mapping

The texture mapping method involves placing 2D images onto an object using that object’s UV coordinates, much like how a globe is represented on a map. This is the traditional way of adding texture information to an object. Using texture maps is quick and easy to iterate upon due to the small file size. The outcome of this process, shown in Figure 5.1, created desired results; the stone texture was showing as the model was receding due to the particle system interactions. The particle system dynamics simulating rainfall were also working accurately as they collided and flowed down the object.

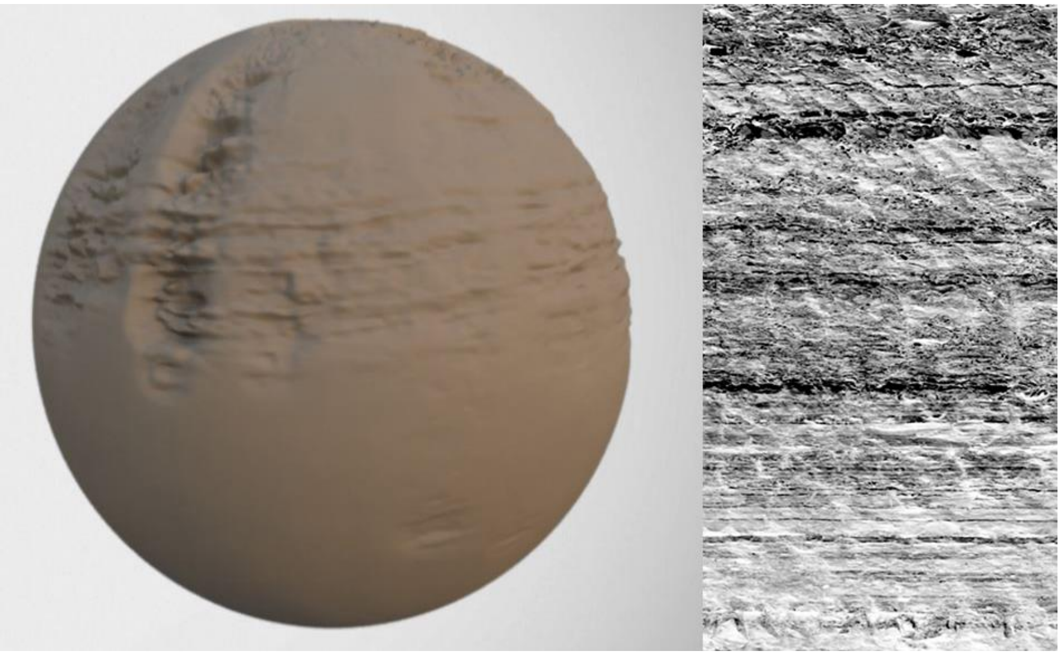


Figure 5.1: Representation of sandstone using the 2D texture map process.

However, by using a texture map the stone is only defined on a surface level. By limiting the information to only the object’s surface, we cannot add in any information to the interior of the object such as other minerals. This restriction greatly reduced the believability of the outcome because the object would still only be reflecting the texture map used. In Figure 5.2, the object is revealing the texture rather than the texture being a result of the process. Because we were only defining the texture from an image of the sample rock’s surface, it was also not an accurate representation of the physicality of the stone.

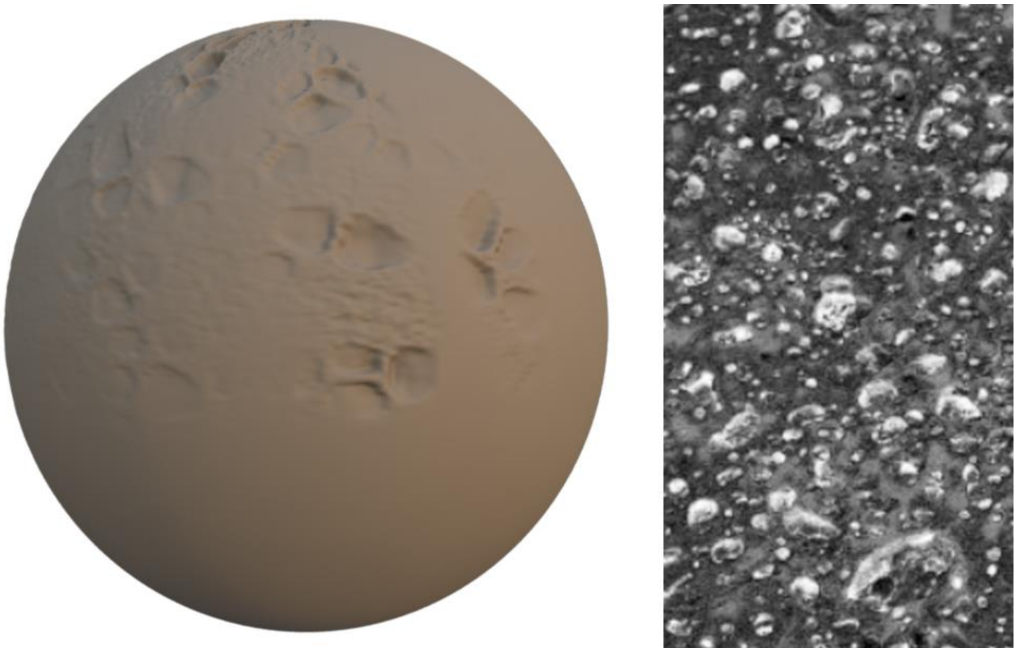


Figure 5.2: Representation of basalt using the 2D texture map process.

5.1.2. Geometry

The next iteration was to use geometry inside of the object to initiate additional manipulations during the simulation. This iteration was a push to define the interior of the stone object as opposed to only using the surface level texture. The interior geometry was used as bounding volumes inside the surface that would transfer values to the deforming surface. Figure 5.3 shows an example of this process. In Figure 5.3a, we define a black cube inside of our deforming object, the green sphere. Visualizing these objects as a point cloud, we can see that the values that fill the sphere are all green and the values that fill the cube are all black. As we run the simulation in Figure 5.3b, we can see that the green sphere weathers away but the black cube stays intact. The black sphere stays intact due to the value associated with it. All vertices that are inside or lay along the black cube’s volume will inherit its value, zero, stopping the displacement in those areas.

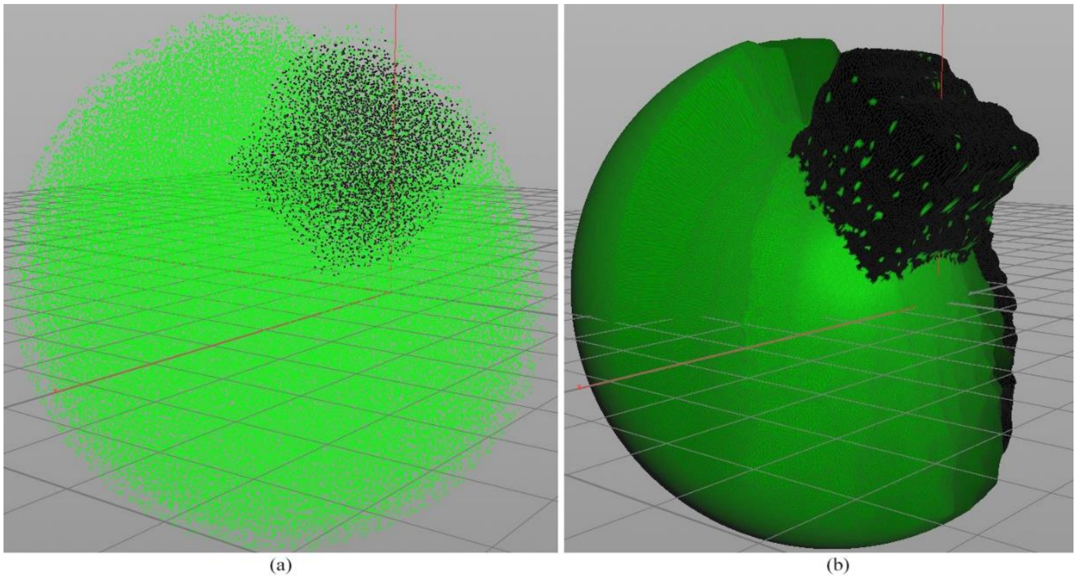
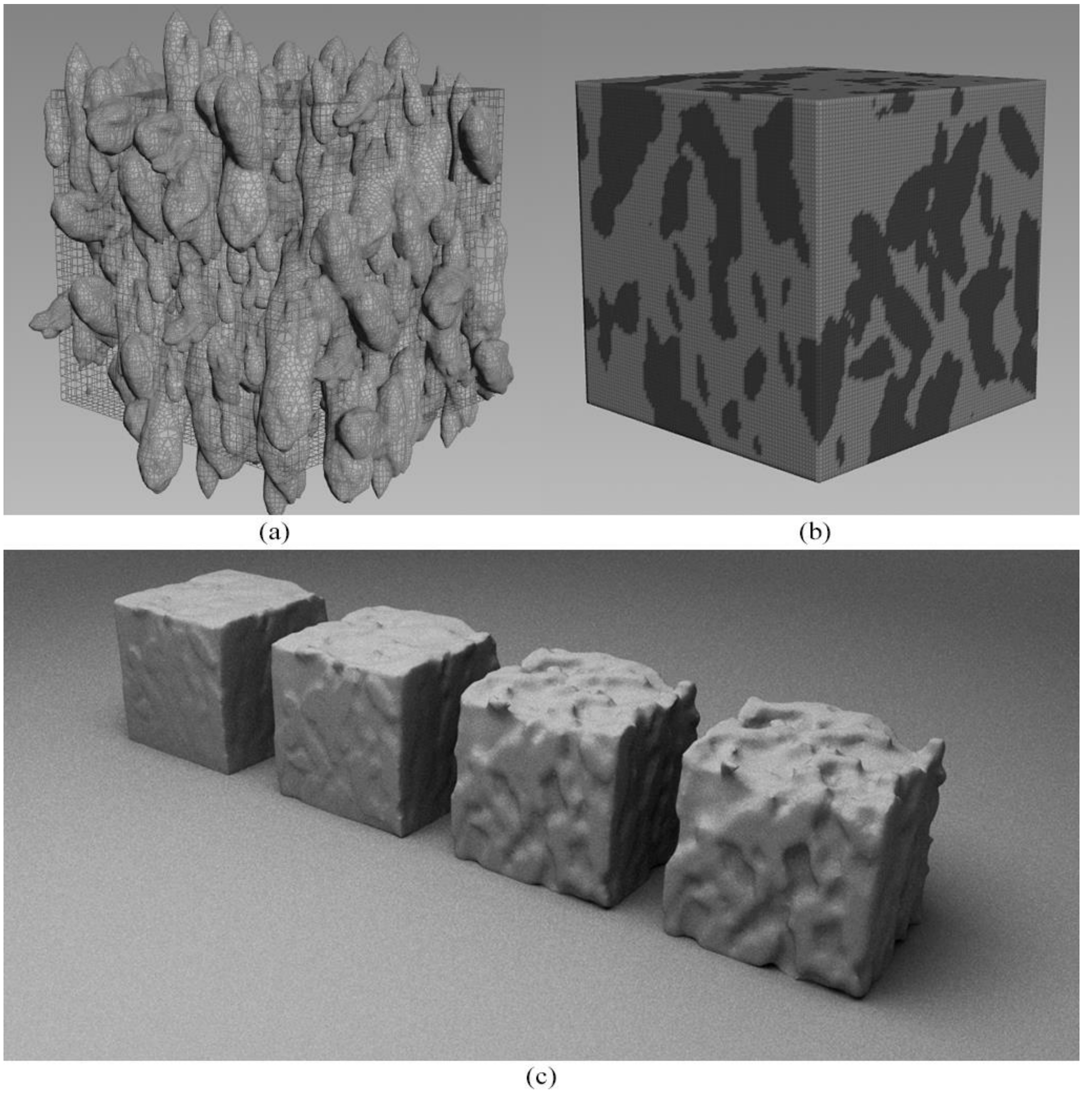


Figure 5.3: (a) Visualization of geometry on the interior of another object. (b) Revealing the interior geometry through the weathering process.

This solution was the easiest to view and create the interior of the stone. The user also had control over how the interior geometry was laid out, sized, and shaped. The example in Figure 5.4 shows an example using this method. In Figure 5.4a, we define our interior elements that are reflected on the surface by using the bounding volumes in Figure 5.4b. The results (Figure 5.4c) create a more believable outcome as opposed to using the texture-based approach as in Figure 5.2. However, the simulation time for the bounding volume method was exponentially longer than the texture mapping method. This time increase was due to the number of particles, geometry, and volumes that are attached to a single object. This process is also missing the smaller detail and grain that we got from using the texture-based approach.



(a)

(b)

(c)

Figure 5.4: (a) Interior geometry. (b) Transfer of attributes from interior geometry to the surface. (c) Results of interior geometry method.

5.1.3. 3D Noise

3D noise was used as a way to combine these two processes: texture mapping and the use of interior geometry. 3D noise was chosen due to its ability to define itself in 3D space rather than just 2D space. Due to the noise's 3-dimentional coordinate system, we are able to use it to define the interior of an object with a texture-based definition. In Figure 5.5a shows the texture on the surface and Figure 5.5b is showing that same texture as a point cloud to visualize the interior of the object. By replacing the use of interior geometry with a 3D texture we eliminate the need for the interior geometry to be defined, saving time and computing power. Through these iterations, it was found that using 3D noise was exponentially faster than any other method tested.

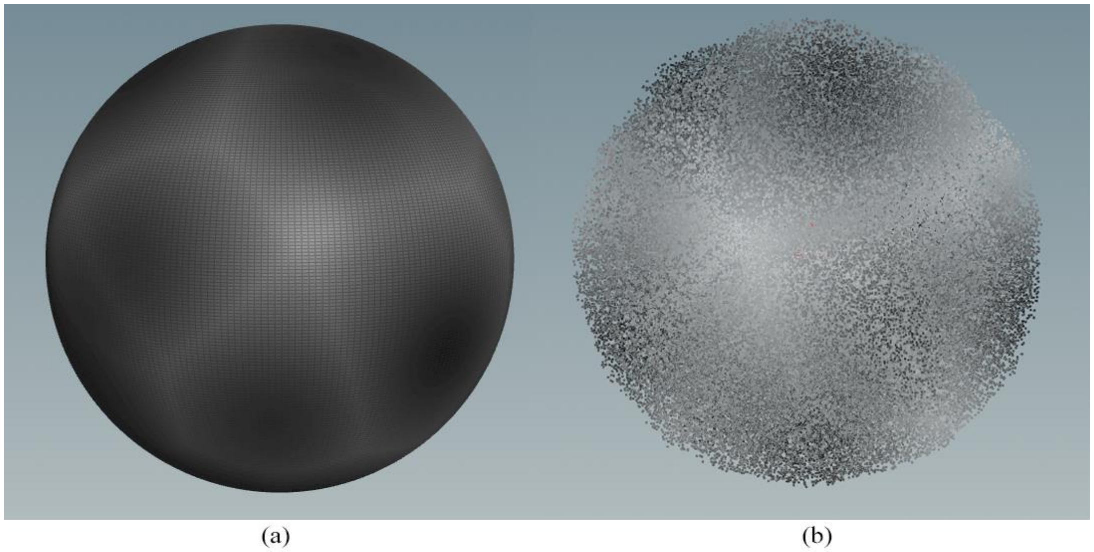


Figure 5.5: (a) Worley 3D noise values on the surface of a sphere. (b) Worley 3D noise values on points scattered within the surface of a sphere.

To replace the 2D textures, we needed to refine the 3D noise used to match the fidelity and visual detail of using the texture maps. Breaking down a sample of pegmatitic granite in Figure 5.6, we start to reveal patterns that are achievable by manipulating 3D noise. The primary components in granite are:

- **Quartz** (Figure 5.6a): larger crystals with an angular shape in the composition. Shaping a sparse convoluted noise, we start to form shapes that represent the quartz. Quartz has a hardness of 7 on the Mohs scale, so these elements will be darker to resist the weathering in calculations.
- **Mica** (Figure 5.6b): smaller flaky minerals with angular shapes. A sparse convoluted noise was used here as well with a higher frequency. Mica has a hardness between 2.5 and 3.5 on the Mohs scale, which will correspond to a lighter element.
- **Alkali feldspar** (Figure 5.6c): different minerals making up the rest of the composition. A cellular noise with a high frequency was used here to give the texture some variation in the shape of smaller grain. These elements have a hardness averaging to around 6 on the Mohs scale.

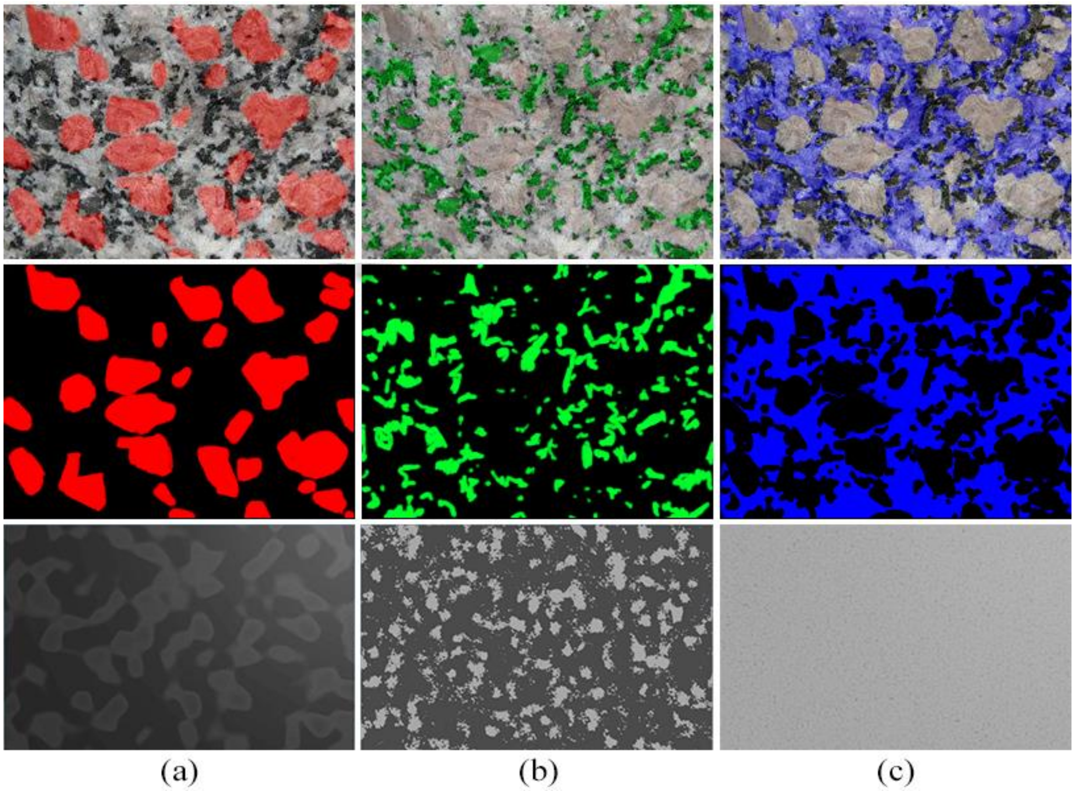


Figure 5.6: Breakdown of the components in pegmatitic granite.

To create our texture library, we used Figures 3.7, 3.8, and 3.10 as reference to separate and match the shapes and patterns visible using a similar process as in Figure 5.6. To distinguish the type of rock, we researched the elements in the composition and assigned hardness values to each layer of noise.

5.1.4. Noise Layers

In order to use multiple layers of noise on the same object, each layer would have to be composited on top of the other. One example of compositing the layers together is by adjusting the transparency of the top layer so that both layers are visible. However, the system becomes limited by only compositing through alpha values. There is also an artistic limitation due to the limited amount of control as well as an accuracy limitation due to the manipulation of the values of the base layer. Figure 5.7 shows an example of the manipulation of values due to compositing with an alpha layer based approach. Figure 5.7a is the base layer used, a value of 0.31 is determined from a central area. In Figure 5.7b, we add a layer with a transparency of 75%. By adding the transparent layer, we start to lose the value we set in our base layer. As we turn our transparency down, Figure 5.7c with a transparency value of 50% and Figure 5.7d a value of 25%, we can start to bring the composited value back to our base value. These alpha manipulations come at a cost of changing the values in both the base and manipulating layer. The skewed values are relevant in our system because we denote the hardness of the elements in the stone through the value of noise, potentially losing accuracy.

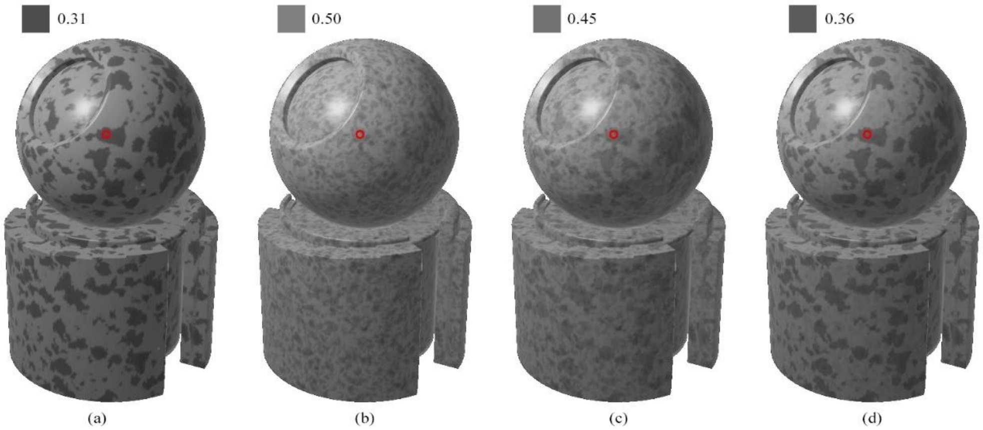


Figure 5.7: Effect of layering noise on the values used for hardness.

To counter these limitations, we implement several ways to blend the layers together. The blending modes are modeled similar to Photoshop’s due to an artist’s familiarity with these systems and options. Adding the blending modes gave the system enough artistic flexibility while still having options that kept the majority of each layer's values.

