5-2020

Simulating Destruction Effects in SideFX Houdini

Ethan B. Elkins

Simulating Destruction Effects in SideFX Houdini

By

Ethan Elkins

The Honors College

University Honors Program

East Tennessee State University

May 9, 2020

Ethan Elkins, Author

James Livingston, Faculty Mentor

Marty Fitzgerald, Faculty Reader

Greg Marlow, Faculty Reader

Abstract

As movies, television shows, and other forms of media have progressed over the last century, the use of destruction sequences as a form of entertainment have seemingly grown exponentially. From ginormous explosions to cities collapsing, more destruction sequences have drawn people's attention in ways that are quite captivating. However, as content producers continue to push the limit of what is possible, the reliance on practical effects starts to dwindle in comparison to the usage of computer generated scenes. This thesis acknowledges the trend and dissects the entire process of how a general destruction sequence is made, from the research and planning process to the actual simulation of the effects. Various methods are discussed in how to attempt the creation of destruction with a singular project in mind. The goal is to not only to complete the sequence, but to do so in an efficient manner that can rival a professional workflow.

Table of Contents

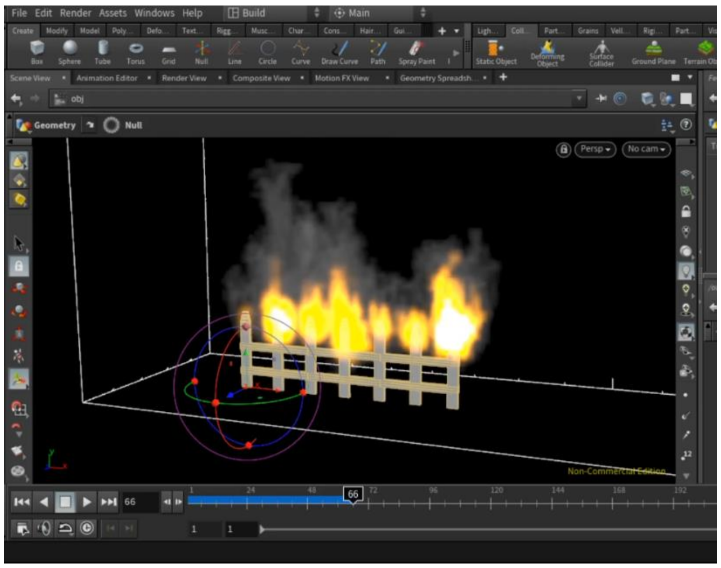
Introduction

There is no denying that humans seem to be naturally drawn to disasters and destruction. While such a statement would require scholarly research on its own right, this statement, from a general view of the entertainment industry, holds its ground. Destruction plays a huge part in our standard entertainment today; for example, many video games, such as *Call of Duty, Halo, Super Mario,* and more, use weapons' explosions, debris, or destructible environments to tell a story and immerse the player. Even social media and streaming platforms have used destruction as a form of entertainment, such as the YouTube-based channel Hydraulic Press Channel, whose videos of obliterating different objects with a pressurized hydraulic mechanism have gathered over 400 million views [1]. No better example of using destruction can be seen than within the film industry, where such examples are innumerable between movies and television shows. Even some of the highest grossing movies of all time, such as *Avengers: Endgame*, *Avatar*, and *Titanic*, have their climactic moments revolve around enormous and devastating destructions. Indeed, there is no mistaking that such scenes are enticing to humankind.

For someone who is interested in visual effects and simulated animation, learning how to form any kind of destruction shot would be ideal for stepping into the industry. This is especially true if said person, like most of mankind, has a certain awestruck wonder with grand explosions and toppling constructions. However, visualizing a destruction shot and creating a computer-generated destruction sequence are two different processes, and the latter is the more complicated process that must be carefully researched, practiced, and understood in order to fully carry through to a finalized scene. Therefore, analyzing each individual step and practice will allow us the opportunity to answer the paramount question: Using SideFX Houdini, what are the necessary steps in building a complete destruction sequence?

Why Use Houdini?

Of course, before this major question can be answered, one must realize why this thesis chooses to utilize Houdini as the tool for solving its problems. While other 3-dimensioinal software have proven effective with special effects, such as Cinema 4D and Maya, Houdini is not only geared towards such effects, but also boasts a procedural, node-based workflow. This visual and literal procedural method allows changes to be made at any time in any area of the pipeline, with little to no consequences. This also allows driven parameters for easy customization after a model or simulation has been created. This type of control, mixed with the power of the software to create reliable, beautiful simulations, makes Houdini the best personal preference



when tackling effects such as destruction simulation. Therefore, Houdini will be the basis on which the methods, process, and future research and development will stand.

