Houdini™ FOUNDATIONS

FOR FILM, TV & GAMEDEV

HOUDINI 16.5

SideFX®

Houdini™

FOUNDATIONS

FOR FILM, TV & GAMEDEV

HOUDINI 16.5

ROBERT MAGEE

SideFX®

HOUDINI FOUNDATIONS [PDF]

Author: Robert Magee

Cover Art: Joshua Matthews

ISBN: 978-1-7753338-0-7

First Published in 2018
by SideFX Software
123 Front Street West, Suite 1401, Toronto, Ontario M5J 2M2

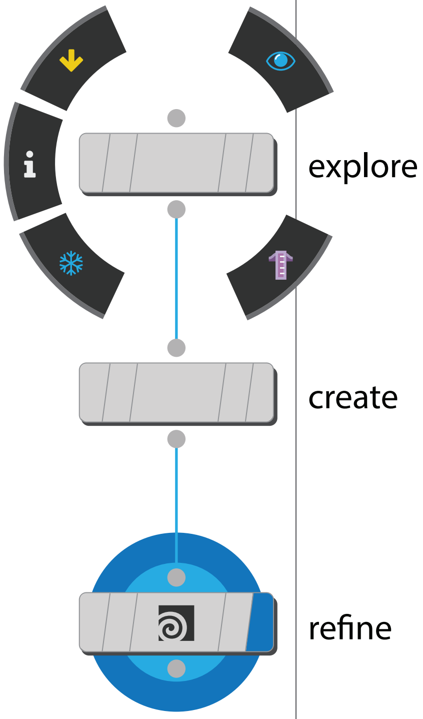
Written for the features found in Houdini 16.5.378+
Document Version 1.1 | March 2018

DISCLAIMER: Every reasonable effort has been made to obtain permissions for all articles and data used in this book, and to make it as complete as possible. This book should be considered "as is" and neither SideFX, nor its employees, officers or directors shall be responsible or incidental for consquential damages resulting from the use of this material or liable for technical or editorial omissions made herein.

HOUDINI FOUNDATIONS

HOUDINI FOUNDATIONS

OVERVIEW



To become a creator of 3D animation and VFX for Film, TV, Video Games and VR, you need a combination of technical and creative skills. Houdini is the perfect tool for bringing these worlds together as you explore, create and refine your projects from concept to final sign off.

While Houdini has a wide variety of tools designed for generating CG content, its **node-based procedural workflow** is what sets it apart. This approach makes it easier for you to **create directable shots**, explore **multiple iterations** and **hit deadlines**. As you learn Houdini, understanding how to work with these nodes and networks will be important to your success.

WHAT YOU WILL LEARN

This overview chapter contains general information about Houdini that will help you become familiar with important concepts and ideas. While you might not understand it all in your first pass, this chapter will be a valuable reference point as you build up your knowledge.

If you already work with 3D software then learning Houdini will be a transfer of existing skills. You will learn how to interactively build-up shots using the scene view and shelf tools, then how to work with the nodes and networks to take advantage of Houdini's procedural nature.

If you are new to 3D and Computer Graphics then Houdini is a great package to start with. The Foundations material assumes some general knowledge, therefore you may want to read up on CG concepts that you are unfamiliar with. In the end, Houdini will help you achieve a deeper understanding of what goes on under the hood of not only Houdini but other 3D apps as well.

Once you have finished the fundamentals, go to **SideFX.com** to find more tutorials. Go to **Learn > Learning Paths** for a comprehensive list of available lessons created by SideFX and members of the wider Houdini community. There is lots of material for you to explore as you build and refine your Houdini skillset.

DOWNLOAD
HOUDINI FOR FREE

SideFX has a free learning edition for you to use as you work through the lessons. The **Houdini Apprentice** edition gives you **FREE** access to all of Houdini's features with a few restrictions such as limited render size and user interface and render watermarks.

You can download Houdini Apprentice from the SideFX website where you can also get the latest versions which are updated regularly:

SideFX.com/download

INDIE
ANIMATORS AND GAMERS

If you want to go beyond the free learning edition, **Houdini Indie** removes the watermarks found in Apprentice and offers higher render resolutions up to 4K x 4K and limited commercial use **[less than $100K USD]** of Houdini.

The Indie program makes Houdini a great tool for developing personal projects and indie games. To learn more you can go to:

SideFX.com/indie

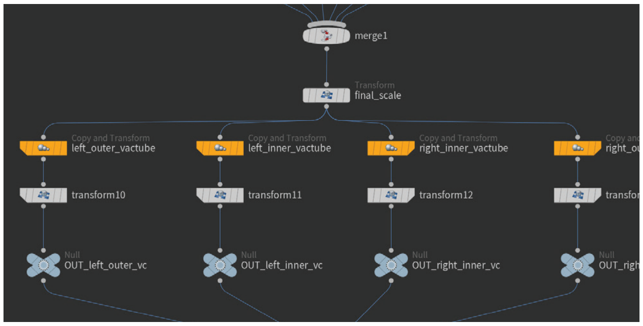
Learning Houdini

Houdini is a CG graphics application which you can use to model, animate, render and simulate. In the process of learning Houdini, you will explore new ways of managing the creative process that involves the interactive manipulation of nodes, networks and assets.

When we say everything is procedural in Houdini, this means that modeling, character rigging, lighting, rendering and visual effects all benefit from this node-based workflow which lets you build up networks of nodes to manage all the steps needed to accomplish a particular task. Networks also "talk" to each other to create even more sophisticated results.

GOING PROCEDURAL

In Houdini, every action is stored in a node. These nodes are then "wired" into networks which define a "recipe" that can be tweaked to refine the outcome then repeated to create similar yet unique results. The ability for the nodes to be saved and to then pass important information, in the form of attributes, down the chain is what gives Houdini its procedural nature.



KNOWN FOR VFX

Visual effects artists have traditionally gravitated to Houdini because this procedural workflow is ideal for working with particles and dynamics. Often visual effects are designed to react to actions that are taking place in a shot and a procedural solution "automates" these reactions. Therefore, Houdini provides studios with higher levels of productivity and more control over the creative process.

Igor Zanic

Houdini is also capable of working with large data sets which is critical as visual effects become more sophisticated with many layers such as rigid body destruction, fluids, and particles all interacting to achieve the final result.

PROCEDURAL CONSTRUCTS

For motion graphics projects, a procedural approach offers lots of visually-stunning eye candy. These special effects are often the result of animating parameters on nodes and adding noise in interesting ways that you wouldn't expect in real life.

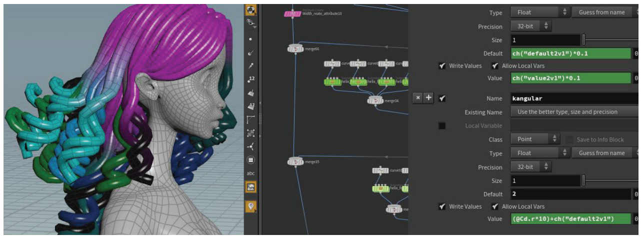
Niels PRAYER

THE WIDER CG PIPELINE

Beyond VFX and motion graphics, Houdini has bread-and-butter tools for all parts of the pipeline from modeling to rendering to character work and gamedev. Its procedural workflow supports you as you create all of your CG content. Along the way, you will benefit from the ability to explore multiple iterations and make changes deep into production.

Andrey Belichenko

While the nodes are what makes Houdini unique and give it its power, there are lots of viewport and shelf tools that are used to work interactively while Houdini builds the networks.


Type Float Guess from name
Precision 32-bit
Size 1
Default ch("default2v1")*0.1
☑ Write Values ☑ Allow Local Vars
Value ch("value2v1")*0.1
☑ Name kangular
Existing Name Use the better type, size and precision
■ Local Variable
Class Point Save to Info Block
Type Float Guess from name
Precision 32-bit
Size 1
Default 2
☑ Write Values ☑ Allow Local Vars
Value (@Cd.r*10)+ch("default2v1")

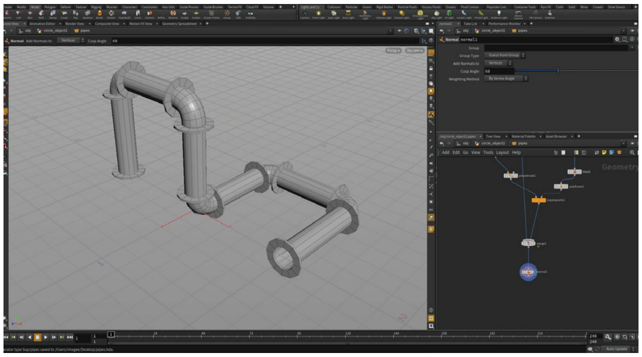
DIRECTABLE RESULTS

The reason you are able to make edits deep into production is because changes made to parameters on Houdini nodes will cascade right through the network to create a unique result. This directability is retained throughout the creative process and can be used to make last minute decisions that would be too costly in a traditional CG pipeline.

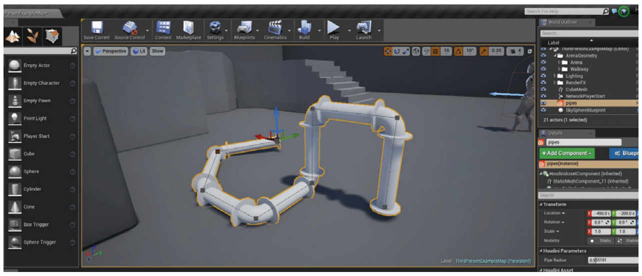
Alex Dracott

TOOL BUILDING

Another benefit of the node-based approach is that it is easy to encapsulate node networks to create custom nodes that are shared with colleagues without writing any code. Houdini's re-usable networks can be wrapped up quickly and easily into special nodes called **Houdini Digital Assets**.

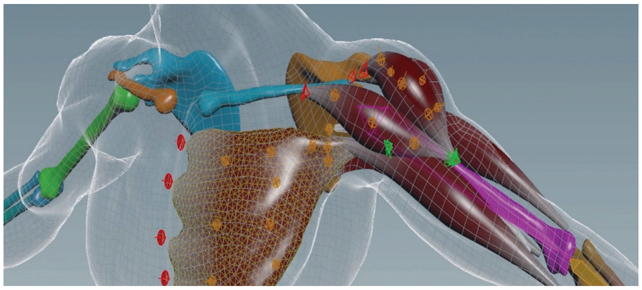


These assets open in Houdini, or in other applications such as **Autodesk Maya, C4D**, **UE4** and **Unity** using **Houdini Engine** plug-ins, with the asset's procedural nature left intact.



FULL ACCESS TO ALL YOUR DATA

As objects move through a typical animation or visual effects pipeline they accumulate information which is often stored as point or primitive attributes such as velocity, capture weights or UV texture coordinates. While other 3D applications hide this information and attempt to control it for you behind the scenes, Houdini gives you tools for working with and managing this data. This results in a much more powerful and flexible approach that makes a huge difference down the line.



A NEW WAY OF THINKING

As you become more proficient with Houdini, you will find new ways to approach a shot or a game level that will make you and your team more productive. Houdini gives you the flexibility to build tools that will support you throughout a project's life cycle and instead of simply reacting to issues and problems, you will be able to anticipate the pain points and use a procedural solution to work much more efficiently.

Andrey Bilichenko

Now that you have chosen to learn Houdini, you will find yourself exploring a versatile application that will redefine how you approach future projects. The key is to embrace this new way of working and be ready to explore CG at a level deeper than you ever imagined.

DO I NEED TO WRITE CODE TO USE HOUDINI?

DEFINITELY NOT! In fact, because of Houdini's node-based workflow, you will often be able to create results interactively that would require writing code in other 3D animation applications. Houdini is very much an artists tool and while it has a technical side that uses scripts and expressions, the out-of-box tools will let you accomplish amazing things. And the nodes let you easily go back and make changes which mimics how the creative process works.

If you do want to work with code then Houdini has a number of languages you can work with inside the Houdini interface. There are Wrangle nodes for working with VEX and Python and PyQT is supported as well. You can also use Houdini's expression language hscript or you can mix all of the together to meet your specific needs.

The Houdini Workspace

Houdini offers a user interface experience that will be familiar for artists coming from other CG applications with the biggest difference being the panes used to manage nodes and networks. The workspace is highly configurable and can be set up to support different ways of working.

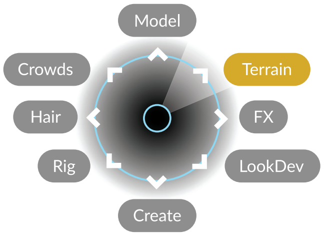
Houdini gives artists many different ways to view the bits and pieces that make up a 3D scene. From the **Scene View** where you look through a camera at your geometry to the **Network view** where you manage the procedural nodes and networks, you will find many different ways to make creative decisions while making sure each shot works at a technical level.

RADIAL MENUS

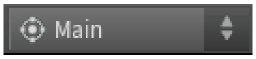
One way to access tools in Houdini is the radial menus which you can access using the **X**, **C** and **V** hotkeys. Each of these brings up a radial menu with lots of options for you to choose from. The main focus of each menu is as follows:

Snapping	X
Main	C
Views	V

Once you learn how a radial menu works, you can access the tool with quick sweep gesture without dwelling on the widget.

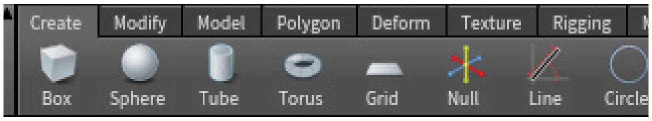


You can change the **custom** menu at the top of the menu bar which says **Main** by default. On OS X this is the **Radial** menu.



SHELF TOOLS

At the top of the workspace, you will find multiple shelves filled with tools for creating and manipulating objects, geometry, cameras, lights and effects.

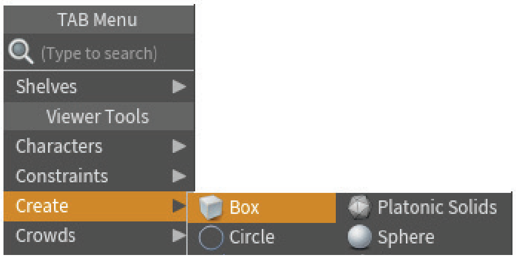


These tools work in the scene view and often involve some sort of scene view interaction. Once you have used one of these tools, one or more nodes will be created which you can then refine in the **Parameter** and **Network** panes.

The shelves offer a very important resource to new Houdini artists because the shelves reduce clicks and often put down networks of nodes that you can learn from.

TAB MENU

Another way to access tools in either the **Scene** view or the **Network** view is to press the **tab** key. This brings up a menu of available tools and nodes you can use in your work.



HOUDINI WORKSPACE

Tool Shelf - The shelf tools let you work with objects and geometry in the scene view.

TOOL BAR:

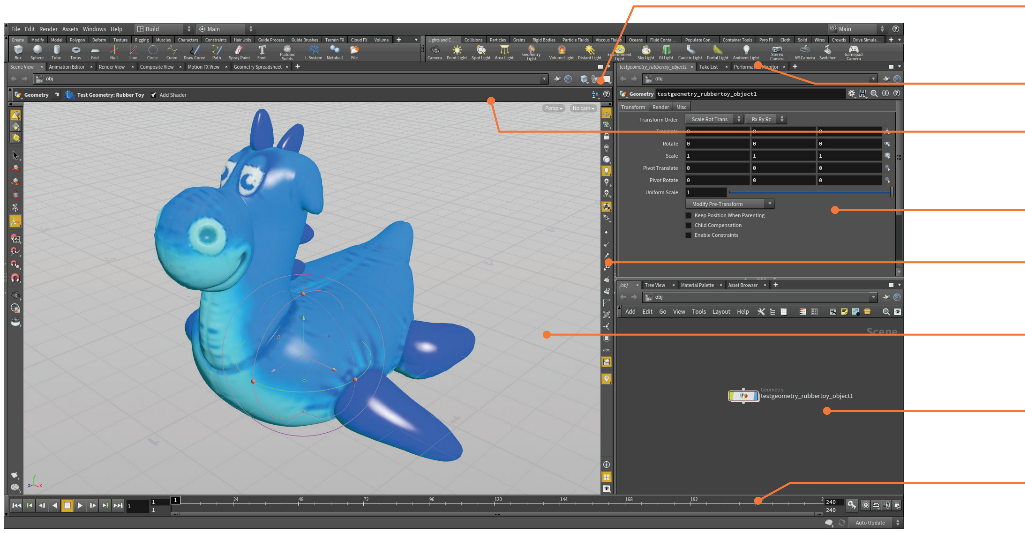
Selection Modes - These modes let you focus on scene, geometry, or dynamic objects.

Transform Tools - Select, Move, Rotate, Scale or Pose or the Handle tool for node-specific controls.

Snapping Tools - Turn on Grid, Primitive, Point or Multi-snapping.

Viewing - Use the View tool to Tumble, Pan and Dolly or Render Region to render in the Scene View.

Output Tools - You can render or flipbook your scene with these tools.



VIEW TOOLS

Here are some of the hotkey combinations available while viewing. You can skip **spacebar/alt** if you are actually in the View tool:

Tumble	Spacebar or Alt[Opt] - Left Mouse Button [LMB]
Pan	Spacebar or Alt[Opt] - Middle Mouse Button [MMB]
Dolly	Spacebar or Alt[Opt] - Right Mouse Button [RMB]

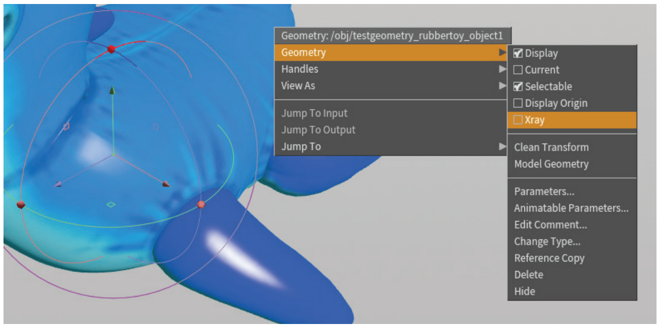
You can find the **View** tool in the toolbar. When you use the **spacebar** or **alt** keys, you temporarily evoke the view tool without interrupting the use of your current tool. This can be quite useful when you are selecting or manipulating in a view and need to quickly change your point of view.

If you want to focus on viewing then you can press **Escape** to go to the **View** tool. In some cases, you will want to home in to get your bearings. There are some hotkeys for that as well:

Home Grid	Spacebar + H
Home All	Spacebar + A
Home Selected	Spacebar + G

RIGHT MOUSE MENUS

There are many right mouse button menus available in Houdini depending on your context. If you RMB click geometry, you will get a different menu than if you RMB click a handle or in empty space. You can also RMB click pane tabs, tools, tool shelves and on nodes in the network editor.



Viewport Display Menus - These tabs let you create and organize multiple panes at the same time.

Pane Tabs - These tabs let you create and organize multiple panes at the same time.

Operation Controls - Use the Handle tool with this bar to access parameters from your selected node.

Parameter Pane - This pane lets you set values, add expressions and keyframe selected nodes.

Display Options Bar - These toggles let you control scene display options such as normals, point numbers or lighting.

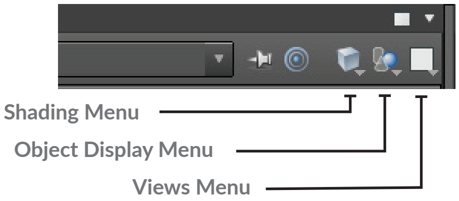
Scene View - Visualize your work and use handles to manipulate objects interactively in your scene.

Network Pane - View and manage networks of nodes to work with the underlying structure of your scene.

Timeline - Set the current time and edit keyframes on selected nodes. You can also use the timeline to copy and paste keyframes.

VIEWPORT DISPLAY MENUS

You can change how objects and viewports appear in the scene view using the menus found in the top right of the scene view panel or using the **V** radial menu.



Shading Menu - Choose from options such as wireframe, flat shaded, smooth shaded or smooth wire shaded. For more options, click on the **Display options** button on the Display bar.

Object Display Menu - As you dive into various networks, this menu sets whether you can hide other objects, view them all or see them ghosted.

Views Menu - This menu lets you split your scene view into various views such as perspective or orthographic views and has an option for syncing up all the orthographic views.

DISPLAY OPTIONS BAR

At the right side of the scene view, the display bar gives you access to options for viewport display. Here are a few examples.

Reference Plane/Ortho Grid - Turn on and off a grid that can be used for grid snapping and provides an implied ground plane while you work.

Construction Plane - Turn on and off a construction plane is referenced when you create or edit an object.

Lock Camera- Lock the current camera to the view so that view changes are modify the camera transform values.

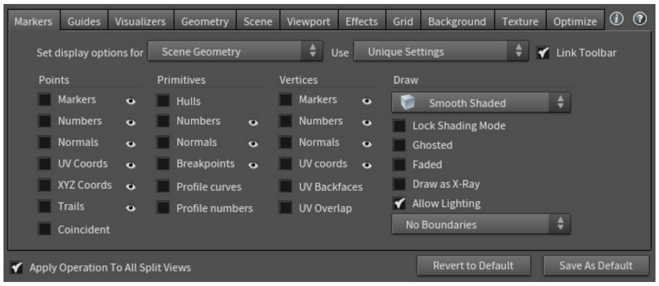
High Quality Lighting with Shadows- Set the best quality of viewport rendering.

Display Primitive Normals - Show the normals belonging to all primitives in the scene to determine their direction.

DISPLAY OPTIONS

The Scene and Network views each have display option panels that offer even more options for working with these panes. You can access them by clicking on the icon at the bottom of the display options bar or using the following hotkey:

Display Options	D



Set display options for Scene Geometry Use Unique Settings ☑ Link Toolbar

Points: □ Markers, □ Numbers, □ Normals, □ UV Coords, □ XYZ Coords, □ Trails, □ Coincident

Primitives: □ Hulls, □ Numbers, □ Normals, □ Breakpoints, □ Profile curves, □ Profile numbers

Vertices: □ Markers, □ Numbers, □ Normals, □ UV coords, □ UV Backfaces, □ UV Overlap

Draw: Smooth Shaded, □ Lock Shading Mode, □ Ghosted, □ Faded, □ Draw as X-Ray, ☑ Allow Lighting, No Boundaries

☑ Apply Operation To All Split Views

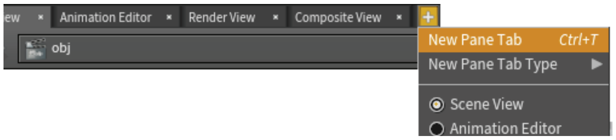
Panes & Desktops

Your workspace is broken up into Panes which offer unique ways of organizing your scene data. You can work interactively in a 3D view or analyze attribute values in a spreadsheet. It is important to learn how these different UI elements can be used to get your work done.

PANES AND PANE TABS

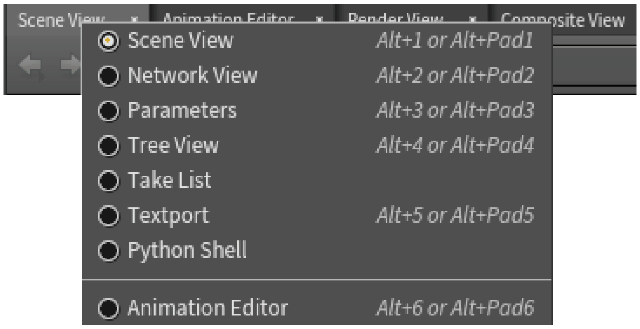
The Houdini workspace is divided into panes so that you can explore your scene data in different ways. Pane tabs let you overlap several panes within the same zone to keep them handy but not visible by default

You can access a pane tab by clicking on it in the workspace. You can close it by clicking on the x. The + menu can be used to change the pane type or add new panes.



PANE TYPES

You can also LMB-click on the pane tab itself to change its type. There are many pane types to choose from. Here are some of them which have hotkeys. Refer to the documentation to find learn more about all of the others.



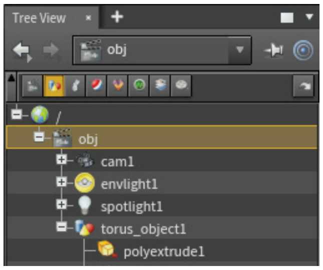
Scene View
Animation Editor
Render View
Composite View
Scene View Alt+1 or Alt+Pad1
Network View Alt+2 or Alt+Pad2
Parameters Alt+3 or Alt+Pad3
Tree View Alt+4 or Alt+Pad4
Take List
Textport Alt+5 or Alt+Pad5
Python Shell
Animation Editor Alt+6 or Alt+Pad6

Scene View - Work interactively in 3D space. This type of view can be set up with one or more viewports. You can have more than one scene view panels open at the same time to look at your scene from different points of view.

Network View- This view lets you see the nodes and networks and connect, rewire, and reorganize them to suit your needs.

Parameters - Set values on parameters, add expressions and control the properties of your nodes.

Tree View- This is a hierarchical view of the nodes. For some people this is a great tool for understanding how scene hierarchies work.

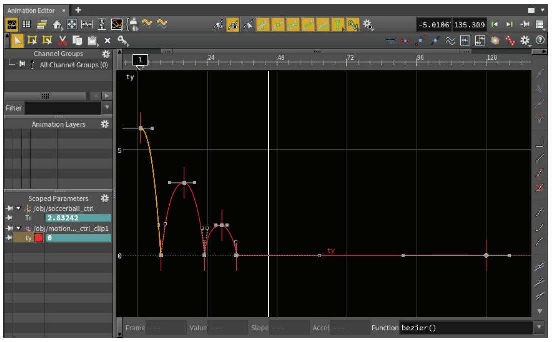


Take List- This list lets you explore different “takes” by making changes to specific parameters. You can then manage the takes to ensure that you have made the right creative choices.