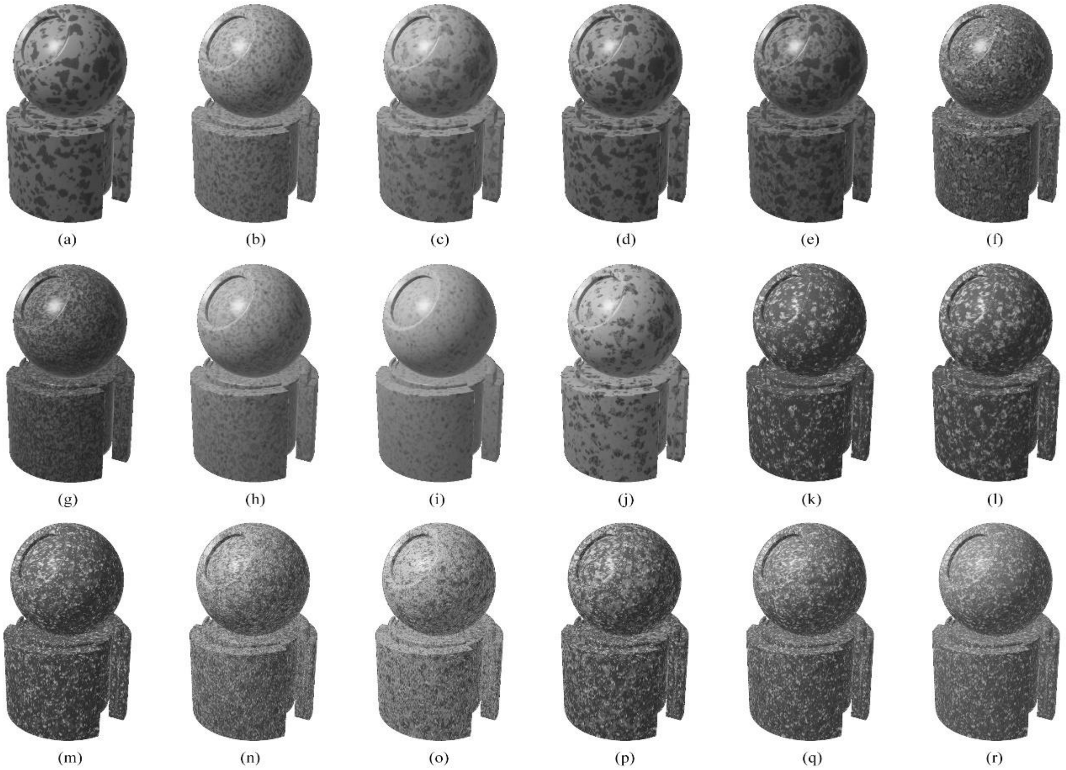


Figure 5.8: Compilation of all possible blending options between layer 1 and layer 2.

Figure 5.8 displays the blending options implemented applied to the same object showing a large angular grain texture with hardness values for both layers at 1 and a texture scale of 1. These blending options [26] are:

- **Layer 1 solo** (Figure 5.8a): Only show layer 1.
    - *layer01 + (layer02 * 0)*
- **Layer 2 solo** (Figure 5.8b): Only show layer 2.

  - *layer02 + (layer01 * 0)*
- **Alpha blend** (Figure 5.8c): Multiply top layer by the alpha value.
  - *layer01 + (layer02 * (1 - alpha))*
- **Darken** (Figure 5.8d): Shows darkest value between layer 1 and layer 2.
  - *minimum(layer01, layer02)*
- **Multiply** (Figure 5.8e): Darkens result, multiplies layer 1 and layer 2.
  - *layer01 * layer02*
- **Color Burn** (Figure 5.8f): Burns (darkens) layer 1 based on values in layer 2.
  - *1 - (1-layer01) / layer02*
- **Linear Burn** (Figure 5.8g): Similar to multiply but linear in application.
  - *layer01 + (layer02 - 1)*
- **Lighten** (Figure 5.8h): Shows lightest value between layer 1 and layer 2.
  - *maximum(layer01, layer02)*
- **Screen** (Figure 5.8i): Brightens result, adds brightness of layer 1 to layer 2.
  - *1 - (1 - layer02) * (1 - layer02)*
- **Color Dodge** (Figure 5.8j): Dodges (lightens) layer 1 based on values in layer 2.
  - *layer01 / (1-layer02)*
- **Overlay** (Figure 5.8k): Multiplies (darkens) when layer 1 is dark and screens (lightens) when layer 1 is light.

- *if layer01 <= 0.5 : {2 * layer01 * layer02}*
- *else : {1 - (2 * (1-layer01) * (1-layer02))}*

- **Soft Light** (Figure 5.8l): Similar to overlay but the application is less intense.
  - *if layer01 <= 0.5 : {(2 * layer01 * layer02) + ((layer01*layer01) * (1-(2*layer02)))}*
  - *else : {((2*layer01)*(1-layer02)) + sqrt(layer01) * (2*layer02 - 1)}*
- **Hard Light** (Figure 5.8m): Similar to overlay but the application is more intense.
  - *if layer01 <= 0.5 : {2 * layer01 * layer02}*
  - *else : {1 - (2 * (1-layer01) * (1-layer02))}*
- **Vivid Light** (Figure 5.8n): Similar to color burn when the value is less than 0.5 and similar to color dodge when the value is greater than 0.5.
  - *if layer01 <= 0.5 : {layer01 / 1 - (2*layer02)}*
  - *else : {1 - (1-layer01) / (2*(layer02-0.5))}*
- **Linear Light** (Figure 5.8o): Similar to linear burn when the value is less than 0.5 and similar to linear dodge when the value is greater than 0.5.
  - *if layer01 <= 0.5 : {layer01 + (2*layer02) - 1}*
  - *else : {layer01 + 2*(layer02-0.5)}*
- **Pin Light** (Figure 5.8p): Similar to multiply when the value is less than 0.5 and similar to screen when the value is greater than 0.5.

    - *if layer01 <= 0.5 : {minimum(layer01, 2*layer02)}*
    - *else : {maximum(layer01, 2*(layer02-0.5))}*
- **Difference** (Figure 5.8q): Subtracts layer 1 from layer 2.
    - *abs(layer01 - layer02)*
- **Exclusion** (Figure 5.8r): Uses the darkness in layer 2 to mask the difference effect.
    - *0.5 - 2*(layer01-0.5) * (layer02-0.5)*

5.1.5. Evaluation During Development

To evaluate the stone texture and parameters during development, a cube and a sphere were used. The test object as seen in Figure 5.7 was used after the texture was iterated on due to the sphere’s and cube’s lower polygon count. The cube provided sharp edges and flat surfaces, making it easy to detect how the edges were being manipulated during the simulation. The sphere provided a curved surface where the texture of the stone could be focused on. The cube and sphere were used interchangeably during development by replacing the root model in the node network, shown in Figure 5.8. Figure 5.8a shows the root geometry node connecting with the weathering network. Figure 5.8b shows the interior of the geometry node. For development, the common models used (the cube, sphere, and test object) were attached to a switch node (Figure 5.8b) that was manipulatable from the root geometry node from the interface, shown in Figure 5.8a.

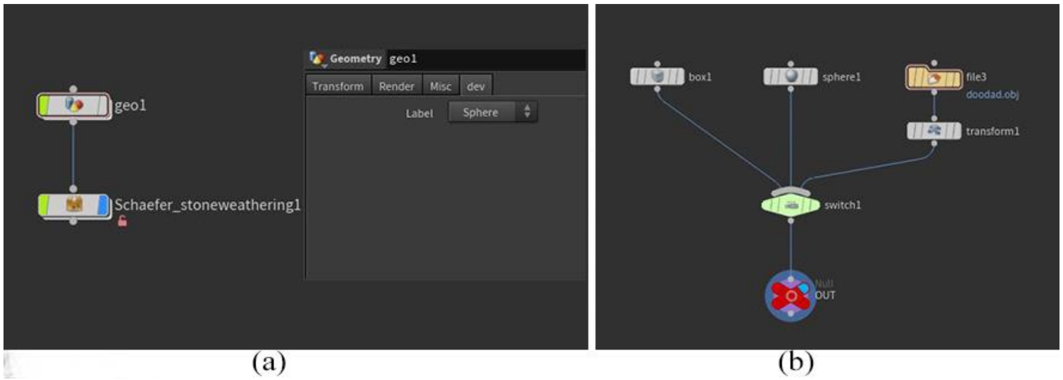


Figure 5.9: (a) Base node network for the system. (b) Test geometry node network.

5.1.6. Vesicular Texture

The vesicular texture was used extensively during the development of the systems due to the distinct pattern in its texture. The Olmec Head, shown in Figure 5.10, was used as reference for the vesicular texture due to its size and variety of curved and sharp surfaces.

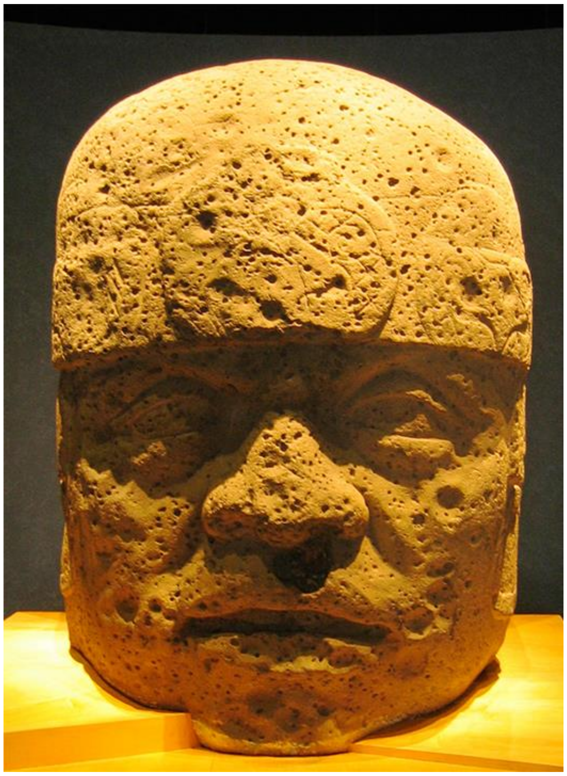


Figure 5.10: Image of an Olmec Head. Used for reference for basalt stone weathering [6].

The pitting in the vesicular texture is caused from air bubbles forming when the rock is formed. Over time, these pores are filled with dirt or sediment, so for this process we set the hardness of these areas to be very low. To create the texture for the pores, we start with a sparse convolution noise. Sparse convolution noise is a sparse form of white noise that will give us separated islands of noise that will represent the pores [23]. The initial pass of the noise in Figure 5.11a needed to be shaped.

To shape the noise, we attach a ramp parameter to the noise. By passing the noise through the ramp, we are able to remap the values and shape the noise. In Figure 5.11b, we remap the values from 0 to 0.5 to be 1, making the original's low points white. We then give the ramp a sharp falloff from 0.55 to 1. The falloff creates a sharp edge around the white areas rather than a smooth fade from white to black. Figure 5.11c and d show further shaping of the noise, adding in additional detail and values.

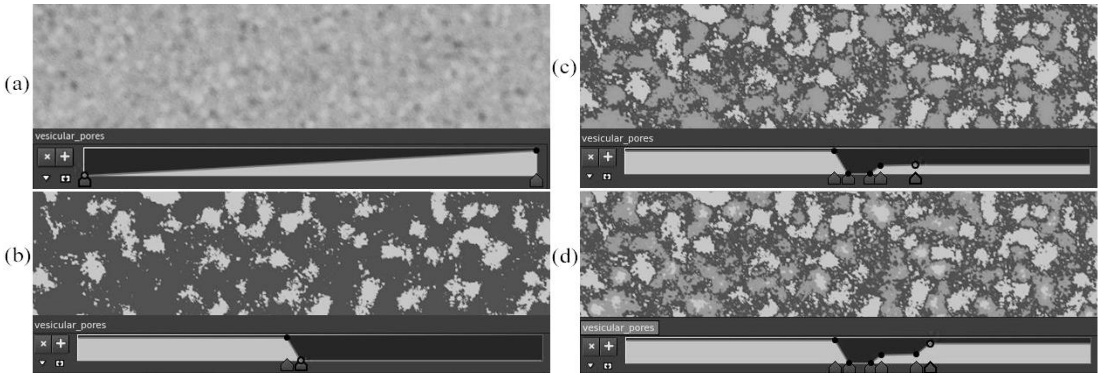


Figure 5.11: Examples of the use of a ramp to shape the noise patterns.

Basalt is primarily composed of plagioclase minerals that have a hardness of around 9 on the Vickers scale. For these minerals a broad Perlin noise, shown in Figure 5.12a, was added and shaped to create additional inconsistencies in the surface. The final layer of noise, Figure 5.12c, is used for the small grain. The grain layer is a Worley noise at a frequency of 25 to simulate a coarse grain. The final texture, Figure 5.12d, blends layer A and B using a difference blending mode. Layer c is then placed on top with an alpha value of 0.5 so that it is visible in dark and light areas of the map.

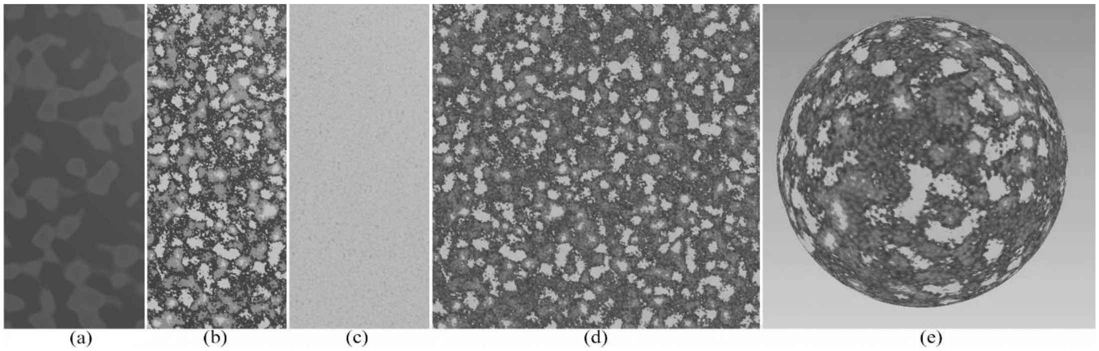


Figure 5.12: Noise layers that contribute to the final vesicular texture.

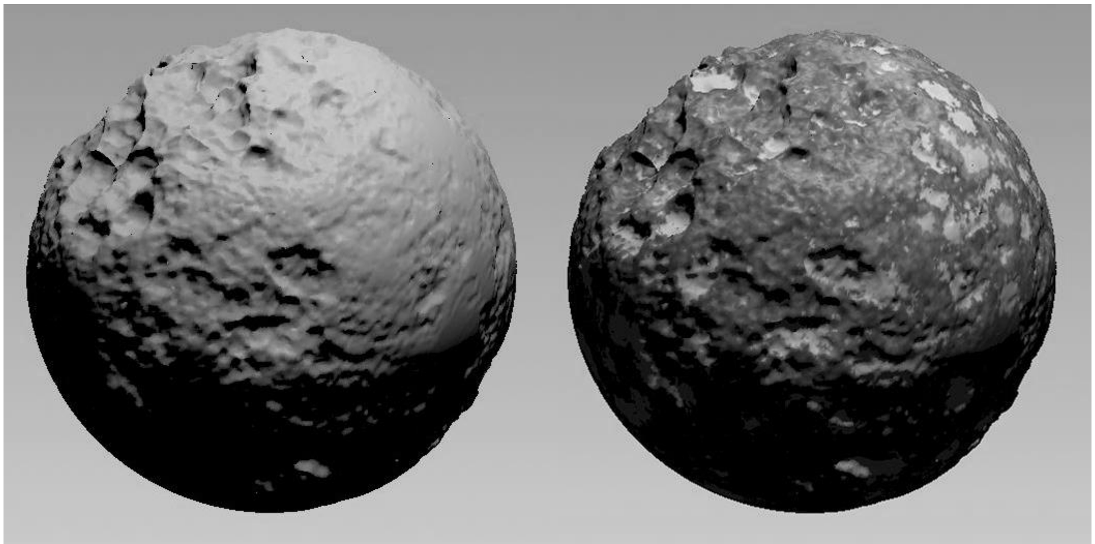


Figure 5.13: Result of the weathering process with the parameters representing basalt.

5.1.7. Layered Microcrystalline Calcite Texture

Layered microcrystalline calcite is a banded sedimentary texture caused by the buildup of different sediments and minerals on top of each other, compressed over time. Because of the banding, a reference with multiple perspectives was used to view the dispersion of materials in the bands. A CT scan of a sandstone statue was found from Scantix [14], shown in Figure 5.14. Scantix offers radiological analysis of sculptures and features several different types of statues. These studies are used view structural faults that need to be repaired to maintain the statue or to view the history of reconstruction. In Figure 5.14, it was found that the dispersion of small minerals was a normal noise pattern from the front but followed the contours of the bands from the side.

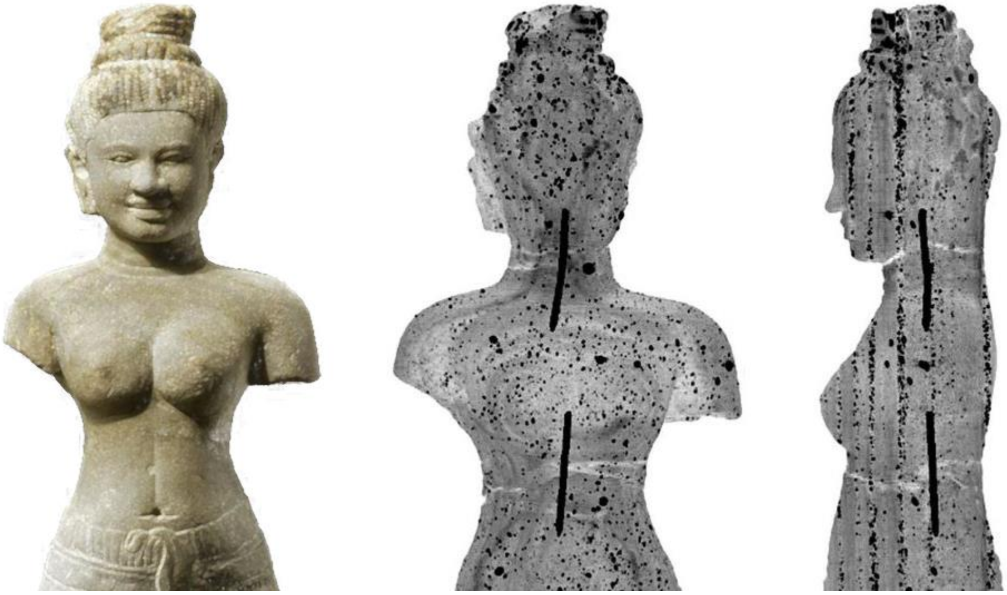


Figure 5.14: CT scan of the Khmer sandstone statue [14].

The creation of the layered texture began similar to the vesicular texture. A sample was evaluated and patterns of minerals were distinguished. To create the smaller interior stones in Figure 5.16b, we start again with a sparse convolution noise. The sparse convolution gives us rounded forms that we then elongate by manipulating the frequency values in a single direction. With the elongated noise, we have the banding structure but referring back to Figure 5.14, the bands are composed of many smaller stones. To achieve this pattern, we use the sparse convoluted noise's output as the position input for a finer, simplex noise. This process is shown in Figure 5.15. Figure 5.15 a is the sparse convolution noise and Figure 5.15b is the simplex noise. Figure 5.15c shows the result of this setup, where the sparse convolution noise acts as a mask to the simplex noise.

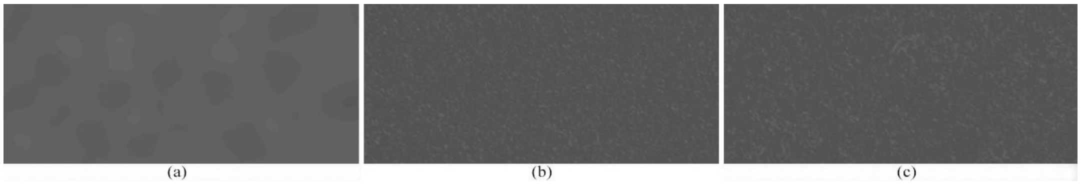


Figure 5.15: Example of using multiple noise patterns to form a more directable noise.

Figure 5.16a displays the broader noise used that follows the format of the noise in Figure 5.16b. This resembles the banding patterns without the harshly separated stones. To achieve the texture in Figure 5.16a, we use a Perlin noise in place of the sparse convoluted noise. The Perlin noise creates uniform areas of noise. The noise is then passed through a broader simplex noise. The texture in Figure 5.16c is similar to the grain texture used for the vesicular texture in Figure 5.12.

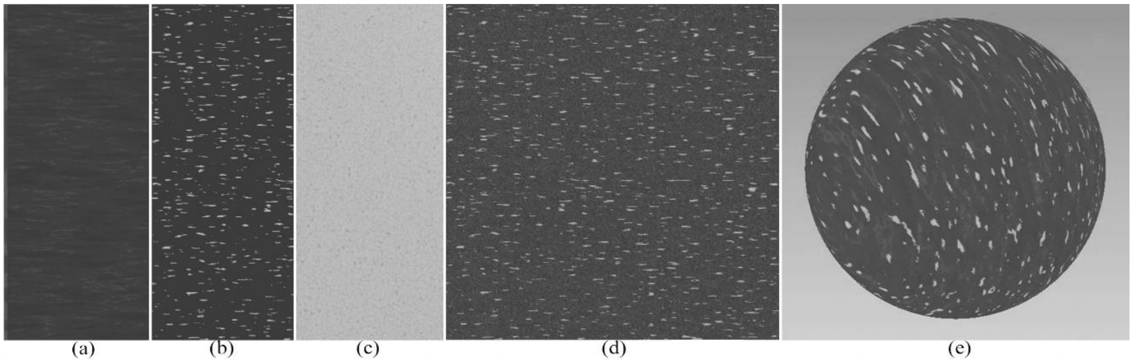


Figure 5.16: Noise layers that contribute to the final layered microcrystalline calcite texture.

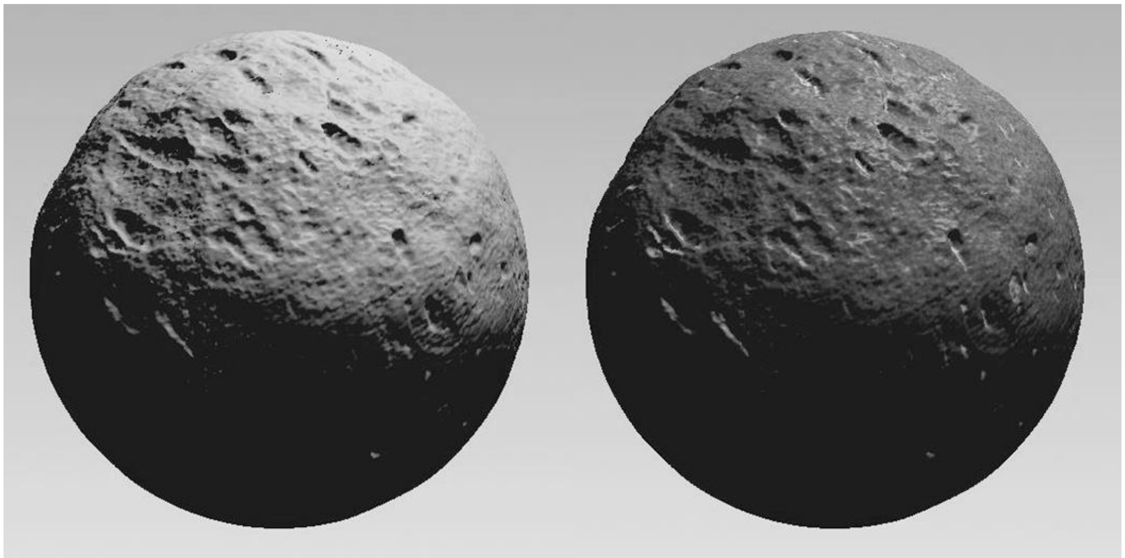


Figure 5.17: Result of the weathering process with parameters that represent sandstone.