Process Breakdown

To say that there is an identical process for every possible destruction sequence would be a faulty statement. The term “destruction sequence” can cover a wide variety of different simulations, from buildings and structures, to earthquakes and explosions. Each scene requires its own variation of steps in order to achieve the desired effect; this thesis focuses on its own goal of destroying a wall comprised of concrete, a wooden window frame, and glass windowpane. However, while the specifics of each simulation will vary drastically, there is a general backbone that each process will follow. This backbone is important because it guides the simulation process in a necessary flow for the most efficiency and success. The general process can be broken down into four steps:

1. Researching the type of materials to destroy
2. Fracturing the geometry using the most efficient method
3. Creating realistic debris through particle systems
4. Creating a pyro simulation for smoke

These four steps are used in most destruction sequences and can help set a concise plan before attempting to destroy any type of mesh. Such a plan sets the difference between completing a scene within a week and struggling for months in a trial-and-error process. Nevertheless, this breakdown still lacks some of the key steps in order to fully complete a destruction scene. Some of the most obvious steps not mentioned include lighting, texturing, and rendering; these steps may be saved for last but can also be intermingled throughout the process. For instance, for the thesis project, texturing was chosen to be done at every level due to the caching process chosen for the project. In addition, both texturing and lighting were combined during the smoke portion of the plan in order to create a realistic simulation and render. Such deviations will be specified and explored later in the thesis. Nevertheless, this thesis will use the four-step plan to dissect the destruction sequence project.

Research

Before any kind of fracturing or simulating has commenced, one must understand what they are wanting to destroy. Breaking a rock in half will appear quite different from splitting a log. On a more complicated level, a wooden house collapsing will involve smaller variations of debris and smoke than a gas explosion in a steel skyscraper. Knowing the type of material as well as the scale will save time during the actual simulating phase, while also creating a convincingly realistic shot. Fortunately, there are a few areas that roughly generalize the types of materials that one would want to destroy, allowing one to follow a base method before digging into specifics. These fields include:

- Basic Fracture (Rock, Cement, etc.)
- Splintered Fracture (Wood)
- Shattered Fracture (Glass)

Firstly, a Basic Fracture is the most common fracture with the easiest base method; often the default fracture for most Houdini nodes creates a rock-like break. A Basic Fracture incorporates all types of hard objects whose fractures create jagged edges along various cracks. While most rock-type fractures look similar, understanding the type of Basic Fracture helps create the specific material desired. Understanding the type of cut and grain within materials such as granite, marble, and brick will drive the customization of one's fracture. On the other hand, Splintered Fractures primarily consist of wood, as these fractures tend to split along the grain in long, pointed edges. While types of wood will not greatly vary in how a Splintered Fracture should be altered, the values that contribute to the widest changes are size, thickness, and length. Pencils and sticks will have a smaller splintering effect than boards of cut wood, while trees and logs may have a greater and more random splintering along the cut. Shattered Fractures hardly vary much outside of glass or glass-like objects (such as hard candy or vases) but have the most unique fracturing pattern out of the three fields. Shattering fractures have an almost spiderweb-shaped break, with the smaller pieces generating at the contact point; this also occurs with thin-walled materials. Shattering is one of the most difficult fractures to do manually inside of Houdini; however, due to its unique pattern, shattering fractures may arguably need the least amount of variation within its own group. While it would be naïve to think that there are no materials that may blend between these fracturing groups, their distinctions are useful in identifying the best course of action moving forward.

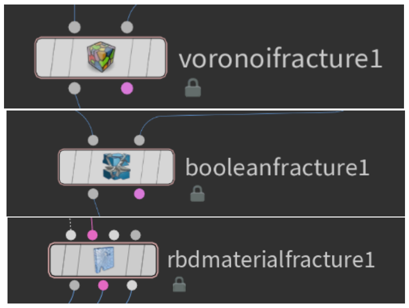
Fracturing

Once the materials are not only identified, but studied in how they act physically upon contact, the next step in completing a destruction sequence is to build and fracture the objects. This is the most important step in the entire process, as it is the focus of the sequence and drives how the entire simulation, including the debris and smoke, works. This is also where the research completed in the first step is put directly into action. Of course, before the fracturing can even begin, models must be created or found. While building or finding the geometry, it is important to do both in accordance with scale. Since "forces in Houdini work in real-world units," with the default scale set to meters, having objects built and destroyed in such units help "create a realistic and reliable simulation" [2]. In the instance of the thesis project, accurate measurements were researched for the average size of all the materials created, such as the wall, window frame, window jamb, and glass pane. Not only does this pre-step allow for a realistic simulation but it also saves time in the future by allowing the user to choose the smoke and debris shelf tools, which automatically calculate their parameters based off of scale. While simulation times may be slightly slower due to more data and polygons, the efficiency in the workflow is a better compensation.