Textport - You can type commands in the panes.

Python Shell -You can type Python commands in this pane.

Animation Editor- Manage keyframes and animation curves. It also has a table view and Dope sheet view.



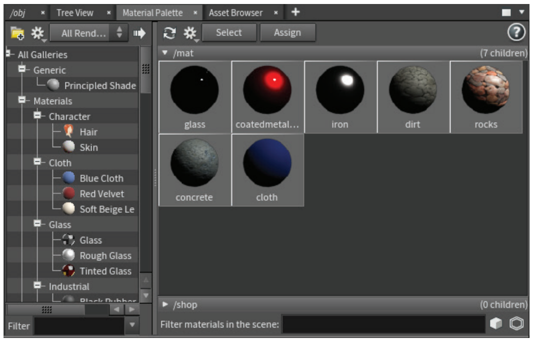
Channel List - Create channel groups and manage the scoped channels as you animate in Houdini.

Motion FX View - This lets you view motion created using Houdini’s channel operator or CHOP nodes.

Render Scheduler - This panel shows you renderings that have taken place and those in progress. You can pause and kill renderings here.

Composite View - View composited images created using Compositing [COP] nodes.

Material Palette - This palette lets you see all the materials in your scene and select and assign them to objects and geometry.

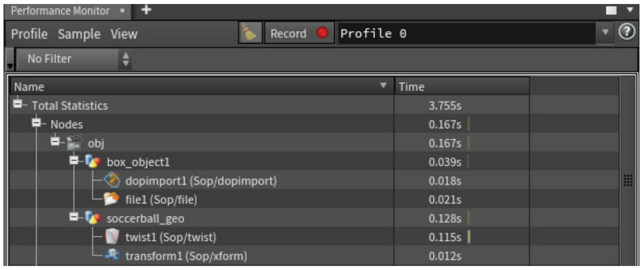


Render View - Start interactive renderings, that will update when you change something in your scene.

Geometry Spreadsheet- A view of the attribute values you have on your geometry. This could include UVs, normals, or custom values you have set yourself.

Data Tree- This view gives you access to a light bank, Material Stylesheet and Object appearance list that can be used for LookDev.

Performance Monitor- This pane lets you explore how long a network chain is taking to cook then take steps to make it more efficient.



Name	Time
Total Statistics	3.755s
Nodes	0.167s
obj	0.167s
box_object1	0.039s
dopimport1 (Sop/dopimport)	0.018s
file1 (Sop/file)	0.021s
soccerball_geo	0.128s
twist1 (Sop/twist)	0.115s
transform1 (Sop/xform)	0.012s

Help Browser - This pane gives you access to online documentation.

Orbolt Asset Browser - This browser lets you access assets from Orbolt.com. To use this pane you need to log in with our orbolt.com account.

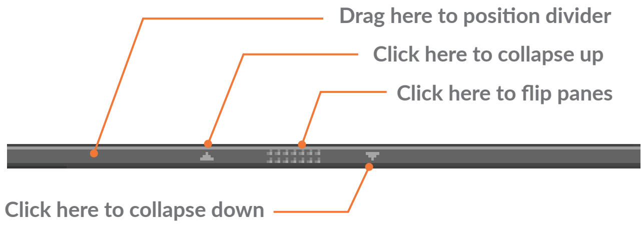
Autorigs- This pane gives you access to tools for building your own rig out of modules for biped, quadruped and facial rigs.

Character Picker - You can use this pane to make it easier to select parts of a character rig. You can have more than one page of selectors.

Pose Library - This pane makes it easy for animators to store and re-use poses and clips for a character.

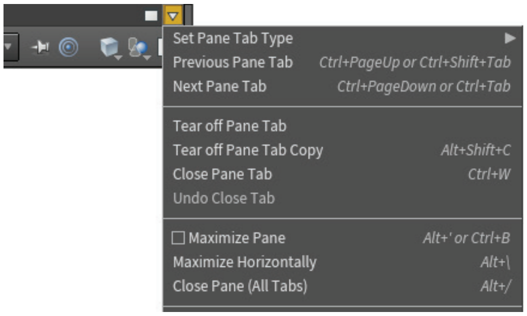
ORGANIZING AND COLLAPSING PANES

Both panes and tool bars can be collapsed and expanded by clicking on the arrows found in their UI. Whole panes can be collapsed to the left or right and you can flip the contents using the center grip. These options let you focus on certain panes by hiding others using a single click of the mouse.


Drag here to position divider
Click here to collapse up
Click here to flip panes
Click here to collapse down

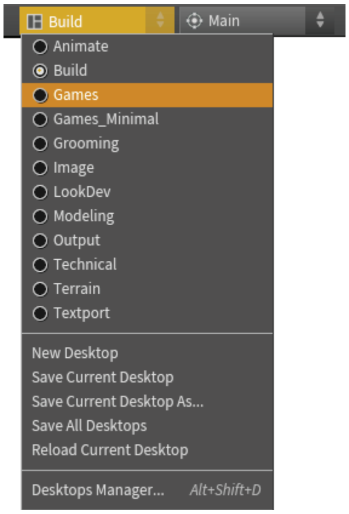
PANE MENU

At the top left of each pane is a button for maximizing and minimizing the pane and an arrow which gives you access to the Pane menu. This menu lets you tear off the pane or a copy of the pane, close, or split your pane. You also have options for determining the UI of each pane.


Set Pane Tab Type
Previous Pane Tab Ctrl+PageUp or Ctrl+Shift+Tab
Next Pane Tab Ctrl+PageDown or Ctrl+Tab
Tear off Pane Tab
Tear off Pane Tab Copy Alt+Shift+C
Close Pane Tab Ctrl+W
Undo Close Tab
□ Maximize Pane Alt+' or Ctrl+B
Maximize Horizontally Alt+\
Close Pane (All Tabs) Alt+/

DESKTOPS

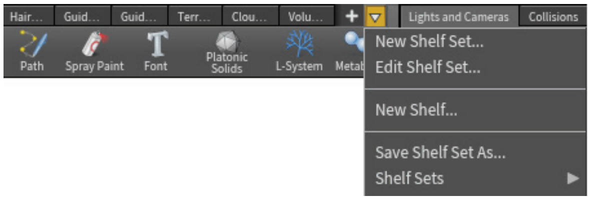
As you open up tabs, add dividers and organize your pane tabs, you start to set up your own workspace. To save any layout, go to the **Desktop** menu (**Windows > Desktop** on OSX) where you can access saved desktops, save your own and manage them as you work. When you save a desktop, it will save the Pane layout, Radial menus and visible Shelf sets.



When you save your scene, it remembers which Desktop you are looking at but not any changes to the pane layout while working. These changes will go away unless you explicitly save them to the desktop or create a new desktop.

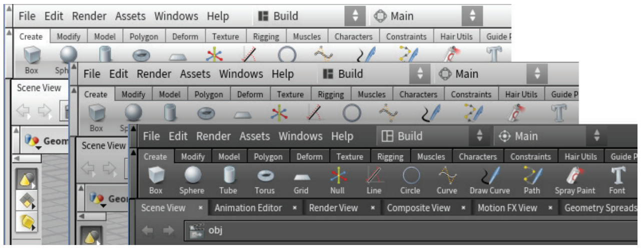
SHELVES AND SHELF SETS

To manage the shelves at the top of the workspace, access the menu found under the arrow icon. You can use this to work with Shelf sets. You can also bring up shelf sets that might be hidden in your desktop.



COLOR SETTINGS

You can customize the look of the Houdini UI by choosing a color scheme for your workspace. Select **Edit > Color Settings** to bring up the option window then you can choose from the default **Houdini Light** or **Houdini Pro** or **Houdini Dark**.


File Edit Render Assets Windows Help
Build
Main
Create Modify Model Polygon Deform Texture Rigging Muscles Characters Constraints Hair Utils Guide
Box Sphere Tube Torus Grid Null Line Circle Curve Draw Curve Path Spray Paint Font
Scene View
Animation Editor
Render View
Composite View
Motion FX View
Geometry Spreads
obj

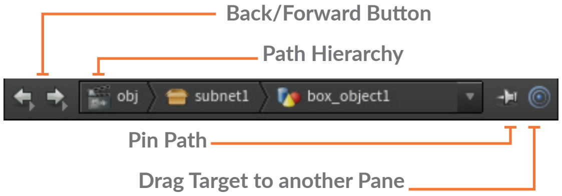
Nodes & Networks

With Houdini's node-based workflow at the heart of its procedural architecture, the ability to work directly with these nodes and networks becomes very important to using it effectively. While the idea of nodes might sound technical, they are actually quite artist friendly and easy to work with.

As you use tools in Houdini, nodes are created and wired with other nodes. The resulting networks offer a history of your actions while providing a simple way to use this flow of information to make changes and refine your work. While it is possible to focus all your effort on the scene view, learning how to work effectively with the node networks will give you more flexibility in the long run.

NETWORK PATHS

Nodes are organized hierarchically with some nodes nested in other nodes known as network managers or subnetworks. To help you manage these hierarchies, a browser-like path is available at the top of most panes.



You can use this path to navigate up and down this hierarchy or to other networks you have been working with. By default, the path changes as you make selections in the scene view although you can pin down a path to keep it focused. You can also drag the target icon to another pane to change its path.

NETWORK TYPES

Houdini includes different kinds of nodes which each work in their own context. The network type is highlighted in the top right corner of the network view and can be used to determine what kind of network you are working with which, in turn, determines what kind of nodes are available. Nodes from each type can connect to other networks. While the different types of nodes are similar in how they are wired together, they each have unique capabilities.

It is important to learn how each type of node works.

Scene	Objects	OBJ
Geometry	Surface Operators	SOP
Materials	VEX Operators	MAT
Motion FX	Channel Operators	CHOP
VEX Builder	VEX Operators	VOP
Outputs	Render Operators	ROP
Dynamics	Dynamic Operators	DOP
Compositing	Compositing Operators	COP/IMG

THE "SECRET" LANGUAGE OF HOUDINI

If you encounter a seasoned Houdini user, you will hear them talk of SOPs, DOPs, VOPs etc. which refers to the node types listed above. As you work though this chapter, you will begin to learn how to use this "secret" language to talk about the node types and how they apply to working procedurally.

NAVIGATING NETWORKS

To jump between network types there are a number of different approaches you can take. Some of these happen naturally as you work with objects in the scene view and others offer shortcuts which can get you to another network more quickly.

Selection Modes - As you select in the scene view, the network editor jumps to the location of the selection. Different selection modes will in turn take you to different network types as you make a selection.

Network path - You can **LMB-click** on the network path to go

NETWORK VIEW

Network Path - The path leading to the current network level. You can also use this bar to navigate to other networks.

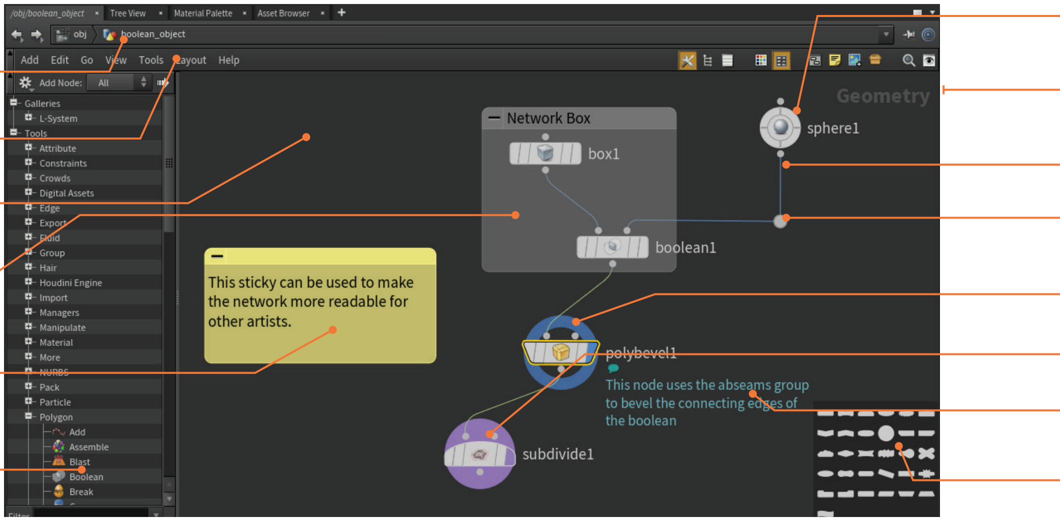
Pane menu - These menus and icons are for organizing your network.

Network Background - You can add an image or set up a grid to help you organize your nodes.

Network Box - Group related nodes then quickly collapse and expand them.

Sticky - Add notes to help other artists read your network or to offer ideas for their networks.

Node Gallery - Drag nodes from here to your network. Use the filter at the bottom to find the node you need.



This sticky can be used to make the network more readable for other artists.

This node uses the abseams group to bevel the connecting edges of the boolean back up the path or **RMB-click** to access parallel networks.

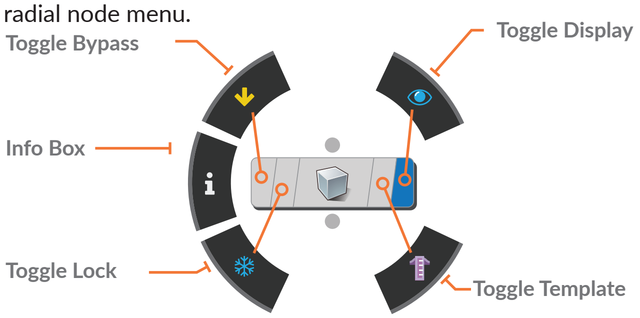
Hotkeys - These hotkeys help you navigate up and down as you work with a selected object.

Dive in	I
Jump up	U
Toggle Objects/Geometry	F8

Jump to Menu - If you **RMB-click** on geometry, then you can use the **Jump to** submenu to go to networks which contain nodes related to the geometry. This could be a material, particle network or dynamic network.

NODE FLAGS

Each of the nodes seen in the network view have various flags which determine if it is displayed, locked or bypassed. You can evoke these by either clicking on the flag itself or using the radial node menu.



Display Flag [R] - This flag lets you choose the display output node for the network and is highlighted with a hollow ring. The **Render flag** [T] sets which node will be output for rendering and is highlighted with a solid circle. You can set this separately from the Display flag by **Ctrl-clicking** on the Display flag.

Template Flag [E] - This flag displays the node for reference or snapping.

Lock Flag - This caches at the locked node and all nodes earlier in the chain are ignored when the network is cooked.

Bypass Flag [B] - This flag lets you ignore the node when the network is cooked.

SELECT AND VIEW HOTKEYS

In the network pane, you will need to pan and zoom around to work with the complete network. Here are the key combinations for these actions.

Pan	MMB
Zoom	RMB
Select Nodes	LMB
Add to Selection	Shift + LMB
Remove from Selection	Ctrl + LMB

CONNECTING AND DISCONNECTING NODES

Here are some other ways of interacting with nodes and connections in the network pane:

Connect Node	LMB-drag from output to input
Insert New Node	RMB connector
Insert Node	LMB drag and drop onto connector wire
Disconnect from Wires	LMB then Jiggle node
Cut Wire	Y drag across connector wire
Move node	LMB-drag
Copy selected nodes	Alt + LMB-drag
Reference Copy	Alt + Shift + Ctrl + LMB-drag

Dot nodes can be used to organize your networks:

Add Dot	Alt + LMB wire
Pin/Unpin Dot	Y drag across connector wire

NODE GALLERIES

The galleries offer quick access to nodes that you want to add to your network directly. The galleries contain those nodes used the most in day-to-day work while the **tab** key gives you access to all the available nodes.

You can create your own galleries using the **Windows > Gallery Manager** and you can add items to your galleries by **RMB-clicking** on a node then choosing **Save to Gallery...**

Nodes saved in the **Mat** network will also be available in the **Material Palette** as long as they are given the proper keywords such as *Mantra* for Mantra materials.

Node - This represents an operation that contributes to the final output of the network.

Network Type - Shows which network type you are working in.

Connector - The connecting lines show how your nodes are linked together and how the data is moving through the network.

Dot - You can add dot connectors to make it easier to organize your nodes.

Display Ring - This small circle shows which node is displayed in the Scene view.

Render Ring - This large circle indicates a render node even if another node is displayed.

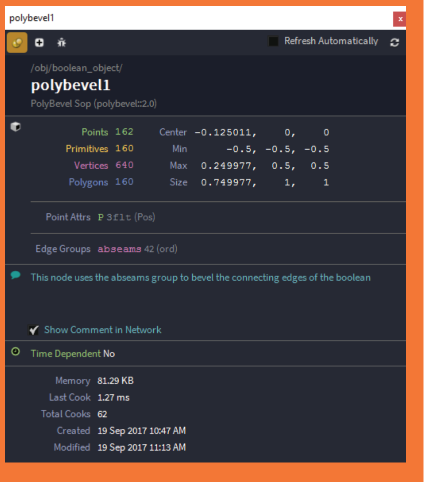
Comments - Node comments can be displayed to help other artists understand your thinking.

Palettes - Buttons up top bring up Palettes that let you set the color and shape of your nodes.

LEARN ABOUT YOUR NODE

Bring up the **Info Box** using either the **Radial Menu** or **MMB-press** on the node This panel gives you info about the node's contents, groups, attributes and other important facts. This panel also explains any errors that are interfering with your workflow.

This panel is temporary but you can click on the pin icon to keep it visible as you work. You can add comments and display them in the Network view using this panel.



```
polybevel1
                                        Refresh Automatically
/obj/boolean_object/
polybevel1
PolyBevel Sop (polybevel::2.0)

     Points 162      Center -0.125011,    0,    0
 Primitives 160         Min      -0.5, -0.5, -0.5
   Vertices 640         Max  0.249977,  0.5,  0.5
   Polygons 160        Size  0.749977,    1,    1

  Point Attrs  P 3flt (Pos)

  Edge Groups  abseams 42 (ord)

This node uses the abseams group to bevel the connecting edges of the boolean

☑ Show Comment in Network

Time Dependent No

      Memory  81.29 KB
   Last Cook  1.27 ms
 Total Cooks  62
     Created  19 Sep 2017 10:47 AM
    Modified  19 Sep 2017 11:13 AM
```

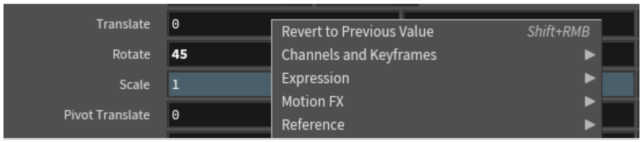
Parameters, Channels & Attributes

All of the nodes in Houdini are driven by parameters, channels and attributes to help you achieve the results that you want. The terminology used in Houdini may differ from other 3D applications therefore it is a good idea to take a moment to understand them in a Houdini context.

PARAMETERS

Parameters refer to the values, sliders, buttons and checkboxes found on Houdini nodes. These are sometimes referred to as attributes in other applications but Houdini uses attributes in a different way.

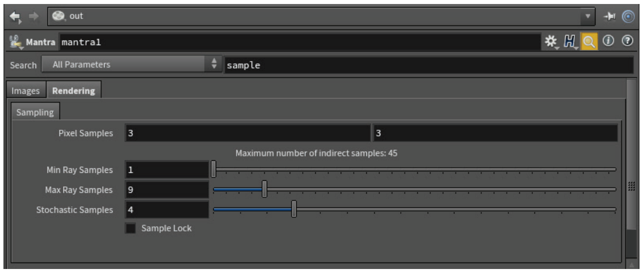
You can change parameter values in the Parameter pane or using handles in the viewport. There is a RMB menu on each parameter that gives you a number of important options such as copying and pasting and reverting to defaults.



Translate 0
Rotate 45
Scale 1
Pivot Translate 0

SEARCHING PARAMETERS

A node might have a large number of parameters and sorting through them all can take time. If you click on the magnifying glass in the top right, you get a search bar that lets you filter the parameters based on name and content. You can find parameters using expressions, overrides and even a raw value.



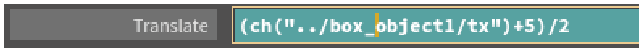
Mantra mantra1
Search All Parameters sample
Pixel Samples 3 3
Maximum number of indirect samples: 45
Min Ray Samples 1
Max Ray Samples 9
Stochastic Samples 4
□ Sample Lock

CHANNELS | KEYFRAMES

You can set a keyframe on a parameter by pressing the **Alt** key and clicking on the name or value field. Once you have set a keyframe the parameter's field changes color and you have created an animated channel. There will now be keyframes associated with the parameter which you can access in the Animation editor.

CHANNELS | EXPRESSIONS

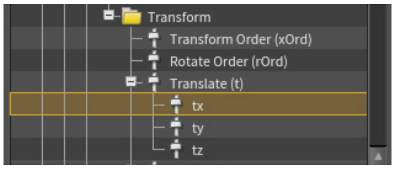
Instead of a raw value, you can add expressions into the parameter using either *hScript* or *Python*. There is a menu in the top right of the Parameter pane for choosing which language you want to use. You can press **Ctrl-E** to bring up an expression editor with a number of scripting tools to make it easier to work.



Translate (ch("../box_object1/tx")+5)/2

REFERENCE SCENE DATA

You can also **RMB-click** on a parameter and **Choose reference > Scene Data** to bring up a window for choosing specifically what you want to link to. Once you have made a choice from any node in your scene, to create a channel reference. This method lets you create references without worrying about the exact syntax needed to write the proper expression.



PARAMETER PANE

Navigation Bar- This bar lets you see where the node is located in the scene hierarchy.

Node Type and Name - Here you can see the node type and set its name. Clicking on the icon gives you a menu for working with the node.

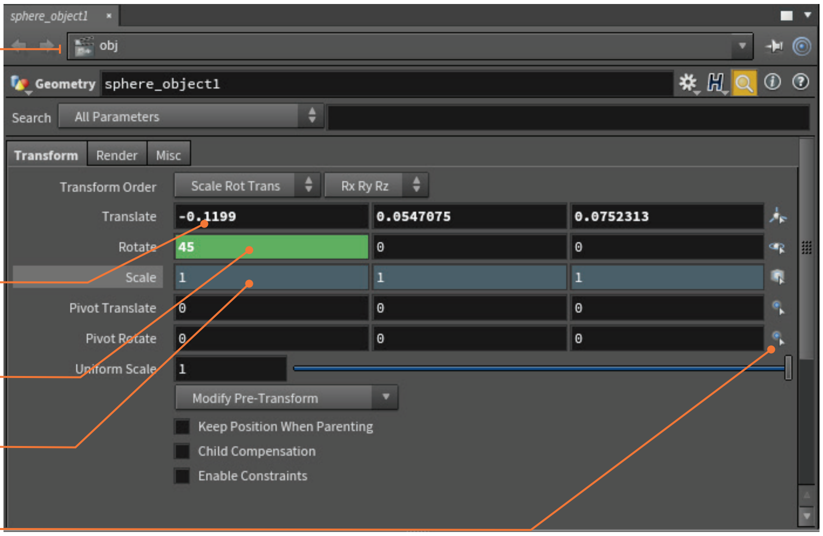
Search Bar - Click on the magnifying glass icon to search parameters by name or by content.

Changed Parameter - When a parameter has been changed from its defaults then its value is bolded. The folder tab name is also bolded

Animated Parameter - When you have keyframed a parameter it is highlighted in green.

Locked Parameter - You can RMB-click on a parameter to lock and unlock it. It will be highlighted in grey

Select to Match - These icons let you match these parameter values to other objects.



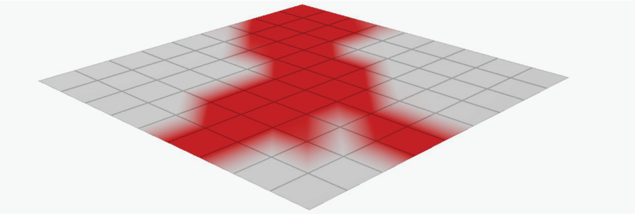
Geometry sphere_object1
Search All Parameters
Transform Order Scale Rot Trans Rx Ry Rz
Translate -0.1199 0.0547075 0.0752313
Rotate 45 0 0
Scale 1 1 1
Pivot Translate 0 0 0
Pivot Rotate 0 0 0
Uniform Scale 1
Modify Pre-Transform
□ Keep Position When Parenting
□ Child Compensation
□ Enable Constraints

CUSTOM PARAMETERS

If you click on the Gear icon in the top right of the Parameter pane, you can choose **Edit Parameter Interface**. Here you can add custom parameters which can then be linked to other parts of your node network.

ATTRIBUTES

Attributes let you attach data to your geometry that can be used by nodes down the chain to complete an operation. A fuel attribute can drive a Pyro FX simulation or a UV attribute sets up texturing. Some attributes are created by Houdini nodes or you can create custom attributes.

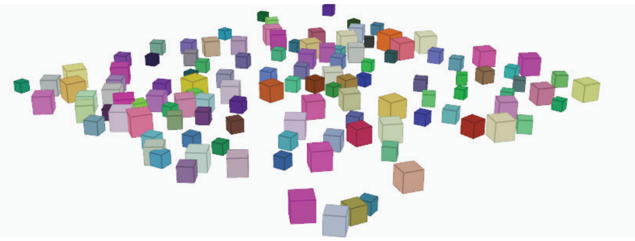


Class - Attributes can belong to Points, Primitives, Details and Vertices. This will affect how they get used down the chain.

Type - You can set up float, integer or string attribute types amongst others.

ATTRIBUTE RANDOMIZE

Attribute Randomize lets you create an attribute and immediately randomize its values. For instance here you can see the Color, rotation and Scale of these boxes being randomized.