5.1.8. Particle System / Dynamics

For this thesis, we will be using a particle system to simulate the rain falling over the stone object over time. A particle system was chosen for several reasons:

- Ability to store multiple parameters in each particle.
- Includes elements of randomness, making each simulation different.
- Manipulatable by realistic forces such as wind and adhesion.
- Ability to manipulate the object and keep the force and direction of the particle system (Figure 5.18a & b).
- Interacts with and flows around objects (Figure 5.18c).

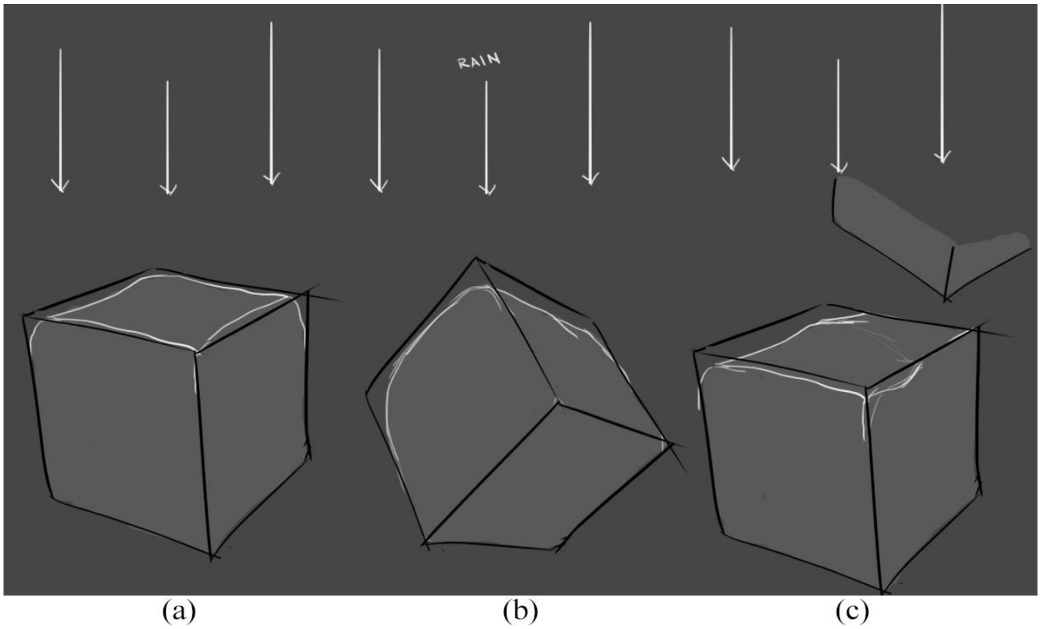


Figure 5.18: Diagram of advantages in using particle systems for this method.

The particle system controls the displacement on the mesh. Attached to each particle is a color and a value that denotes how much of the mesh to displace. These colors, in RGB format, are evaluated when the particle collides with the surface and result in adding or subtracting geometry. Shown in Figure 5.19, the red channel pushes the mesh away while the blue channel brings the mesh in. The value of the color influences how much the particle displaces. For example, if the particle has an RGB value of (0,0,1), that particle will displace into the surface at the maximum amount. The particle may also have a value of (0,0,0.5), which will only displace half of the maximum amount. By using a color ramp, we can change these values over time to simulate the loss of energy each particle would have in the weathering process.

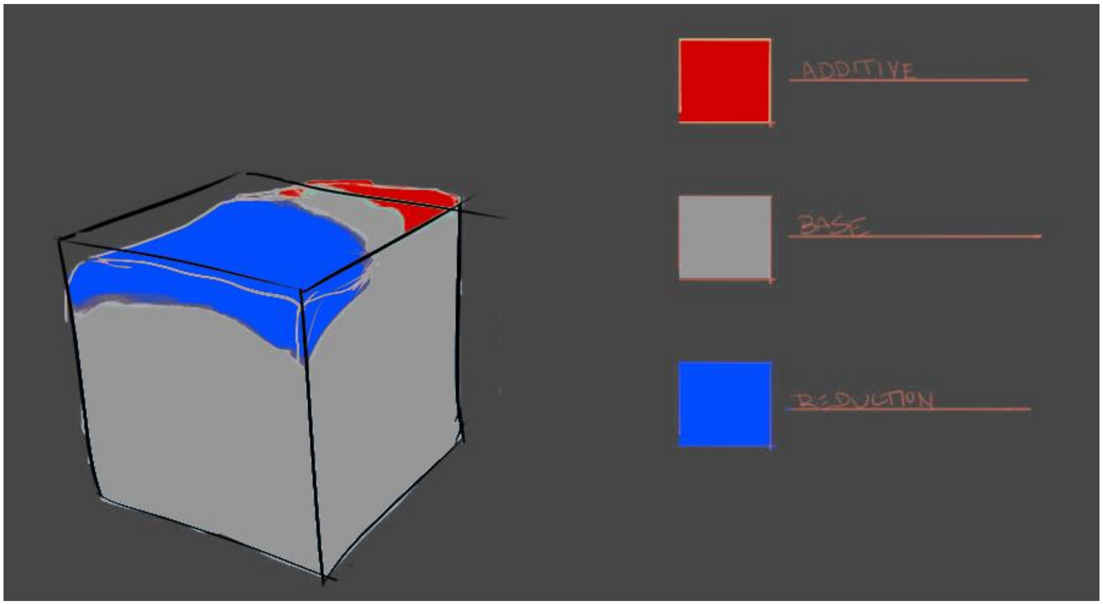
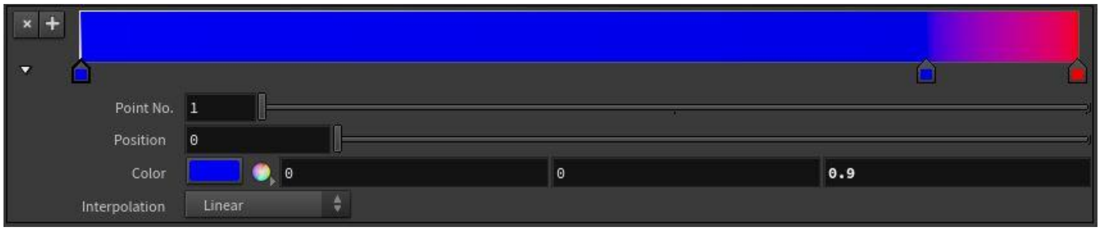


Figure 5.19: Representation of the use of the red and blue channels for addition and reduction in the mesh.

The color ramp is a parameter assigned to each particle in the system. When a particle collides with a surface, the ramp starts at the 0 position and shifts to the 1 position over time, shown in Figure 5.20. To simulate the loss of energy over time, the user would define the blue value at 0 and decrease that blue value over time. By shifting the value from blue to red as in Figure 5.21, we can simulate the sedimentation and deposition processes that occur in weathering.



Point No. 1
Position 0
Color 0 0 0.9
Interpolation Linear

Figure 5.20: Representation of the addition and reduction in the mesh through a color ramp.

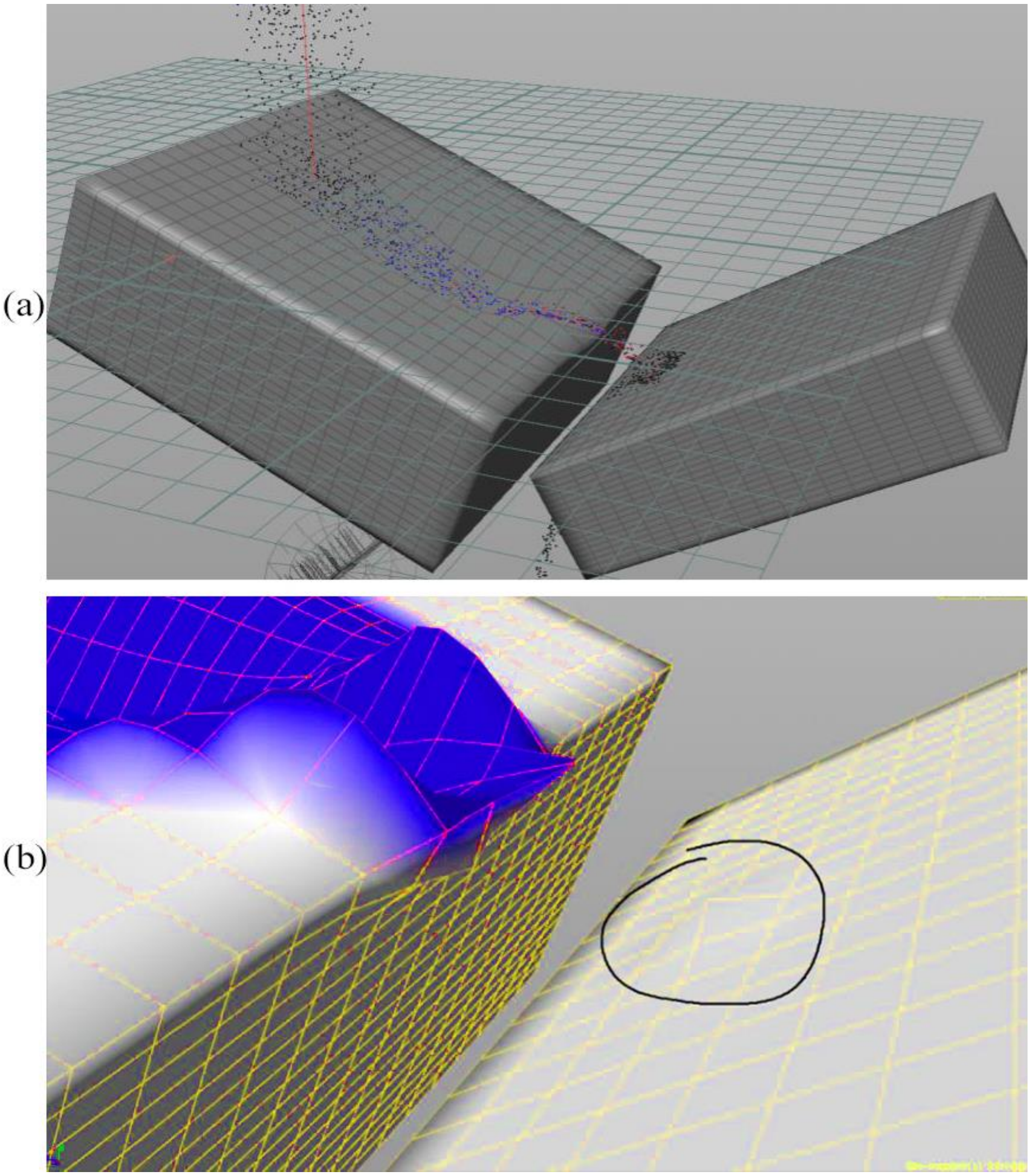
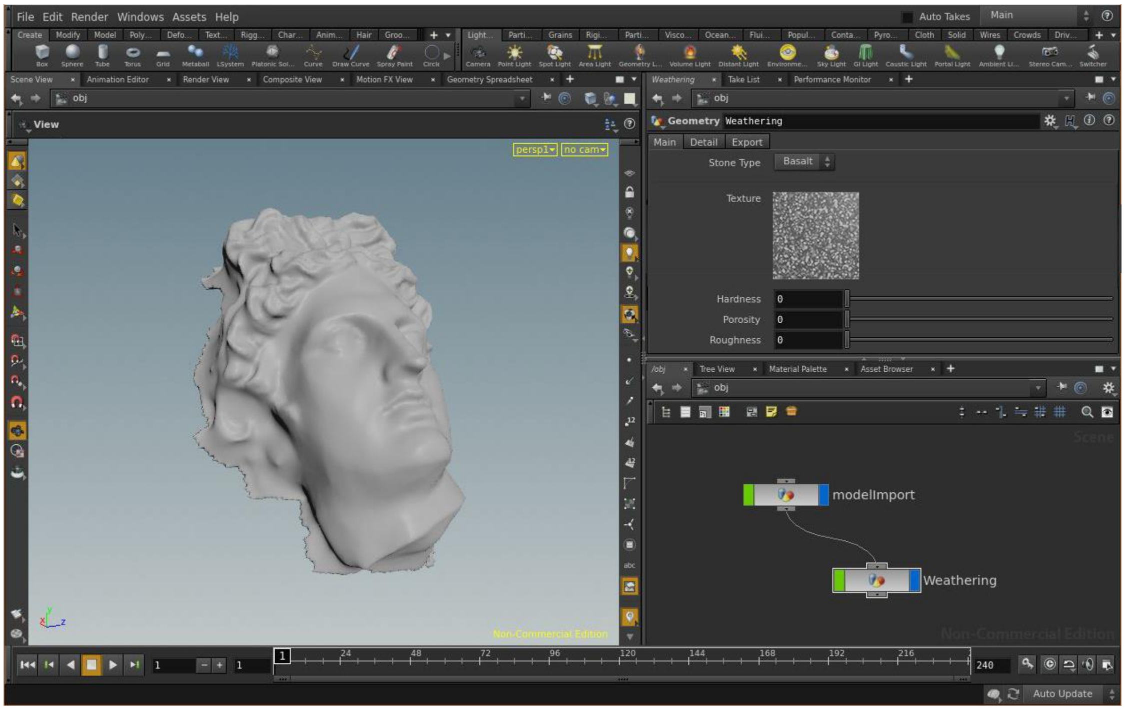


Figure 5.21: Results of the color ramp from Figure 5.20. (a) The setup of the sedimentation/deposition tests. (b) Close-up of the mesh sedimentation, left, and deposition, right.

5.2. User Interface

The user interface created makes this system useable by an artist. The interface needed to be designed in a way that allows the artist to have the tools they need to execute the simulation while keeping the irrelevant parameters hidden. One of the initial UI passes is shown in Figure 5.22. Figure 5.22 shows the initial setup when the system is loaded into Houdini. The system is loaded by an .otl file format and creates a single node upon load. By clicking on this node, a custom UI is loaded into Houdini’s parameter interface. To use the system, the user attaches the geometry they will work with, adjust parameters, and run the simulation.


Geometry Weathering

Stone Type: Basalt

Texture

Hardness: 0

Porosity: 0

Roughness: 0

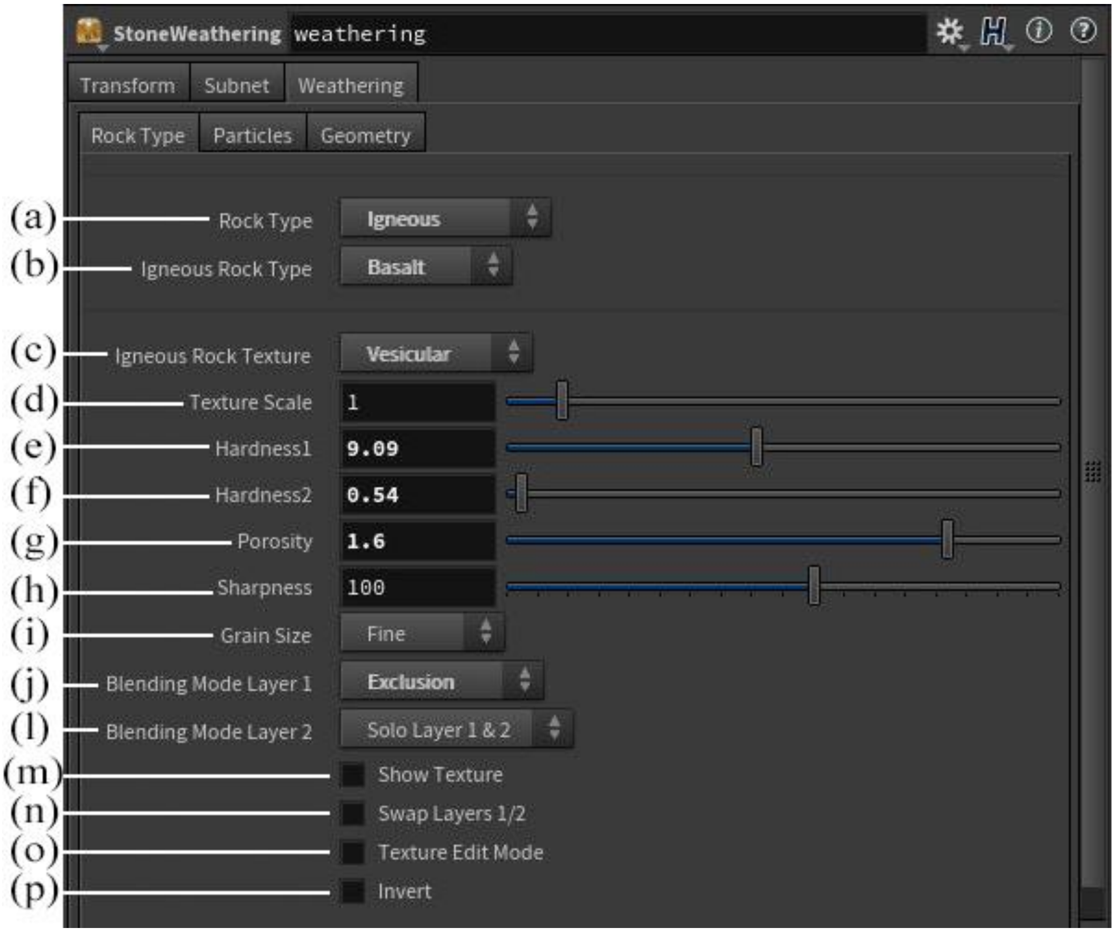
Figure 5.22: UI mockup in Houdini.

To reduce the number of parameters shown at once, the interface is split up into three tabs. These tabs are:

- **Rock Type**
- **Particles**
- **Geometry**

5.2.1. UI Tab: Rock Type

The rock type UI tab is how the user defines the composition of the stone object. It contains the controls that affect the texture and that texture's influence in the weathering process.



StoneWeathering weathering

Rock Type: Igneous

Igneous Rock Type: Basalt

Igneous Rock Texture: Vesicular

Texture Scale: 1

Hardness1: 9.09

Hardness2: 0.54

Porosity: 1.6

Sharpness: 100

Grain Size: Fine

Blending Mode Layer 1: Exclusion

Blending Mode Layer 2: Solo Layer 1 & 2

□ Show Texture

□ Swap Layers 1/2

□ Texture Edit Mode

□ Invert

Figure 5.23: UI tab: Rock Type

- **Rock Type** (Figure 5.23a): Specifies the family of rock.
- **Family Rock Type** (Figure 5.23b): Specifies the type of rock. Contextual with the rock type so sedimentary rocks are restricted to only show when sedimentary is selected in Figure 5.23a. The rock type also presets all following options to

correspond to the specific rock. There is also an option for a custom type of rock that sets no presets.

- **Family Rock Texture** (Figure 5.23c): Sets the texture of the rock.
- **Texture Scale** (Figure 5.23d): Scales the texture. This option is used to match the texture's scale with the object's scale.

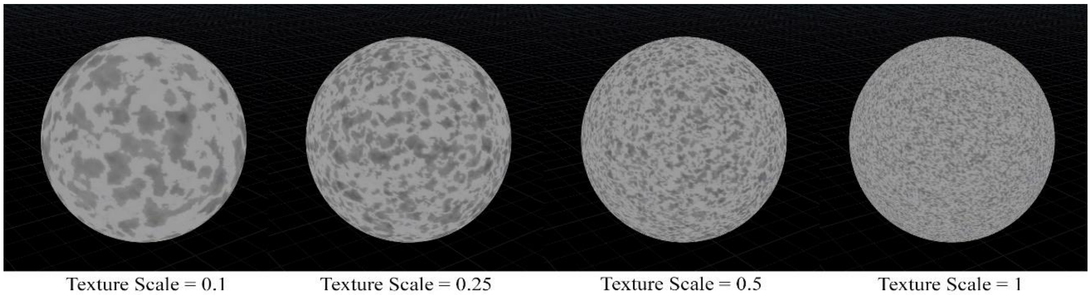


Figure 5.24: Effects of texture scale on the noise pattern.

- **Hardness 1** (Figure 5.23e): The hardness value of the first layer of noise. Higher hardness values result in more resistance to weathering.

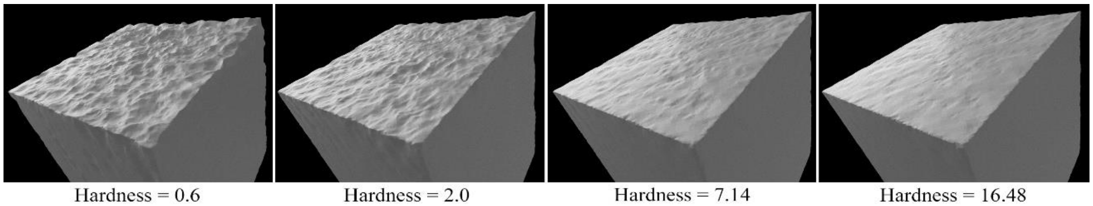


Figure 5.25: Effects of the hardness parameter on a mesh.

- **Hardness 2** (Figure 5.23f): The hardness value of the second layer of noise.

- **Porosity** (Figure 5.23g): Works similar to the hardness values, but has a different scale. Applied to the third layer of noise. Higher porosity values result in less resistance to weathering.

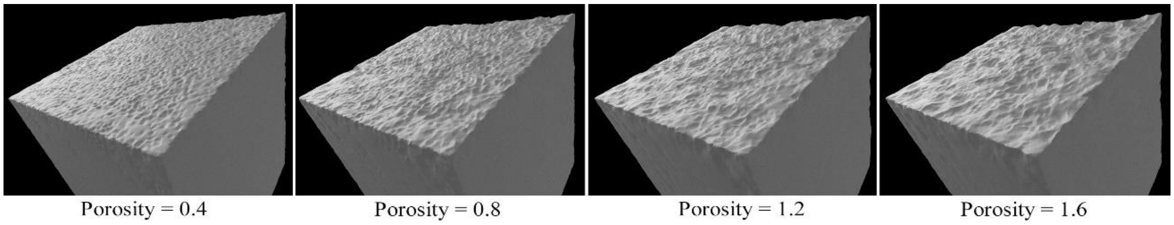


Figure 5.26: Effects of the porosity parameter on a mesh.

- **Sharpness** (Figure 5.23h): The maximum angle to keep before smoothing the edge out after a particle collision. Higher sharpness values result in sharper edges.