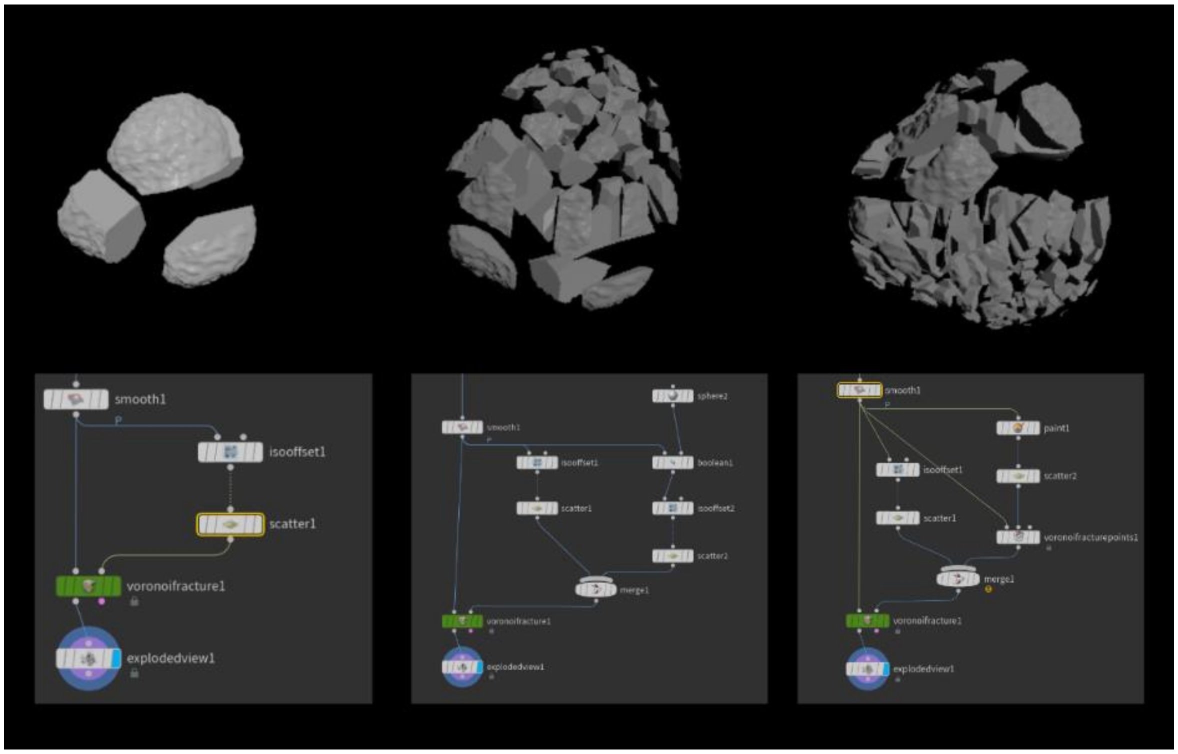
With the models in place, the fracturing step can finally begin. Fracturing the objects into controlled yet random pieces can be achieved in a few different ways, and each depend on the type of destruction sequence which needs to take place. Three different nodes are generally used to slice apart objects:

- Voronoi Fracture
- Boolean Fracture
- RBD Material Fracture

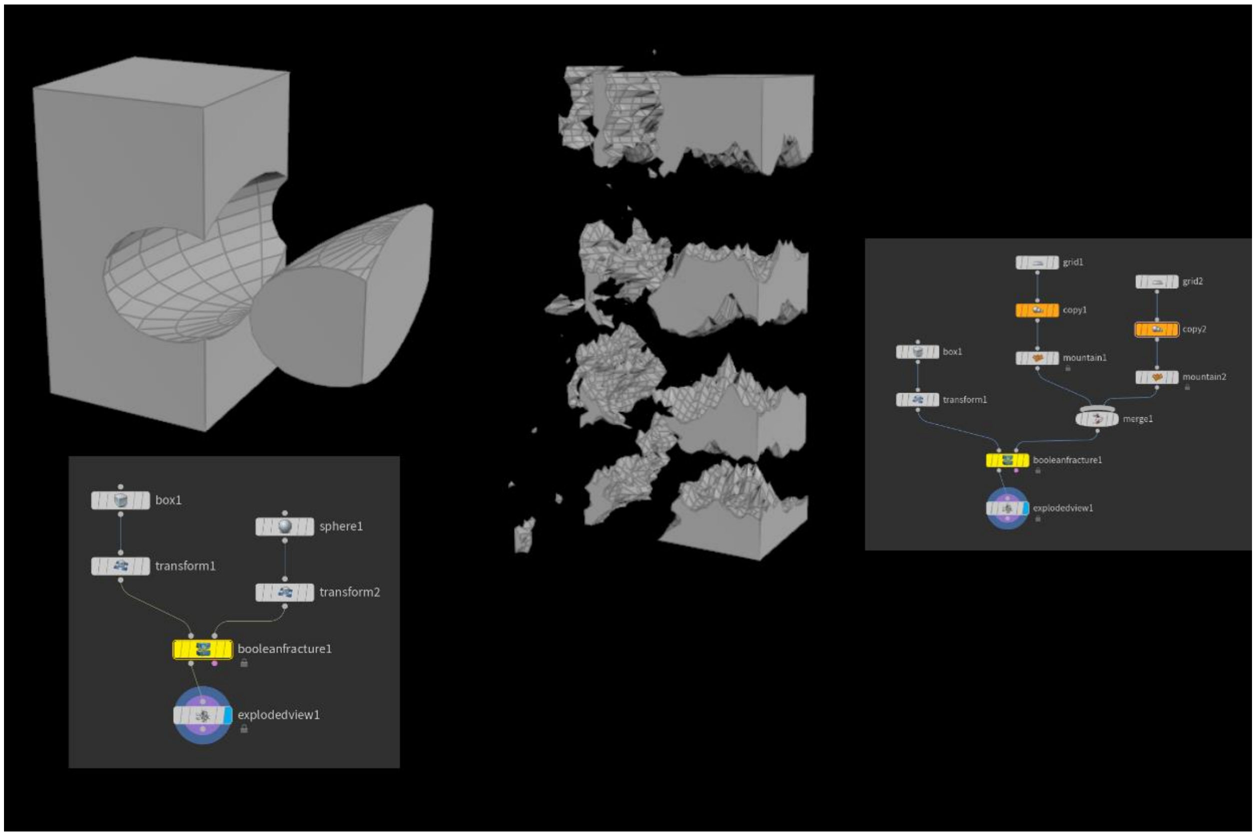
The major difference between these three methods lies in how they fracture objects and their different user interfaces. Yet despite these differences, the ways in which a visual effects artist can customize their fractures to get a more realistic simulation stays relatively the same among all three nodes; this is due to the fact that these methods only focus on the calculation of the fracturing and not the artistic direction. All three methods are quite effective, meaning that choosing between them should be based on the most efficient and accurate way of fracturing the desired materials.



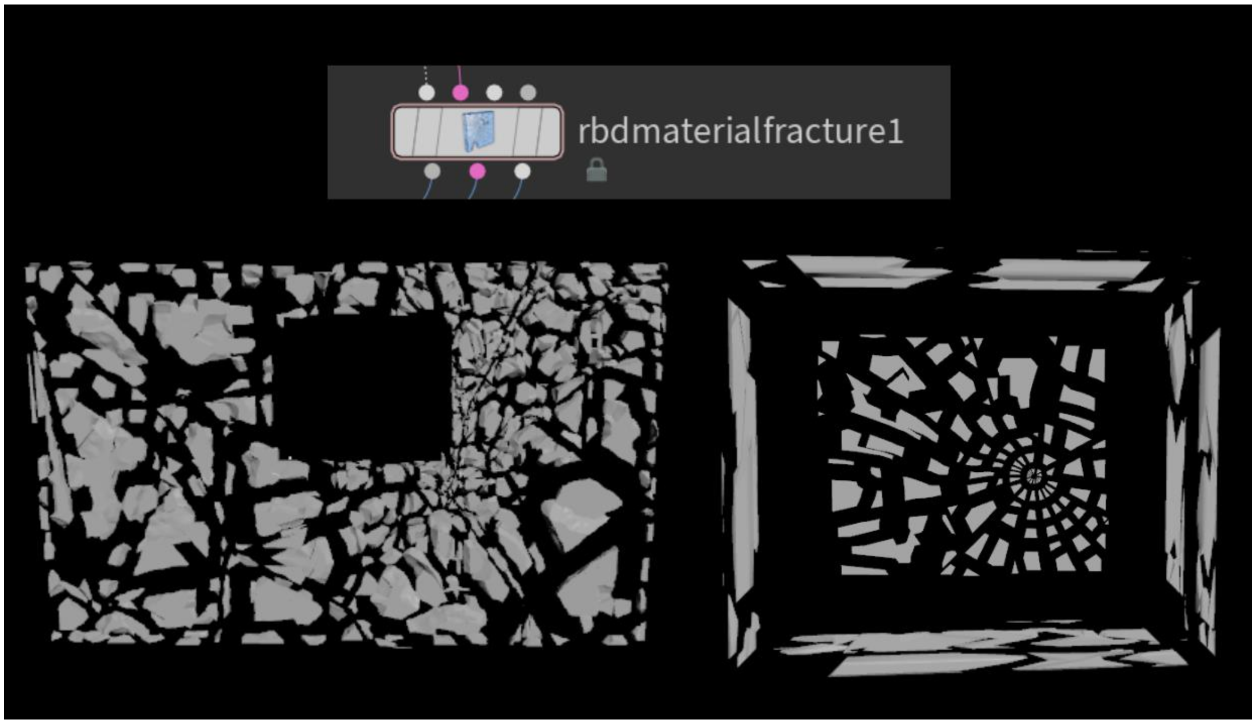
Voronoi Fracturing, for instance, is a great way to get a quick fracture for any type of Basic Fracture. The node uses points scattered along the geometry to calculate the edges between each point, effectively slicing up the model. By turning the model into a volume and scattering the points throughout, as seen in the picture below, one can effectively create a rock-based fracture in a matter of minutes. Utilizing points also offers numerous ways to customize where they scatter and create edges, allowing for artistic decisions such as contact points, randomized sizes, and chipping. While the first example in the figure below demonstrates a quick fracture, the second and third examples put those creative decisions to work. The second rock uses an extra sphere with points scattered about it, allowing for the combining of points between the original model and the smaller sphere to create diversity among the size of the pieces. The third rock uses a similar process but focuses on color to decide where the points cluster rather than another geometry. The third customization also uses a node called Voronoi Fracture Points, which gives the user more control over how the new cluster spreads its points in the selected area [3]. Such customization in the two latter methods can also be used in the other fracturing nodes to varying degrees; however, in terms of the Voronoi Fracture node itself, it is perhaps the most flexible fracturing method.