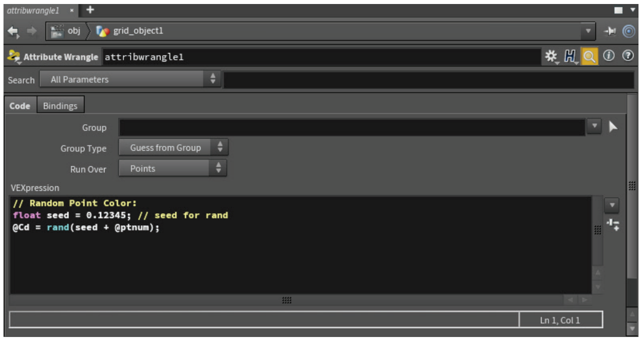


ATTRIBUTE TRANSFER

Within a node chain, attributes are attached to geometry then used by other nodes. You can also pass attributes to other pieces of geometry using **Attribute transfer**. Here the sphere is passing color attributes to the boxes based on a defined threshold value.

ATTRIBUTE WRANGLE

Houdini has a wide variety of nodes that let you create and work with attributes. You can also use the **Attribute Wrangle** node to use a script-based approach to this work. For a lot of Technical Directors this may be the most comfortable way to work.



Attribute Wrangle attribwrangle1

Search: All Parameters

Group:

Group Type: Guess from Group

Run Over: Points

VEXpression

```
// Random Point Color:
float seed = 0.12345; // seed for rand
@Cd = rand(seed + @ptnum);
```

For artists, this might not be the preferred method, therefore working with nodes will make it easier to deal with this kind of information. A lot of Houdini's power is found in the proper use of Attributes and you will eventually need to learn about them.

GEOMETRY SPREADSHEET

You can view all attribute values, even invisible ones using the geometry spreadsheet.

Navigation Bar- This bar lets you see where the node is located in the scene hierarchy.

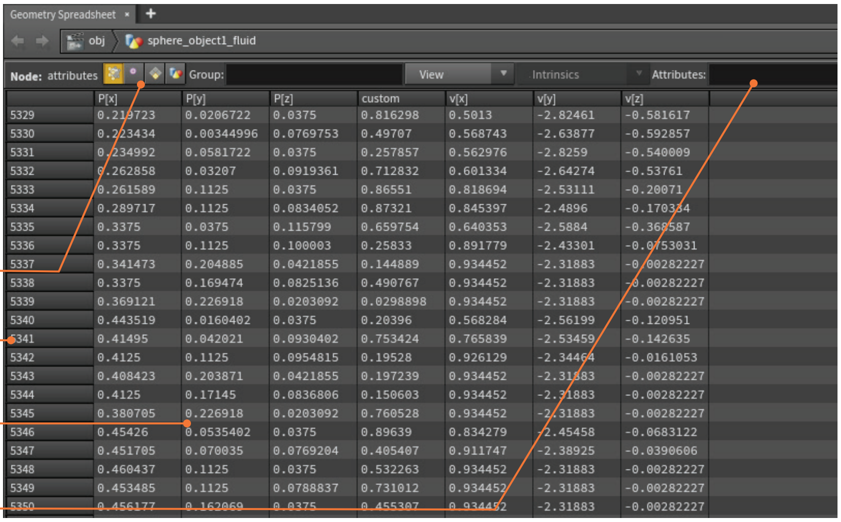
Node Name - This shows you which node is currently selected and which node is generating these attribute values.

Attribute Class Buttons - Use these buttons to filter which kind of attribute you are looking at.

Point Number - Here are the geometry point numbers to help you determine where the attribute is on your model.

Attribute Values - These are the values at this point in the node network chain. .

Filter - You can type in parameter names in here to filter the list when you are working with too many parameters.



Node: attributes

Group:

Attributes:

	P[x]	P[y]	P[z]	custom	v[x]	v[y]	v[z]
5329	0.219723	0.0206722	0.0375	0.816298	0.5013	-2.82461	-0.581617
5330	0.223434	0.00344996	0.0769753	0.49707	0.568743	-2.63877	-0.592857
5331	0.234992	0.0581722	0.0375	0.257857	0.562976	-2.8259	-0.540009
5332	0.262858	0.03207	0.0919361	0.712832	0.601334	-2.64274	-0.53761
5333	0.261589	0.1125	0.0375	0.86551	0.818694	-2.53111	-0.20071
5334	0.289717	0.1125	0.0834052	0.87321	0.845397	-2.4896	-0.170334
5335	0.3375	0.0375	0.115799	0.659754	0.640353	-2.5884	-0.368587
5336	0.3375	0.1125	0.100003	0.25833	0.891779	-2.43301	-0.0753031
5337	0.341473	0.204885	0.0421855	0.144889	0.934452	-2.31883	-0.00282227
5338	0.3375	0.169474	0.0825136	0.490767	0.934452	-2.31883	-0.00282227
5339	0.369121	0.226918	0.0203092	0.0298898	0.934452	-2.31883	-0.00282227
5340	0.443519	0.0160402	0.0375	0.20396	0.568284	-2.56199	-0.120951
5341	0.41495	0.042021	0.0930402	0.753424	0.765839	-2.53459	-0.142635
5342	0.4125	0.1125	0.0954815	0.19528	0.926129	-2.34464	-0.0161053
5343	0.408423	0.203871	0.0421855	0.197239	0.934452	-2.31883	-0.00282227
5344	0.4125	0.17145	0.0836806	0.150603	0.934452	-2.31883	-0.00282227
5345	0.380705	0.226918	0.0203092	0.760528	0.934452	-2.31883	-0.00282227
5346	0.45426	0.0535402	0.0375	0.89639	0.834279	-2.45458	-0.0683122
5347	0.451705	0.070035	0.0769204	0.405407	0.911747	-2.38925	-0.0390606
5348	0.460437	0.1125	0.0375	0.532263	0.934452	-2.31883	-0.00282227
5349	0.453485	0.1125	0.0788837	0.731012	0.934452	-2.31883	-0.00282227
5350	0.456177	0.162069	0.0375	0.455307	0.934452	-2.31883	-0.00282227

Selecting Geometry

Working in Houdini involves the selection and manipulation of many different elements. There are a number of tools and options available to help you work efficiently with objects and geometry components such as points, edges and primitives.

SELECT TOOL

The Select tool lets you focus on making selections therefore it doesn't have any manipulation handles. When working with other tools such as **Move** or **Rotate**, your can select freely or you can lock it using secure selection in which case you must evoke the Select tool with hotkey **S** to make a new selection.

Select Tool	Tap S
Evoke the Select tool while in other Tool	Press and Hold S
Toggle Secure Selection	~

SELECTION TYPES

There are different shortkeys for adding, subtracting or toggling your selection as well as for selecting all or selecting none. You will use these techniques a lot in your work.

Select	LMB
Add to Selection	Shift + LMB
Remove from Selection	Ctrl[Cmd] + LMB
Toggle Selection	Ctrl[Cmd] + Shift + LMB
Select All	N
Select None	Shift + N

SELECTION TECHNIQUES

To select in the viewport, you can choose from four different types of selection that give you distinct ways of accessing your geometry.

Box Select	F2
Lasso Select	F3
Brush Select	F4
Laser Select	F5

There are also some selection filters that let you focus on visible geometry or select groups. You have a wide range of selection options to help make this easier as you work.

Select Visible Geometry Only	Shift + V
Select Fully Contained Geometry Only	Shift + C
Select Groups or Connected Geometry	9

SELECTION MODES

Selection modes, give you access to objects and components. They also let you easily jump from object level to geometry level using a the buttons in the toolbar or the hotkey.

Objects - The object network level is where you work with an object's transforms. In any tool other than the **View** tool, the following hotkey will bring you back to the Object level:

Objects	1

Geometry - You can use any of the following hotkeys, when not in the **View** tool, to jump into the geometry level with the chosen components available for selection.

Points	2
Edges	3
Primitives [Faces]	4
Vertices	5

SLOPPY SELECTION

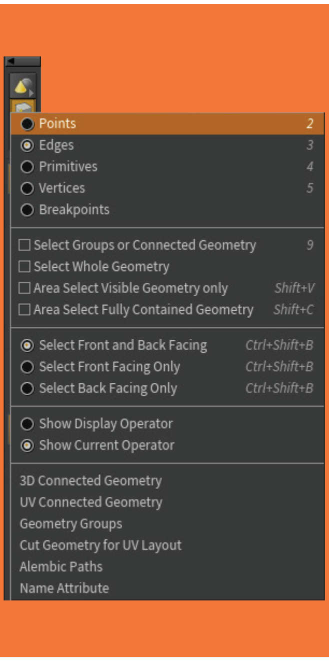
In most cases, only one of the geometry selection modes can be active at a time. But if you are working with an **Edit** node then **Sloppy Selection** lets you choose any combination of points, edges and primitives on the fly.

SELECT MODE MENUS

Each of the selection modes comes with options which let you alter how you interact with your scene. You can access these options by either **LMB** or **RMB-clicking** on each mode's icon.

While working with components, this menu lets you choose to show **Display** or **Current** Operators. These same options are available at the top of the Scene view when working with the **Edit** node.

This menu is different at the object level where it includes some filters for different kind of objects as well as options for more easily selecting materials, constraints and Digital Assets.



○ Points 2
◉ Edges 3
○ Primitives 4
○ Vertices 5
○ Breakpoints

☐ Select Groups or Connected Geometry 9
☐ Select Whole Geometry
☐ Area Select Visible Geometry only Shift+V
☐ Area Select Fully Contained Geometry Shift+C

◉ Select Front and Back Facing Ctrl+Shift+B
○ Select Front Facing Only Ctrl+Shift+B
○ Select Back Facing Only Ctrl+Shift+B

○ Show Display Operator
◉ Show Current Operator

3D Connected Geometry
UV Connected Geometry
Geometry Groups
Cut Geometry for UV Layout
Alembic Paths
Name Attribute

SELECTION OPTIONS

Edit | Components You can choose which components you want to work with from this collection of buttons. Here edge selection has been chosen.

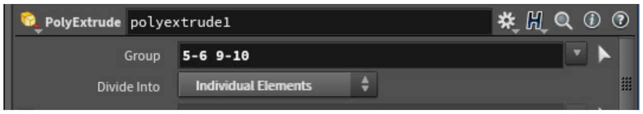
Select Tool - The **Select** tool lets you translate the selection. Using the handle shown in the Scene view. You can access it with the S hotkey.

Selection Types - You can use this top bar to change your type of selection. You can choose from Box, Lasso, Brush or Laser. There are also some filter options.

Edge Loop - To select an edge loop you can **double-click** while selecting edges. You can also select point loops or primitive loops using the same technique. To select partial loops, select one edge then press **A** and then an end edge. Again this works with points and primitives.

HOW SELECTIONS ARE USED BY TOOLS

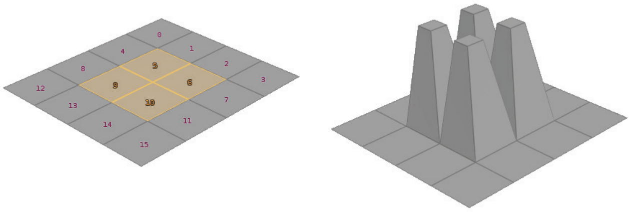
When you make a selection in the viewport then use a tool, a new node is created and the selected points, primitives or edges are listed in the node's **Group** parameter.


PolyExtrude polyextrude1

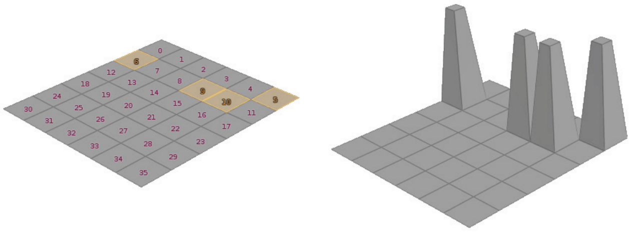
Group 5-6 9-10

Divide Into Individual Elements

For example here we see primitives **5, 6, 9 and 10** are being used by a *polyextrude* node. You can see them listed in the Group node and then used to extrude the faces.



If you were to change the topology of the incoming geometry node then there might be more faces and the extrusions will have moved to a different location. This may not be what you want and you may need to reselect faces.



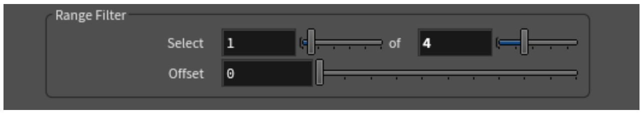
To do this, you can select *polyextrude*, press **Enter** to go to the **Handle** tool and press ` to go into re-select mode. Select new primitives then press **Enter** and your new selection will be used in the **Group** parameter.

SELECT ALL

If you want to select all of the primitives on the incoming shape, then you would leave the **Group** parameter blank. Even if the topology of the incoming geometry changes, all the faces will be operated on by the node. Using **Select All** [N] in the viewport will usually ensure that this field remains blank when a tool is used. **Select None** [Shift N] will deselect everything.

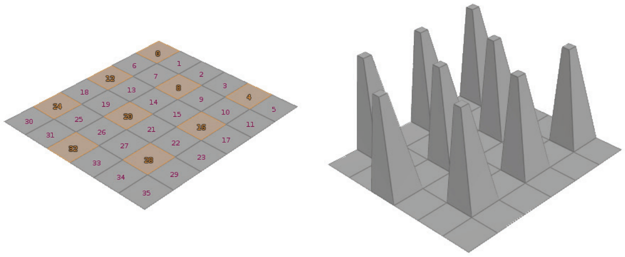
THE GROUP NODES

Groups let you refer to a defined selection of points, vertices, polygons, or edges by **name**. You can define a group interactively, by selecting components in the viewer, or mathematically, using ranges or an expression. The group name can then be assigned to the **Group** parameter instead of using point or primitive numbers.


Range Filter

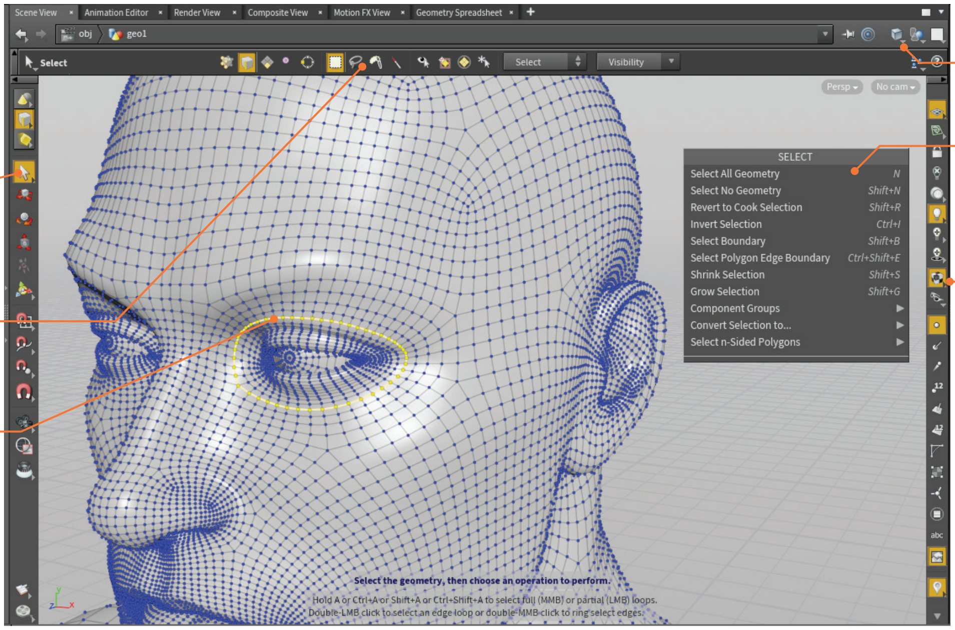
Select 1 of 4

Offset 0



Here are some **Group** nodes for you to choose from:

- **Group Create** - You can use an interactive selection, a bounding box, face normal direction or edge angles to populate the group.
- **Group by Range** - This lets you choose a range and a simple pattern to populate the group.
- **Group Expression** - With this node you can use a vex expression to define the membership of the group.
- **Group Paint** - This node lets you use an interactive paint interface to choose the geometry for the group.



Select the geometry, then choose an operation to perform.

Hold A or Ctrl+A or Shift+A or Ctrl+Shift+A to select full (MMB) or partial (LMB) loops.
Double-LMB click to select an edge loop or double-MMB click to ring select edges.

Shading Options - The shading options determine what you see in the Scene view. In this case, we are using Smooth Wired Shading.

RMB Menu - While in the Select tool, this menu gives you access to selection options such as inverted selections, boundaries or growing and shrinking your selection.

Display Filter - This filter lets you turn off things you don't need such as bones, null objects, lights or cameras to let you focus on the work at hand.

Display Options - While the selection modes show you edges or points when you have nothing selected, you should use these options if you want to see the particular item, such as points shown here, all the time.

Transform and Edit

From basic transformation tools for objects, to the pose tool for animation rigs and the edit node for reshaping geometry, there are a number of different tools that let you use interactive handles in the viewport. In Houdini, these handles are closely to the node you are working with.

TRANSFORM TOOLS

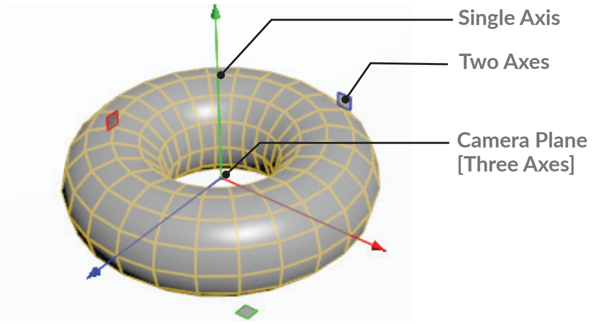
The transform tools give you handles that you can use to manipulate objects or reshape geometry. When you transform objects, the parameters at the object level are updated to reflect your changes.

Move	T
Rotate	R
Scale	E
Pose	Ctrl-R

While using these tools, you can also re-select by **pressing and holding S**, making a new selection then **releasing S** and continuing to transform.

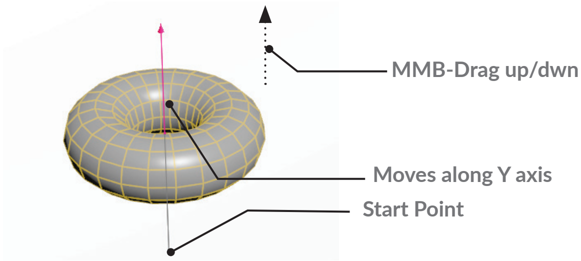
TRANSFORM HANDLES

When you are working with a **Move** handle you can work with a single axis, two axes or move along the camera plane using the center. **Rotate** and **Scale** handles offer similar controls.



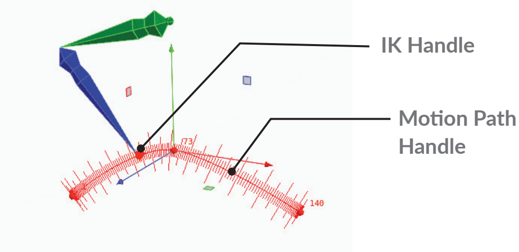
MMB TRANSLATION

If you don't want to click directly on handles, you can use the **Middle Mouse Button** in open space combined with a drag along one of the axes to translate or scale in that direction. **MMB** dragging with a rotate handle rotates in all directions.



POSE TOOL

When you are animating, you can use the Pose tool to work with bones and to display motion path handles that reveal the motion of an object through space. You can then use tangent handles and keyframe points to modify the motion in the viewport.



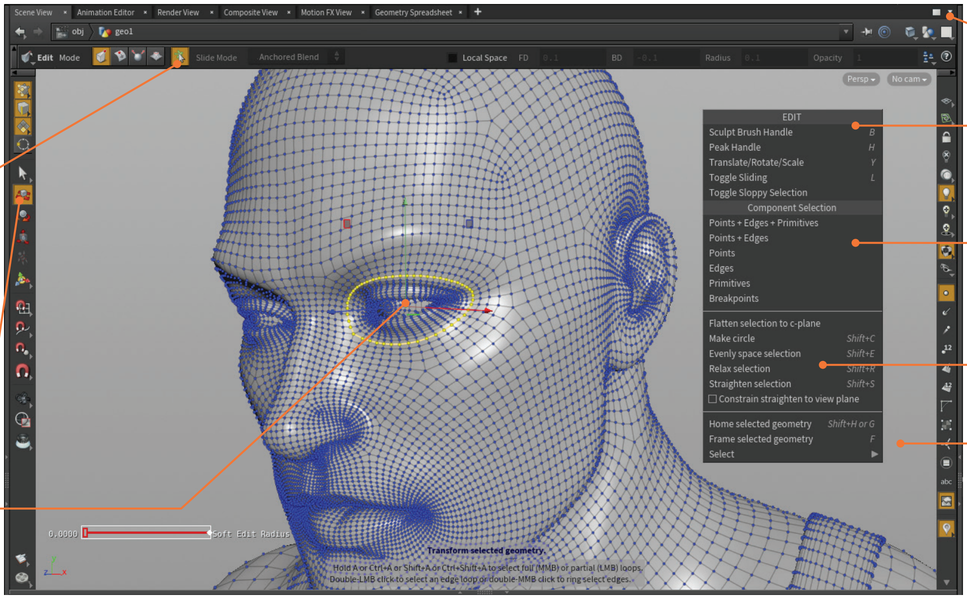
EDITING GEOMETRY

Edit | Components You can choose which components you want to work with from this collection of buttons.

Sloppy Selection - Three of the component buttons are selected at the same time because the Edit node is using Sloppy selection which makes all of them available at the same time for a more fluid selection process.

Move Tool - The **Move** tool lets you translate the selection using the Scene View handle.

Handle - This handle lets you move along one axis using the lines or two axes using the square dots. **RMB-click** on the Handle to access the handle options.



Transform selected geometry.

Hold A or Ctrl+A or Shift+A or Ctrl+Shift+A to select full (MMB) or partial (LMB) loops.
Double-LMB click to select an edge loop or double-MMB click to ring select edges.

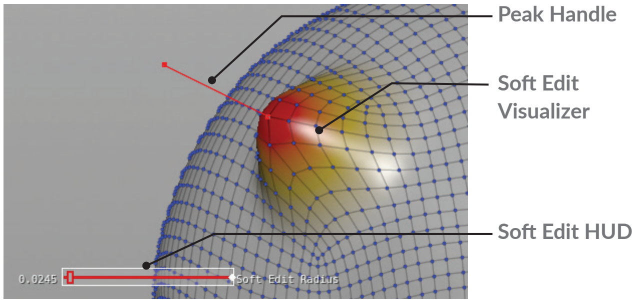
EDIT NODE

If you try to move geometry components then an **Edit node** is placed down to accept your transformations. In addition to transforming the geometry, you can slide on surface, work perpendicular to the normals or sculpt the surface.

Edit	T/R/E
Slide on Surface	L
Peak	H
Sculpt	B

SOFT FALLOFF

When you are transforming points, you can use the **Soft Edit Radius** to create a falloff. There is a visualizer that is evoked to show you where the falloff is occurring on the surface.



EDIT OPTIONS

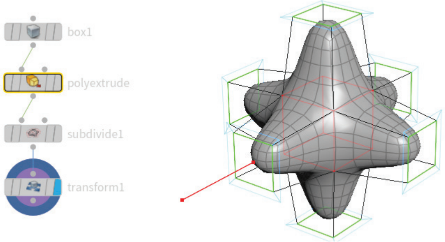
If you **RMB-click** while in the Edit node, you can access some options for transforming your selection in specific ways. You can make a circle or straighten the selection. These options work with points and edges but not on primitives.

Make Circle	Shift-C
Evenly Space Selection	Shift-E
Relax Selection	Shift-R
Straighten Selection	Shift-S

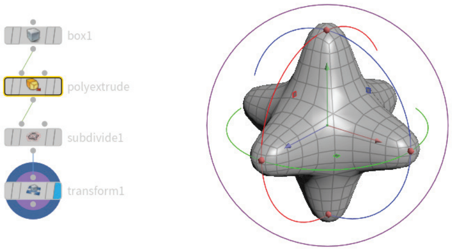
HANDLE TOOL

After using a shelf tool, you often find yourself in the **Handle** tool. Or you can select a node in the network and press the **Enter** key in the Scene view to go into the **Handle** tool. This brings up a handle which focuses on the specific parameters for the selected node such as the **distance** parameter on a *polyextrude* node.

Show Current Operator - By default when you select a node other than the display node, it becomes the current node and you get a wireframe display of the geometry. You can then use the handle to manipulate this intermediate node while evaluating the results on the shaded surface.



Show Display Operator - Another option is to always show the Display Operator. In this case, selecting a node in the chain does not show the wireframe and the handles stay focused on the display node.



You can change parameters in the parameter pane for the current node but the handles will continue to work with the parameters on the display node.

Shading Options - The shading options determine what you see in the Scene view. In this case, we are using Smooth Wired Shading.

RMB Menu - This menu gives you access to Edit tool options such as which type of edit you want to focus on. This information is also available on the top bar of the Scene view.

Component Selection - You can also select the component type using this menu. This offers you the same options you would find on the toolbar.

Display Options - You can use this menu to **Make circles** and **Straighten** the selection.

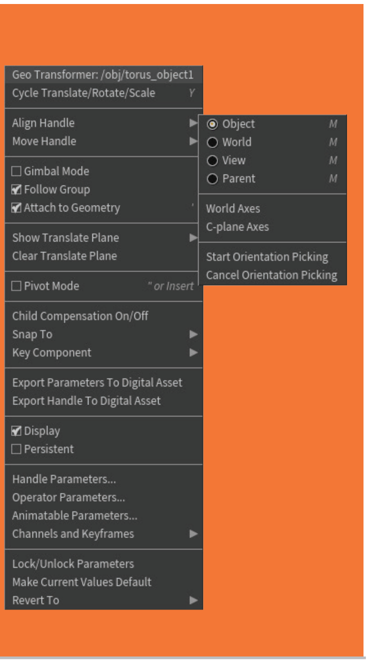
Selection Options - The selection options found in the RMB of the select tool are available here in this submenu to support selection while you are modeling.