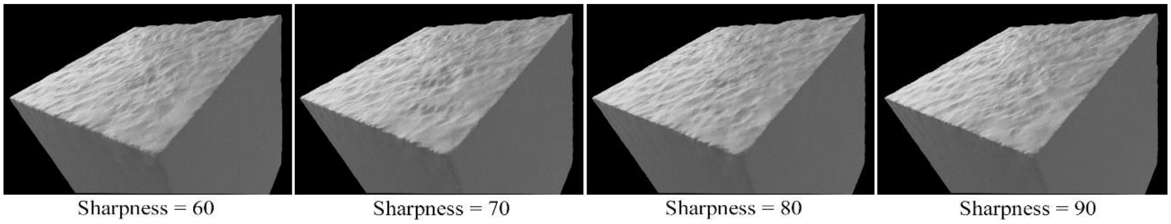
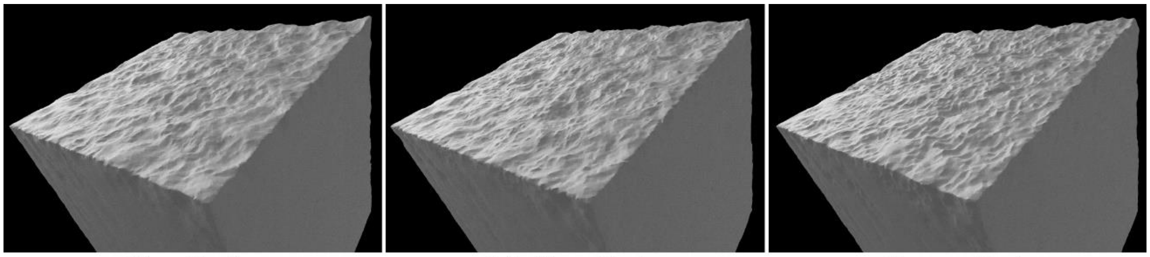


Figure 5.27: Effects of the sharpness parameter on a mesh.

- **Grain Size** (Figure 5.23i): The size of the noise in the third layer.

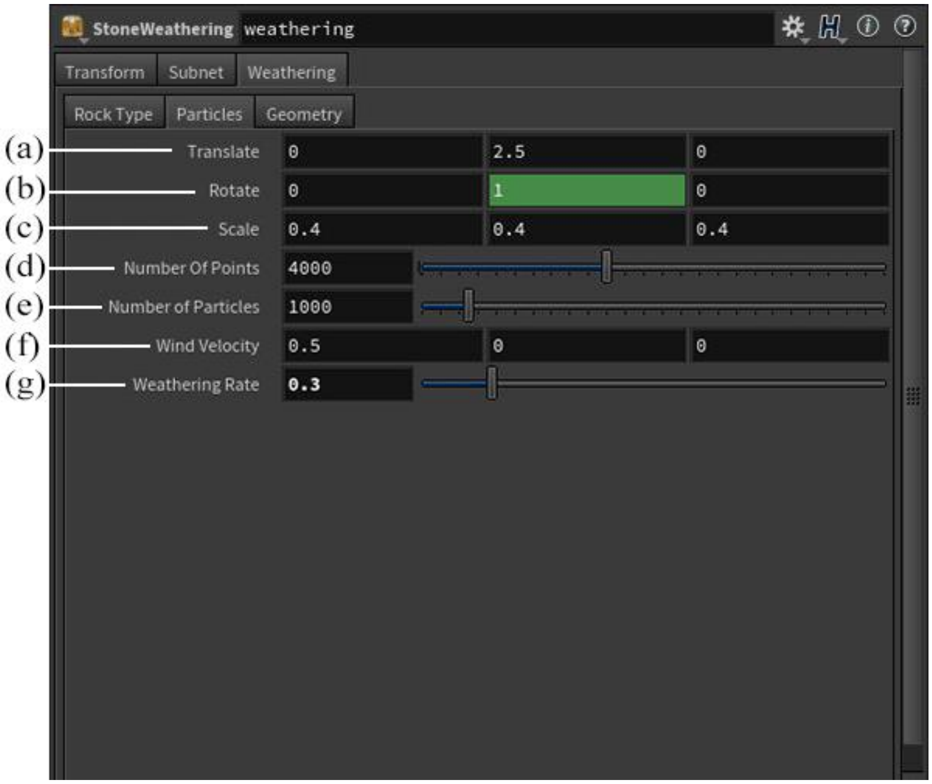


Fine Grain Medium Grain Coarse Grain

Figure 5.28: Effects of the grain size parameter on a mesh.

5.2.2. UI Tab: Particles

The particles UI tab is how the user manipulates the attributes for the particle system.



StoneWeathering weathering

Translate 0 2.5 0

Rotate 0 1 0

Scale 0.4 0.4 0.4

Number Of Points 4000

Number of Particles 1000

Wind Velocity 0.5 0 0

Weathering Rate 0.3

Figure 5.29: UI tab: Particles.

- **Translate, Rotate, Scale** (Figure 5.29a/b/c): The manipulation tools for the particle emitter.
- **Number of Points** (Figure 5.29d): Controls how many different places the particles can spawn
- **Number of Particles** (Figure 5.29e): Controls how many particles come out of a point.
- **Wind Velocity** (Figure 5.29f): Adjusts the force of wind on the particles. This can be used to shape and direct the way the particles fall towards the surface.
- **Weathering Rate** (Figure 5.29g): Defines how much of the mesh is displaced when a particle collides with the surface.

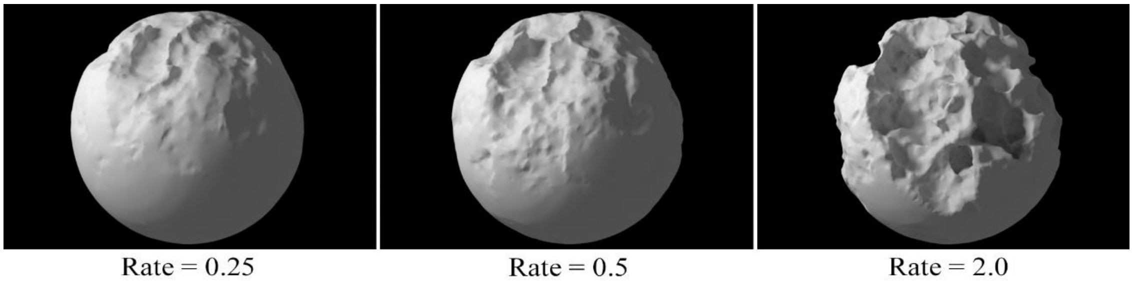
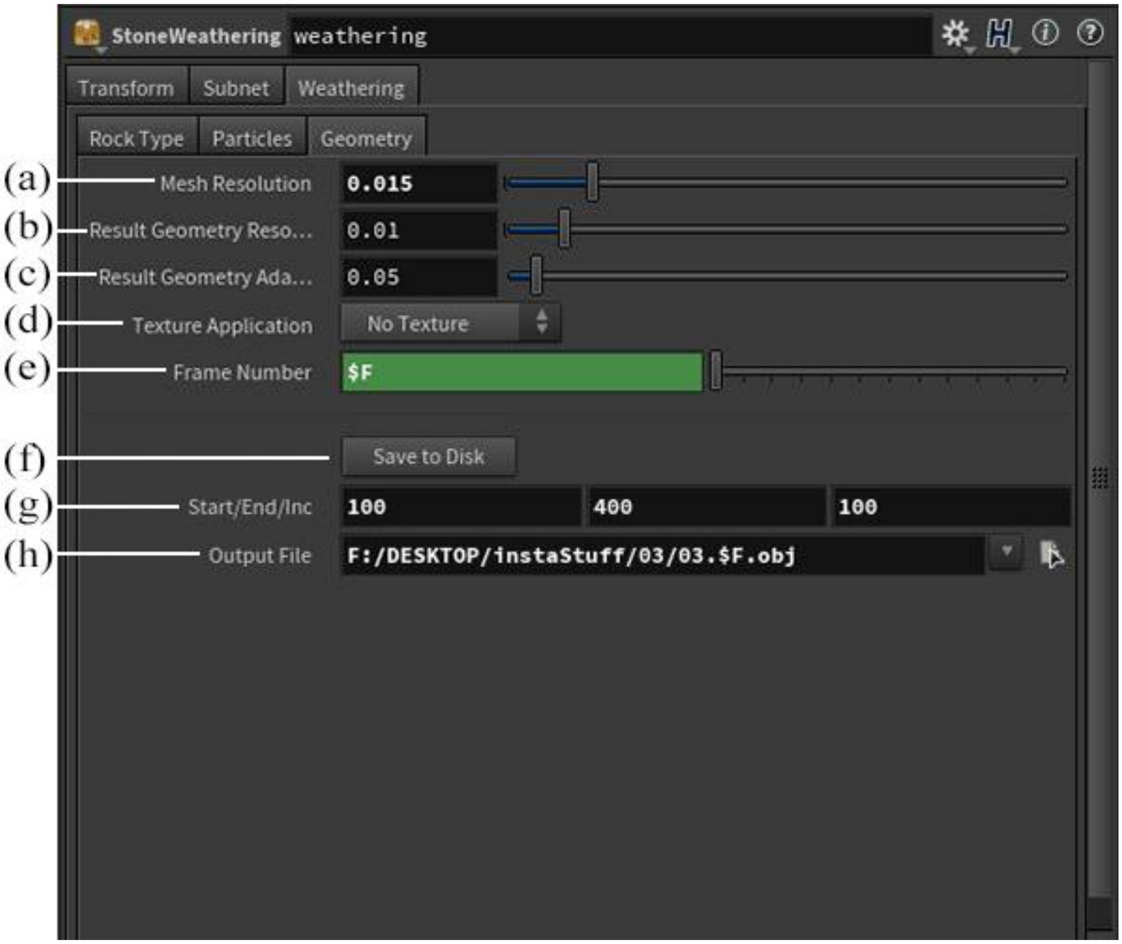


Figure 5.30: Effects of the rate parameter on a mesh.

5.2.3. UI Tab: Geometry

The final tab of the UI deals with the resolution of the geometry, how the geometry is exported, and the time in the weathering process.



StoneWeathering weathering

Mesh Resolution 0.015

Result Geometry Reso... 0.01

Result Geometry Ada... 0.05

Texture Application No Texture

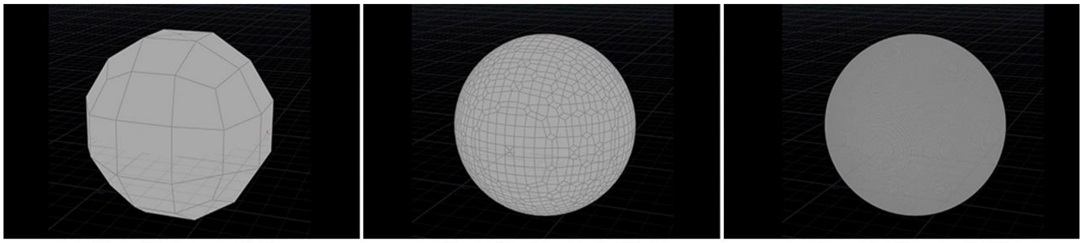
Frame Number $F

Start/End/Inc 100 400 100

Output File F:/DESKTOP/instaStuff/03/03.$F.obj

Figure 5.31: UI tab: Geometry.

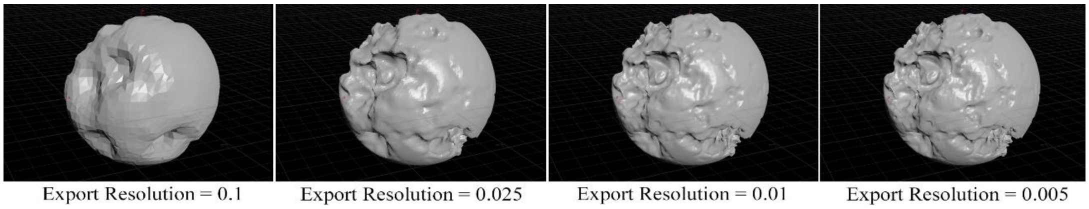
- **Mesh Resolution** (Figure 5.31a): Controls the density of the mesh. Smaller values create denser, more detailed meshes at a cost of processing time.



Resolution = 0.1 Resolution = 0.05 Resolution = 0.01

Figure 5.32: Effect of the mesh resolution parameter on a mesh's wireframe.

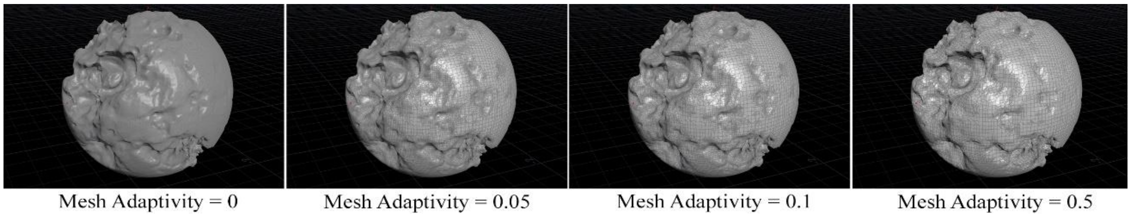
- **Result Geometry Resolution** (Figure 5.31b): Similar to the mesh resolution in Figure 5.31 a, but this is applied to the output geometry.



Export Resolution = 0.1 Export Resolution = 0.025 Export Resolution = 0.01 Export Resolution = 0.005

Figure 5.33: Effect of the export resolution parameter on a mesh.

- **Result Geometry Adaptivity** (Figure 5.31c): Adaptivity affects how the model's density is distributed. Higher adaptivity values take some of the mesh density out of uniform areas but keeps density in areas with more contrast.



Mesh Adaptivity = 0 Mesh Adaptivity = 0.05 Mesh Adaptivity = 0.1 Mesh Adaptivity = 0.5

Figure 5.34: Effect of the mesh adaptivity parameter on a mesh.

- **Texture Application** (Figure 5.31d): This option is used to apply texture maps to the object. Although 2D textures are not used in the system anymore, the option was kept to allow the user the ability to apply a texture mask to mask parts of the model to resist weathering.
- **Frame Number** (Figure 5.31e): This controls the frame number the system will simulate to. The frame number is set up to be independent of the main timeline

so that the user can work in the same scene as the simulation without manipulating the weathered object through moving the time frame. A command of $F may also be passed through to allow scrubbing through the timeline.

- **Save Button** (Figure 5.31f): The button that takes the saves the specified file or files designated in the frame range to the file location.
- **Save Frame Range** (Figure 5.31g): The three fields correspond to the start frame, the end frame, and the increment amount. In the figure, the object at frame 100, 200, 300, and 400 will be saved. This is because the start frame is 100, the end frame is 400, and will save every 100 frames.
- **Output file location** (Figure 5.31h): This is a basic file path. The user can click on the button to the right of the field to open up a GUI menu rather than typing in the file path.

6. IMPLEMENTATION

6.1. Houdini

For this project, Side Effect's Houdini FX version 15.0 was used. Houdini was chosen due to its ability to handle mass amounts of data efficiently. Houdini also uses a non-destructive, node based system. The node based system allows us to present the user with many variables that can be changed or adjusted without being destructive to the process as a whole [41].

6.2. Using 3D Noise as Hardness

The noise corresponds to the texture pattern as well as the hardness values for the minerals in the composition. The noise generated for each texture is a black and white value map with values that range from 0 to 1. This map is applied to the object in 3D space by vertex color. Vertex color was chosen to keep UVs and 2D texture maps out of the process. This ensures that the object always has a seamless texture and is not distorted on the object. Figure 6.1 shows the use of 3D textures on multiple objects and scales, allowing manipulations in the objects without a loss in the texture. The drawback of using the vertices to store the texture information is that the mesh density has a direct relationship with the quality of the texture. This relationship requires higher density geometry to achieve higher weathering details, which increases the processing time.

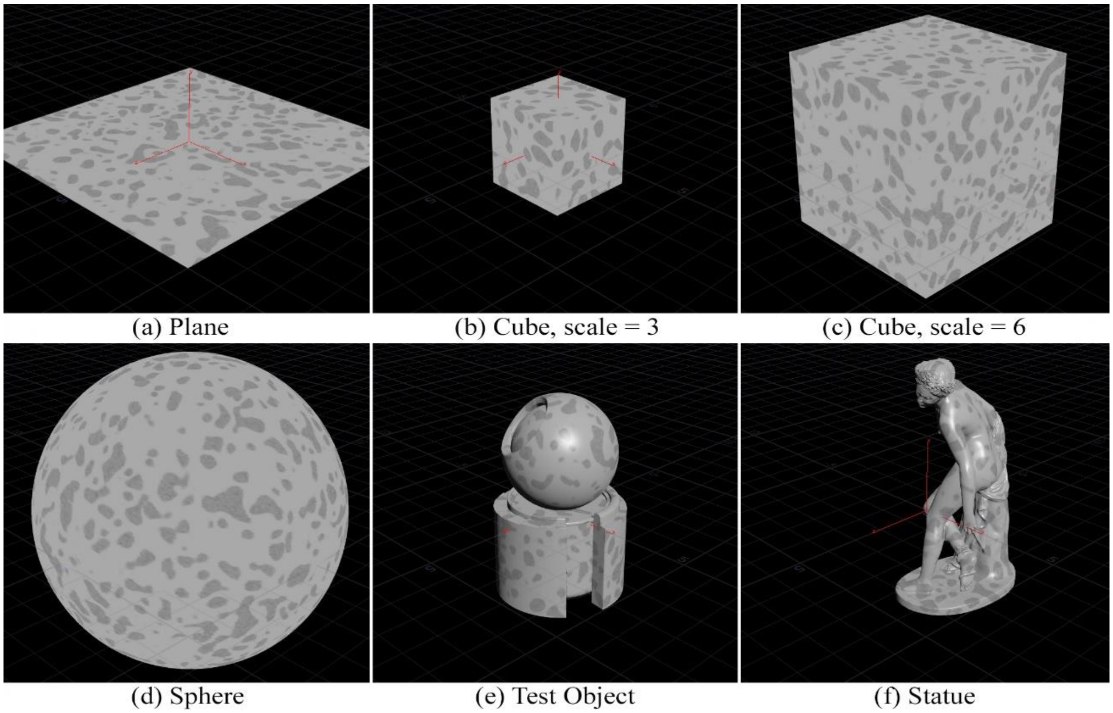


Figure 6.1: Application of the same 3D texture on multiple different meshes.

Figure 6.2 below shows the node network used to apply and create the 3D noise textures. These sections are broken into:

- **Point import** (Figure 6.2a)
- **Noise layers** (Figure 6.2b)
- **Rock type switch** (Figure 6.2c)
- **Hardness** (Figure 6.2d)
- **Layer input** (Figure 6.2e)
- **Blending modes** (Figure 6.2f)
- **Output** (Figure 6.2g)

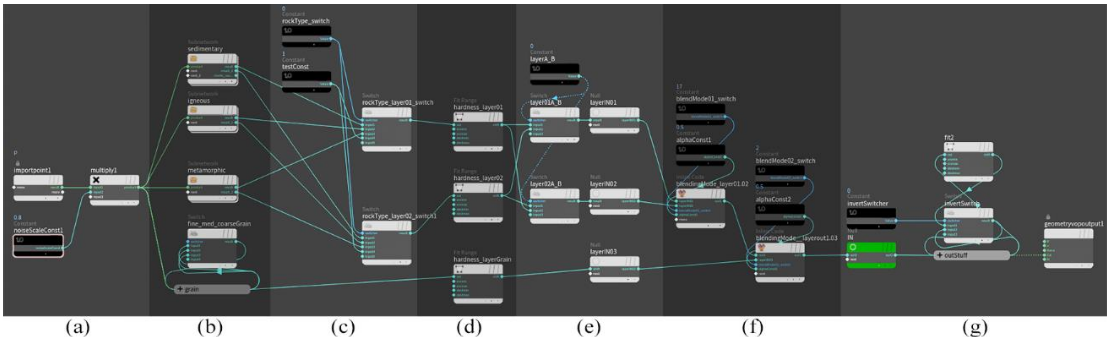


Figure 6.2: 3D noise node network.

6.2.1. Point Import

The section in Figure 6.2a imports the position parameter of the point being looked at. The point is taken from the geometry input of the root node that the network is inside of. In Houdini, each object has a list of points that are numbered starting from 0. Each point has attributes such as location, color, alpha, texture, weight, and normal direction [41]. After importing the points that make up the geometry, we multiply it by the texture scale parameter set in the UI to adjust the size and scale of the resulting texture.

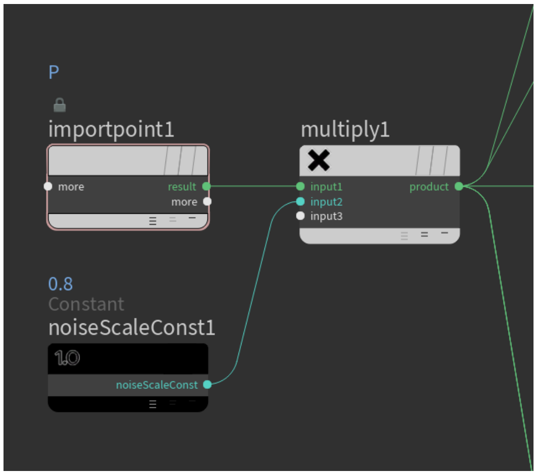


Figure 6.3: Point import nodes.

6.2.2. Noise Layers

Due to the amount of different options for the stone texture, this section was broken up into subgroups by using the subnets in Houdini. A subnet acts like a folder with input and output options that contain the nodes specified. Figure 6.4a shows the root of all the texture nodes split into sedimentary, igneous, metamorphic, and grain size. Inside the igneous node in Figure 6.4b, we see all of the options given for igneous. The rock type is changed using switch nodes. The switch node is controlled by changing a constant value which changes when the Family Rock Texture parameter is changed the UI. Figure 6.4c dives into the vesicular texture node where it is split into the two layers used in the texture. Each stone texture follows the same format of applying a noise node, attaching a ramp parameter to that noise, and then further manipulation such as multiplies or power nodes to add contrast. These nodes are then output to the right of the node in Figure 6.4a as result1 and result2.