If Voronoi Fracturing is perfect for use on a general fracture, then Boolean Fracturing would be its counterpart. Boolean Fracturing is geared more towards creating specific edges and piece shapes through means of other geometry. Instead of using geometry to create points, it instead uses the standard boolean function found in most 3-dimensional software to create pieces in the shape of the intersecting geometry. The figure below demonstrates two different approaches of using this node, with the first box using a deformed sphere to cut out a shape, and the second box using a series of planes with noise to slice apart the mesh. In fact, the second method is a great way to divide geometry into grain-like pieces for a Splintered Fracture. Still, no matter the geometry used to cut into the original mesh, the usefulness of the Boolean Fracture lies in a procedural pipeline. By allowing this node to set the base shape of the major pieces, an effects animator can further fracture each piece using any of the three fracturing nodes. This would allow the Boolean Fracture node to be the ultimate tool for shaping pieces for further breaking, giving the user complete control over the entire fracturing process.



Finally, the RBD (Rigid Body Destruction) Material Fracturing method allows for versatility when dealing with multiple types of materials, while also combining some of the methods and tools from the previous two nodes. Built into the RBD Material Fracture node is a user interface that allows the artist to select the type of material they are wanting to fracture (concrete, wood, and glass). Based on the selection, the node will pre-fracture the geometry and set up parameters for customization. Not only are the controls extensive within the node, but for the concrete and glass setting, further customization can come from outside of the node via points; this opens the door for using previously discussed methods such as the Voronoi Fracture Points clustering. Furthermore, the node itself allows for connectivity between other RBD Material Fracture nodes and RBD Glue Constraint nodes. An entire fracturing process can be done within this pipeline, and while customization may not be as unlimited as the Voronoi and Boolean processes, it is an ideal method when working with multiple materials and meshes. For the sake of this thesis project, the RBD Material Fracture node was used in order to fracture the wall, window frame, and glass pane. Though all three materials were used by the same node, the Voronoi Fracture Points method was used to customize the clustering of smaller pieces near the collision point of the wall fracture.