HANDLE OPTIONS

All handles have a menu that can be accessed by **RMB-clicking** on any part of the handle.

This menu gives you options for aligning the handle, detaching it from the parameters of the node, pivot mode and more. You can use these options to customize how the handle works.

You can also keyframe parameters on a handle and promote all parts of the handle to a Digital Asset. By promoting the parameter the handle will be accessible at the asset level.



Modeling Tools

Houdini has a lot of tools for creating, shaping and deforming geometry to achieve a desired look. Here are just a few of the many tools you will use on a regular basis when building models in the geometry, or SOP, context of Houdini.

CREATION

To start creating geometry, you can start with some basic shapes or draw a curve. In each case you get an object with a geometry/SOP node inside with the tool's name. You can access these on the **Create** shelf or in a radial menu.

Primitives - Houdini includes Box, Sphere, Tube and Torus primitive shapes along with a variety of Platonic solids.

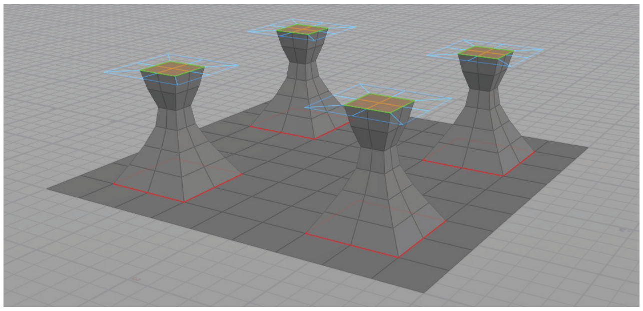
Grid - The grid tool offers a great starting point for a wide range of models. You can set its shape and size at the geometry level.

Curve - Draw a curve by laying down control points then create a poly, NURBS or Bezier curve.

POLYGON MODELING

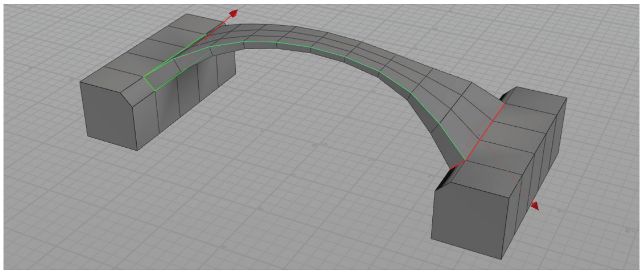
Polygons are one of the most popular geometry types especially in video game projects where they are mandatory. Houdini has a comprehensive set of poly modeling tools which you can use to develop your models.

PolyExtrude - Push or pull on one or more polygons to reshape the geometry. Control the extrusion profile to get a wide variety of shapes.



PolyBevel - Bevel selected edges to create chamfered or rounded bevels. You can often use the output group from a previous node such as Polyextrude or Boolean to automatically find the right edges.

PolyBridge - Connect two sets of polygons with control over the shape of the bridge.



PolySplit/Edge Loop/Knife - These tools let you split polygons to add more detail to your model.

PolyExpand 2D - Take curves and edges that sit on a 2D plane and creates geometry based on a desired offset value.

PolyReduce - Create different levels of detail by reducing the number of polygons while preserving quads and UVs.

UTILITY NODES

Because of Houdini's procedural nature, modeling actions like copy, clip and mirror create nodes in your network. This can makes it easier to go back and make changes later on.

Clip - Cut your model based on a clipping plane. You can set the direction of the clip and whether you keep one half, the other or both.

Mirror - Mirror the geometry based on a clipping plane. There is an option for fusing the points after mirroring to connect the geometry.

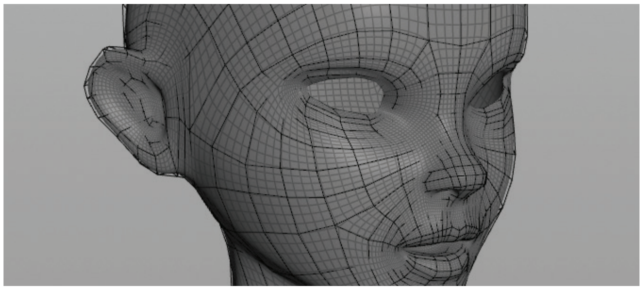
Copy and Transform- This node will let you create multiple copies based on transformation values.

Blast - This node lets you delete polygons from your model. You can choose to remove or keep the selected polygons. If you press the **Delete** key when a polygon is selected it will be blasted.

Dissolve - This view lets you remove edges without breaking up the surrounding geometry. Pressing the Delete key when an edge is selected will dissolve it.

SUBDIVISION SURFACE MODELING

In Houdini, you can model with polygons then **display** and **render** them as subdivision surfaces using options found on the **Render** tab on the object's parameter pane. You can also create a **Subdivide** node at the geometry level to add polygons to give you a more detailed topology to work with.



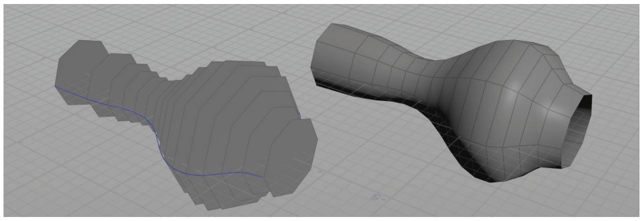
SURFACING TOOLS

There are tools in Houdini that will take profile curves and build surfaces. Those input curves can be either polygonal or NURBS or a mixture of the two.

Revolve- Create geometry by revolving a profile curve around an axis. There is a handle available for tweaking the results.

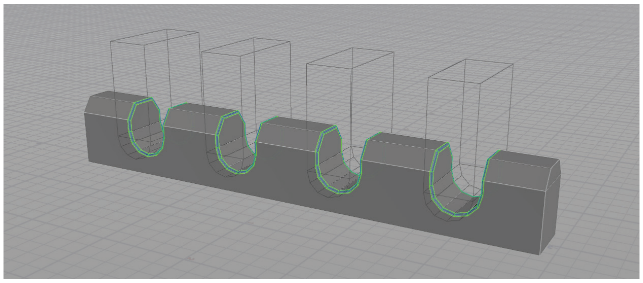
Skin - Take a series of profile curves and turn them into a surface.

Rails - Copy one or more profile curves along two or more rail curves, then Skin the results to get a surface.



BOOLEANS

Subtract, **Union** or **Intersect** geometry using the Boolean tool. This node can handle very complex topologies and can be used to break up a surface for destructions using rigid body dynamics. This often creates more realistic results compared to the Voronoi-based **Shatter** node.

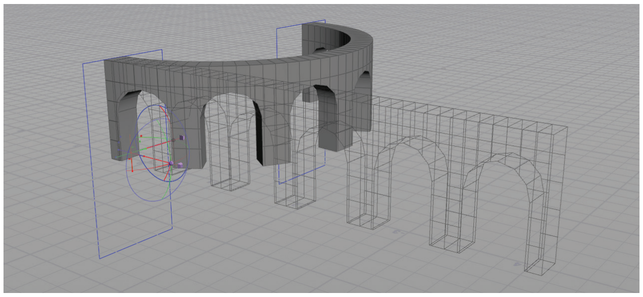


The Boolean tool can create output groups that you can use to feed other nodes such as the **Polybevel** node. This way any updates you make to the boolean will update properly when it feeds into the second node.

DEFORM TOOLS

While you can shape your geometry by editing points directly, there are times that you need a more generalized approach. The following nodes provide options for shaping your geometry procedurally.

Bend - This node lets you set a capture range and direction then bend, twist, taper and squish the encompassed geometry.



Lattice - This builds a lattice around your geometry then lets you edit points on the cage to reshape it. You can also use a custom cage.

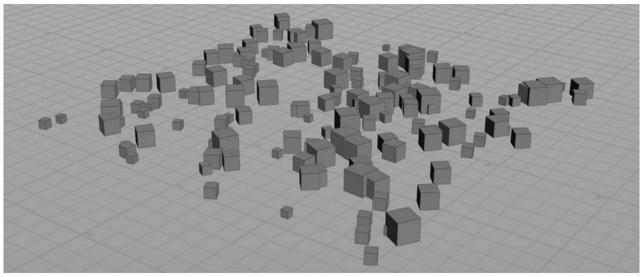
Mountain - Apply a noise function to deform the surface to create a random result. The points are actually being moved with this node.

Ripple - This node creates a ripple shape in your geometry.

Waves - This node adds noise functions to create a wave-like pattern that animates over time. Perfect for creating realistic oceans.

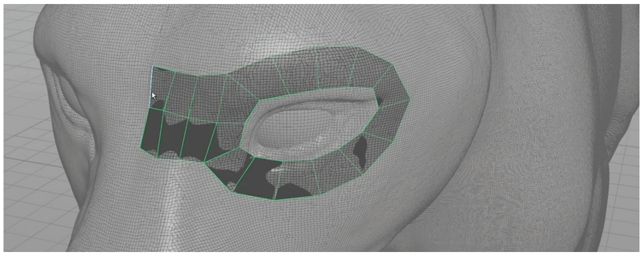
COPY TO POINTS & SCATTER

A typical Houdini workflow is to scatter points on a surface then to copy objects to those points. Attributes for scaling and rotating the objects can then be applied to create a more organic result. This is often used to populate landscapes with trees and rocks.



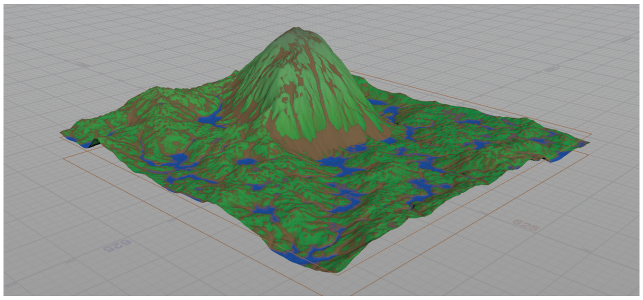
TOPOBUILD

Houdini has a topobuild node that lets you draw polygons directly onto high-resolution geometry that you either scanned or created in an application such as Pixologic's Z-Brush. You can create a cleaner topology for animation then bake the details from the original model into a normal map.



TERRAIN TOOLS

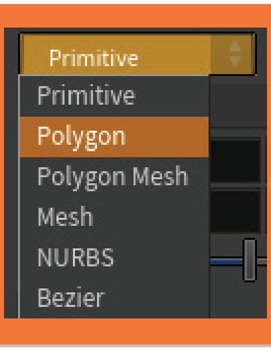
A wide variety of Terrain building tools are available that use **heightfields** to add noise, landforms and even cause erosion. The results can be used to render and can act as collision surfaces for different kinds of dynamic simulations.



GEOMETRY TYPES

Houdini supports a number of different geometry types including **Primitives**, **Polygons**, **NURBS** and **Beziers**. You can **Convert** back and forth between them and you can have more than one geometry type merged together in a single object.

Polygons models can be set up to display and render as **Subdivision Surfaces** using **PIXAR's OpenSubdiv** standard. Both Subdivisions and NURBS will render in Mantra as perfectly smooth without relying on any tessellation settings.



Time and Motion

Animation involves changes happening over time. Whether we are talking about the position of an object or its overall shape, you are animating if moving forward in time creates a visible change. Houdini has all the necessary tools for a keyframe-based workflow in addition to a procedural node-based workflow called **CHOPs** for more advanced manipulation of time and motion.

SETTING KEYFRAMES

Keyframes are set whenever you want to set a specific parameter value at a specific point in time. As these keyframe values change, the objects in your scene will become animated. You can then use animation curves created in-between the keys to determine the quality of the motion.

Here are a few main hotkeys used to set keyframes on your objects while working in the scene view:

Set Keyframe	K
Toggle AutoKey	Alt + K
Key Handle	Ctrl + K
Key Translate	Shift + T
Key Rotate	Shift +R
Key Scale	Shift + E

You can also set keyframes in the parameter pane by pressing the **Alt** key and clicking on either a parameter name or parameter field or by **RMB-clicking** on a parameter and selecting **Channels and Keyframes > Set Keyframe**. This lets you set keyframes one parameter at a time

THE TIMELINE

At the bottom of Houdini's main workspace is the Timeline which lets you see your frame range and set your current time. Houdini measures time in frames with a default frame rate of 24 frames per second. The Timeline can also be used to edit keyframes.

At the left are the playback controls. Here are some hotkeys for quickly setting up playback and moving through time.

Play Forward	↑
Play Back	↓
Next Frame	→
Previous Frame	←
First Frame	Ctrl + ↑
Next Scoped Key	Ctrl + →
Previous Scoped Key	Ctrl + ←

You can also **RMB-click** on the frame range in the timeline to access options such as **Cut**, **Copy** and **Paste** of keys along with special pasting such as **Replace**, **Cycle**, **Repeat** and **Stretch**. All these options also have their own hotkeys which you can find on the menu. You can accomplish a lot working in the Timeline before moving to the **Animation Editor**.

SCOPED CHANNELS

When you set a keyframe or look at animation curves in the Animation editor, you are working with what are called **Scoped channels** with a channel being an animated parameter. If you select an object then its animated channels are scoped and existing keyframes loaded into the timeline or channel editor. If you deselect the object then the channels will no longer be scoped unless you pin them in the channel list.

CHANNEL LIST

The **Channel List** can be used to scope channels for objects that aren't necessarily selected. You can use the list to create groups of channels so that they can be more easily accessed. You can also use the list to pin channels so that they are scoped even if you select different objects. This is a useful pane if you are setting keyframes on characters.

FLIPBOOK

As you animate your scene, you will want to preview the results in motion. The **Flipbook** tool found on the toolbar on the left side of the Scene view lets you capture frames from the viewport then playback the results as a movie in realtime.

TIMELINE

The timeline is where you scrub through time and create and edit keyframes. The timeline can also be used to make quick edits, while the Animation editor is used for more comprehensive refinements.

Playback Controls

These controls let you quickly play, pause, and move to next key.

Current Time

The current time is shown in the field and on the black marker in the frame range. The marker can be used to scrub through the timeline.

Frame Range

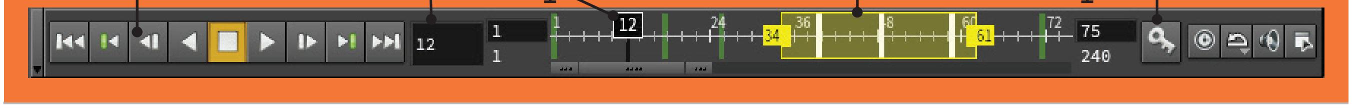
The overall range is defined by the Animation controls button at the far left. Then the visible range can be reduced using the handles at the bottom of the range.

Edit Keys

As you set keys, your scoped channels are shown in the timeline. Press the **shift** key to select keys with the **LMB** and then edit the keys with the **MMB**.

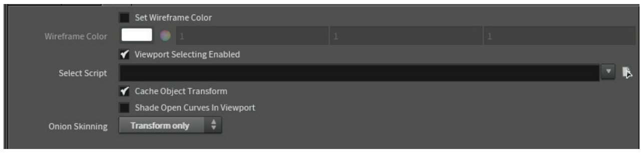
Auto Key

The **Auto Key** button will automatically set keyframes when you change a parameter value. This works well for a straight-ahead animation style.



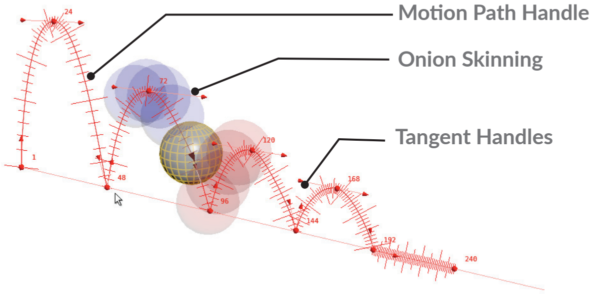
ONION SKINNING

Onion Skinning lets you display a ghosted version of your object on the frames before and after the current frame. Turn it on using the **Misc** tab on your object while the onion skinning options such as Frames Before, Frame After and Frame Increment can be found in the viewport's Display panel [**d**] under the **Scene** tab.


□ Set Wireframe Color
Wireframe Color 1 1 1
☑ Viewport Selecting Enabled
Select Script
☑ Cache Object Transform
□ Shade Open Curves In Viewport
Onion Skinning: Transform only

MOTION PATH HANDLES

When using the **Pose** tool to animate, you can click on the **Motion Path** option to bring up handles that let you see the animation of the selected object over time. You can also use the handle to manipulate the shape of the motion.



SCOPED CHANNELS

Scoped channels are loaded into the animation editor where they are represented as keyframes and animation curves, as a spreadsheet or as a dope sheet. The keyframes can be selected and edited and the curves can be shaped using tangent handles. The curves define the motion in-between the keyframes and play a key role in defining the quality of the motion.

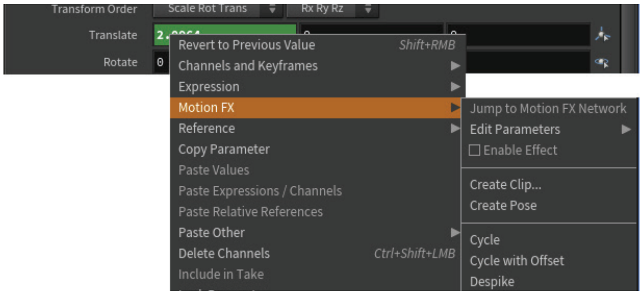
While working with channels you can change which keys you are viewing using these hotkeys:

View All/Home	H
Pan	MMB
Zoom	RMB

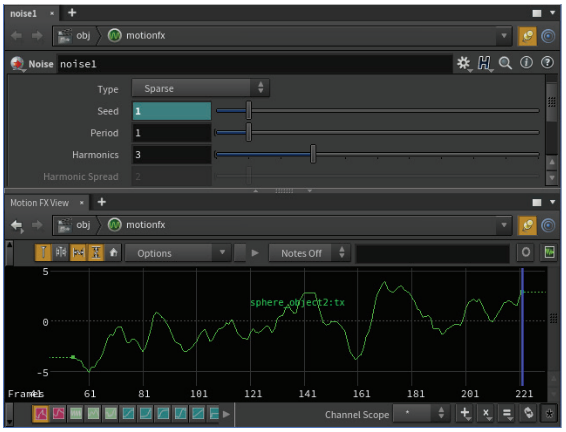
MOTION FX

While keyframes and animation curves are stored in the parameters of your nodes, you can also use **channel operators (CHOPs)** for a more procedural node-based approach to working with motion.

The easiest way to create a channel operator is using **Motion FX** which can be found when you **RMB-click** on a parameter and select from the **Motion Effects** sub-menu. You can also apply these effects to channel groups using the Channel List.



Motion FX can be applied to keyframed motion which is extracted and stored in a **Channel CHOP**. You can then apply effects such as cycle, noise, smooth, limit or lag to the existing motion. On the **Constraints** shelf, you have tools which let you have one parameter either lag or jiggle behind another.


Noise noise1
Type: Sparse
Seed: 1
Period: 1
Harmonics: 3
Harmonic Spread: 2

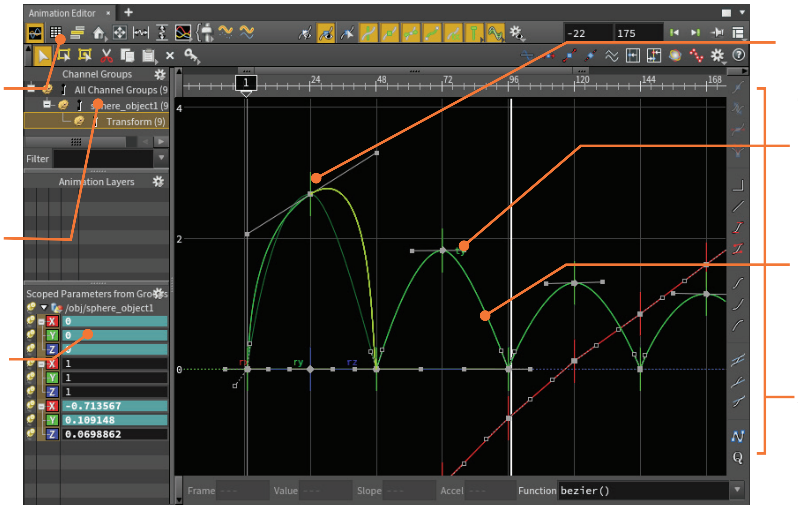
This non-linear approach to working with motion offers a different way of working that can be very flexible.

ANIMATION EDITOR

Editor Options - This editor can be shown as a graph, a dope sheet or a table. You can choose the type of editor which is most appropriate.

Channel List - This area of the graph shows you channel groups which make it easier to select and pin your scoped channels.

Scoped Channels - Channels that have been scoped show up in this area. You can then select the names of channels you want to see in the graph.


Scoped Parameters from Groups
/obj/sphere_object1
X 0
Y 0
Z 0
X 1
Y 1
Z 1
X -0.713567
Y 0.109148
Z 0.0698862
Frame --- Value --- Slope --- Accel --- Function bezier()

Key Handles - Move the key in time using the vertical bar and edit its value using the box.

Tangent Handles - They define the tangents coming in and out of a keyframe and they help you shape the surrounding animation curves.

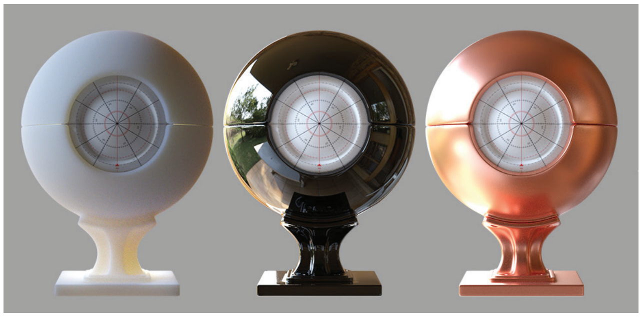
Curve - The animation curves define the motion in-between the keyframes, which defines the quality of the motion.

Curve Functions - They let you change the shape of the in-between curves and can determine if you are speeding up or slowing down.

Shaders & Materials

To render objects in your scene, you must assign materials, also known as shaders, to your geometry. In Houdini, these materials and shaders are created in the mat and vex builder networks. The ability to build up your materials using nodes is a powerful tool when defining the look of a shot.

Houdini organizes different types of nodes into network types and for materials you will use the `/mat` network. This is where you can set up VEX operators or VOPs. You can either work with pre-built shaders or build your own by using a custom network of VOP nodes.



CREATING AND ASSIGNING MATERIALS

You can add materials using the **Material Palette** then assign them to objects by clicking and dragging. This pane has a workspace for managing the materials in your scene which is organized into tabs that represent different subnetworks where material nodes can be created. This pane gives you a flattened view of all of the materials in your scene, so they can easily be found in whichever subnetwork they are located.

THE MATERIAL GALLERY

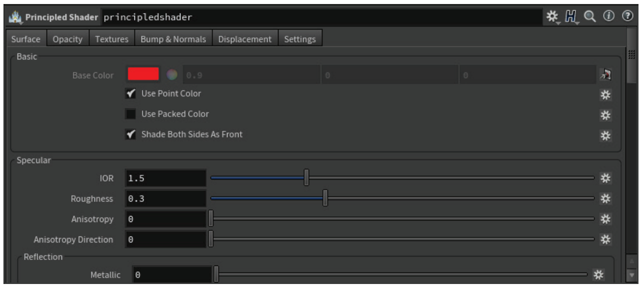
On the left side of the palette is the gallery where you will find pre-built materials that you can assign to your objects. You can assign a material directly from the gallery to the viewport or you can move it to area on the right and assign it later. As you learn how to create your own materials, you can build custom galleries and add them to this list.

PRINCIPLED SHADER

In the Material Palette, you will find the principled shader, a material based on the Disney "principled" BRDF by Brent Burley. This shader is "principled" rather than physical in order to make it easier for artists to work with.

The **Principled Shader** has been pre-built to include the ability to assign textures directly and set up with other features such as bumps, normals, displacements and more. The texture maps you assign will be displayed in the viewport and you can achieve a wide variety of looks right out of the box.

You can extend this material by feeding it with other VOPs but that isn't necessary in all cases. Many of the materials found in the gallery are variations on this shader.



Principled Shader principledshader

Surface | Opacity | Textures | Bump & Normals | Displacement | Settings

Basic

Base Color 0.9 0 0

☑ Use Point Color

☐ Use Packed Color

☑ Shade Both Sides As Front

Specular

IOR 1.5

Roughness 0.3

Anisotropy 0

Anisotropy Direction 0

Reflection

Metallic 0

PRINCIPLED SHADER CORE

The **Principled Shader Core** node sits inside the **Principled Shader** and contains the main features of the shading model but doesn't have all of the texturing features built-in.

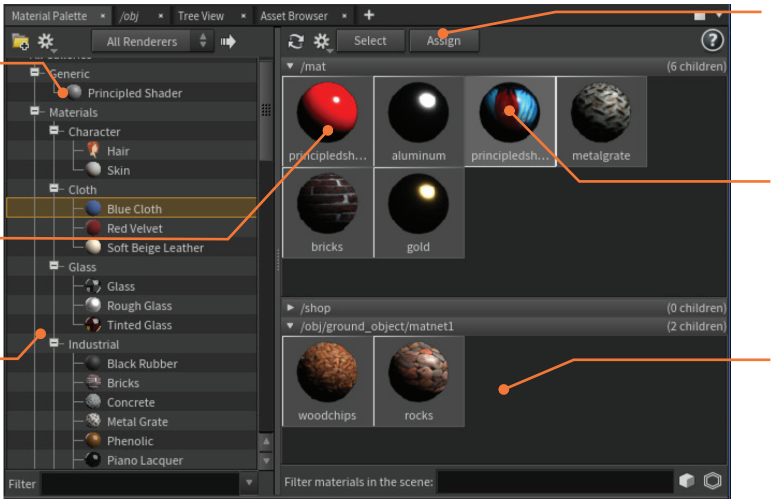
To build a robust shader from scratch using this node, you would need to add VOP nodes using Houdini's shader building tools. You can accomplish this by either wiring together nodes in the Node view or adding them using the Shader FX menus.