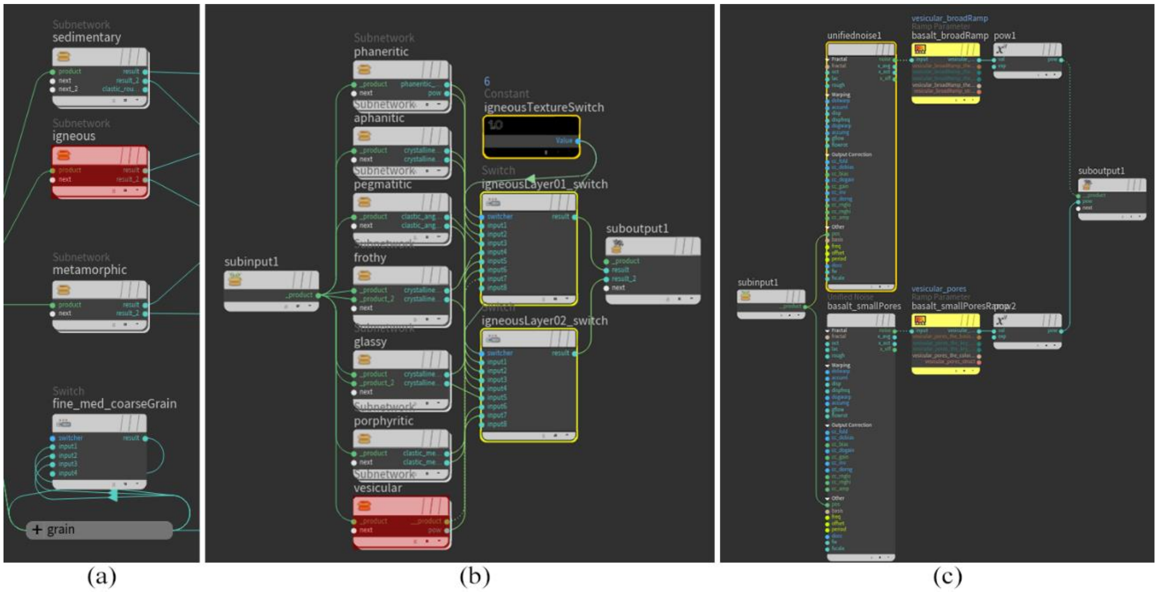


Figure 6.4: Noise layer nodes.

6.2.3. Rock Type Switch

The switch in Figure 6.2c switches the rock type between igneous, metamorphic, and sedimentary. The rock type switch performs similarly to the switch used in Figure 6.4b to change the type of texture applied to the rock. The rockType_switch variable is changed by changing the Rock Type in the UI.

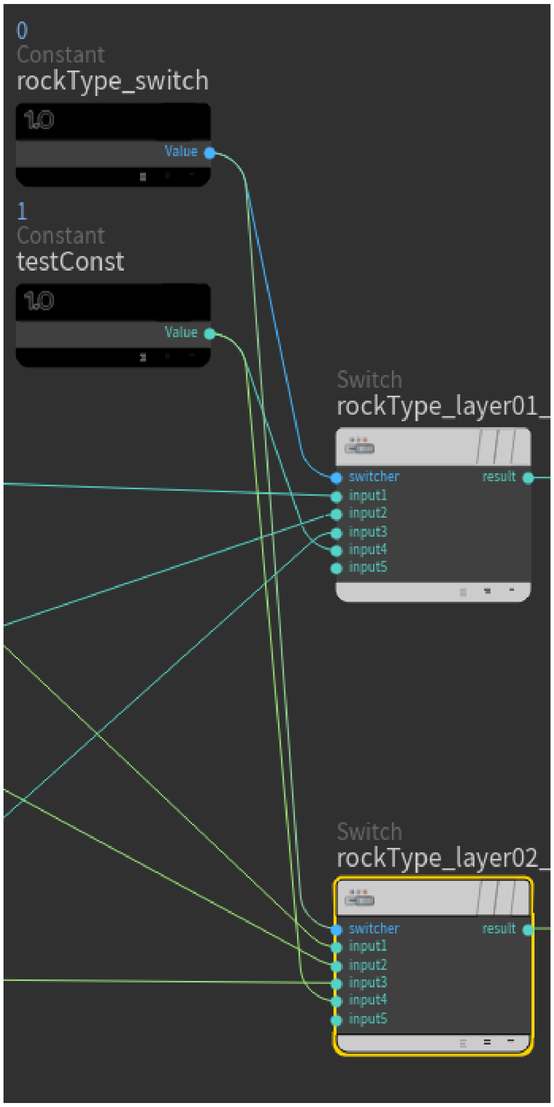


Figure 6.5: Rock type switch nodes.

6.2.4. Hardness

After the rock type and texture is decided, the hardness value for each layer is applied in Figure 6.2d. For this thesis, the hardness value is the greyscale value assigned to a certain point in the 3D texture. Hardness values in the map only range from 0 to 1, so to recontextualize these values we use a fit range node. This node takes existing values and scales them to fit within another set of values. For example, Figure 6.6a shows the noise as it comes into the fit range node. In Figure 6.6b, we have a rock with a hardness value of 10, that becomes the noise's source maximum (srcmax). The destination minimum and maximum (destmin, destmax) are 0 and 1. When the srcmax is set to 10, all values that go through the fit range node are pushed to be between 0 and 1. In this case, the value of 1 from Figure 6.6a becomes 0.1 in Figure 6.6c. The hardness values in the UI change the srcmax values in these nodes.

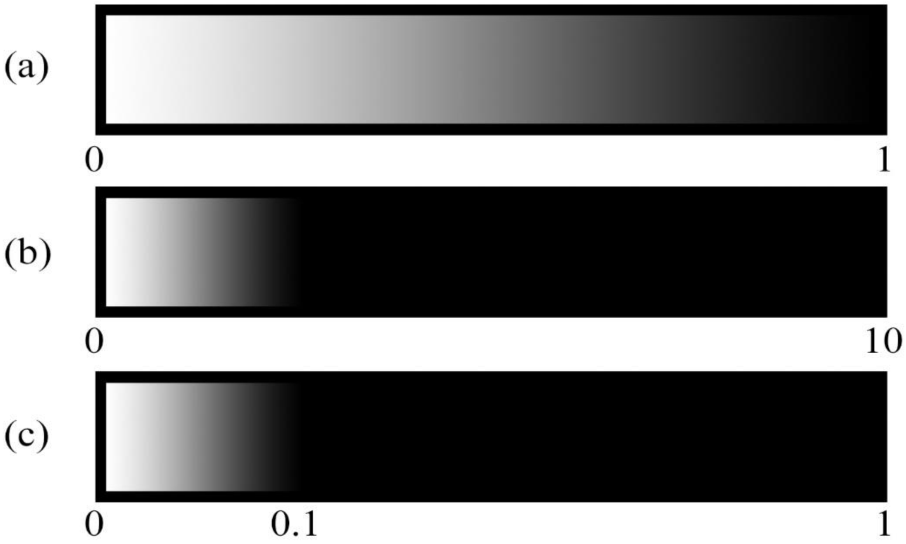


Figure 6.6: Representation of the value change after using a fit range node.

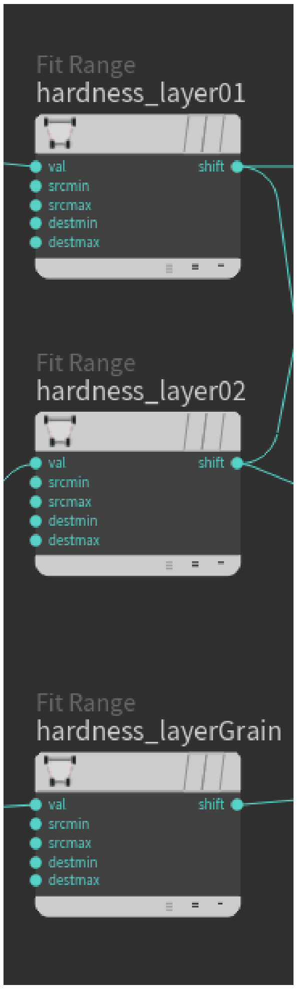


Figure 6.7: Hardness nodes.

6.2.5. Layer Input

The layer input block at Figure 6.2e collects the texture information to pass on to the layer blending nodes. The switch option allows the user to switch which layer is on top, often changing the way the blending options work. We use the null nodes to rename the outputs before they reach the next section.

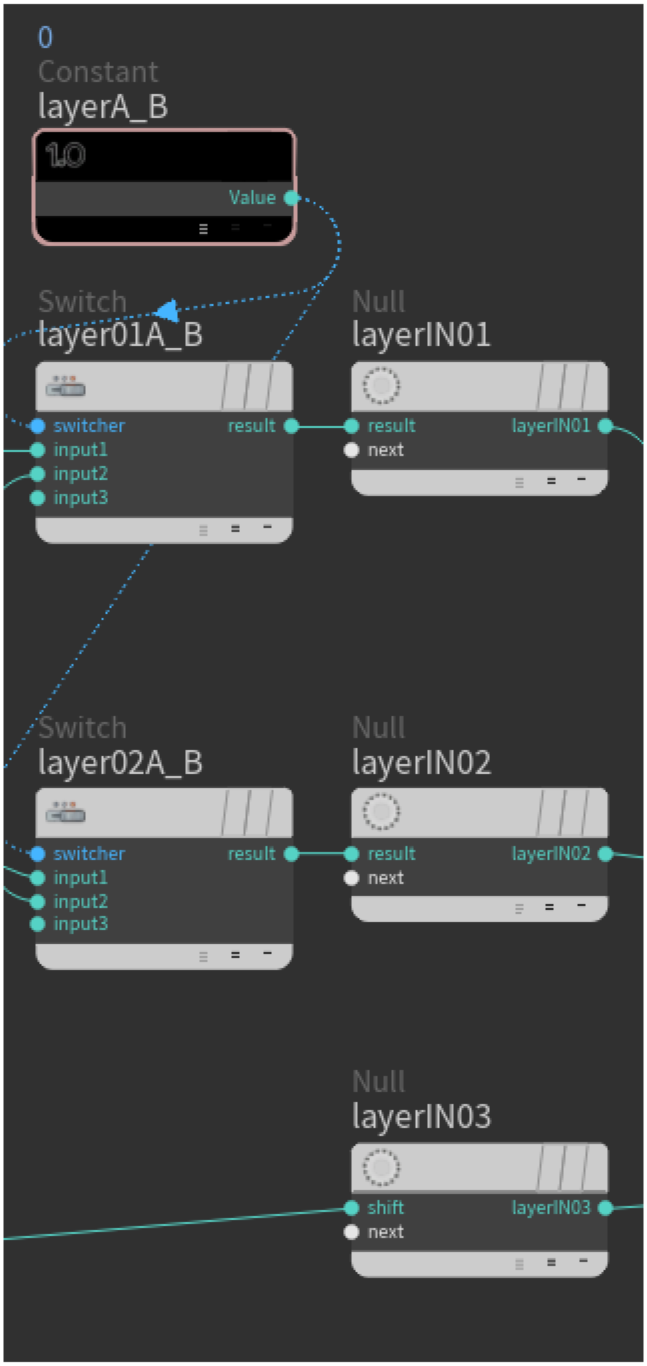


Figure 6.8: Layer input nodes.

6.2.6. Blending Modes

Figure 6.2f applies the blending mode nodes. Like the variables before, they are changed by the switch nodes and chosen from the UI. The alpha value is also changed from the UI and changes the alpha variable in the script.

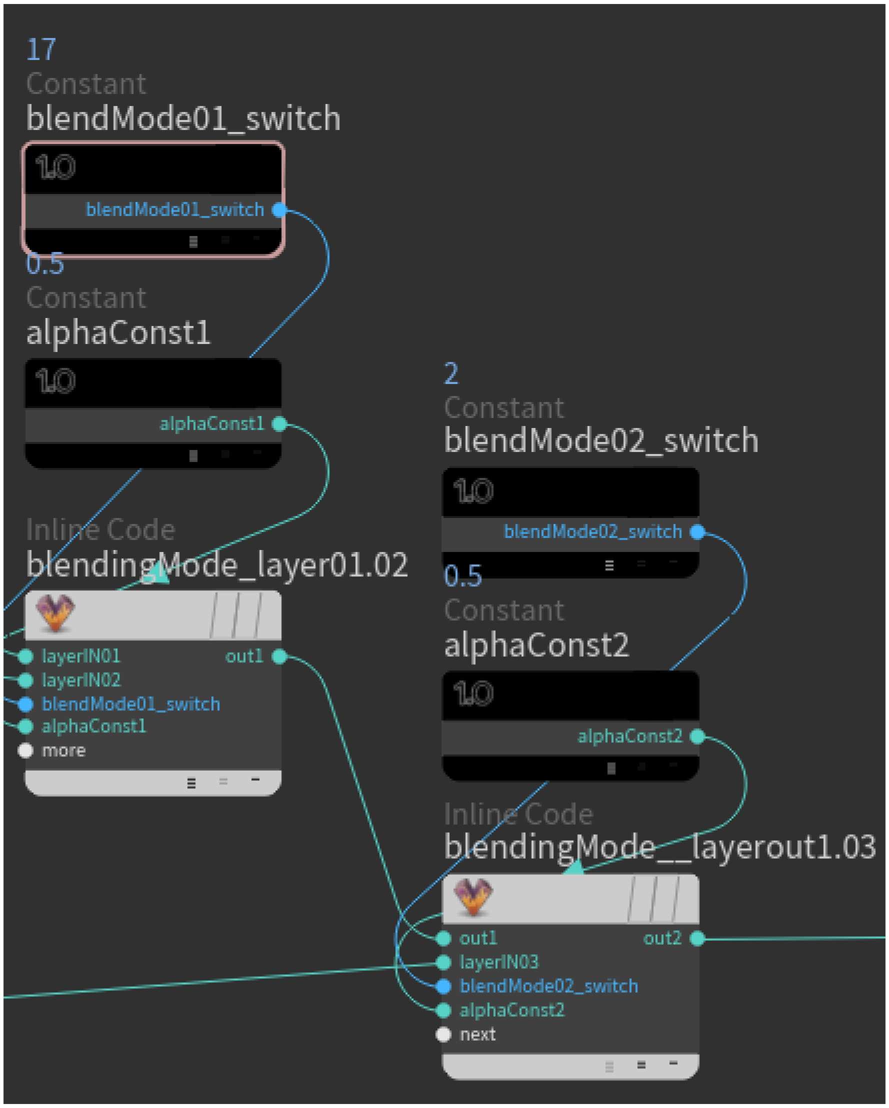
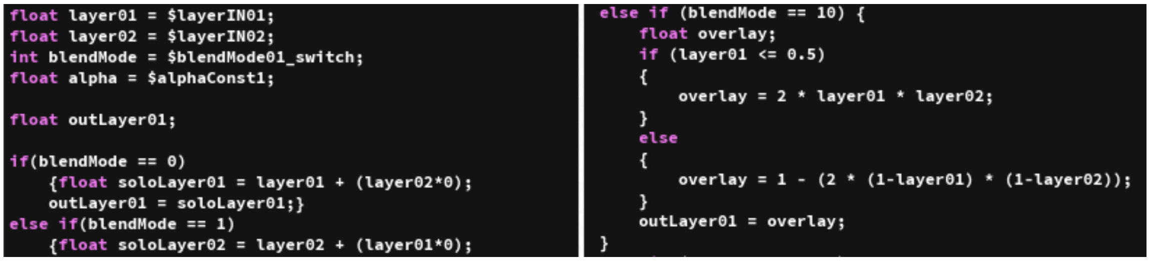


Figure 6.9: Blending mode nodes.

To manipulate the layers to composite onto each other using the blending modes, we use an Inline Code node to input custom VEX scripts. The VEX code starts by initiating parameters for the layers, blending mode constant from the UI, the alpha value from the UI, and the output variable. The script then calls the blendMode integer variable and checks its value. Depending on the value, the script will run a different formula that corresponds to the blending mode set by the user.



```
float layer01 = $layerIN01;
float layer02 = $layerIN02;
int blendMode = $blendMode01_switch;
float alpha = $alphaConst1;

float outLayer01;

if(blendMode == 0)
    {float soloLayer01 = layer01 + (layer02*0);
    outLayer01 = soloLayer01;}
else if(blendMode == 1)
    {float soloLayer02 = layer02 + (layer01*0);
```

```
else if (blendMode == 10) {
    float overlay;
    if (layer01 <= 0.5)
    {
        overlay = 2 * layer01 * layer02;
    }
    else
    {
        overlay = 1 - (2 * (1-layer01) * (1-layer02));
    }
    outLayer01 = overlay;
}
```

Figure 6.10: VEX code snippet used to create the blending modes.

6.2.7. Output

The final block in the texture application, Figure 6.2h, is clean up and color output. This section runs the current output through an absolute value node so that there are no negative values. Then an overall ramp is applied to the system to give the user an option to apply a broad change to the compiled noise. The switch node setup looks for a toggle in the UI to invert the noise as a whole. The inversion is done by using another fit range function and setting the destination minimum to be 1 and the destination maximum to be 0. The output is then put into the geometry’s Cd, or color, attribute.

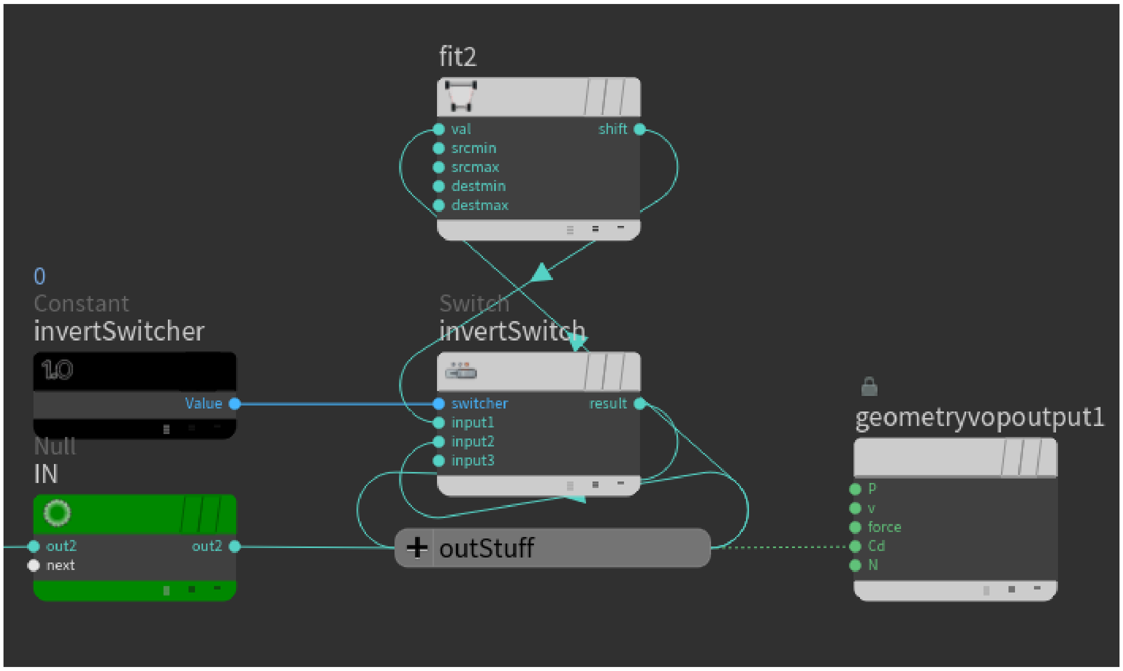


Figure 6.11: Output nodes.

6.3. Displacement

Figure 6.12 displays the node network that displaces the points on the surface of the object. The node network containing the application of the hardness values is used as the input to the node network controlling the displacement. The hardness values are used to manipulate values in this system. This section is broken up into:

- **Texture import** (Figure 6.12a)
- **Particle color** (Figure 6.12c)
- **Multiply particle velocity** (Figure 6.12d)
- **Displacement output** (Figure 6.12b)

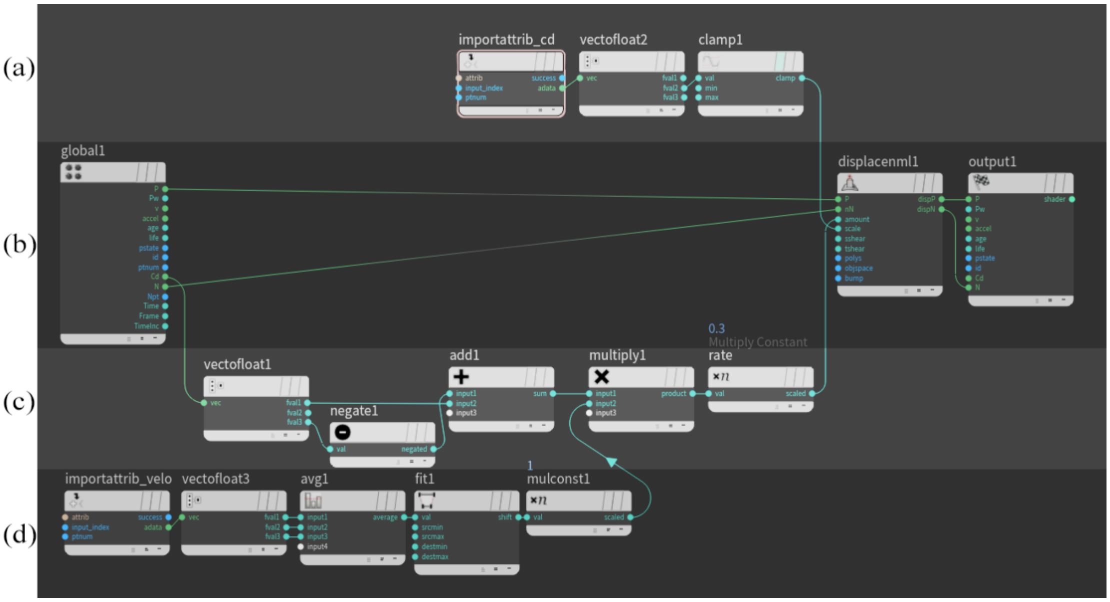


Figure 6.12: Displacement node network.

6.3.1. Texture Import

The nodes in section Figure 6.12a import the RGB values from the hardness node's input. The green channel (fval2) is only used here due to the use of the red and blue channels later in the system. Due to the noise textures only setting up black and white values, the green channel does not affect the output. The clamp node is in place as a fallback if the values imported are not between 0 and 1.

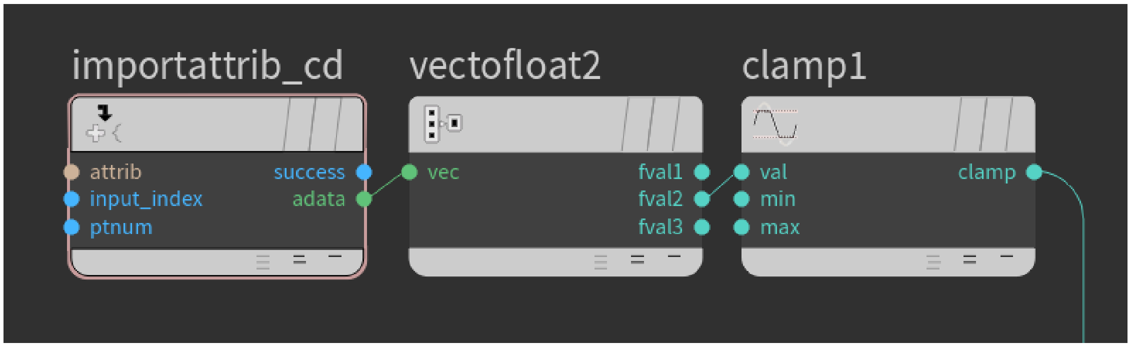


Figure 6.13: Texture import nodes.