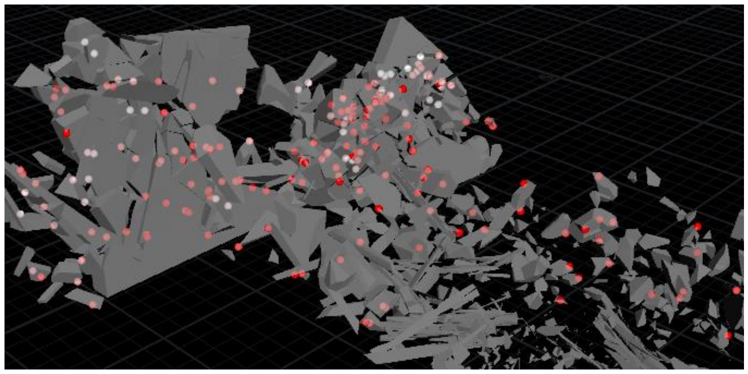
Before the fracturing stage of the destruction sequence can be completed, there are a couple loose ends; in this instance, the texturing and glue constraints need to be set in place. In texturing, the process is slightly more flexible due to preference in playback. The most important part is setting up individual UV's for each complete mesh before the initial fracture, and then setting up UV's for the inside geometry in the pieces. This allows for most cracks to be camouflaged with textures in the final project while allowing a more procedural workflow. If one chooses to texture now, the process may be done after the actual simulation is calculated. In doing this now, the user has the option of caching out a low-resolution version with no textures, and a high-resolution version with textures. Both allow for consistent and reliable fracturing in the simulation so that other simulations that rely on its destruction and collision, such as the smoke and debris, will always have the same results. In having two different resolutions cached, an effects animator can have optimization in switching between them for simulating or final lighting purposes.

Glue constraints, on the other hand, are a necessity in guiding how the pieces will come apart in the actual simulation. Setting up glue constraints within Houdini hold more variations and possibilities than the fracturing process, and an extensive look into their creation could be an entirely separate thesis. However, much like the fracturing process, there are a few different glue constraint options that are the most utilized, and the one that was used within the thesis project belongs to the RBD Material family. RBD Constraint Properties directly link up the same way that the RBD Material Fracture nodes do and can work straight from the fracture nodes. With a user interface that helps distinguish between regular glue, soft body, and hard body constraints, this method proves to be both efficient and effective for this project. Once the glue constraints are set, the simulation can finally be calculated.

Creating Debris

With the fracturing process complete, the next step in creating a destruction sequence requires setting up a particle simulation for debris. Debris simulation holds a consistent set-up for any destruction shot, with creative decisions only influencing debris count, type of geometry stamped, and the simulation physics. Indeed, creating debris falls more into a technical aspect; debris relies on particle physics, and default particles only carry attributes revolving around their location in the 3-dimensional space and their velocity. In order to achieve randomness in size and geometry, as well as accurate rotation, the user must add attributes through node-based and VEX (Vector Expression) coding. Fortunately, the process for both is what stays the same throughout all debris simulations, so knowing how to do it once means that any effects artist can copy the process for other destruction sequences. In the instance of the thesis project, three different debris simulations were formed, but most of the coding and node networks were copied and pasted into each simulation with successful results.

Before particles can be created for the debris, the source must be created. This is perhaps the easiest part of the simulation, as there is a single node aptly named Debris Source. By using the previous fracture, the node creates source particles along the cracks when the initial break happens. Through the user interface, an effects artist can determine how many source particles are formed and how long they are active; this will determine how much debris is formed. Once created, the source particles are applied and run through a debris simulation that creates the particles. All that is needed are the previous collisions used for the fracture, ground plane, contact object, and any other object that the particles may hit. With the collisions connected, a basic debris simulation has been formed. What lacks is the realism that completes this part of the sequence: geometry to replace the point particles and accurate rotation of the debris. Both of these key attributes can be done in either order, as the main simulation has already been completed, and these actions either add onto the calculations or mask over what has already been done.



Since setting up accurate rotation happens in the same area where the physics are set up for the particles, it is easier to transition into that explanation. In order to calculate each particle's rotation, one must first build a particle's orientation. Three POP Wrangle nodes are connected to the particle solver for the simulation, each with its own set of VEX code. The first wrangle node uses the code to set the particle scale and find the distance traveled from the source particle; the second node uses the found distance value to determine the velocity's direction and assigning that value as the particle's individual X-Axis; the last node uses the X-Axis to find the other two axes and rotates the particle along its own Z-Axis [4]. With the orientation now built, these three newly coded nodes now calculate each particle's rotation based on its velocity and where it formed. Since the formulas can work for any scenario involving particle-based debris, the nodes for this thesis project copied onto the other two particle simulations.

Despite having the correct physics and rotation of realistic debris, the simulation is still comprised of particles. Thus, geometry must replace all of the points to complete this stage of the destruction process. This is the stage of debris simulation that requires the most amount of efficiency, as calculations and polygon counts tend to skyrocket in this process. In preparation for copying geometry to the particles, the source geometry must be gathered. This can be a low-resolution copy of the original fractured mesh or newly created simple fractures. Most importantly, the debris must match the type of material used by the fractured geometry. Using that same fractured geometry allows for random pieces to be copied, which in turn varies the types and sizes of the debris. However, when using fractured geometry, the user must set each piece to the world origin. Having each piece centered at the origin allows them to be completed centered on each particle, eliminating drastically floating debris. Setting each piece to origin can easily be done with a For-Each Piece Loop, which takes a set of transformations and individually applies them to each piece.