MATERIAL PALETTE

Material in Gallery - The Materials listed here are saved to disk in a gallery file. You can drag these into the scene area found on the right or onto objects in the viewport to assign them.

Material in Scene - This is where you find materials that are part of your scene file. They can be assigned to objects by dragging from here into the viewport.

Material Gallery - There are some default galleries that come with Houdini. You can build your own and add them to this area for your own projects.



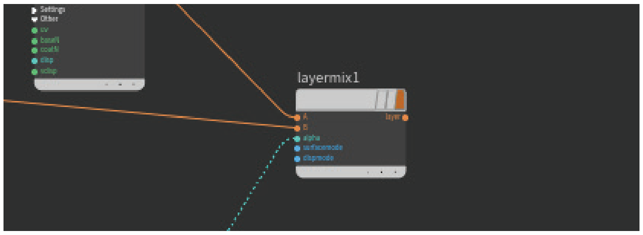
Assign Material - If you select your object in the scene and the material in the palette then you can use this button to assign the material.

Double Click to Edit - If you double-click on any of these materials, you will jump to a Node View where you can make edits at the /mat level.

Material Subnets - Sometimes a material subnet is used to keep materials relative to the geometry they are being assigned to. All of the subnets in your scene will be listed here.

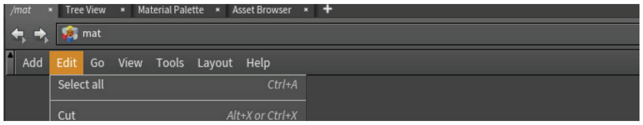
LAYERING MATERIALS

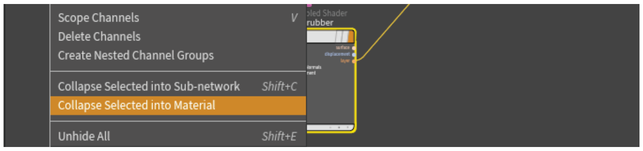
At the `/mat` level, you can layer materials to create a unique look for your object. Using a **Layer Mix** node, you can combine two different materials into a single look. For instance, a shiny metal material and a matte rust material can be layered using this technique. You can then texture an alpha channel and you can choose to mix the surface, the displacement or both.



MATERIAL BUILDER

If you want to turn your layered nodes into a new material then you can select them and choose **Edit > Collapse Selected into Material**. This puts the nodes inside a **material builder** where you can continue to tweak it. At this level, there are *output* and *collect* nodes to make the network work efficiently.



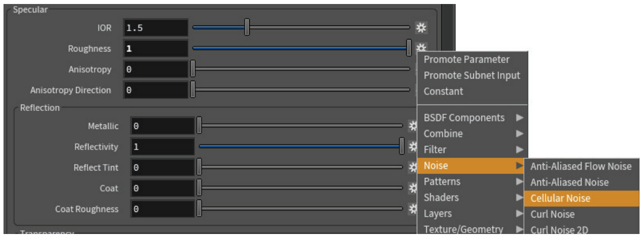

Scope Channels
Delete Channels
Create Nested Channel Groups
Collapse Selected into Sub-network Shift+C
Collapse Selected into Material
Unhide All Shift+E
rubber

MATERIALS AS DIGITAL ASSETS

You can make materials even more efficient by saving them out as **Houdini Digital Assets**. In the **Asset Properties** pane, you can go to the **Save** tab and choose **Save Cached Code** so that the material is pre-compiled when you render with Mantra. HDA's make it easier for you to share materials with your team.

SHADER FX MENU

While working with material VOPs, you can add nodes in the Node view and wire them in or you can access the **Shader FX Menu** by clicking on the icon at the far right of each parameter. This menu lets you focus on the parameter that you want to work with and choose the appropriate node in context.



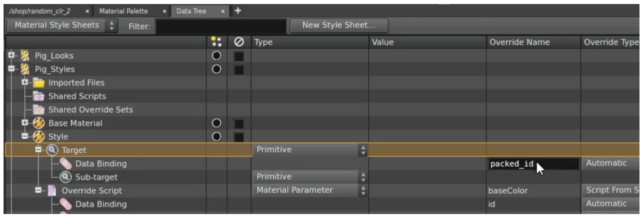
In the Parameter Pane, you can see what type of connection each has by looking at the icon on the far right:

- No Connection
- Parameter Node
- Connected with other Nodes
- Hidden Connection

OTHER WAYS TO ASSIGN MATERIALS

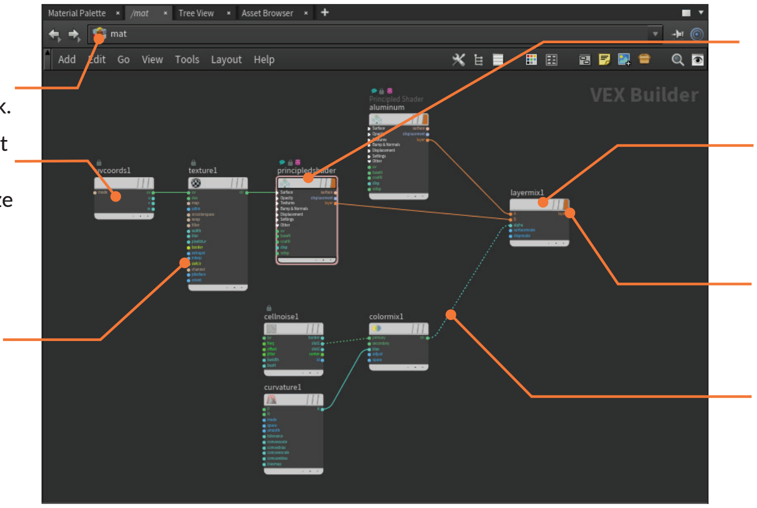
Materials can be assigned to the object where it will be listed under the **Render** tab. There is also a **Material SOP** which allows you to assign materials to part of a model at the geometry level. You will often use groups to make it easier to assign materials this way.

The **Data Tree** panel gives you access to **Material stylesheets** which you can use to assign materials and textures to large groups of objects or imported Alembic files. This is a production-level tool that makes it more efficient for artists to work with large shots. Refer to the Houdini docs for more information.



			Type	Value	Override Name	Override Type
Pig_Looks	○	■				
Pig_Styles	○	■				
Imported Files						
Shared Scripts						
Shared Override Sets						
Base Material	○	■				
Style	○	■				
Target			Primitive			
Data Binding					packed_id	Automatic
Sub-target			Primitive			
Override Script			Material Parameter		baseColor	Script From S
Data Binding					id	Automatic

SHADER BUILDING



Node Path - This will confirm your current path and help you navigate into and out of the material network.

VOP Nodes - In the material context you can start with material nodes then wire in VOP nodes to customize the texturing of the material. Once you are finished you can choose to collapse all of this back into a single material.

Node Connectors - You can add nodes to this area using the Shader FX menu which is can be accessed by **MMB-clicking** on the dot. **RMB-clicking** on the dot gives you a full node menu.

Principled Material - This is a typical material which could be assigned on its own or fed into the Layer mix.

Layer Mix - The Layer output on the materials can be fed into this mix node. You can assign this to your geometry.

Material Flag- If you want the layer mix to show up in the material palette then check this flag.

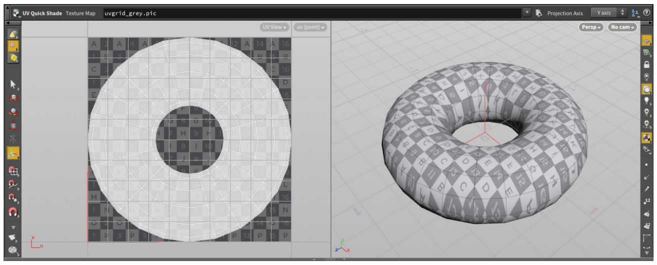
Alpha - Here the layer mix node's alpha is being fed by VOP nodes to create the alpha mask for the two layered materials.

UVs & Textures

To make 2D maps fit properly onto 3D objects, UV coordinates can be used to define a flattened view of your geometry. When you first create any geometry it will not have any UVs. Even primitive objects do not have any built-in UVs. This means that you will need to add them at the geometry level using one or more SOP nodes.

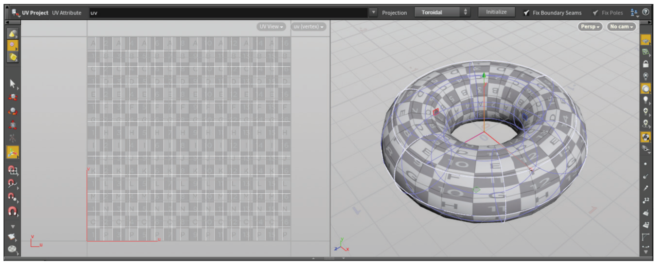
UV QUICKSHADE

When you first start adding UVs to your geometry a **UV Quickshade node** will set up some projected UVs, which you will probably replace with a subsequent UV node, as well as assign a **UVgrid texture** that you can use to evaluate your work. Later when you want to assign a real texture map to the object, you may want to bypass or delete this node to hide the UV grid.


UV Quick Shade Texture Map uvgrid_grey.pic Projection Axis Y axis

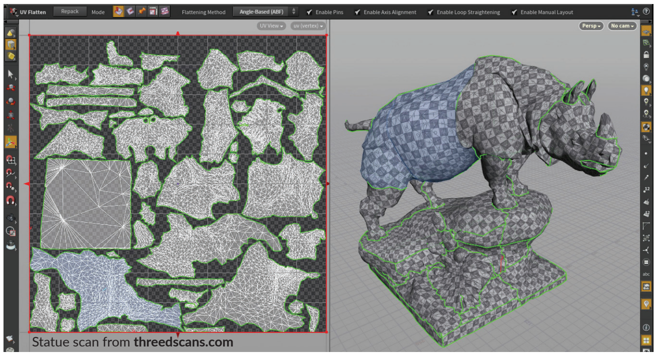
UV PROJECT

Assign UVs using one of several projection techniques. Once you choose your projection type you can then "initialize" the projection to match your object. This may invert the UVs and you may need to a transform value such as -90 instead of 90 for rotate x. This method only works for the simplest cases and will cause distortion in models that have more organic shapes.


UV Project UV Attribute uv Projection Toroidal Initialize ✓ Fix Boundary Seams ✓ Fix Poles

UV FLATTEN

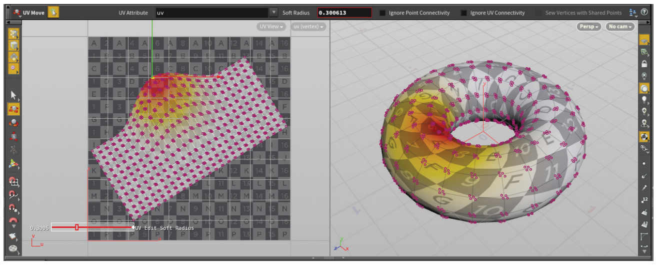
UV Flatten unwraps your geometry based on predefined boundaries based on selected edges or edge groups. The results can then be tweaked by pinning points in the **UV view** and adjusting the islands to get the look you want.


UV Flatten Repack Mode Flattening Method Angle-Based (ABF) ✓ Enable Pins ✓ Enable Axis Alignment ✓ Enable Loop Straightening ✓ Enable Manual Layout

Statue scan from **threedscans.com**

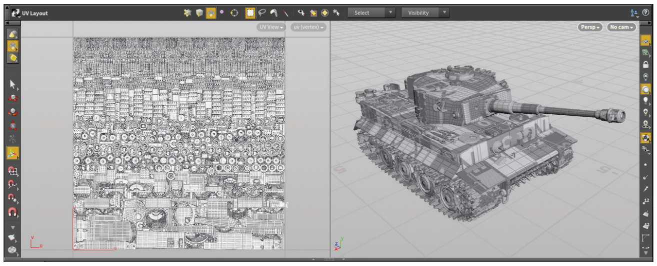
UV EDIT

To edit individual vertices or groups of vertices you will use either the **UVedit** or **UVtransform** nodes. The UVedit node lets you perform many edits using a single node while UVtransform allows one edit per node which can provide a more procedural result. You can use **Display > UV Distortion** from the **UV Viewport menu** to see whether you are editing too much.


UV Move UV Attribute uv Soft Radius 0.300613 Ignore Point Connectivity Ignore UV Connectivity Sew Vertices with Shared Points

UV LAYOUT

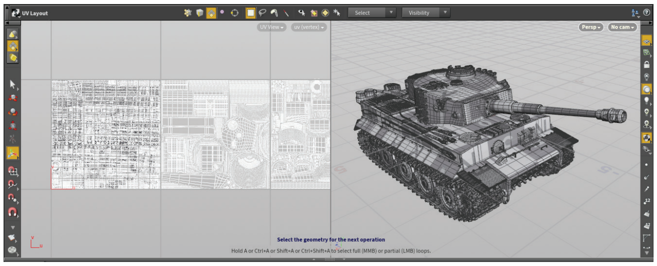
UV layout will let you create UV islands and pack them into UV space as efficiently as possible. This lets you maximize how much of a texture is being used on your geometry which is important when optimizing for both rendering and gameplay.



You can use the region handle to put a the UV layout into a particular part of your UV space. Subsequent layouts can then work around this layout using the **Pack Islands in Cavities** option.

UDIMS

In addition to working with a Single UV tile, you can use UDIMs to spread your UVs over many tiles. This technique lets you create more detailed texture maps because your UV islands are not packed in too tight. Properly numbered texture maps will then be assigned to the appropriate tile.


Select the geometry for the next operation

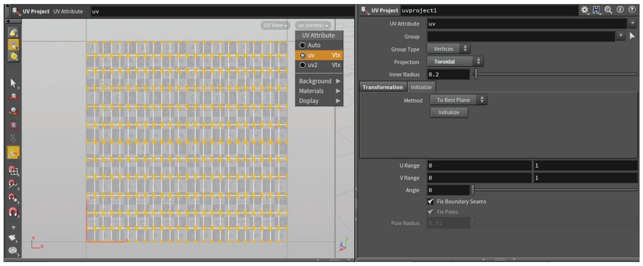
Hold A or Ctrl+A or Shift+A or Ctrl+Shift+A to select full (MMB) or partial (LMB) loops.

UV ATTRIBUTES

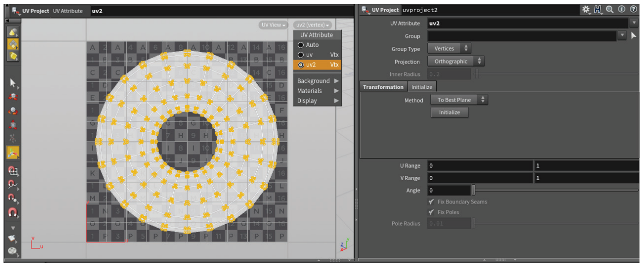
Earlier you learned about how attributes can be assigned to geometry and carry important information that can be used down the line. UVs are vertex attributes that are used to wrap texture maps around your model and are created by the UV SOP nodes that you apply at the geometry level.

These attributes are visualized in the UV viewport and analyzed in the geometry spreadsheet. These attributes work with various SOP nodes including the attribute wrangle node which lets advanced TDs manage their UVs using scripting.

It is possible to create more than one UV set on the same geometry. When you use the UV nodes, you can set the UV attribute that you want to work with. By default this is *uv* but you could create a *uv2* to create a second set. These different UV sets are used when you assign textures in VOPs so that different texture maps use different UV attributes to define their layout.



UV Project uvproject1
UV Attribute: uv
Group:
Group Type: Vertices
Projection: Toroidal
Inner Radius: 0.2
Transformation | Initialize
Method: To Best Plane
Initialize
U Range: 0 1
V Range: 0 1
Angle: 0
☑ Fix Boundary Seams
☑ Fix Poles
Pole Radius: 0.01

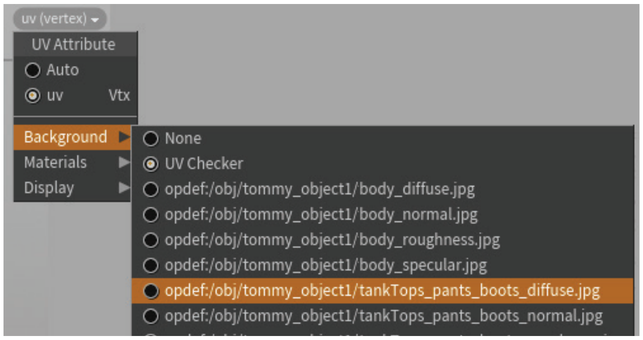


UV Project uvproject2
UV Attribute: uv2
Group:
Group Type: Vertices
Projection: Orthographic
Inner Radius: 0.2
Transformation | Initialize
Method: To Best Plane
Initialize
U Range: 0 1
V Range: 0 1
Angle: 0
☑ Fix Boundary Seams
☑ Fix Poles
Pole Radius: 0.01

In the two images shown here, the first uses a toroidal projection and is assigned to a *uv* **UV attribute** while the second uses a planar projection and is assigned to a *uv2* **UV attribute**. These UV attributes can have any name, for instance it could be *bob* instead of *uv2*.

UV VIEWPORT MENU

The UV viewport menu lets you display UVs based on the UV attribute. You can also use this menu to figure out the background image which can either be the default UV grid or a texture map pulled from an assigned material.

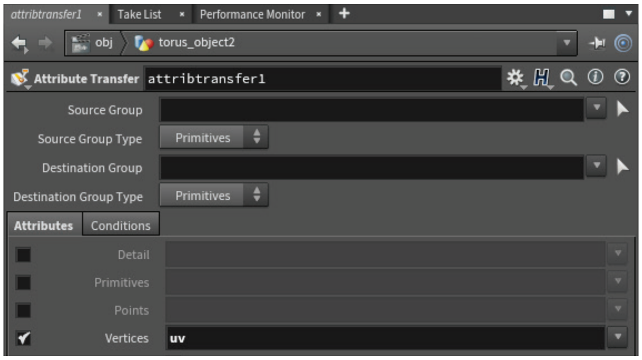


uv (vertex)
UV Attribute
○ Auto
● uv Vtx
Background ▶
Materials ▶
Display ▶
○ None
● UV Checker
○ opdef:/obj/tommy_object1/body_diffuse.jpg
○ opdef:/obj/tommy_object1/body_normal.jpg
○ opdef:/obj/tommy_object1/body_roughness.jpg
○ opdef:/obj/tommy_object1/body_specular.jpg
○ opdef:/obj/tommy_object1/tankTops_pants_boots_diffuse.jpg
○ opdef:/obj/tommy_object1/tankTops_pants_boots_normal.jpg

This menu also has display options for displaying **UV Overlap**, **UV Backfaces** and **UV Distortion**. These can be helpful when evaluating your UVs to decide if more tweaking is required.

ATTRIBUTE TRANSFER

One of the SOP nodes that can be used to manage attributes is **Attribute Transfer** which lets you take the UV attributes from one piece of geometry and transfer them based on proximity to another. This can be useful when the topology of a model has changed but you want to preserve some of the work you did creating UVs for the original model.



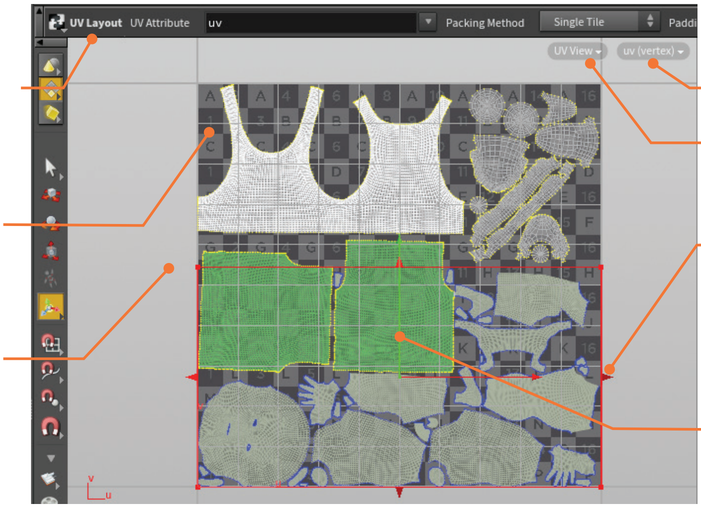
attribtransfer1 × | Take List × | Performance Monitor × | +
obj > torus_object2
Attribute Transfer attribtransfer1
Source Group:
Source Group Type: Primitives
Destination Group:
Destination Group Type: Primitives
Attributes | Conditions
□ Detail
□ Primitives
□ Points
☑ Vertices: uv

SCENE VIEW | UVS

Current Tool - At the top of the Scene View, you will find the selected node when the Handle tool is active.

Background - The background of the main tile can be set using options found in the UV menu.

Outside Main Tile - polygons positioned in the area outside the main tile would overlap the same area on the main tile because the texture repeats unless you are using UDIMS which cover more than one tile.



UV Layout UV Attribute: uv
Packing Method: Single Tile

UV menu - When you are in the UV view, this menu gives you a number of different options for working with UVs.

UV view - Use this menu to choose the UV view for this viewport.

Layout Handle - This handle is part of the UV Layout node and lets you focus the UVs to a certain part of the tile.

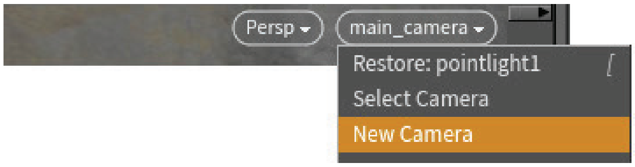
Pack Around - The UV layout will pack your UVs around any existing geometry that has already had its UVs set.

Rendering

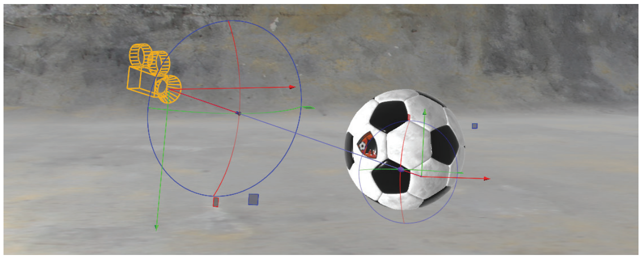
When you render a shot, you essentially take picture of your 3D objects using cameras and lights then save it out as an image or sequence of images. Game artists may also use rendering to render game cinematics or to bake textures from a high resolution to a low resolution object.

CAMERAS

Cameras can be accessed from the **Lights and Cameras** shelf or you can choose **New Camera** from the menu in the top left of the viewport. Cameras are placed into the scene as objects and have special handles to adjust its parameters.



There is a **lock camera to view** button in the Display Options bar that lets you use the **View tools** to reposition the camera. If you want to set up camera cuts then you would wire multiple camera nodes into a **Switcher** node then keyframe the **Switch Camera** parameter.



In addition to a regular cinematic camera, Houdini also has special cameras for different kinds of projects.

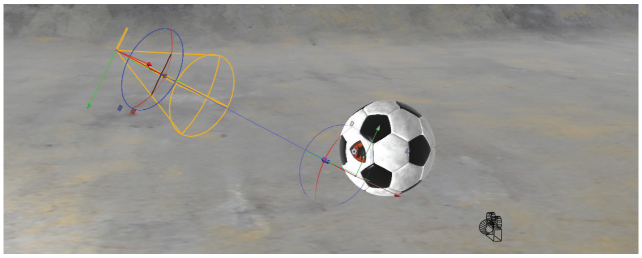
Stereo Camera - This camera provides a stereo rig with left/right cameras that has controls for parallax and interaxial distance.

VR Camera - For rendered VR projects, the VR camera provides the options necessary to .

Gamepad Camera- An interactive camera that lets you walk around your perspective view using an external gamepad controller.

LIGHTS

Lights can also be accessed from the shelf and have similar handles to help you position them. If you look through the light then you can also **lock light to view** and use the **View tools** to position the light.



There are a number of different light types in Houdini to choose from.

Environment Light- Casts light into the scene as if from a surrounding hemisphere or sphere.

Point Light - Emits light from a point in all directions and is similar to a light bulb.

Spot Light - Radiates a cone shaped beam of light from a point in a certain direction.

Distant Light- Emits parallel rays of light, which are similar to the rays of the sun.

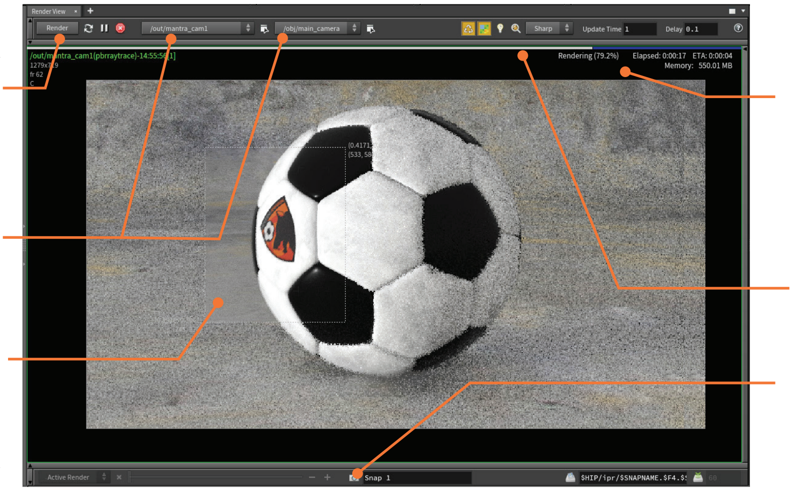
Area Light- Automatically distributes a number of light sources over a specified area. There are five area light shapes to choose from: Line, Tube, Grid, Disk an Sphere

Sky Light- This tool creates an Environment light node with the Sky Environment Map enabled.