6.3.2. Particle Color

Figure 6.12c reads the color of the object's surface. When a particle hits the object's surface, a Transfer Attribute node transfers the color of the particle to the vertices on the surface that particle collides with. The color is read from the global import node and is converted into three float values that correspond to the red, green, and blue channels. The red channel (fval1) moves on while the blue channel (fval3) gets negated, or multiplied by -1. The red and blue channel values get added together after they separate in case the color of the particle becomes white or a shade not needed to keep the values between 0 and 1. We then put the result of this into a multiply node where we multiply it with the velocity of the particle. Then the total value is multiplied by the rate of weathering defined in the UI.

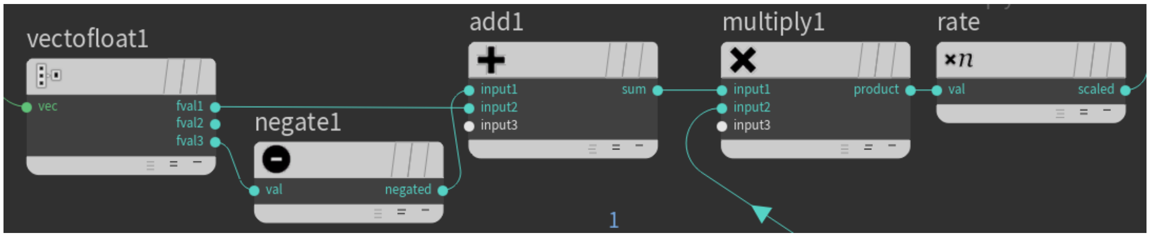


Figure 6.14: Particle color nodes.

6.3.3. Particle Velocity

The nodes in Figure 6.12d read in the velocity values of the particle on and during collision with the object. The velocity is used to simulate the loss of energy in the sedimentation process when the particle is moving slower. When a particle collides with the surface, the velocity is imported and the vector of that velocity is averaged into a float value. This float value is fit in between 0 and 1 to keep the results predictable and then multiplied by the particle color value. By using the particle's velocity to manipulate the displacement values, we can ensure that if particles collect in an area they will have no effect on the system.

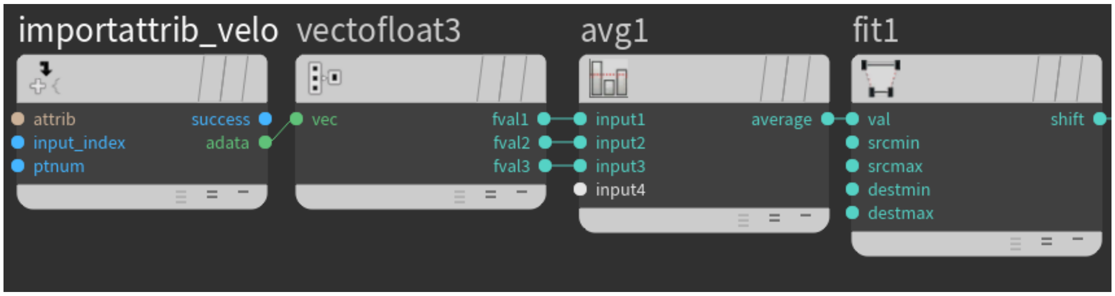


Figure 6.15: Particle velocity nodes.

6.3.4. Displacement Node

The displacement node collects the values and displaces the mesh based on them. The position (P) and normal (nN) are pulled from the imported mesh. The amount is the result of the particle color and particle velocity. The scale multiplies the amount by the hardness values imported from the color of the mesh. The updated position and normal values are then exported to displace the mesh.

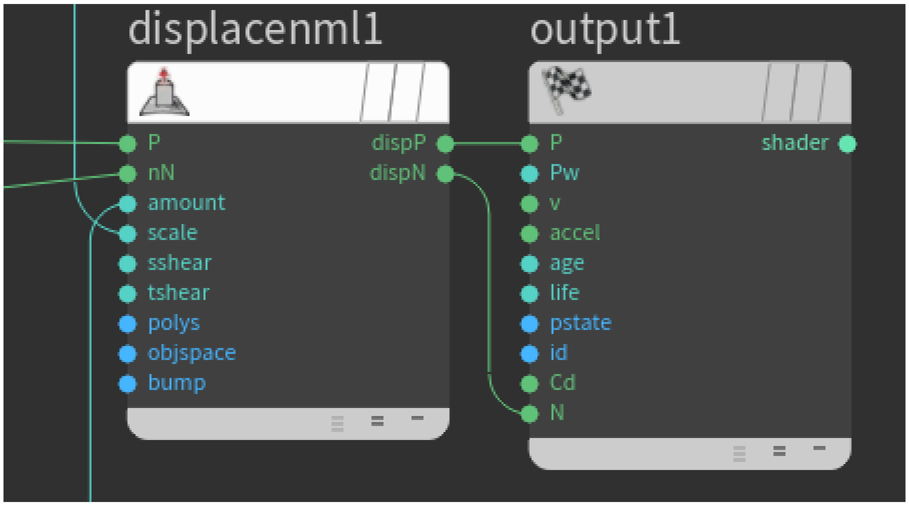


Figure 6.16: Displacement nodes.

6.4. Sharpness

When a stone is weathered, depending on the hardness of the materials, sharp edges become smoothed out. To recreate this, we needed to find a way to smooth out the edges of the objects as the particles flow over the object during the simulation. Initially, the system was pushing geometry through itself on the sharp edges. This artifact, shown in Figure 6.17, comes from using the normal of the mesh to inform the direction of displacement.

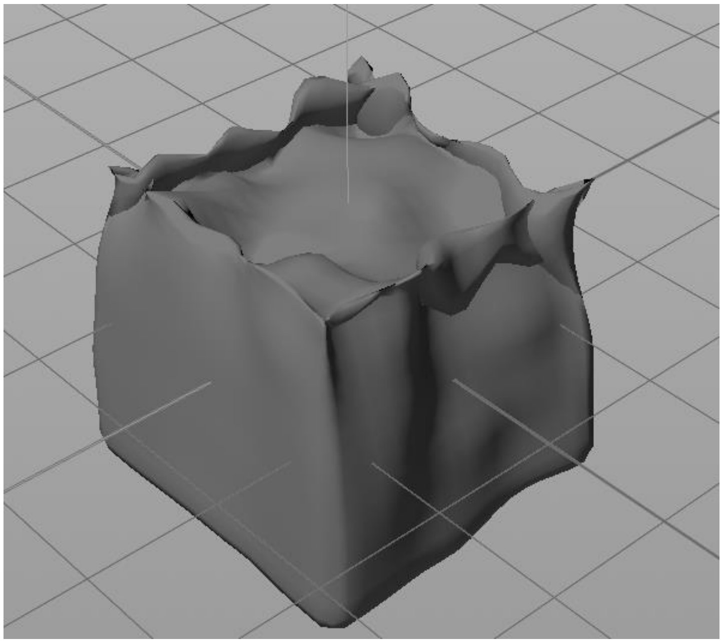
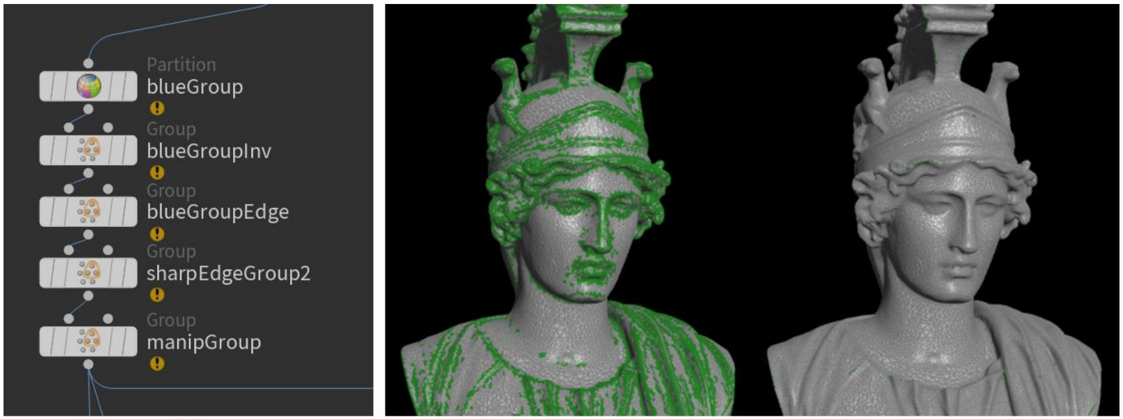


Figure 6.17: Result of using the initial system on sharp edges.

To solve the intersection of mesh, we can define an angle to check the object for and change those angles before the mesh intersects with itself. To define this angle, we use a group node to add edges with a minimum angle of the value specified. This is visible in Figure 6.18b and c, where the faces in green are faces that will be added to the group. Once the faces are added to a group, we can call that group later in the network to manipulate only members of that group. Figure 6.18a is the network in the system. Since we only want to affect the faces that have been hit by a particle, we first check to see if any parts of the mesh have a value in the blue channel. We then adjust the minimum angle to the sharpness value specified in the UI and combine the groups. If there are any faces that result from this process, they are stored in the manipGroup group.



(a) (b) Min edge angle = 10 (b) Min edge angle = 50

Figure 6.18: (a) Edge detect node network. (b) Effect of the minimum edge angle shown by highlighting the edges in the mesh.

The mainpGroup is then passed onto a switch node, shown in Figure 6.19, that alternates every 5 frames. On frames not divisible by 5, the edges in the manipGroup are smoothed. On every 5th frame, the mesh is retopologized by a VDB process. The VDB triggers every 5 frames to keep the computational times lower.

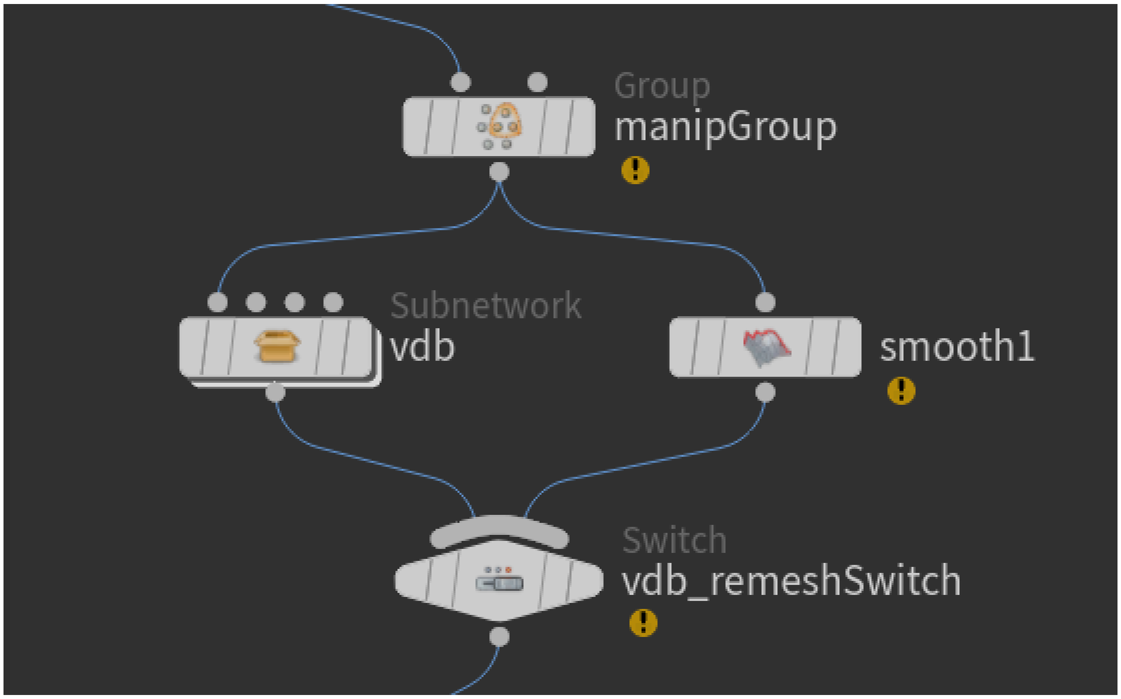


Figure 6.19: Smoothing nodes affecting the edges included in the sharpness group.

6.5. VDB

An issue with displacing the mesh is keeping the mesh density consistent when larger deformations are introduced. If a substantial deformation is done on a mesh without adding more mesh density to the changed area, it will result in visual artifacts on the model. The lack of detail in the deformations also restricts the further deformations in that area due to the area having fewer vertices to attach information to. To solve this problem, we convert the mesh into a VDB volume and then convert it back into a mesh. This creates a consistent density in the mesh without losing details and filling any holes that may of come up in the process. VDB is a volumetric solver that models a virtually infinite 3D space and allows fast data access into high resolution volumes [31]. The VDB operation used for this project converts the mesh into voxels. In the conversion, the size of the voxel affects the amount of detail kept in the VDB. The VDB works much like a volume, so the form fills the space of the mesh rather than being restricted to the surface of the mesh. Figure 6.20 shows the resulting mesh as the voxel size is brought down, allowing more detail in the mesh.

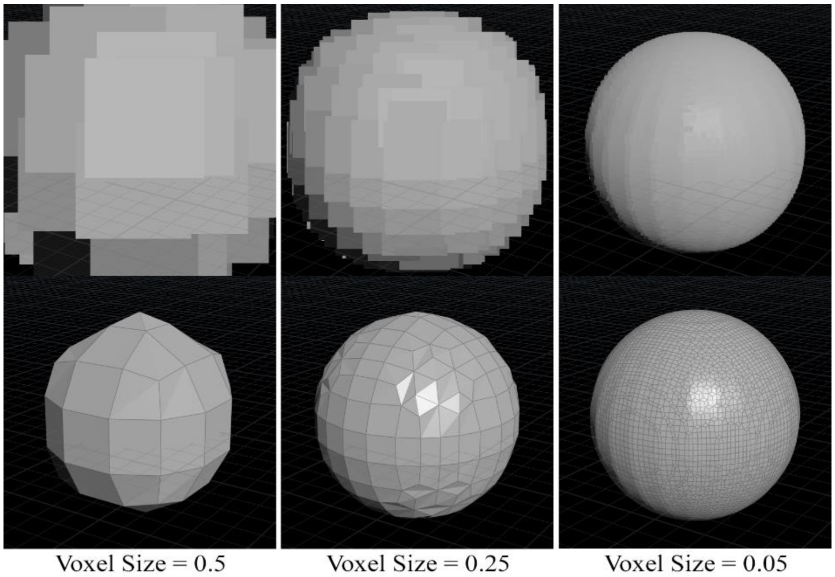


Figure 6.20: Effect of the voxel size parameter on mesh density.

Figure 6.21 shows the contents of the VDB subnetwork in Figure 6.19. These nodes start by deleting (blast node) and replacing geometry (polycap node) where the manipGroup was applied. It then checks for separated mesh with the connectivity node and separates them into groups with the partition node. The foreach node works much like a for loop in a scripting language.

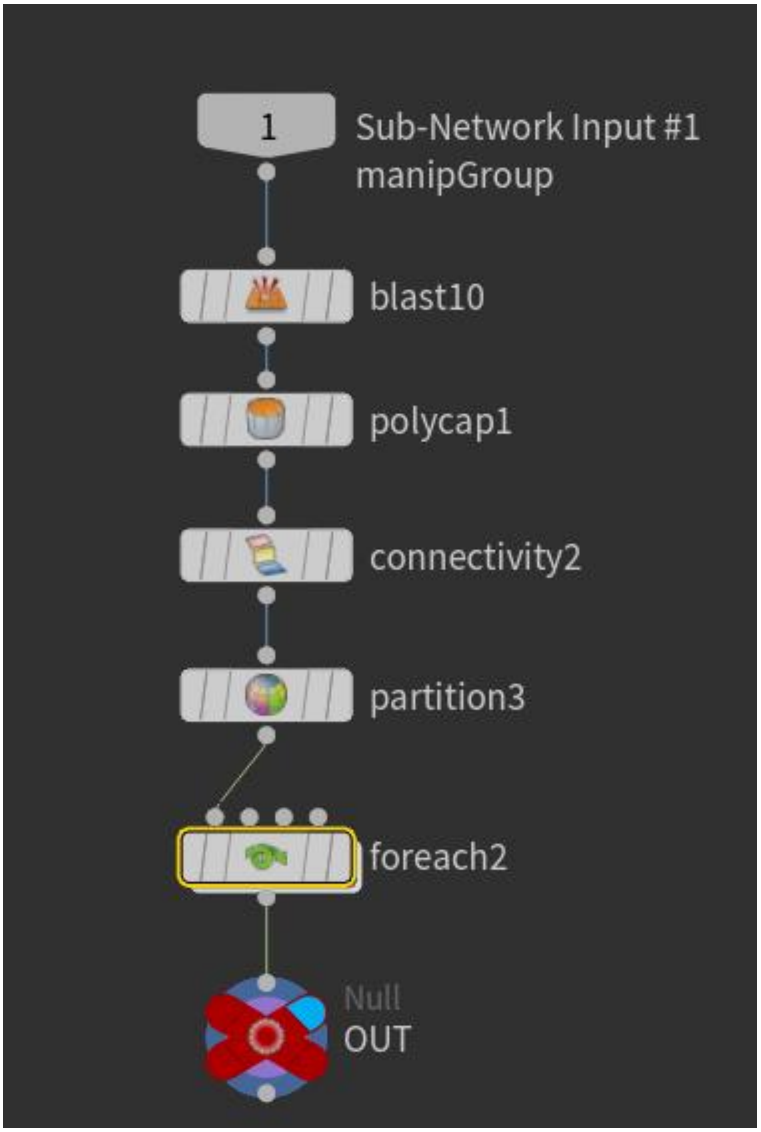


Figure 6.21: Contents of the VDB subnetwork from Figure 6.19.

Figure 6.22 shows the contents of the foreach node. The each node initiates the number of times this loop will run and what objects it will affect each time. The number of times the loop runs is based on the number of separated groups from the partition node. Each of these groups are ran through individually by identifying the groups name, PIECE, followed by the separated object’s number. The vdbfrompolygons node converts the polygon mesh into a VDB volume as in the top row of Figure 6.20. The voxel size in the vdbfrompolygons node is set from the UI. The convertvdb node converts the VDB into a polygonal mesh. To keep the sharper edges of the mesh, we attach the original object as a reference surface in the convertvdb node to sharpen the edges and corners based on that object. The result of this process is a retopologized mesh with no holes and a uniform density.

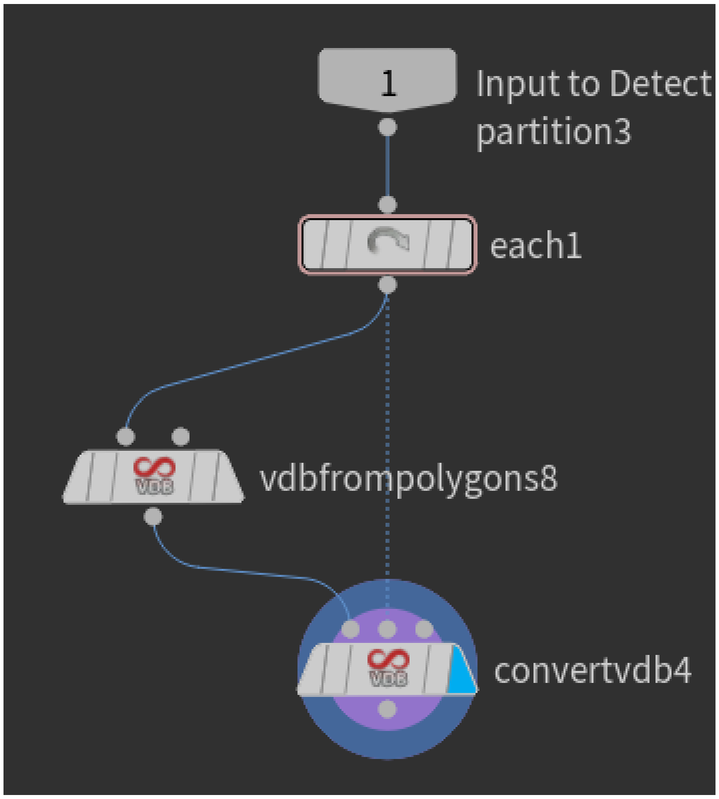


Figure 6.22: Contents of the foreach node in figure 6.21.

6.6. Multiple Objects

The foreach loop from the VDB process becomes important when more than one object is used in the simulation. Figure 6.23 shows a stone temple made of multiple stone blocks. Figure 6.23a is the result of the VDB process without the foreach loop. The VDB process converts all of the geometry into a volume and, on conversion, combined all of the volumes into a single mesh. The result of Figure 6.23a was not a realistic representation of weathering on multiple surfaces, so the foreach loop was introduced. The result of the loop is shown in Figure 6.23b, which feels more organic and physical than the object in Figure 6.23a.

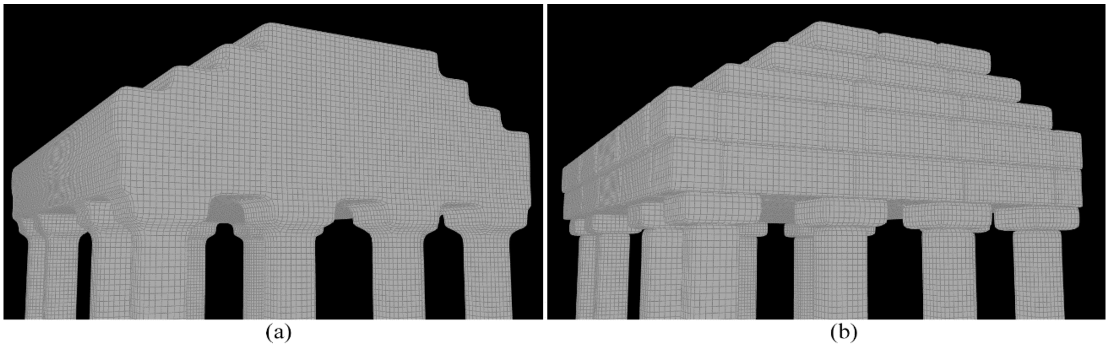


Figure 6.23: (a) Effect of the VDB process without separating each object. (b) Effect of the VDB process when separating each object.