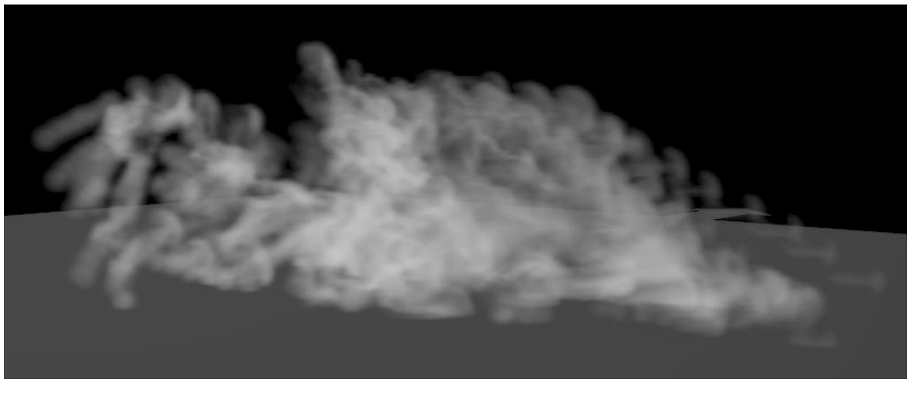
Once the geometry is prepared the copying can begin. There are a few different ways of assigning random geometry to points, with the most commonly known method involving the Copy-Stamp node. Unfortunately, this method recalculates the copying process for every frame, and with more complicated fractures, it has the possibility of drastically slowing down or crashing Houdini. Instead of the Copy-Stamp node, this thesis utilizes the Copy-to-Point node for better efficiency and control [5]. In order to achieve random assignment for the geometry, the Copy-to-Point node is encased in a For-Each Point Loop. By using the metadata for each individual point, an effects artist can receive a unique value for each point based on increment. This unique value can then be used to generate a random value within a fixed range that can correlate to the piece number of the geometry. Once the random expression is put inside the Copy-to-Points node, a random piece is selected for each point and is replaced. With the geometry now copied, the debris simulation is ready to be cached and textured for the final render.

Creating Smoke

In terms of rigid body destruction, the sequence is now officially complete. Nevertheless, the goal of this thesis is to create a full destruction sequence, and the creation of smoke may be needed. Often times, whether when dealing with fiery explosions or large breaking rock-type materials, smoke is necessary to finish the realism needed for the shot. As for the thesis project, one smoke simulation is used for the concrete wall. Crafting smoke is quite different from creating debris; where debris simulation's difficulty lies in technical work, smoke simulation's difficulty involves creative decisions. Shaping, lighting, and texturing smoke is where the vast majority of the work lies. If performed incorrectly, the entire scene can be ruined.

Just like the setup for the debris, the initial formation of smoke is quite simple. The Debris Source node becomes useful again as the easiest way to form source points. In fact, the node used for the actual debris can be copied if the birth rate and source point count match what the user wants, but in the case of the thesis project, a separate one was created to pull back the source count. From there, attributes need to be added to the particles and turned into rasterized volume sources for smoke to be simulated. While this can be done manually, the shelf tools in Houdini can work as a suitable starting point as long as the models and other simulations are done to scale. Regardless, once the volume source is created, it is ready to go through the actual simulation process.

Within the actual simulation node is where most, if not all, of the shaping will take place. When dealing with destruction, smoke is usually formed as part of the dirt and debris that gets caught in the air. In other words, the smoke in Houdini needs to be thick with a long dissipation process. This can be accomplished within the actual Pyro Solver node, which guides how the smoke is simulated. The parameters which control how the smoke forms are dissipation, disturbance, and turbulence; essentially, dissipation controls how fast the smoke fades, disturbance adds small scale noise, and turbulence determines the big curls and formations. In order to get realistic results, fine-tuning is needed for both the individual values and the fall-off ramps. While such controls may seem simple, getting the exact desired shape may be tedious depending on simulation times. Other than shape, a couple more factors need attention within the simulation node. Wind, for example, can help direct the smoke and add a final layer of noise to the overall shape, and the voxel size parameter needs accurate balance between smoke quality and comprisable render times. The more time spent at this stage finding the perfect values for a smoke simulation, the better result of the render and final sequence. Of course, the smoke cannot be complete without the proper texturing and lighting. Both of these steps influence the smoke's appearance in brightness and density, while also creating internal shadows for denser smoke simulations. These last steps are the difference between wispy vapor and debris-like smoke.