Geometry Light- This object emits light into the scene using a geometry object's surface shader for the colors of the emitted light.

Portal Light- This tool creates an Environment light and sets the Portal geometry parameter to the geometry object you select.

RENDER VIEW



Render Button - start a new IPR rendering. Once it has started, parameter changes on objects and shaders will initiate a re-rendering of the shot.

ROP/Camera menus - These menus let you choose a ROP or a camera - if you don't choose anything then default settings will be used.

Shift-Select - You can shift-select an area to focus the rendering to that area. Shift-select in the black area to go back to a refining the full image.

Progress HUD - While a quick noisy version of the rendering shows up quickly, this progress HUD shows you how much time is expected to complete a finished image and how much memory is needed.

Progress Bar - You can also refer to the bar at the top to determine the progress.

Snapshots - At the bottom of the view you can take snapshots to make it easier to compare different renderings.

MANTRA

Mantra is a renderer developed by SideFX that implements scanline, raytracing, and physically-based rendering. The physically-based rendering engine is the most advanced and is recommended for most situations. Mantra is deeply integrated into Houdini with highly efficient rendering of geometry, instances and volumes.

In general, rendering in Houdini uses a **camera** defining the viewpoint to render from, **lights** to illuminate the scene, and a **render node** representing the renderer and render settings to use. However, you can also make preview renders using the current view, a headlight, and default render settings.

PREVIEW RENDERING

While working, you will want to preview your work within the Houdini interface. There are a few different places where you can start up a rendering and evaluate the results.

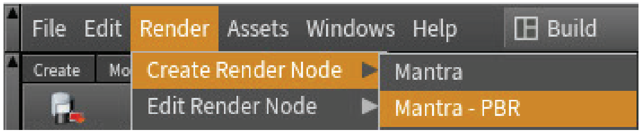
Render Region - This tool lets you draw a bounding box in the viewport and render the chosen area. You can find this tool on the toolbar. As you make changes to parameters, this region will automatically update.

Render to MPlay - This button found at the bottom of the toolbar. It will start up a rendering and send it to Mplay which is a separate application for reviewing rendered images.

Render View- You can also use the render view panel which offers Interactive Photorealistic Rendering (IPR) to allow for fast review of your work as it renders. This view will quickly render the whole image at a lower quality then layer in higher quality results while you make creative decisions about the image.

RENDER OUTPUT NODES

To render out a shot, you will need to create a render output node. You can do this by choosing **Render > Create Render Node > Mantra - PBR**. You can then rename this node and use it to render to disk or to Mplay. This node contains many of the parameters that you need to control the final image such as sampling, noise level and overall quality.

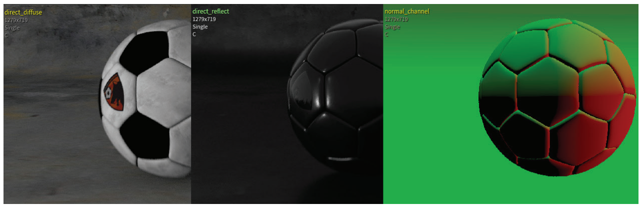


It is possible to have a different ROP per object or group of objects and you can use it to select Mattes and Phantom objects. You can create ROP dependencies by wiring different nodes together. If you press the render button on the last node in the chain then all the other nodes will render first.

The **Bake Texture** ROP would be used to generate a texture from the rendered appearance of a model.

RENDER LAYERS & COMPOSITING

On the ROP, you will find controls for setting up **Image Planes** to create render layers for *Direct lighting, Indirect lighting, Shadows, Depth* etc. These channels can be loaded in Houdini's compositing context or in an external compositor such as Nuke.



3RD PARTY RENDERING

Houdini supports third party renderers such as **RenderMan, Arnold, OctaneRender, Redshift** and **Maxwell**. You can assign materials and render to the appropriate third party ROP and then set up one ROP to render some objects with a third party renderer and another to render objects with **Mantra**.

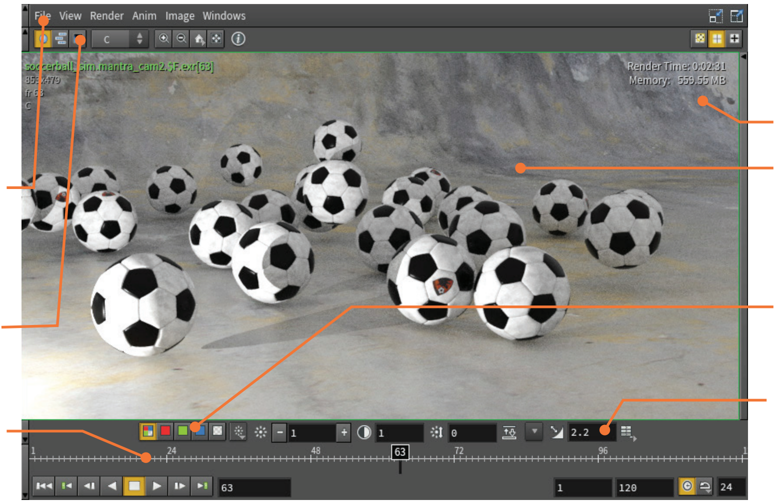
MPLAY

MPlay lets you view still images, or a sequence of frames as an animation.

Main Menu - This menu lets you load images or sequences of images to preview them. You can also save them out to another format if needed.

Render Layers - This menu lets you display different render layers such as *color, normal, diffuse_direct* or *reflect-direct*.

Timeline - If you have loaded up a sequence of images then you can use this timeline to play and scrub through the sequence.


File View Render Anim Image Windows
Soccerball_sim.mantra_cam2.$F.exr[63]
Render Time: 0:02:31
Memory: 559.55 MB
63
120
24
2.2

Render Time - Since you are sometimes rendering directly to this view, render time info will be displayed.

View Options - **MMB-drag** to pan and **RMB-drag** to zoom in and out of the image.

Channels - You can click on these buttons to focus on red, green, blue or alpha channels or to see them combined.

Gamma Settings - Set the brightness, contrast and gamma for the viewport. By default a gamma of 2.2 is used to support a linear workflow.

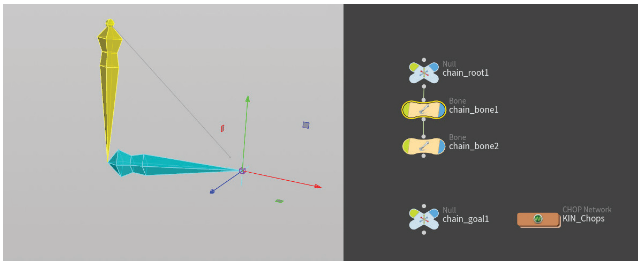
Character Rigging & FX

Houdini includes a range of tools for creating character rigs that you can animate and eventually render. There are also tools for adding Character FX such as hair, fur, muscles, cloth and crowds. All of these elements are combined to achieve the final look of any digital actor.

BONES

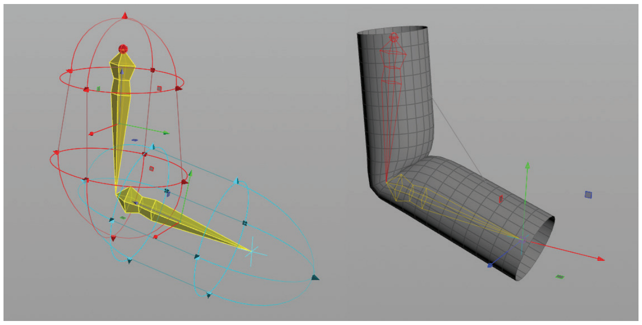
In Houdini, you draw and edit bones using the **Bones** tool and the **Bones from Curve** tool found on the **Rigging** shelf. Each bone chain is made up of a chain root and bones. Whereas other 3D apps are joint-based, Houdini uses **Bone** nodes that have parameters for **Length** and **Rest Angle**.

You can also use the **Bones** tool to add inverse kinematics to the chain which also adds an **end effector** and in some cases a **twist effector**. The kinematics are driven by **Channel Operator** or **CHOP** nodes which exist in their own subnetwork.



CAPTURING GEOMETRY

The character's geometry can then be **captured** to the bones to create the deformations needed to convey realistic motion. Houdini bones include **Capture Regions** that you can set up to encompass your geometry while creating the right overlap at the joints. This process results in weighted attributes being assigned to the points that get fed into a **Deform** node that controls the geometry when the bones are moved and rotated.



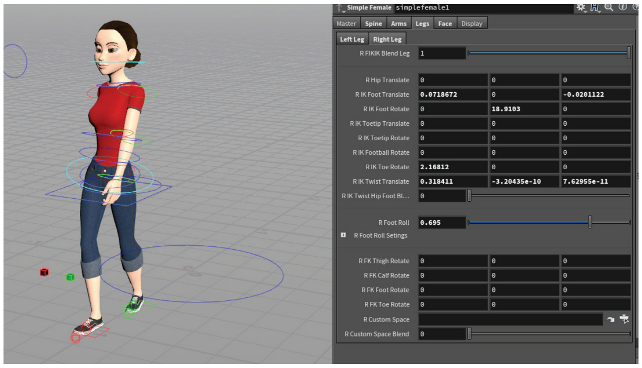
At first, captured geometry might not get you exactly the look you want, therefore you would use various tools to **Edit and Paint Capture Weights**. This lets you smooth out the weighting at the joints to create more realistic bending. You can also smooth out the effect of point deformations using a **Delta Mush** node that wires into the **Deform** node.

A new technique called **Bone Capture Biharmonic** allows you to capture geometry without requiring extensive point weights to get a desirable look at the joints. This method sets up biharmonic functions on a tetrahedral mesh to create a much more holistic solution.

DIGITAL ASSET CHARACTERS

To rig a Houdini character and share it with the animation team, you will need to wrap up the bones, geometry and materials into a **Houdini Digital Asset**.

This creates a file on disk that can be easily referenced by animators into multiple shots. Handles and key parameters are made available at the top level so that Animators can set keyframes without worrying about the inner workings of the rig. You can also save Pose Library and Character picker setups into the asset for quick access.



Simple Female simplefemale1

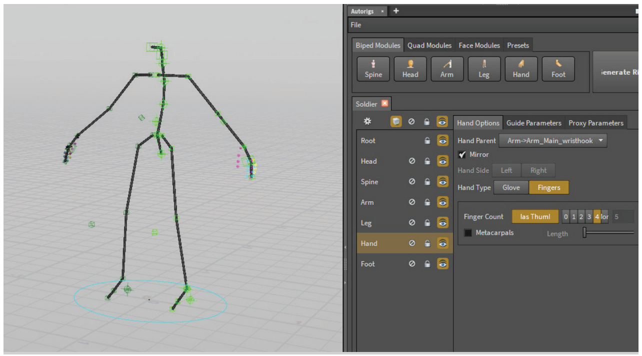
R FIKIK Blend Leg	1		
R Hip Translate	0	0	0
R IK Foot Translate	0.0718672	0	-0.0201122
R IK Foot Rotate	0	18.9103	0
R IK Toetip Translate	0	0	0
R IK Toetip Rotate	0	0	0
R IK Football Rotate	0	0	0
R IK Toe Rotate	2.16812	0	0
R IK Twist Translate	0.318411	-3.20435e-10	7.62955e-11
R IK Twist Hip Foot Bl...	0		
R Foot Roll	0.695		
R Foot Roll Setings			
R FK Thigh Rotate	0	0	0
R FK Calf Rotate	0	0	0
R FK Foot Rotate	0	0	0
R FK Toe Rotate	0	0	0
R Custom Space			
R Custom Space Blend	0		

Changes made to any part of that character are saved into the asset where all shots will be updated. This creates a robust character pipeline that is easy to manage.

AUTO-RIG TOOLS

Using the **Autorigs** pane, you can set up biped and quadruped rigs by simply determining the location of the joints on your character. Once you have these marked, the Autorig tools will build up a full rig with bones, kinematics, control nulls and a Digital Asset interface ready to go.

The network includes your base geometry, feeding into the capture and deform nodes. This character can then be put into production as a sharable Digital Asset.

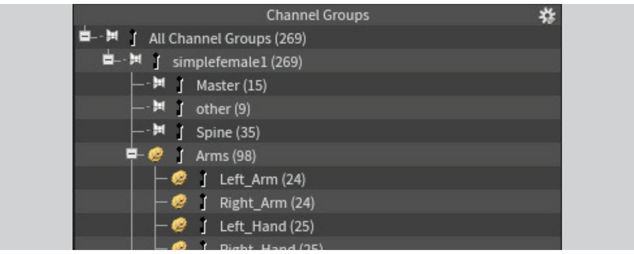


Hand Parent: Arm->Arm_Main_wristhook
☑ Mirror
Hand Side: Left Right
Hand Type: Glove [✓ Fingers]
Finger Count: las Thuml 0 1 2 3 [✓ 4] lor 5
□ Metacarpals Length

CHANNEL GROUPS

When you animate in Houdini, channels that are scoped can be keyframed and displayed in the **Animation Editor.** Generally these channels are based on your current selection. You can also put together **Channel Groups** that let you scope and pin down channels to assist with keyframing.

When you have a character set up as a Digital Asset, you can click on its icon at the top left of the parameter pane and choose **Parameters and Channels > Create Nested Channel Groups** to create groups using the asset's folder hierarchy as a guide. A well designed character asset makes this easy.



Channel Groups

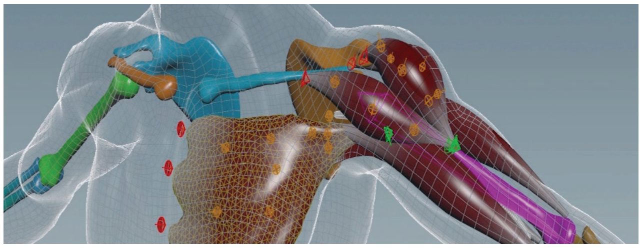
- All Channel Groups (269)
  - simplefemale1 (269)
    - Master (15)
    - other (9)
    - Spine (35)
    - Arms (98)
      - Left_Arm (24)
      - Right_Arm (24)
      - Left_Hand (25)
      - Right_Hand (25)

CHARACTER FX | HAIR & FUR

Houdini has a hair and fur toolset that you can use to setup and groom your character starting with the **Add Hair** tool. These tools also let you work with guide hairs and then animate them using wire sims to create added realism.

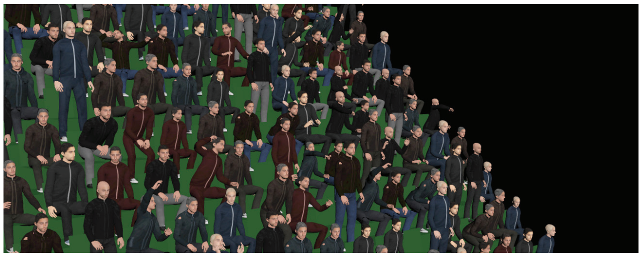
CHARACTER FX | MUSCLES & SKIN

Houdini makes it easy to add muscles to an animated creature and then apply them as a skin deformer without requiring any simulation. Start by creating simple muscle forms using the **Muscle Tool Shelf**. You then adjust the muscle's shape and placement, attach it to your character rig then enable automatic secondary animation or jiggle. Houdini's muscle system has been designed to serve both FEM (dynamics simulations) and non-FEM (skin deformer) workflows while using a unified set of Digital Assets.



CROWD SIMULATIONS

A crowd simulation begins with agents that are made up of a character skeleton, skin geometry, and animation clips. These are assigned to points in a simulation similar to particle systems, where simple rules can combine to create complex behaviors and the agents can interact with other dynamic elements. For example an agent might be struck by a passing car and become a rag-doll or a crowd might be triggered to react to an action on the field.



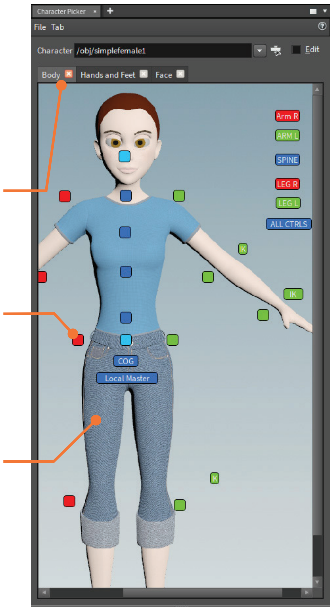
CHARACTER PICKER

This pane lets you create an interface for choosing parts of your rig. This can then be saved into a file that you add to your Digital Asset file on disk.

Tabs - You can set up multiple tabs for different areas of the body such as hands, feet or face.

Controls - You can place markers for the different handles on your rig then add text or color to help you distinguish between them.

Background Image - You can also add a visual representation of your character to properly associate the markers with the parts of your character.

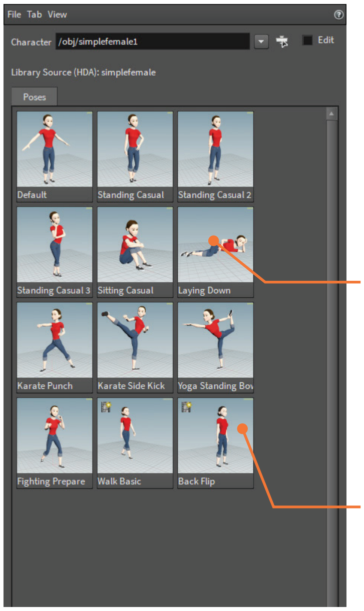

Character Picker

File Tab

Character /obj/simplefemale1

Edit

Body ☒ Hands and Feet ☒ Face ☒


File Tab View

Character /obj/simplefemale1

Edit

Library Source (HDA): simplefemale

Poses

POSE LIBRARY

The Pose Library pane lets you capture poses and clips from your character for future reference. You can go to a frame in your timeline and apply the pose by clicking on it here.

Pose - These save out a pose taken from a single frame. All of the parameter settings at that pose would be applied to the character in your current scene. You would then use this to interpolate to another pose.

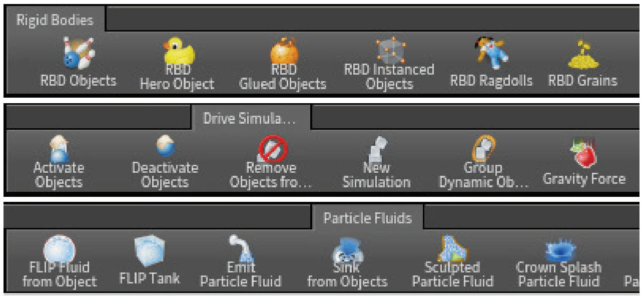
Clip - A clip contains a set of keyframes for a longer time period. These might be a walk cycle or a particular movement like a back clip.

Dynamic Simulations

Whether you are creating Bullet Rigid Body destruction, Pyro FX fire and smoke or FLIP fluid oceans or liquids, Houdini lets you work in an integrated dynamics environment. Different solvers know how to talk to each other to allow for more directable results.

SHELF TOOLS

Setting up dynamic simulations involve a network of nodes in the **Dynamic or DOPs** context, as well as nodes at the **Geometry or SOPs** context. It is always a good idea to use the shelf tools because they will add all of these nodes for you and reduce the number of clicks needed to set up a sim. You can then dive into the networks to explore all of the nodes.



The shelf tools are great for setting up groups of nodes automatically. Seeing how these shelf-built networks are put together can be very useful later if you choose to set up DOP networks from scratch.

DYNAMIC SOLVERS

At the center of any simulation is the dynamic solver. It is the brain of the simulation and takes all of the dynamic objects, forces and collision objects and integrates them to create the final result. The shelf tools put these solvers into a Dynamic Network and wire up the nodes for you.

Rigid Body Solver- Simulate rigid objects falling and colliding using the efficient Bullet solver or Houdini's built-in solver.

Static Solver- For situations where you want objects to work as collision geometry but not be affected by the simulation.

Flip Solver - This solver creates FLIP Fluid simulations to create splashing and wave effects.

Whitewater Solver - After completing a FLIP solve, you can run this solver to create foam, spray and bubbles.

POP Solver- Used for particles and grains, this solver simulates a wide range of different particle-based scenarios. Grain simulations can also be used for soft body and cloth-like simulations.

Wire Solver- You can use this solver for hair and fur or other wiry objects such as the rigging of a ship or the branches of a tree.

Finite Element Solver- Simulates the physics of continuous materials or solids as determined by tetrahedrons. This solver is used for muscles, soft body sims and destructions shots such as breaking wood.

Cloth Solver- Create cloth simulations that can collide with deforming geometry such as a character.

Crowd Solver- Uses logic to animate crowd agents based on rules while working with targets, paths obstacles and foot placement.

SOP Solver- Use a SOP network to evolve an object's shape over time such as a wall being dented as it gets hit by objects.

OPENCL

You can use the GPU for faster sim times using **OpenCL** on solvers such as the **POP Grain node**, the **Pyro solver** (**Advanced** tab) and the **FLIP solver** (**Volume Motion > Solver** tab).

FORCES

To create dynamic motion, forces are needed to "get the ball rolling." The most basic of forces is gravity although other external forces such as fans, fluids and magnets can also play a role in initiating motion in your simulation.

Gravity Force A downward force on objects which works as if they were inside a gravity field.

Drag Force - Applies force and torque to oppose an object's existing motion to slow it down or dampen its momentum.

Uniform Force- A precise amount of force and torque that can be augmented by a noise DOP to add turbulence.

Fan Force- Applies a cone-shaped force on objects.

Fluid Force- Deform soft bodies such as cloth or wires with fluids.

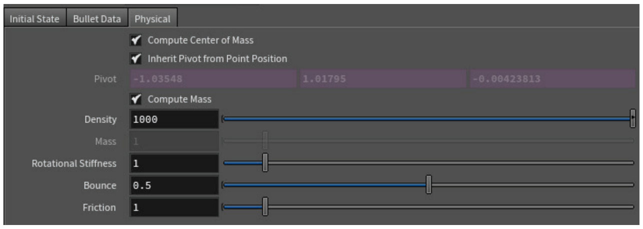
Wind Force- A pushing force that will increase the velocity of objects up to but not beyond its own speed.

Magnet Force- Attracts or repels objects using metaballs to represent a force field.

Vortex Force- Creates a vortex-like force that causes objects to orbit around a curve much like objects around a tornado.

DYNAMIC OBJECTS

When you select an object and use a shelf tool to add it to your simulation, Houdini creates a Dynamic Object that uses the object's geometry and adds dynamic properties such as *density*, *friction*, and *bounciness*.



☑ Compute Center of Mass

☑ Inherit Pivot from Point Position

Pivot -1.03548 1.01795 -0.00423813

☑ Compute Mass

Density 1000

Mass 1

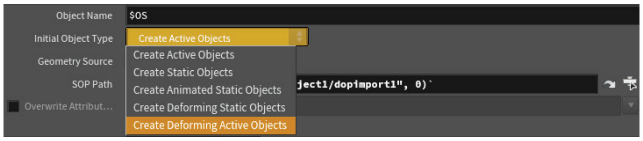
Rotational Stiffness 1

Bounce 0.5

Friction 1

ACTIVE VS STATIC

Active dynamic objects are affected by forces and collisions while Static objects are not. If you want to use animated or deforming geometry then you must define this on the dynamic object using either the **Initial Object Typ**e menu or the **Use Deforming Geometry** checkbox



Object Name $OS

Initial Object Type Create Active Objects

Geometry Source

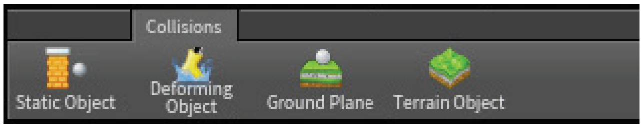
SOP Path ject1/dopimport1", 0)`

■ Overwrite Attribut...

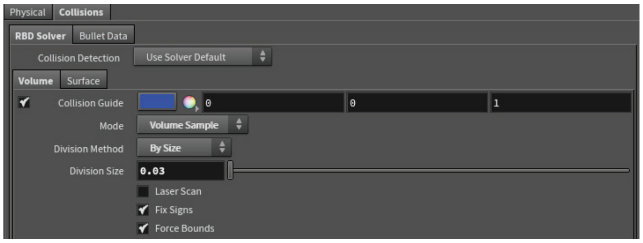
Create Active Objects
Create Static Objects
Create Animated Static Objects
Create Deforming Static Objects
Create Deforming Active Objects

COLLISIONS

Collision objects are also a big part of any simulation. You can set up a **Ground Plane** to create a continuous surface for collisions or use either a static or deforming object.



On each **Dynamic Object**, there are also settings for displaying and optimizing the collision volume. While you often want collisions to be as accurate as possible, you need to balance that with the need for faster simulation times.