6.7. Particle System

The node network in Figure 6.24 creates the parameters given to the particle system, making the particles readable by the system. It is broken down into:

- **Particle import** (Figure 6.24a)
- **Activate trigger** (Figure 6.24b)
- **Update attribute** (Figure 6.24c)
- **Give particle color** (Figure 6.24d)

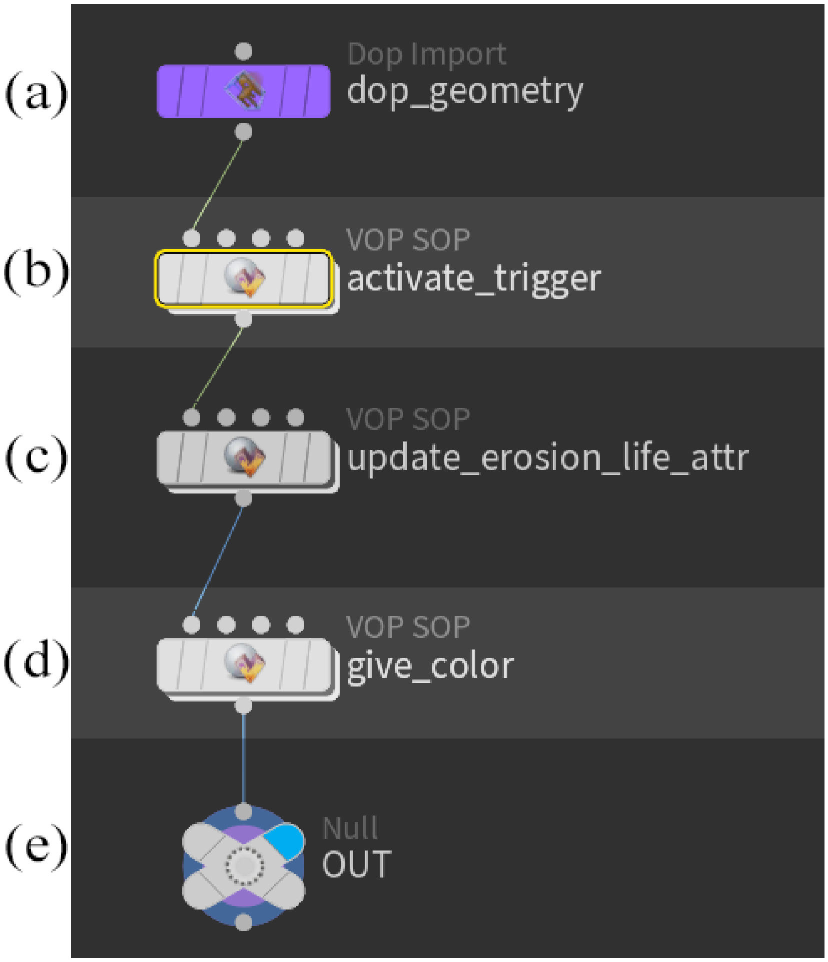


Figure 6.24: Particle system attributes node network.

6.7.1. Activate Trigger

The network shown in Figure 6.25 states that if the particle is colliding and the erosion_life attribute is inactivated, we activate the erosion_life attribute for the particle.

The pstate attribute acts as a toggle, initiating as 0 and increasing to 1 when the particle is colliding with a mesh. The erosion_life attribute was given to the particle on birth and initiates at -1. Erosion_life is used later in the process to increment the ramp values for the color of the particle.

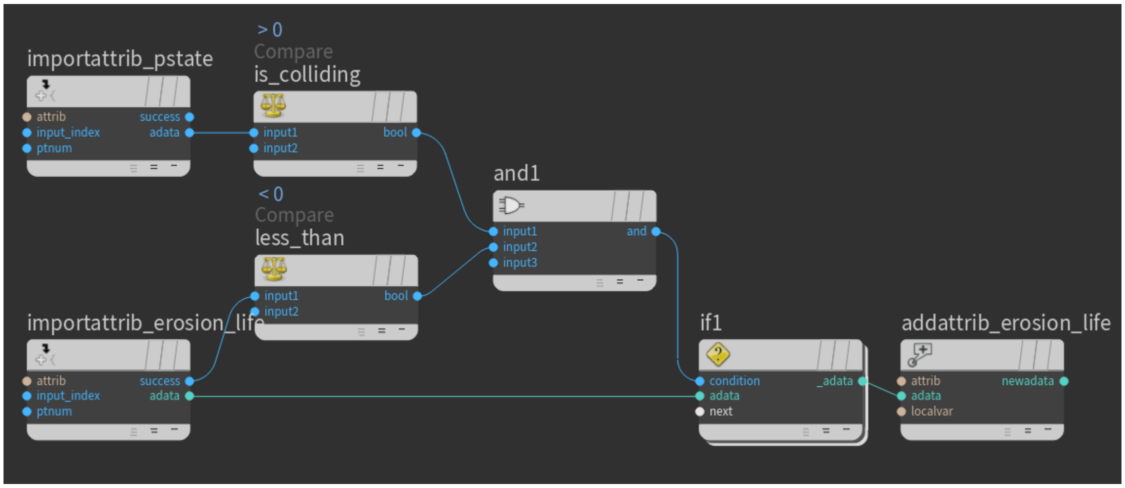


Figure 6.25: Activate trigger nodes.

6.7.2. Update Attribute

The nodes in Figure 6.26 state that if the erosion_life attribute was activated in the previous step, then we add the time increment to the attribute. When erosion_life is activated, it starts at a value of 0 that corresponds to the 0 value in the color ramp. Because we want the values to run through the color ramp, we add a value to erosion_life based on a time increment that starts when the attribute is activated. We prevent erosion_life from incrementing before colliding with a mesh by using the greater than or equal to 0 check in these nodes.

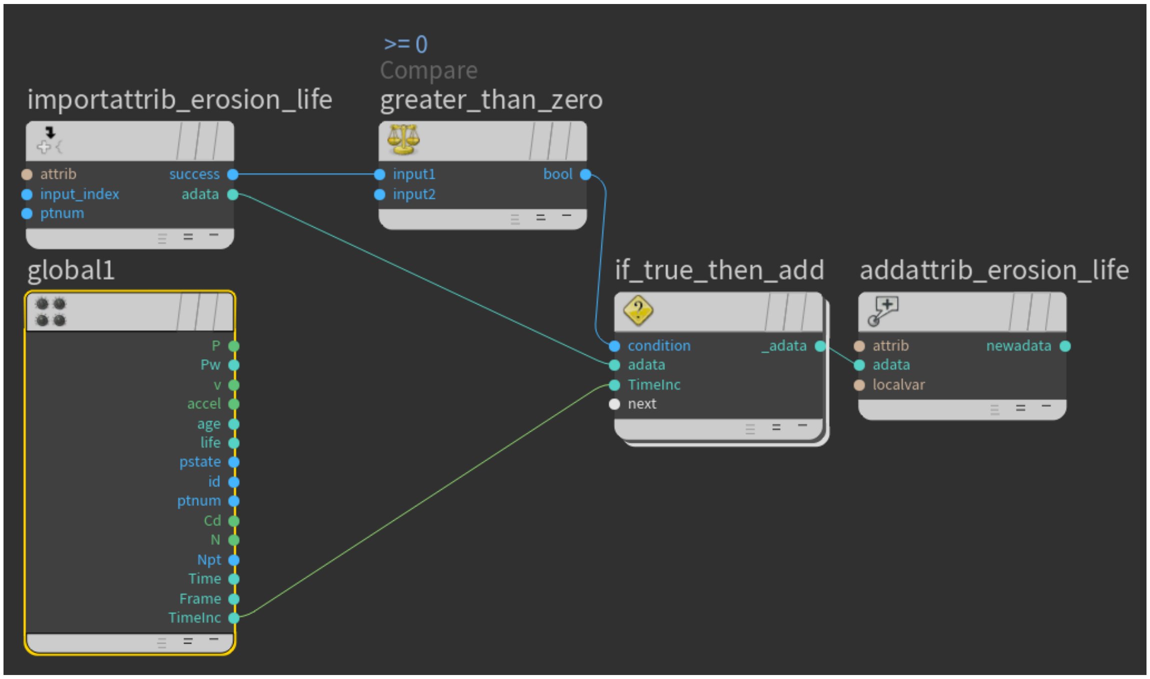


Figure 6.26: Update attribute nodes.

6.7.3. Give Color

The nodes in Figure 6.27 import the value from erosion_life, fit it between 0 and 1, and apply color of the particle. The network starts by importing the erosion_life attribute and runs it as the input value for the color ramp. The color ramp has a range from 0 to 1, so we fit the value between 0 and 1 before we input it into the ramp. To give the particles slight variations between each other, we use VEX code to generate a random float number and multiply it by the resulting ramp value. The final value is then output to the Cd attribute of the particle.

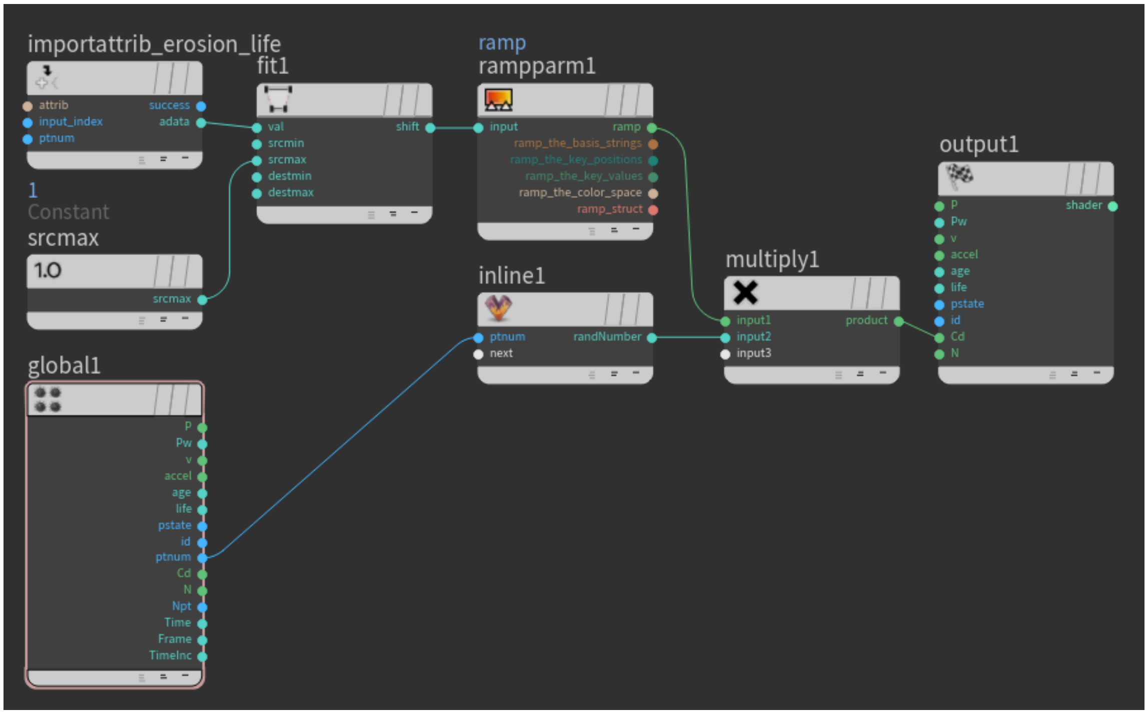
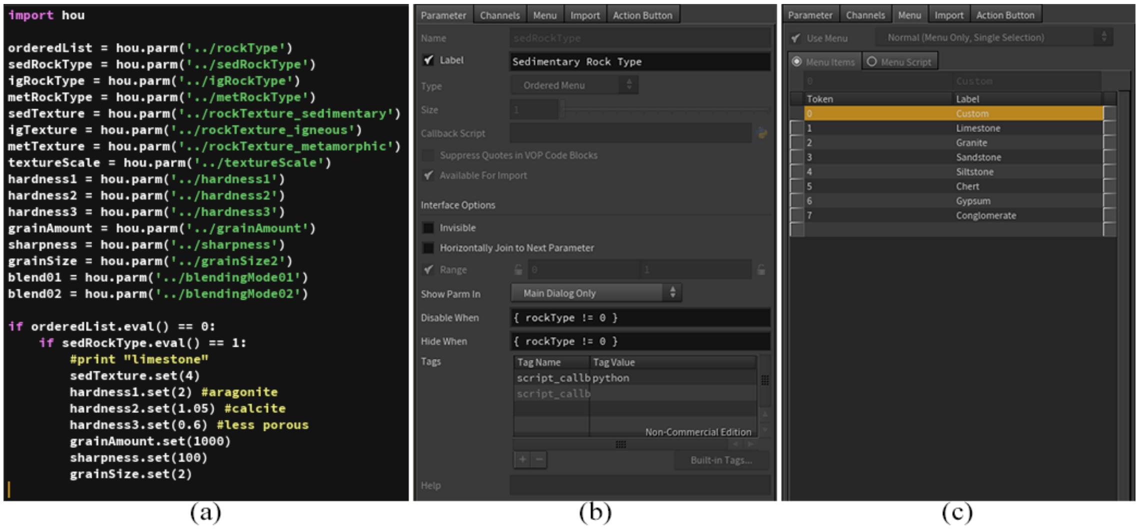


Figure 6.27: Give color nodes.

6.8. Menu Script

The Python script used for the menu sets the values based on the type of rock the user selects. The script in Figure 6.28a starts by defining variables that connect the UI parameters with the script. It then checks the Rock Type field in the UI for sedimentary (0), igneous (1), or metamorphic (2). After the check, it goes into the corresponding if statement and does a check for the Family Rock Type value. It then sets the texture type, hardness values, porosity, sharpness, and grain size that correspond to the type of rock. Figure 6.28b shows the parameters for the Sedimentary Rock Type menu. The menu is made contextual by setting the Disable When and Hide When fields to disable and hide whenever the rockType menu is not equal to 0. Figure 6.28c shows the ordered list menu used that assigns integers to the strings in the UI menu.



```
import hou

orderedList = hou.parm('../rockType')
sedRockType = hou.parm('../sedRockType')
igRockType = hou.parm('../igRockType')
metRockType = hou.parm('../metRockType')
sedTexture = hou.parm('../rockTexture_sedimentary')
igTexture = hou.parm('../rockTexture_igneous')
metTexture = hou.parm('../rockTexture_metamorphic')
textureScale = hou.parm('../textureScale')
hardness1 = hou.parm('../hardness1')
hardness2 = hou.parm('../hardness2')
hardness3 = hou.parm('../hardness3')
grainAmount = hou.parm('../grainAmount')
sharpness = hou.parm('../sharpness')
grainSize = hou.parm('../grainSize2')
blend01 = hou.parm('../blendingMode01')
blend02 = hou.parm('../blendingMode02')

if orderedList.eval() == 0:
    if sedRockType.eval() == 1:
        #print "limestone"
        sedTexture.set(4)
        hardness1.set(2) #aragonite
        hardness2.set(1.05) #calcite
        hardness3.set(0.6) #less porous
        grainAmount.set(1000)
        sharpness.set(100)
        grainSize.set(2)
```

(a)

Name: sedRockType
☑ Label: Sedimentary Rock Type
Type: Ordered Menu
Size: 1
Callback Script:
□ Suppress Quotes in VOP Code Blocks
☑ Available For Import

Interface Options

□ Invisible
□ Horizontally Join to Next Parameter
☑ Range: 0 1
Show Parm In: Main Dialog Only
Disable When: { rockType != 0 }
Hide When: { rockType != 0 }
Tags:

Tag Name	Tag Value
script_callb	python
script_callb	

(b)

☑ Use Menu: Normal (Menu Only, Single Selection)
● Menu Items ○ Menu Script

Token	Label
0	Custom
1	Limestone
2	Granite
3	Sandstone
4	Siltstone
5	Chert
6	Gypsum
7	Conglomerate

(c)

Figure 6.28: (a) Python code snippet used for the UI. (b) Parameters for the sedimentary rock type menu. (c) Ordered list menu for the sedimentary rock type menu.

7. RESULTS

7.1. Object Weathering

Figures 7.1 through 7.3 show 9 resulting rock types after 200 frames of simulation. The object was exported from Houdini into Maya 2015, textured with Mari, and rendered with Renderman 20.0. The test object was chosen for this demo to show a range of surfaces that the simulation was able to work on. The results of this test turned out well, there was enough variation in each object to differentiate one from the other. The hardness of each object is clearly defined by the level of erosion on each object. The decision to apply materials to the objects was to make the deformations of each object stand out more than only using a flat color. Overall, the results of this test were successful; the time it took each object to simulate was about 10 to 15 minutes, it was extremely easy to manipulate the texture from the UI, streamlining the process. Additionally, bringing the results into other programs from Houdini was a quick and easy process as well.

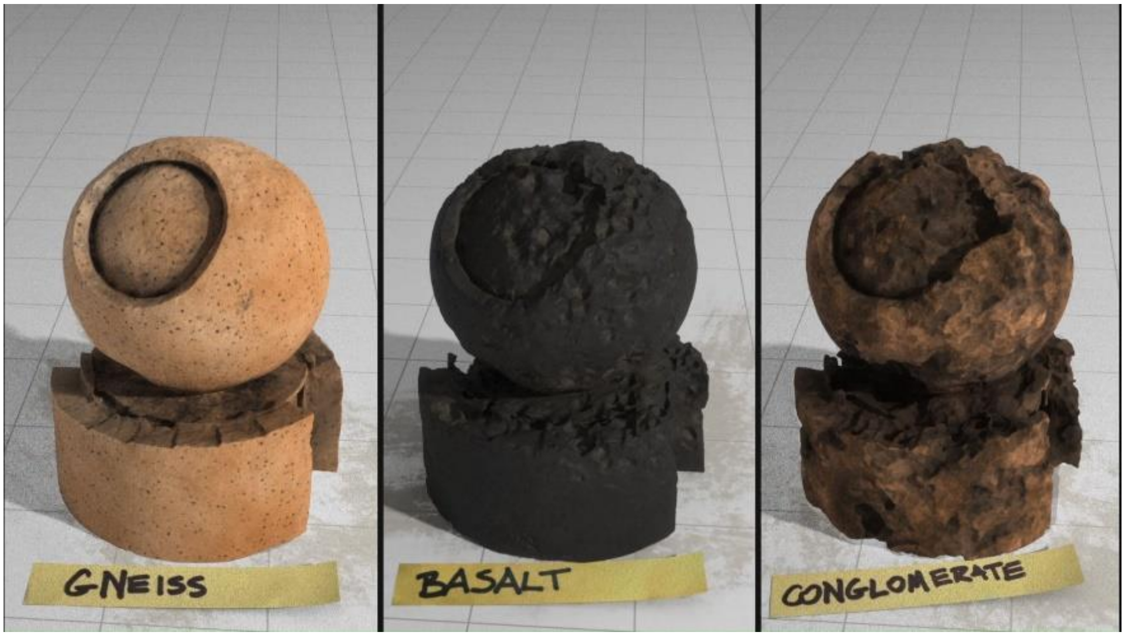


Figure 7.1: Results on the test object for gneiss, basalt, and conglomerate.

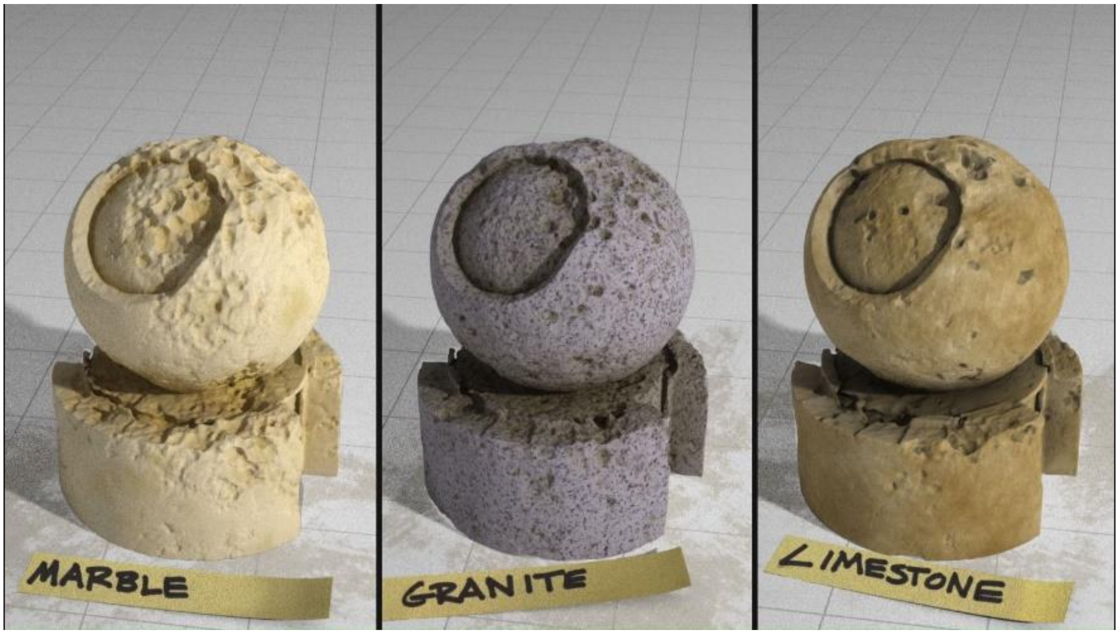


Figure 7.2: Results on the test object for marble, granite, and limestone.

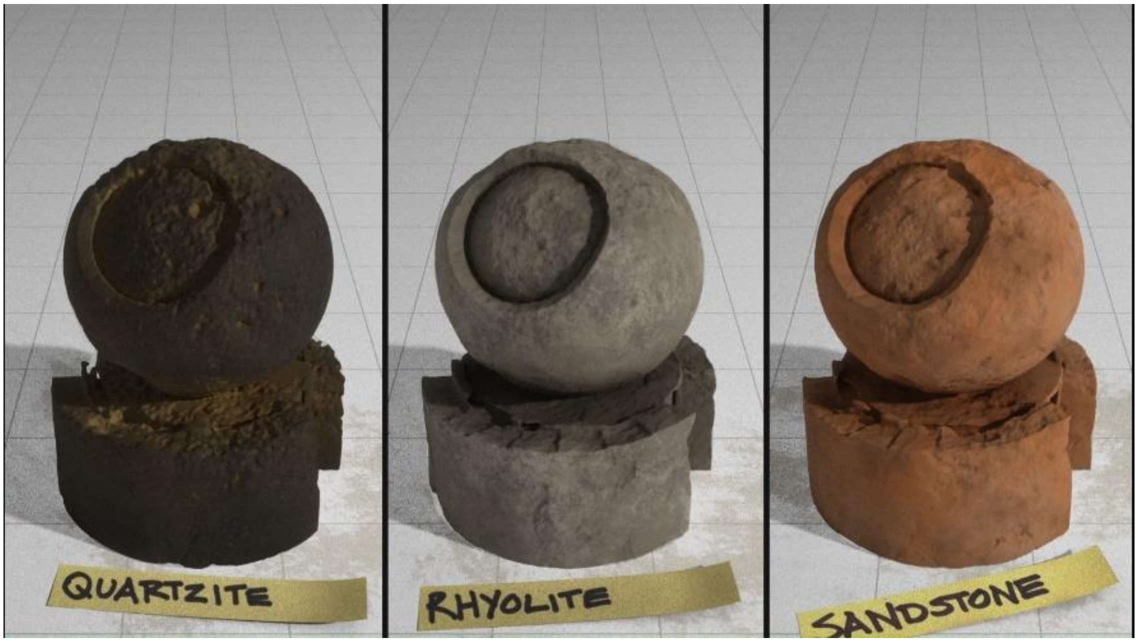


Figure 7.3: Results on the test object for quartzite, rhyolite, and sandstone.