Conclusion

With every process cached, textured, and lit, the rendering process can begin. Understanding how to render efficiently not only saves an artist time, but allows for better workflow on a professional standard. Finding the balance between quality and speed lies in sampling within the Mantra render and the lighting system, as well as voxel count for the smoke simulation. For the thesis project, the final sequence took several hours to render out overnight, which was ideal for the desired timeframe and quality. The pictures below show scenes from the final destruction sequence. While this sequence may be simplistic in nature, the methods that were developed and utilized can be applied on a grander scale. The project lays the groundwork for all destruction sequences, and since it contains multiple and diverse simulations, it also creates a broader understanding of how effects animation works inside of Houdini. Ultimately, the thesis research applied to the sequence made it a success.

It is important to remember that these steps focus on the general process of creating a destruction sequence and does not incorporate every altercation and detail possible. With new techniques and Houdini software updates created every year, there is ample reason to continue researching in order to fully understand every aspect of destruction. This includes working with large scale destruction pieces, to unique materials such as cloth and metal. There is even a fourth method of fracturing that was found within the past year that makes larger simulations more efficient by replacing most of the particle work with smaller geometry. This method known as RBD Smart Activation takes any type of previous fracturing methods and further fractures the edges of the existing pieces; by stitching these smaller pieces together along the initial fracture seams, they create a more dynamic simulation that resembles debris without creating thousands of particles [6]. This method proves that a base groundwork for destruction simulation does not always cover every specific scenario.

Regardless of the need for further research, the general process for a complete destruction sequence still holds the same flow of work that is required for every scenario. With this knowledge, any amount of extra research would only add a fraction of time to the overall process, allowing for the main goal of efficiency to still stay intact. Therefore, this project succeeds in its function as a foundation for all modern-day, computer-generated destruction.

References

[1] Hydraulic Press Channel, "About: Hydraulic Press Channel," YouTube, 6 October 2015. [Online]. Available: https://www.youtube.com/channel/UCcMDMoNu66_1Hwi5-MeiQgw/about. [Accessed 28 April 2020].

[2] P. Rutkowski, "Destruction Simulation: Tips and Tricks," 80LV, 21 June 2017. [Online]. Available: https://80.lv/articles/destruction-simulation-tips-and-tricks/.

[3] Fifty50, Director, *ART DIRECTING YOUR DESTRUCTION IN HOUDINI // SERIES #01 GEOMETRY.* [Film]. United Kingdom: Vimeo, 2017. Available: https://vimeo.com/198266808

[4] P. Rutowski, "Rolling Debris with VEX Tutorial," 29 June 2018. [Online]. Available: https://www.pawelrutkowski.com/rolling-debris-tutorial.

[5] H. 123, Director, *Copy Stamp vs For Loops (and random rotation).* [Film]. United Kingdom: YouTube, 2018. Available: https://www.youtube.com/watch?v=5ja8yyj6_SQ

[6] A. Nardini, Director, *Houdini Masterclass: RBD Smart Activation.* [Film]. Italy: VFX HIVE, 2020. Available: https://www.youtube.com/watch?v=s9DjkDt-6Js&t=1273s

[7] J. Cameron, Director, *Titanic.* [Film]. United States of America: Twentieth Century Fox, 1997.

[8] A. Russo and J. Russo, Directors, *Avengers: Endgame.* [Film]. United States of America: Marvel Studios, 2019.

[9] J. Cameron, Director, *Avatar.* [Film]. United States of America: Twentieth Century Fox, 2009.

[10] H. Tutorials, Director, *Houdini Font Crumble with Smoke and Debris Shards.* [Film]. United States of America: YouTube, 2016. Available: https://www.youtube.com/watch?v=zupKCDgzScw&t=770s

[11] Fifty50, Director, *ART DIRECTING YOUR DESTRUCTION IN HOUDINI // SERIES #03 DEBRIS & PYRO.* [Film]. United Kingdom: Vimeo, 2017. Available: https://vimeo.com/204553931

[12] Fifty50, Director, *ART DIRECTING YOUR DESTRUCTION IN HOUDINI // SERIES #02 GLUE NETWORK.* [Film]. United Kingdom: Vimeo, 2017. Available: https://vimeo.com/200513702

[13] 3dsmaya, Director, *Shaping RBD Material Fracture Input Points.* [Film]. United States of America: YouTube, 2019. Available: https://www.youtube.com/watch?v=YxAp9QNjxzo

[14] S. Houdini, Director, *RBD Tools Update | H17 Masterclass.* [Film]. United States of America: Vimeo, 2019. Available: https://vimeo.com/332035825