Physical | Collisions

RBD Solver | Bullet Data

Collision Detection: Use Solver Default

Volume | Surface

☑ Collision Guide: 0 0 1

Mode: Volume Sample

Division Method: By Size

Division Size: 0.03

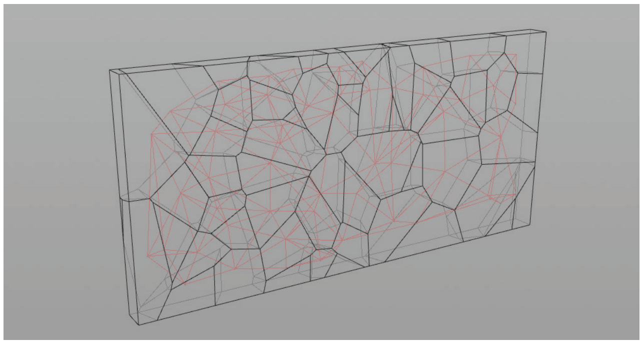
□ Laser Scan

☑ Fix Signs

☑ Force Bounds

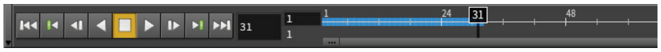
RIGID BODY CONSTRAINTS

On the Rigid Bodies shelf, you will find a number of constraints that can also be used to influence a simulation. These include **Pin**, **Spring** and **Slider** Constraints. You can also use **Glue Objects** when you set up a rigid body sim to hold objects together until you either "loosen up" the glue or a collision occurs.



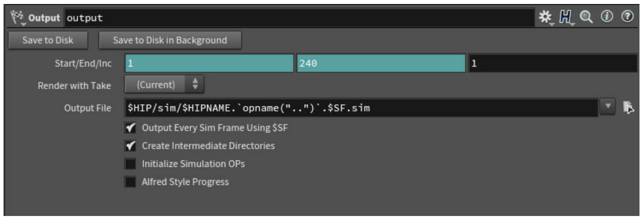
TIMELINE FEEDBACK

To start a simulation, you press **Play** in the timeline. As the simulation progresses, the timeline is highlighted to show how much of the sim has been cached to memory. You can then scrub through that area without re-simming.



CACHING TO DISK

Once you have completed a simulation, you can either lock it down by saving out a **sim** file from within DOPs or more commonly, write out the simulated geometry to a **bgeo** sequence. This will make it easier to work with the results of a sim during the lighting and rendering stages of production.



Output: output

Save to Disk | Save to Disk in Background

Start/End/Inc: 1 240 1

Render with Take: (Current)

Output File: $HIP/sim/$HIPNAME.`opname("..")`.$SF.sim

☑ Output Every Sim Frame Using $SF

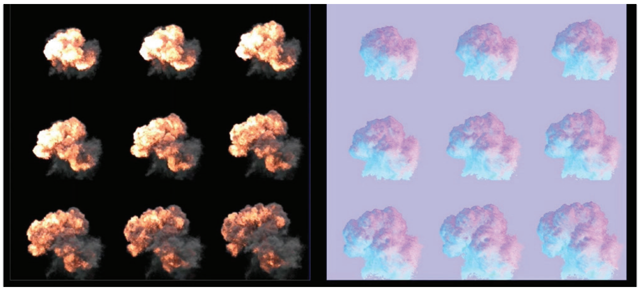
☑ Create Intermediate Directories

□ Initialize Simulation OPs

□ Alfred Style Progress

REALTIME FX FOR GAMES

In games, you need effects, such as explosions, to be optimized for real time in the game engine. Check out the **Game Development Tools** to learn more about converting different kinds of Houdini sims such as rigid bodies, Pyro FX and Fluids into game ready art.



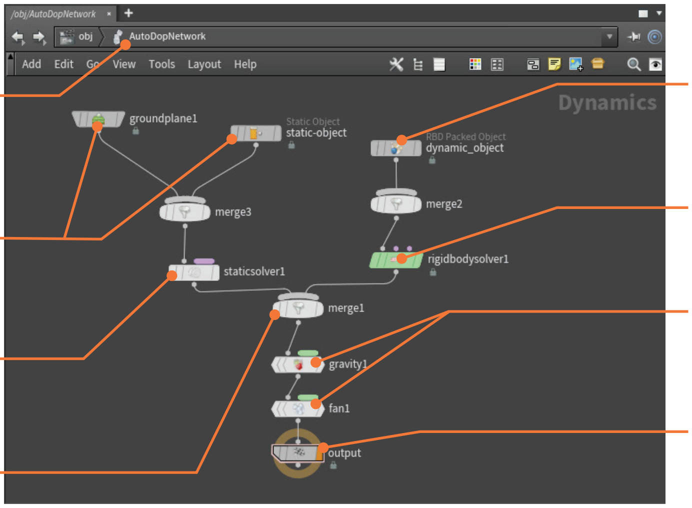
AUTODOPNETWORK

Any time you use a shelf tool to create a dynamic object, collision object or force, the AutoDopNetwork is created to combine all of the parts.

Static Objects - These nodes set up the properties of the ground plane and a static collision object.

Static Solver - This solver keeps the incoming objects still while dynamic objects interact with them.

Merge Node - Brings together parts of a dynamic system. During simulation nodes are evaluated up and down the chain so that everything interacts.



Dynamic Object - This node brings geometry into the DOPs environment and assigns basic properties.

Rigid Body Solver - The solver that generates the simulation of the participating objects.

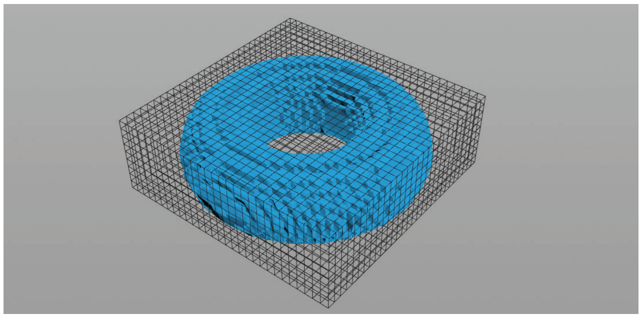
Forces - The nodes that influence the dynamic objects using forces such as gravity or wind.

Output node - you can use this node to output .sim files if you want to cache out the simulation.

Cloud FX and Volumes

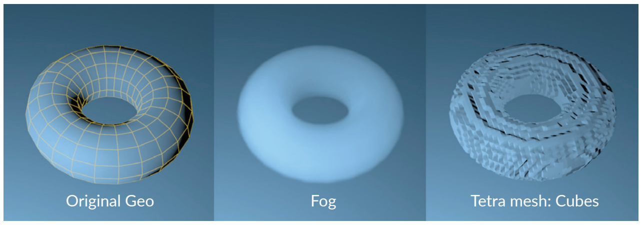
A big part of visual effects in Houdini is the use of volumetric data. In Houdini, volumes often sit under the hood to help tools get the job done but it is a good idea to learn what they are and, in time, learn how to work with them directly.

With volumes, you describe objects using *voxels* rather than points and polygons. A voxel is a 3D pixel, a cubic grid where each voxel contains information that informs how the volume will be displayed which makes it ideal for wispy cloud-like shapes. The visual quality of a volume-based object is defined by the resolution of that 3D grid. With more resolution, the results are of a higher quality but performance may be affected.



ISO OFFSET

The **Isooffset** node found in the geometry context lets you take any manifold Polygon geometry and construct a Houdini Volume for Houdini to use. You can choose from a variety of different Output types to see the shape as *fog* or a *tetra mesh*.



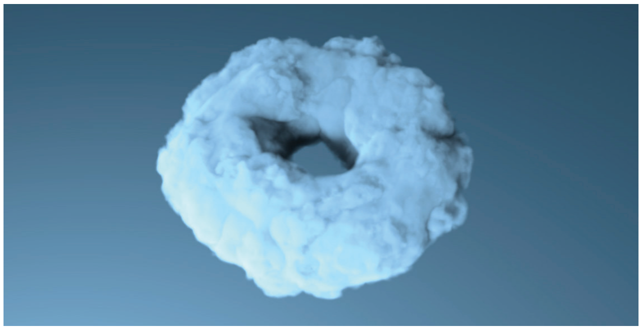
Original Geo

Fog

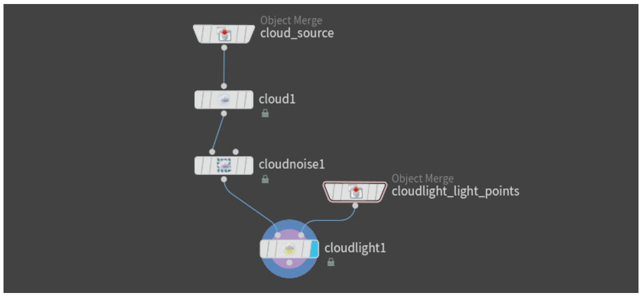
Tetra mesh: Cubes

CLOUD FX

This toolset converts geometry into a cloud-like VDB volume complete with lighting. The **Cloud Rig** tool can be useful for shaping an individual cloud, or simply as a way to better understand lower-level tools such as **Cloud**, **Cloud Noise** and **Cloud light** that all contribute to the final look.



The resulting network imports the cloud source then applies these other nodes to create the cloud-like effect using Houdini Volumes and VDBs. Houdini also comes with a **Sky Rig** tool that fills the sky with volumetric clouds.



To create a cloudscape for a game engine such as **UE4** you can use a Sky rig in Houdini, then convert it to a mesh and use it as a spawn surface in Unreal. There is a great tutorial on the SideFX website by **Andreas Glad** that teaches this approach.

OPEN VDB

"OpenVDB is an Academy Award-winning open-source C++ library comprising a suite of tools for the efficient storage and manipulation of sparse volumetric data discretized on three-dimensional grids. It is developed and maintained by DreamWorks Animation for use in volumetric applications typically encountered in feature film production." - **openvdb.org**

Houdini has a variety of OpenVDB Volume nodes available in geometry [SOP] networks that convert geometry into volumes.

VOLUMES UNDER THE HOOD

Many of the tools in Houdini make use of volumes under the hood where you can't see them. Here are some of the areas where volumes are making contributions to your work.

- **Colliders** - By default, volumes convert geometry into colliders for dynamic simulations.
- **Simulation Fields** - Volumes define fields such as density or velocity that contribute to dynamic simulations.
- **Hair and Fur tools** - These tools use volume data to assist with grooming calculations.
- **Terrain** - Heightfield tools use 2D volumes where each voxel contains the height of the terrain at each grid point.
- **Rendering** - Volumes create water depth and fog effects in Mantra.

Terrain and Heightfields

Procedural terrain generation in Houdini is possible using a collection of heightfield nodes that let you layer shapes, add noise and run erosion simulations to define the look for your digital landscapes. This a workflow that is similar to compositing but you do all your work with 3D shapes.

Houdini provides a variety of geometry nodes for generating and shaping terrain. These tools represent terrain using 2D volumes where each voxel contains the height of the terrain at that grid point, called **heightfields**. The data passing through a geometry network can contain multiple heightfields. You can access these tools using the **Terrain** desktop.

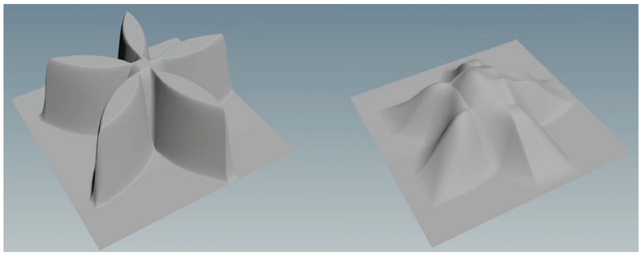
The Houdini viewport lets you visualize the 2D heightfield as a 3D surface, and the mask field is displayed as a red tint on the 3D surface. There is a dedicated Mantra procedural for rendering heightfields and they can be used as collision surfaces for dynamic simulations.

ARON KAMOLZ

PATTERNS

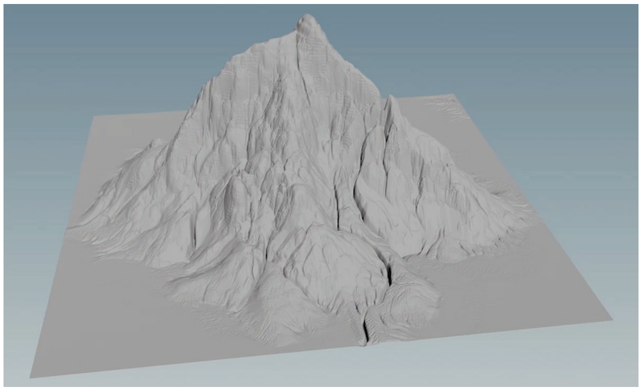
After laying down a **Heightfield** node to define your base resolution, the **Heightfield Pattern** node gives you access to a number of starter shapes. You can set up linear, concentric, or radial ramps, linear steps, radially symmetrical shapes such as stars and voronoi cells.

These shapes can then be blurred and distorted to get shapes that you can use to begin your terrain. You can also combine and layer elements to achieve even more sophisticated results.



NOISE

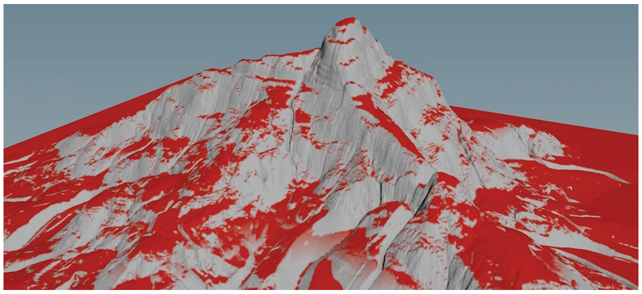
As you build up your terrain, you can then add noise to layer in a natural look. You can choose from a variety of different types of noise including, *Perlin*, *Sinusoid*, *Worley* and more. This adds realism to your terrain and by combining different shapes with different kinds of noise, you can achieve a wide variety of hyper realistic results.



MASKS

The heightfield tools also use a secondary type of 2D volume, where each voxel contains a **mask layer**. Most terrain nodes take a second input that can contain a mask layer to control which parts of the terrain the node will modify.

You can use a variety of different methods to create masks and then use them to assist you as you add detail and shape the terrain. You can also draw or paint masks onto the heightfield.



EROSION

The **Heightfield Erode** node uses rainfall, the erodibility of the soil, and entrainment rates as variables to simulate erosion and deposit buildup. This node works iteratively during playback. It will appear to have no effect on the first frame. You need to press play in the timeline to watch as it sims the erosion.

EXPORT OPTIONS

There are a couple of different ways to export your terrain for use in another application such as a game engine. You can use the **Heightfield Output** node to export height and/or mask layers to disk as an image then bring these in as textures.

You can also create **Houdini Digital Assets** that will open up in applications such as UE4 using the Houdini Engine plug-ins. In UE4, you can even set up Digital Assets that will interact with Unreal's built-in Terrain tools.

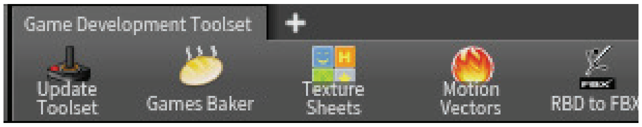
Game Development Tools

The Game Development Toolset is a collection of high level tools aimed to speed up games related workflows in Houdini. There are a growing number of game tools being developed that range from UVing to generating Motion Vectors from simulations.

While all of Houdini can be used to generate content for games, this toolset addresses games-specific tasks that are not currently available in Houdini,.These tools are developed separately from the regular Houdini development cycle and become available the moment they are ready for testing. They are currently deployed through our **github** page, and can be downloaded directly from within Houdini.

DOWNLOADING THE TOOLS

Houdini has a Games Development shelf tab with gamedev specific tools. This tab is included in the **Games desktop**, or you can add it to the shelf set in other desktops. Click on the **Update Toolset** button. This will pop up a dialog prompting you with which branch to download and which release.

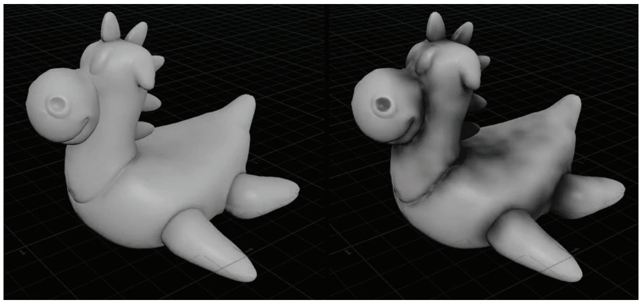


This toolset is always evolving and should be updated regularly. Here are some of the kinds of tools you can expect to find in this collection.

GEOMETRY TOOLS

These tools and utilities are useful for working at the geometry level of Houdini.

Calculate Occlusion - Calculates vertex Ambient Occlusion at the SOP level. Lots of controls for intensity, falloffs and blur.



Delete Small Parts - Utility SOP to remove small disconnected parts based on a size threshold.

Edge Group to Curve - Convert an edge selection (group) to a polygon curve. Useful if you want to sweep something along your model or many other applications.

Skinning Converter - Skinning Converter is a Houdini Digital Asset (HDA) that can convert any non-changing topology deforming mesh sequence into a bone based animation.

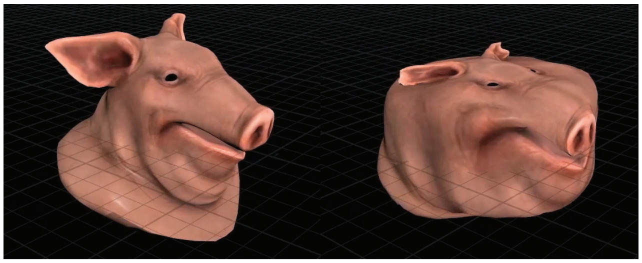
Group Edge Loop - Procedurally generate Point, Prim and Edge Loops. There is also an option to skip every N units.

Group Expand - Expands or contracts a group by a number of iterations. Positive iterations expand, negative iterations contract

Instant Meshes Bridge - Instant Meshes is a great open source Quad Remesher. This sops simply wraps around its command line interface for a more native Houdini experience.

Soften Normals - Utility script for softening the normals of an object and optionally hardening the UV seams.

PolyDeform - Deforms the first input to match the shape of the second input. Incredibly useful for modifying scanned data.



Quick Material - QuickShade on steroids. Not only apply diffuse textures, but all of the different PBR material settings right at the SOP level

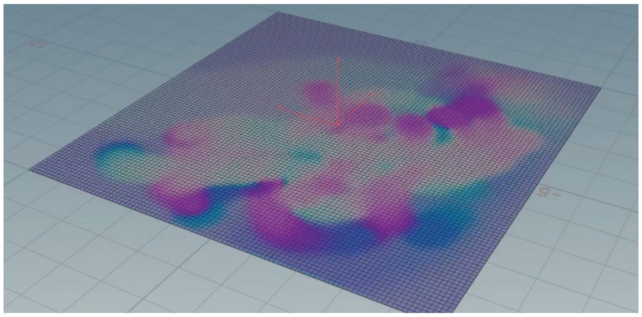
Thicken - Give planar surfaces thickness with this node. It also has a slightly smarter logic on how to thicken an object in order to avoid geometry errors. Great for making things water tight!

Unreal Pivot Painter - Tool to generate the appropriate data necessary for Unreal 4's pivot based animation nodes. Which allows you to do complex shader animations with a very slight overhead compared to a traditional joint setup.

FLOWMAP TOOLS

Flowmaps allow you to simulate the movement of liquids inside a game environment. These tools help you convert your Houdini Fluid sims into flowmaps.

Flowmap - Initializes the correct attributes on a mesh in order to export a flowmap.



Flowmap Guide - Uses a curve as an additional input and bends the velocity field of the flow map based on said curve. Very useful for non destructively combing the flow field.

Flowmap Obstacle - Uses an object as an additional input and warms the velocity field around said object. Useful for very quickly bending the flow around objects.

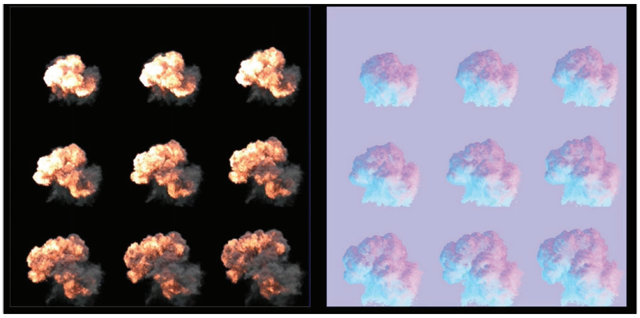
Flowmap To Color - Converts the worldspace velocity vectors into Tangent (uv) space colors. This allows the user to then bake the velocities out as vertex colors to be used in a game engine.

REALTIME FX TOOLS

Visual Effects in video games must be as light as possible to not interfere with gameplay. These tools help you convert FX from Houdini into game-friendly assets.

Make Loop - Makes a Volume or a mesh with constant topology automatically loop. It does so by offsetting the animation by half and cross dissolving the animation onto itself.

Texture Sheets - This node helps with the generation of Flipbooks/ Texture Sheets. Including the generation of normals (both based on the volume data as well as a traditional 6 light rig) and the ability to channel pack your texture.



Vertex Animation Textures - Export simulation data to texture files. This allows for very effective vertex shader playback of your high resolution simulation. There are 4 types of simulation supported

Destruction Cleanup - Optimizes RBD simulations based on whether the objects have moved or not.

RBD to FBX - Export a destruction sequence to FBX. It takes a packed primitive destruction setup and moves all of the point data up to object level, which allows FBX to interpret that as top level motion. Most game engines will interpret this data as a skeletal mesh animation that can be played back natively.

Static Fracture Export - Exports pre-fractured pieces for realtime destruction. Useful in Unreal's destruction System and NVidia's PhysXLab

Motion Vectors - Renders out the velocity vectors to camera space, which allows them to be used as frame blending flow maps.

Vector Field - Vector Fields are velocity grids used to control GPU particles in UE4. This tool exports them to the FGA file format that Unreal consumes.

UV MAPPING

Texture UVs are a big part of creating game art and these tools augment Houdini's existing UV toolset to make you faster and more efficient.

Auto UV - Automatically generates the seams for an object and immediately runs a UV Flatten after the fact. There are 2 main techniques:

- **Shortest Path**: Uses the shortest path algorithm to define seams based on the surface's curvature
- **Cluster**: Subdivides the meshes into quadrants and uv islands.

Mark Seams - This nodes allows for a more intuitive way of managing edge groups for common workflows like UVing. It is essentially an uber group node which lets you store selections as well as convert primitive selections into edge selections of said primitive borders

Axis Aligned - Similar to the UV Unwrap, but does a better job at clustering smaller islands

UV Stack - Stacks similar UV shells together. Great for hair workflows and general UV management. Also contains some extra controls for splitting stacks randomly for variation.

UV Unitize - Simple Utility tool to make every face of a mesh to be UV'd from 0-1

UV Visualize - Helper script to visualize UVs. Including features such as: Visualize Seams, Warp between UV space and Model Space, Modify the tiling of the grid texture and Visualize Islands

BAKING

These tools wrap up Houdini's baking tools to make them easier for artists to work with.

Games Baker - This node simplified the baking process by allowing you to bake the common maps easily, as well as any point attribute without additional setup. Bake multiple high>low pairs at the same time.

Simple Baker - A Wrapper for the Games Baker at the SOP level. Takes a low mesh and a high mesh as the 2 inputs

NORMAL MAPS

This tools let you use Houdini's compositing context to work with normal maps.

Normal Map - Generates a normal map from a grayscale image, with additional options to soften the normals and an intensity scale

Normal Combine - Combines 2 normal maps together while keeping them properly normalized

Normal Invert - Inverts individual channels of the image, usually for flipping that pesky Y channel

Normal Normalize - Ensures the image is a valid normal map with all of the 3 channels properly balanced

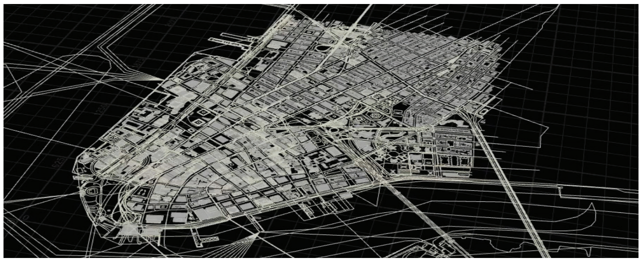
Normal Rotate - Rotates a normal map while keeping all of the normal vectors correctly oriented.

INTEROPERABILITY

These tools make it easier to import and export data into Houdini and out to game engines.

Impostor Texture - The impostor tool creates texture sheets to fake 3D objects in your game engine. The tool will create full 3D, fixed axis and animated impostors.

OSM Import - Open Street Map is a great database for city street data. This node efficiently loads the OSM files into Houdini as well as all of the different tagged attributes on the buildings and streets.



OSM Buildings - Open Street Map is a great database for city street data. This node parses the data brought in from the OSM Import node and generate building geometry from the profile curves.

FBX ROP - Utility node to bring the FBX export capabilities into the SOP context. You can currently export geometry and alembic from SOPs but there wasn't an equivalent version for FBX.

CSV Exporter - A simple example ROP to demonstrate how you can extract data from geometry and write it out with Python. The bulk of the code is in the Scripts Tab in the Type Properties Dialog.

Volume Textures - Converts a volume into a flipbook representation of a 3D texture. The resulting texture is usable in Ray Marching shaders which render your Houdini in Realtime!

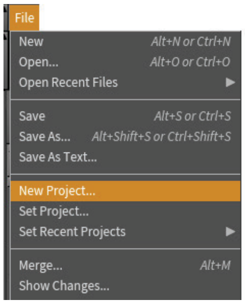
File Management

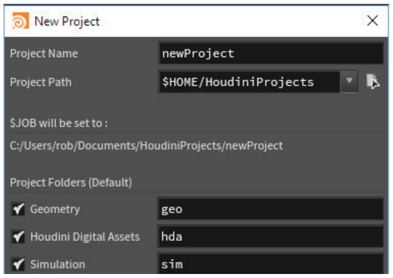
Understanding how to manage all of the files that you create while working with Houdini is very important to your success as an artist. A typical scene file can have outside dependencies on disk and managing these is important especially if you are moving your files to a different computer.

There are a number of file formats that you will use on a regular basis when working with Houdini. Some of these are unique to a Houdini pipeline while others are industry standards such as Alembic or EXR.