Figure 7.4 and 7.5 show the use of the system on stone statues. Figure 7.4 was an attempt to match the reference of an Olmec Head. The base model was downloaded from Thingiverse [44] and was reconstructed to eliminate the wear in ZBrush. Overall, the results in Figure 7.4 were successful. However, many of the smaller pits are missing from the result, possibly due to the limitations from the mesh's density. The hardness values used for Figure 7.4 appear to be accurate as the overall shape is similar to the reference photo. Figure 7.5 shows the results of the marble stone settings on a downloaded statue of a marble player from Lincoln 3D Scans [25]. The results from this test felt natural, the form of the base statue made it easier to differentiate the areas that were eroded from the areas hidden from erosion. The arms and legs of the statue were also depicted accurately from the overall recession applied to them. The right arm's wrist also weathered down and split due to the width of the wrist as well as its exposure to the particle system.

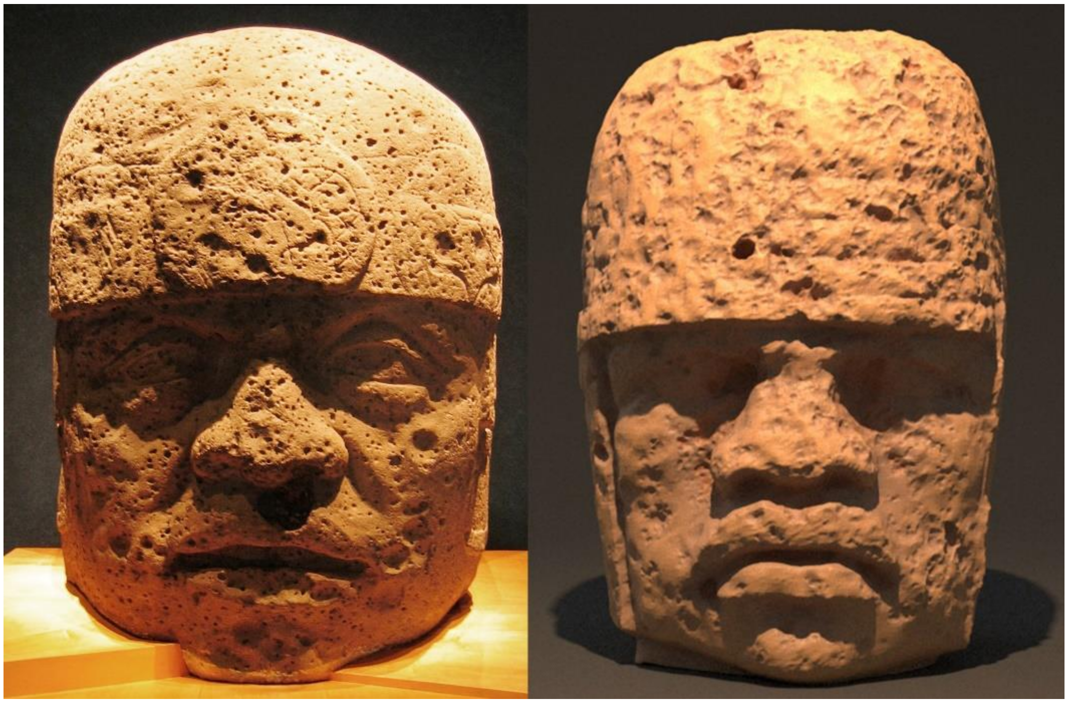


Figure 7.4: Comparison [6] of reference for Olmec head with Olmec head ran through this thesis' system.

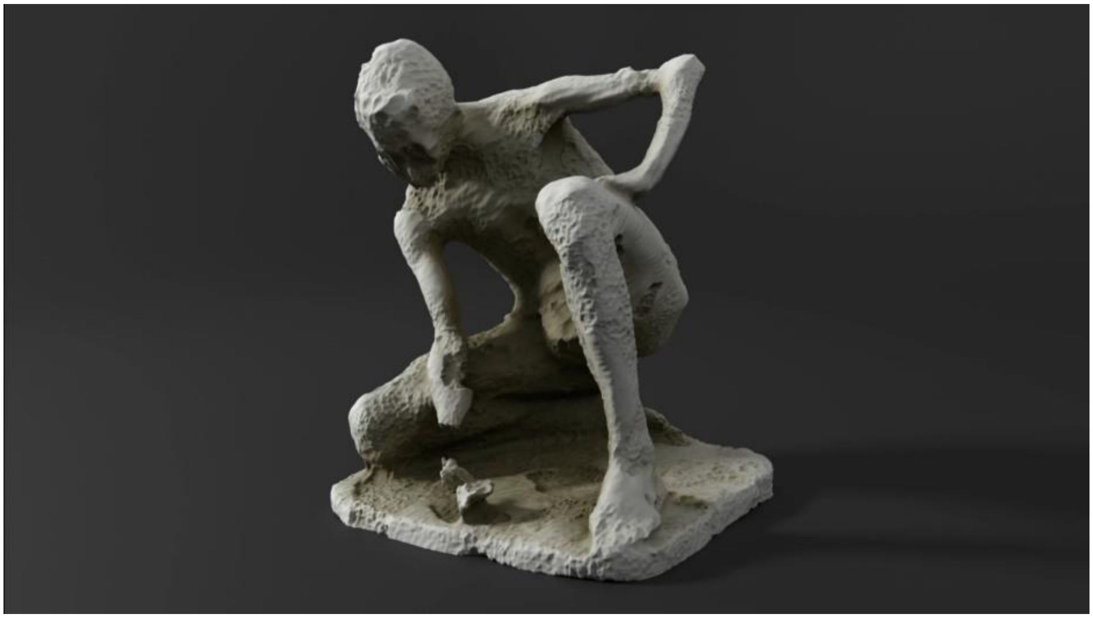


Figure 7.5: Results of marble parameters on a stone statue of a marble player.

Figure 7.7 shows the system applied to architecture. Primary tests of the Donon Temple were similar to the test shown in Figure 6.23a, with the individual bricks meshing together. Working with this model prompted the addition of the individual pieces workflow. The outcome of the weathered edges was desired and made a realistic depiction of individual blocks of stone weathering. The texture on the front facing stones for the temple appears too broad, which would be a matter of changing the texture's scale in the UI. The stones in the foreground are not at the fidelity they should be. The conversion from the VDB to polygons is apparent in the sharp edges of the foremost stone. However, naturally occurring stones were not a focus in this thesis due to the stone's unknown beginning stage of weathering.

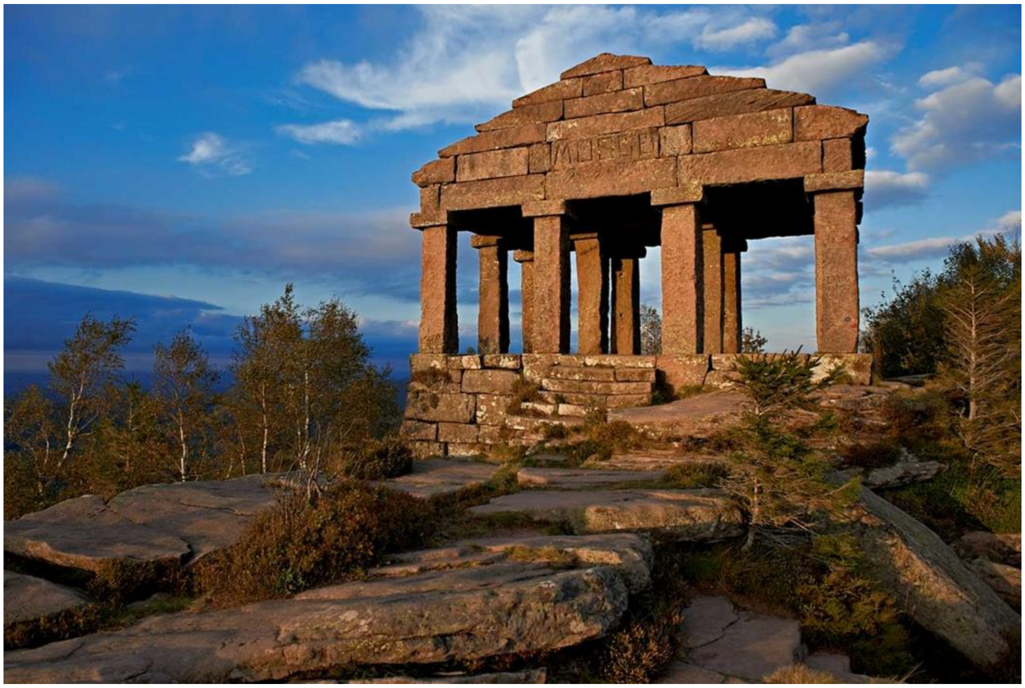


Figure 7.6: Reference for the Donon Temple. [41]

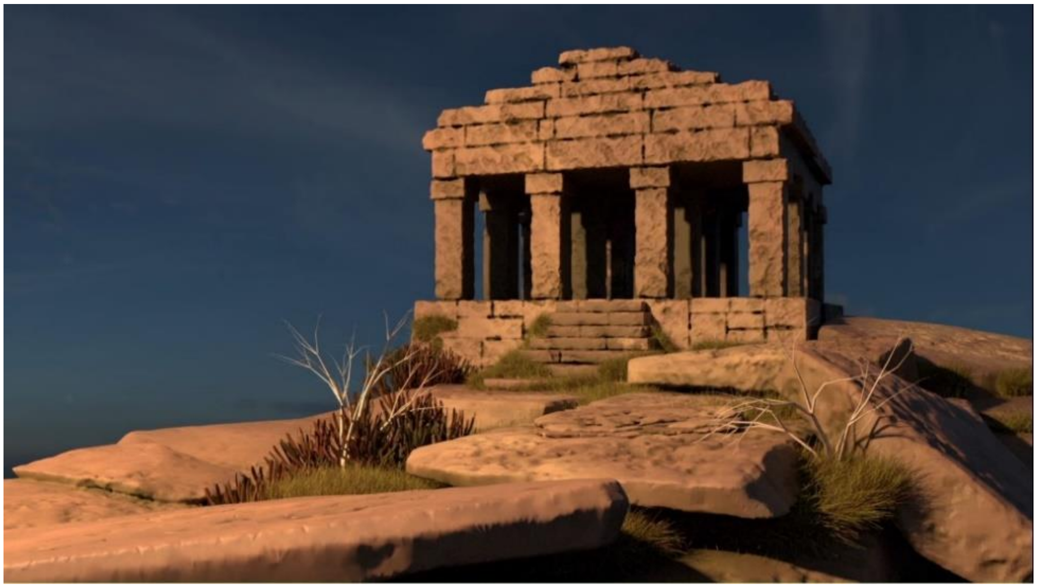


Figure 7.7: Results from using this thesis' system to match Figure 7.6.

7.2. Performance and Art Directability

The performance of the system proposed was quick to iterate on an object many times. Although the system could be optimized more, the optimizations and changes that were put in place drastically sped up the simulation times. The ties to mesh density are what keep the performance of the system from being ideal.

The number of options and the flexibility of the UI keep the system art directable. The ability to view the texture of the stone without running the simulation allows the user to make informed decisions without wasting time waiting on the simulations. The system was also used in personal work, Figures 7.8 and 7.9. The workflow was to decide on an object and find a pattern that I liked in the noise pattern, run the simulation, and export several options throughout the weathering process into Maya to view it with the lighting setup. Changes were made in the simulation after viewing it in the Maya scene. The output was not going to be a realistic formation of stone, so the texture was scaled and positioned to get the shapes wanted in the output model.

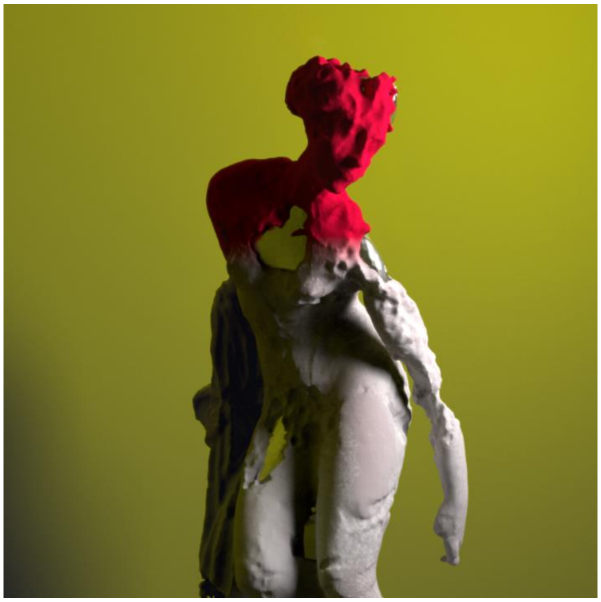


Figure 7.8: Art directed result of this thesis' system on a statue of a woman.

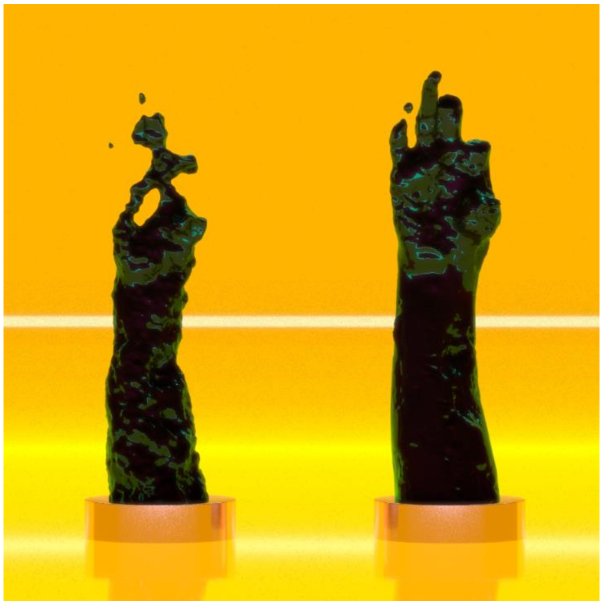


Figure 7.9: Art directed result of this thesis' system on a statue of an arm.

8. SUMMARY AND CONCLUSIONS

A new method for generating weathering patterns on stone statues through physically based systems was created in this thesis. By using this method, the user is quickly able to create weathering patterns that adhere to the real-world principles of the rock specified. The user is given a variety of parameters and values, gathered from this research, to define the type of rock. These parameters interact with the system realistically, but also allow the user to artistically determine and change the results.

This method was tested on a range of different objects including: stone statues, architecture, and small objects. The wide range of different test objects were to illustrate the capabilities and range of this method.

The method could be expanded on by including fractures in the stone. Fracturing is an important part of the weathering process for stone statues, breaking off or exposing interior elements of the stone. The addition of fracturing would make the system more realistic and could be used to predict the steps that would be needed to take to preserve an object.

The current version of this system is optimized, but steps could taken to optimize the it further. These optimizations would increase the performance of the system. By increasing the performance, it would be possible to use this system in a real-time game engine such as the Unreal Engine [11]. Adapting the system into a real-time environment would allow for instantaneous manipulations of the object and quick visualizations of the weathering process.

REFERENCES

[1] Allegorithmic (2017). *Substance Designer*. Retrieved from https://www.allegorithmic.com/products/substance-designer.

[2] Bishop, A. C., Hamilton, W. R., & Woolley, A. R. (2005). *Guide to minerals, rocks & fossils*. Firefly Books Limited.

[3] Boggs Jr, S. (2006). *Sedimentology and stratigraphy*. Pearson Education.

[4] Broz, M. E., Cook, R. F., & Whitney, D. L. (2006). Microhardness, toughness, and modulus of Mohs scale minerals. *American Mineralogist*, *91*(1), 135-142.

[5] Butcher, K., & Frey, M. (1994). Petrogenesis of metamorphic rocks. *Springer-Verlag*.

[6] Che, Xuan. (2006). *Olmec Colossal Head.* Retrieved from https://www.flickr.com/photos/rosemania/.

[7] Coe, A. L. (Ed.). (2010). *Geological field techniques*. John Wiley & Sons.

[8] Day, M. J. (1980). Rock hardness: field assessment and geomorphic importance. The Professional Geographer, 32(1), 72-81.

[9] Dorsey, J., Edelman, A., Jensen, H. W., Legakis, J., & Pedersen, H. K. (2006, July). Modeling and rendering of weathered stone. In *ACM SIGGRAPH 2006 Courses* (p. 4). ACM.

[10] Dorsey, J., Pedersen, H. K., & Hanrahan, P. (2005, July). Flow and changes in appearance. In *ACM SIGGRAPH 2005 Courses* (p. 3). ACM.

[11] Epic Games. (2017). *Unreal Engine.* Retrieved from https://www.unrealengine.com/

[12] Geology.com. (2005). *Granite*. Retrieved from http://geology.com/rocks/granite.shtml

[13] Gerberich, W. W., Ballarini, R., Hintsala, E. D., Mishra, M., Molinari, J. F., & Szlufarska, I. (2015). Toward Demystifying the Mohs Hardness Scale. *Journal of the American Ceramic Society*, *98*(9), 2681-2688.

[14] Ghysels, Marc. (2003). *Khmer Sandstone Statue.* Retrieved from https://www.scantix.com/case-studies/stone-statues/khmer-sandstone-statue/.

[15] Greeley, R., Marshall, J. R., White, B. R., Pollack, J. B., Marshall, J., & Krinsley, D. (1984). Abrasion by aeolian particles: Earth and Mars.

[16] Grippo, Alessandro. (2009). *Igneous Rocks.* Retrieved from homepage.smc.edu/grippo_alessandro/rocksigneous.html.

[17] Grippo, Alessandro. (2009). *Metamorphic Rocks.* Retrieved from homepage.smc.edu/grippo_alessandro/rocksmetamorphic.html.

[18] Grippo, Alessandro. (2011). *An Introduction to Sedimentary Rocks.* Retrieved from homepage.smc.edu/grippo_alessandro/rockssedimentary.html.

[19] Harada, M., Witkin, A., & Baraff, D. (1995, September). Interactive physically-based manipulation of discrete/continuous models. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques* (pp. 199-208). ACM.

[20] Krumbein, W. C., & Pettijohn, F. J. (1938). Manual of sedimentary petrography.

[21] Labus, M., & Bochen, J. (2012). Sandstone degradation: an experimental study of accelerated weathering. *Environmental Earth Sciences*, *67*(7), 2027-2042.

[22] Lagae, A., Lefebvre, S., Cook, R., DeRose, T., Drettakis, G., Ebert, D. S., ... & Zwicker, M. (2010, December). A survey of procedural noise functions. In *Computer Graphics Forum* (Vol. 29, No. 8, pp. 2579-2600). Blackwell Publishing Ltd.

[23] Lagae, A., Lefebvre, S., Drettakis, G., & Dutré, P. (2009, July). Procedural noise using sparse Gabor convolution. In *ACM Transactions on Graphics (TOG)* (Vol. 28, No. 3, p. 54). ACM.

[24] Lewis, J. P. (1989, July). Algorithms for solid noise synthesis. In *ACM SIGGRAPH Computer Graphics* (Vol. 23, No. 3, pp. 263-270). ACM.

[25] Lincoln 3D Scans. (2012). *Marble Player.* Retrieved from http://lincoln3dscans.co.uk/marble-player/

[26] Lynch, R. (2011). *The adobe photoshop layers book*: Taylor & Francis.

[27] McGreer, M. (2003). Weathering Testing Guidebook. *Chicago, USA: Atlas Electric Devices Company*.

[28] Meierding, T. C. (1993). Marble tombstone weathering and air pollution in North America. *Annals of the Association of American Geographers, 83*(4), 568-588.

[29] Mistrot, J. M. (2004). *Particle staining: physically based texture generation* (Doctoral dissertation, Texas A&M University).

[30] Mok, Carrie. (2016). *ZBrush Character Sculpting Tutorial.* Retrieved from https://www.3dartistonline.com/news/2016/12/zbrush-character-sculpting-tutorial/

[31] Museth, K. (2013). VDB: High-resolution sparse volumes with dynamic topology. *ACM Transactions on Graphics (TOG), 32*(3), 27.

[32] Nichols, G. (2009). *Sedimentology and stratigraphy*. John Wiley & Sons.

[33] Parneix, Paul. (2015). Houdini FLIP shark & sealion splash – Vray / 3DSMax render. Retrieved from https://vimeo.com/113552442.

[34] Parneix, Paul. (2015). Thinking Particles – Structure Demolition R&D (Rayfire, SC, Procedural Joints). Retrieved January 05, 2017, from https://vimeo.com/106340112.

[35] Pellant, C. (2002). *Smithsonian Handbooks Rocks and Minerals*. Dorling Kindersley.

[36] Pixologic. (2017). *ZBrush.* Retrieved from pixologic.com.

[37] Pope, G. A., Meierding, T. C., & Paradise, T. R. (2002). Geomorphology's role in the study of weathering of cultural stone. *Geomorphology, 47*(2), 211-225.

[38] Porck, H. J., & European Commission on Preservation and Access. (2000). *Rate of paper degradation: the predictive value of artificial aging tests* (p. 27). Amsterdam: European commission on preservation and access.

[39] Powers, M. C. (1953). A new roundness scale for sedimentary particles. *Journal of Sedimentary Research*, *23*(2).

[40] Shepard, C. U. (1857). *A treatise on Mineralogy*. CU Shepard.

[41] SideFX Software. (2017). *Houdini*. Retrieved from https://www.sidefx.com/

[42] Spach, Stéphane. (2014). *Le Donon, Montagne Sacrée.* Retrieved from http://www.valleedelabruche.fr/fr/decouvrir/visiter/sites-naturels-et-jardins/F215000338_le-donon-montagne-sacree-grandfontaine.html

[43] Taylor, E. W. (1949). Correlation of the Mohs's scale of hardness with the Vickers's hardness numbers. *Mineralogical Magazine*, *28*(206), 718-721.

[44] Thingiverse. (2016). *Olmec Head 3D Scan.* Retrieved from https://www.thingiverse.com/thing:1645908

[45] Walker, M. (2015). *Realistic Aging of Materials in Computer Graphics* (Doctoral dissertation, Texas A&M University).

[46] Witkin, A. (2001). Physically based modeling particle system dynamics. *ACM SIGGRAPH Course Notes*.

[47] Witkin, A. P., & Heckbert, P. S. (1994, July). Using particles to sample and control implicit surfaces. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques* (pp. 269-277). ACM.