PROJECT DIRECTORIES

While Houdini can work with files scattered all over your hard drive, this will make it harder to share your work and manage file dependencies. It is better to set up project directories using **File > New Project** or use **File > Set Project** to choose an existing project directory as the home base for your work. This will make it easier to set up local dependencies with respect to all of the required project files.




New Project

Project Name: newProject

Project Path: $HOME/HoudiniProjects

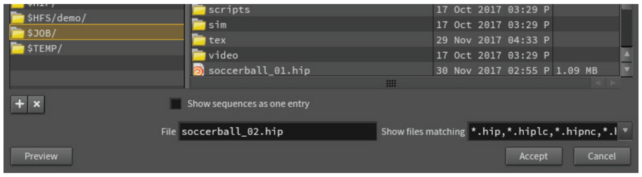
$JOB will be set to :
C:/Users/rob/Documents/HoudiniProjects/newProject

Project Folders (Default)

- ☑ Geometry: geo
- ☑ Houdini Digital Assets: hda
- ☑ Simulation: sim

SCENE FILES | .HIP

The main file type when working with Houdini is the **.hip** file. This file contains all your nodes and networks and is the file type used when you save your work.

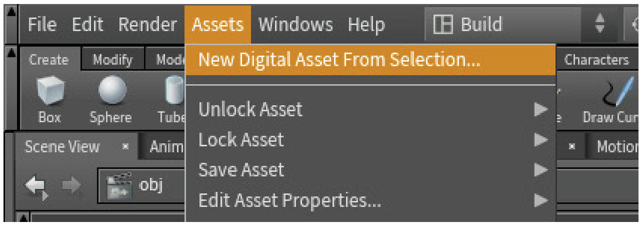

```
$HFS/demo/      scripts            17 Oct 2017 03:29 P
$JOB/           sim                17 Oct 2017 03:29 P
$TEMP/          tex                29 Nov 2017 04:33 P
                video              17 Oct 2017 03:29 P
                soccerball_01.hip  30 Nov 2017 02:55 P 1.09 MB
```

□ Show sequences as one entry

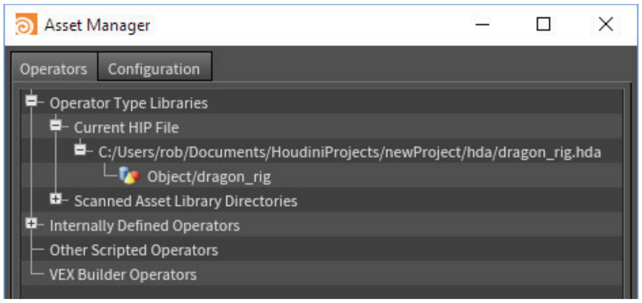
File: soccerball_02.hip   Show files matching: *.hip,*.hiplc,*.hipnc,*.h

HOUDINI DIGITAL ASSETS | .HDA

You can also encapsulate then save out Houdini networks into **Houdini Digital Asset** or **.hda** files. Parameters from inside the asset can then be promoted to the top level to create a custom ui for the asset. These files can be easily shared with other artists and provide a robust referencing architecture as your assets evolve through the life cycle of a project.

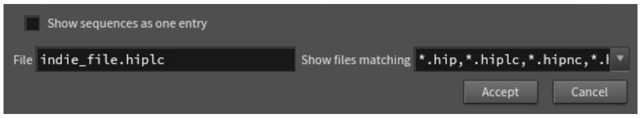


To create and load assets you can use the **Asset** menu. You can also manage assets loaded into your scene using the **Asset Manager** found on that menu. If you have two HDA files loaded into your scene that have the same name, Houdini will choose one of them based on rules set up in the manager. Changes made to an asset definition in an HDA file will be automatically pulled into scenes that reference that file. Note that older Digital Asset files may have a .otl extension which will work exactly the same as .hda files.



APPRENTICE AND INDIE FILES

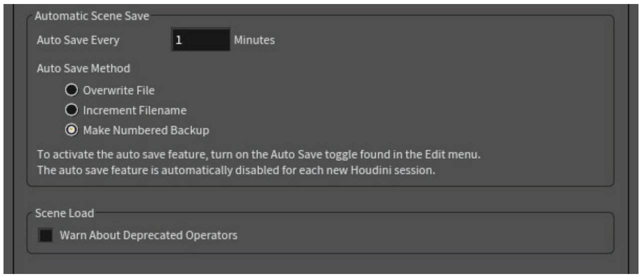
The free **Houdini Apprentice** and the **Houdini Indie** products use different file types that cannot be opened in a commercial version of Houdini. Apprentice files will use **.hipnc** (non-commercial) and Indie will use **.hiplc** (limited commercial). HDA files and image files have similar extensions.


□ Show sequences as one entry

File: indie_file.hiplc   Show files matching: *.hip,*.hiplc,*.hipnc,*.h

BACKING UP YOUR WORK

By default, Houdini creates a numbered backup of your scene files and Digital Asset files every time you save. This gives you a file to go back to if you want to review an early iteration or if something happens to your working file. You can also set up Houdini to **AutoSave** in the **Edit > Preferences > Save and Load Options**. Just remember that all those backup files take up disk space and you will want to clear them from time to time.


Automatic Scene Save

Auto Save Every: 1 Minutes

Auto Save Method

- ○ Overwrite File
- ○ Increment Filename
- ● Make Numbered Backup

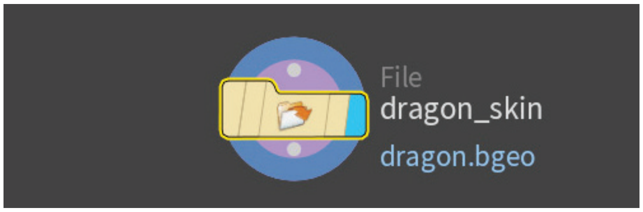
To activate the auto save feature, turn on the Auto Save toggle found in the Edit menu.
The auto save feature is automatically disabled for each new Houdini session.

Scene Load

□ Warn About Deprecated Operators

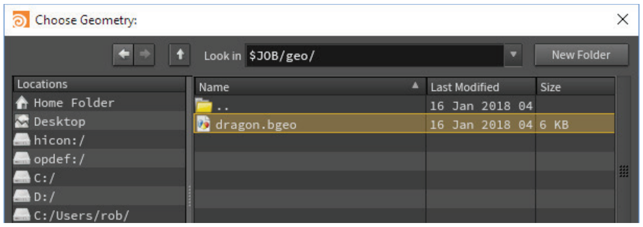
FILE SOP

When you import geometry into Houdini using **File > Import > Geometry** it puts down a **File** node at the geometry (SOP) level. This file maintains a connection to the file on disk and changes made to that file will also update in your Houdini scene. If you want to break this connection then you would need to lock the File node.



FILE DEPENDENCIES [$HIP/$JOB]

When you work with nodes that reference files on disk such as geometry or texture files, the path you use will determine what happens if you move the project directory to another computer or to the cloud. A direct path will break if you move the files therefore you should either use **$HIP** which uses the scene file as the "home base" for the path or **$JOB** which uses the project directory. You can use **Render > Preflight Scene** to check to make sure that your scene file is set up properly.



Look in $JOB/geo/

Locations
Home Folder
Desktop
hicon:/
opdef:/
C:/
D:/
C:/Users/rob/

Name	Last Modified	Size
..	16 Jan 2018 04	
dragon.bgeo	16 Jan 2018 04	6 KB

DISK SPACE MANAGEMENT

Large scene files, backup files and large simulations can take up **lots of disk space**. Be sure that you are not filling up too much space on your computer which could lead to instability issues. Try to use external drives to save out the largest files and leave the main disk on your computer with enough space to accomplish your day-to-day work.

INTEROPERABILITY

To import and export from Houdini, there are a wide variety of file formats that you can use. Here is a list of some of the main formats you will work with in a typical Houdini pipeline.

Houdini Files - Here are some file formats, other than .hip and .hda that work exclusively in Houdini.

.bgeo - This format saves geometry along with related attributes such as UVs, velocity and normals. Animations and simulations can be saved out as numbered bgeo files to save out motion. A bgeo.gz file is a compressed version of this format.

.sim - These files let you save out simulation data to cache the sim to disk. Some people use these files while others use .bgeos to cache sims.

.ifd - This is a scene description format that is created when rendering to Mantra. Typically these are created while rendering in Houdini but sometimes they are saved to disk to be rendered by Mantra directly.

.pic - This is an image file format that was used by Houdini in the past. It was replaced as the default format by the open source EXR.

.rat - This image format is ideal for texture maps being rendered in Mantra. All textures get converted to this format anyway so it speeds up rendering to convert to this format using Mplay.

Image Formats - These industry-standard formats are used to render out shots and for texture maps.

.exr - OpenEXR is a high dynamic-range (HDR) image file format developed by Industrial Light & Magic that is now the default format for saving out renderings from Houdini.

.jpg/.png - These formats are used to publish images to the web

.tga/.tif - These formats are often used to texture map video games.

Geometry Formats - When importing and exporting geometry, these formats are the most popular.

.abc - Alembic is an open computer graphics interchange framework that is in CG pipelines to share data between applications.

.fbx - This format owned by Autodesk is popular when exchanging data with game engines and other 3D applications. It can hold geometry, rigging, motion and shader information.

.obj - This is an simple geometry format originally created by Wavefront.

PREFLIGHT PANEL

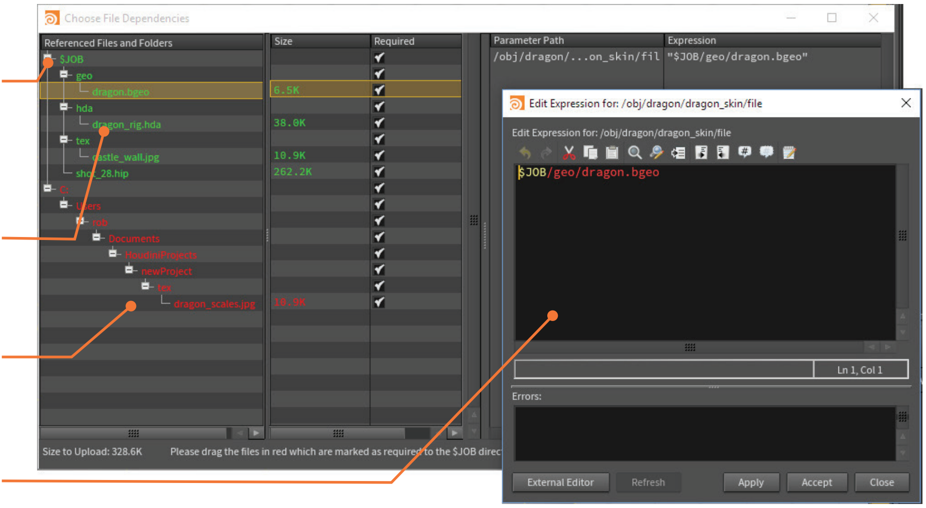
From the **Render** menu, select **Preflight Scene** to evaluate your scene setup.

Referenced Files - The preflight panel can either reference $HIP or $JOB when verifying file references for your scene file.

Greenlit Reference - If the reference is relative to either $HIP or $JOB then it will be displayed in green to indicate it is working.

Incorrect Reference - If a file reference is a direct path and not relative to $HIP or $JOB then it will be displayed in red and will need to get fixed before sharing your project with other artists or on the cloud.

Edit Expression - Click on any file name then on the right click on the expression to open up the edit expression window.



Referenced Files and Folders	Size	Required
$JOB		✓
geo		✓
dragon.bgeo	6.5K	✓
hda		✓
dragon_rig.hda	38.0K	✓
tex		✓
castle_wall.jpg	10.9K	✓
shot_28.hip	262.2K	✓
C:		✓
Users		✓
rob		✓
Documents		✓
HoudiniProjects		✓
newProject		✓
tex		✓
dragon_scales.jpg	10.9K	✓

Parameter Path	Expression
/obj/dragon/...on_skin/fil	"$JOB/geo/dragon.bgeo"

Size to Upload: 328.6K Please drag the files in red which are marked as required to the $JOB direc

Edit Expression for: /obj/dragon/dragon_skin/file

```
$JOB/geo/dragon.bgeo
```

Ln 1, Col 1

Errors:

Expressions & Scripting

Houdini is a production-level solution which means that scripting will eventually play a role in your work. Artists can usually get by with simply writing expressions while technical directors will spend more time using these techniques. Houdini includes support for Hscript, Python and VEX.

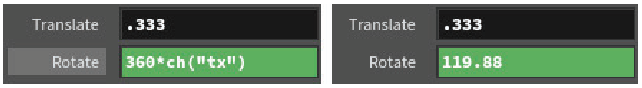
Animation, visual effects and gamedev are creative fields built on a highly technical foundation and it is important to balance your artistic aspirations with technical know-how.

Houdini's node-based workflow is designed to make CG technology more artist-friendly, while providing you with the flexibility to shape things any way you want.

Scripting and expressions offer another way to enhance the creative process by giving you a deeper level of control.

HSCRIPT EXPRESSIONS

HScript is designed to be a fast and concise way to retrieve and manipulate information that can be used to write expressions. An expression is typically any value that is not either a simple string or number. This can be something as simple as a variable, a math equation or an expression function.



Translate	.333
Rotate	360*ch("tx")

Translate	.333
Rotate	119.88

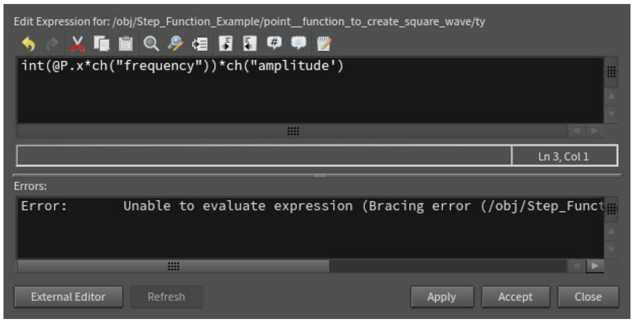
You can enter expressions directly into a parameter by simply typing into a field. When you press **Enter** the field highlights in green. You can click on the parameter name to toggle back and forth between the expression and the result of the expression.

If you are creating channel references then you can **RMB-click** on a parameter and choose **Copy Parameter** then go to the parameter you want to link it to and choose **Paste Relative References**.

You can also achieve this by RMB clicking on the second parameter then choosing **Reference > Scene Data**. This opens up a panel where you can choose data from other objects and nodes and an expression will be built for you. This method can even set expressions on multiple parameters.

EXPRESSION EDITOR

Depending on the complexity of your function, or the type of parameter, you may instead choose to use the **Expression Editor**. The expression editor can be opened by **RMB-clicking** on a parameter and selecting **Expression > Edit Expression,** or by placing the mouse over the parameter and pressing **Alt - E**.



Edit Expression for: /obj/Step_Function_Example/point__function_to_create_square_wave/ty

```
int(@P.x*ch("frequency"))*ch("amplitude')
```

Ln 3, Col 1

Errors:

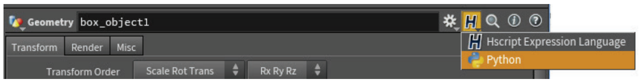
```
Error:      Unable to evaluate expression (Bracing error (/obj/Step_Funct
```

PYTHON

Python is a popular scripting language that is well known in the CG industry that supports integration and standardization. This makes it perfect for tool development.

Python in Houdini is built on the Houdini Object Model (HOM) which is an API that lets you get information from and control Houdini using the Python scripting language. In Python, the `hou` package is the top of a hierarchy of modules, functions, and classes that define the HOM. The `hou` module is automatically imported when you are writing expressions in the parameter editor and in the hython command-line shell.

You can also use it to write expressions in Houdini. To do this, change the expression language option at the top of the parameter pane for the node.

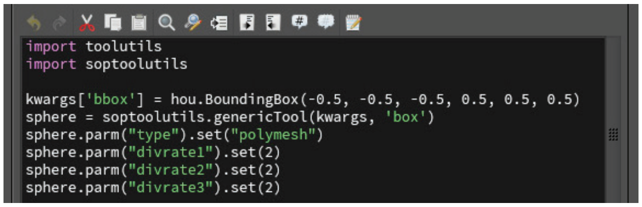


Geometry box_object1

There is also a **Python Shell Panel** which you can use to enter Python commands. You can also import the `hou` module into a regular Python shell to integrate Houdini into your existing Python-based scripts.

TOOL SHELVES

The shelf tools are also set up using Python. You can see this code by **RMB-clicking** on any shelf tool and choosing **Edit Tool.**

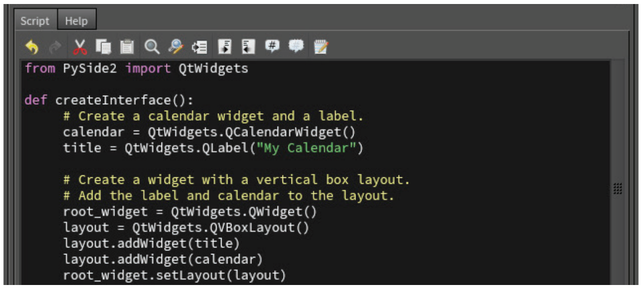


```
import toolutils
import soptoolutils

kwargs['bbox'] = hou.BoundingBox(-0.5, -0.5, -0.5, 0.5, 0.5, 0.5)
sphere = soptoolutils.genericTool(kwargs, 'box')
sphere.parm("type").set("polymesh")
sphere.parm("divrate1").set(2)
sphere.parm("divrate2").set(2)
sphere.parm("divrate3").set(2)
```

PYSIDE/PYQT

The **Python Panel Editor** pane lets you create, edit and delete PySide2 or PyQt5 interfaces. The editor also lets you manage the entries in the Python Panel interfaces menu as well as the entries in the Houdini pane tab menu. The panel comes with some sample code that you can try out for yourself.



```
from PySide2 import QtWidgets

def createInterface():
     # Create a calendar widget and a label.
     calendar = QtWidgets.QCalendarWidget()
     title = QtWidgets.QLabel("My Calendar")

     # Create a widget with a vertical box layout.
     # Add the label and calendar to the layout.
     root_widget = QtWidgets.QWidget()
     layout = QtWidgets.QVBoxLayout()
     layout.addWidget(title)
     layout.addWidget(calendar)
     root_widget.setLayout(layout)
```

VEX

VEX is a high-performance expression language used in many places in Houdini, such as writing shaders. VEX evaluation is typically very efficient, giving performance close to compiled C/C++ code.

VEX is not an alternative to scripting, but rather a smaller, more efficient general purpose language for writing shaders and custom nodes. VEX is loosely based on the C language, but takes ideas from C++ as well as the RenderMan shading language.

VEX is used in several places in Houdini:

Modeling – The VEX SOP allows you to write a custom surface node that manipulates point attributes. This can move points around, adjust velocities, change colors. As well, you can group points or do many other useful tasks.

Rendering – Mantra uses VEX for all shading computation. This includes light, surface, displacement and fog shaders.

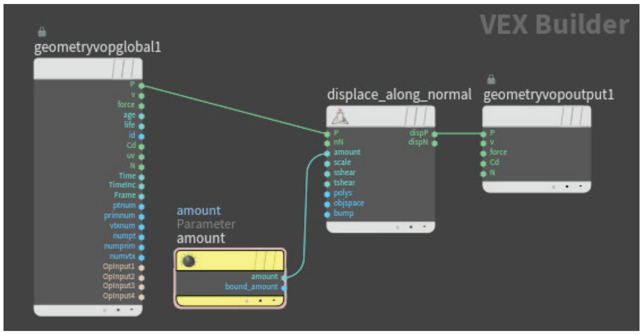
Compositing – The VEX Generator and VEX Filter COPs allows you to write complex custom COPs in VEX. The expressions evaluate very close to C/C++ speeds and run 1000's of times faster than the Pixel Expression COP.

CHOPs – The VEX CHOP lets you create custom CHOPs. The CHOP functions can manipulate arbitrary numbers of input channels and process channel data in arbitrary ways. In some cases, the VEX code can run faster than compiled C++ code.

Fur – Procedural fur behavior is implemented with VEX.

VOPS

If you want to use VEX but don't want to write the code then you can use the VOP context to use a node-based interface. You can do this in the SOP context using an Attribute VOP node that lets you dive in and use VOPs to create VEX code. You can take input geometry and manipulate it.



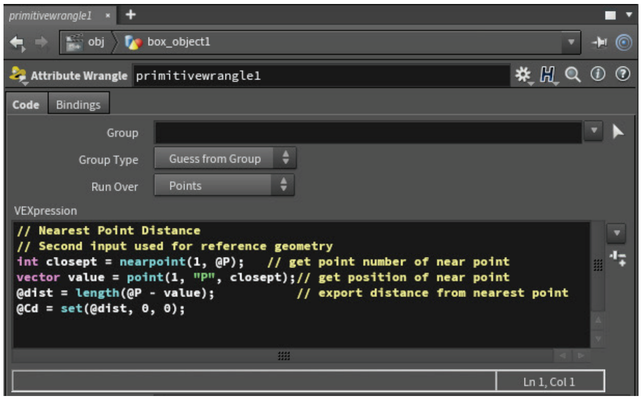
You can use parameter vops to build interface element such as float sliders that are then available at the SOP level. This way you can execute the VEX code without diving back down to the VOP level.

Displacement Amount 0

The VOPs context is designed to give artists an interactive way of creating VEX code. For people with a scripting background, it might make more sense to write the code directly into a Wrangle node.

WRANGLE NODES

You can also use a wrangle node such as the **Attribute Wrangle** which provides a low-level node that lets coders who are familiar with VEX tweak attributes. There are also wrangle nodes for working with channels, volumes and deformations.



Attribute Wrangle primitivewrangle1

Group

Group Type: Guess from Group

Run Over: Points

VEXpression

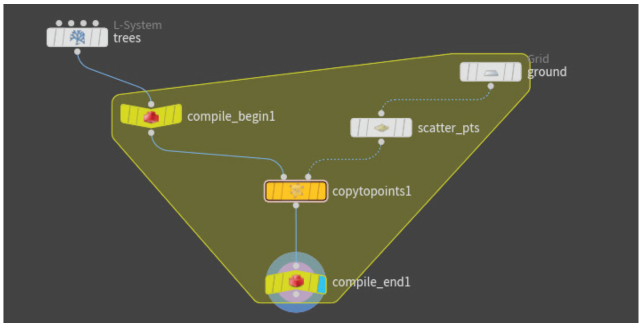
```
// Nearest Point Distance
// Second input used for reference geometry
int closept = nearpoint(1, @P);   // get point number of near point
vector value = point(1, "P", closept);// get position of near point
@dist = length(@P - value);          // export distance from nearest point
@Cd = set(@dist, 0, 0);
```

Ln 1, Col 1

If you are interested in learning how to work with wrangle nodes, you should take a look at **Entagma.com** where you will find lots of great tutorials that generally take a more technical approach to creating content but with an artist's mindset.

COMPILE BLOCKS

In geometry networks [SOPs], you can put a part of the network inside a compiled block that makes it function as efficiently as if you had written code. This imposes a number of restrictions on how the network can work, but can potentially deliver big benefits in the right circumstances.



HOUDINI DEVELOPERS KIT | HDK

An even deeper way to work with Houdini is to use the HDK which is the same comprehensive set of C++ libraries that SideFX programmers use to develop the Houdini family of products. With the HDK, you can create plug-ins which customize different areas in the Houdini interface. Here are some examples of what you can do with the development kit:

- Add custom expression functions
- Add custom commands (hscript or HOM)
- Add custom operators (SOPs, COPs, DOPs, VOPs, ROPs, CHOPs, and even Objects)
- Add output nodes to support a non-standard Renderer
- Add custom lighting or atmospheric effects to the renderer

To learn more about working with the HDK, go to the SideFX website and choose **Support > Documentation > HDK**.

HOUDINI DIGITAL ASSETS
Procedural Tool Building

Networks of nodes give Houdini its procedural nature and define a recipe that can be applied over and over. Houdini Digital Assets let you wrap up these networks to create custom tools and smart assets. These artist-built tools can be used repeatedly to increase productivity across your studio.

One of the things that Houdini's node-based workflow is great at is allowing artists to avoid repetitive steps and to generate multiple iterations by simply making changes to an existing network of nodes. This lets you achieve results that are unique without starting the whole process from scratch.

Houdini Digital Assets take this one step further by letting you encapsulate a network or collection of networks into a single node with parameters that have been promoted to the top level. This node is then saved to disk which creates a shareable file that other artists can load into their scenes.

ARTIST BUILT TOOLS

The process of creating a Houdini Digital Asset works with the interactive tools in Houdini. You build a high level interface by dragging parameters from nodes to an asset properties panel which allows you to create custom tools without writing any code. This means that technical directors and even artists can build custom tools then deploy them quickly to colleagues.

A Houdini Digital Asset might be a procedural prop such as a staircase or a piece of furniture, a visual effect such as an explosion, or a more generalized tool such as a populate tool for scattering objects over a surface. Whether you are creating content specifically for your current project or building a larger toolset for all your projects, your artists can build a collection of Houdini Digital Assets to meet your production needs.

PIPELINE FRIENDLY

When a Houdini Digital Asset is loaded into a scene file, it references the .hda file on disk. This means that changes made to the asset will be picked up automatically by everyone who is referencing that file.

This makes it very easy to deploy updates throughout your pipeline. Now artists can point to a single asset on disk knowing that once it gets updated with the most current iteration, they will immediately have access.

Houdini Digital Asset files can also hold more than just the asset definition. You can store images, geometry files and scripts that are used by the asset. This ensures that all the relevant parts are available when other people work with the asset.

ORBOLT

Orbolt is an online asset marketplace where members of the global Houdini community post a variety of 3D assets, from fully-rigged props, to render-ready visual effects, animatable characters, game assets and more. Some of these are available for free and some are for sale.

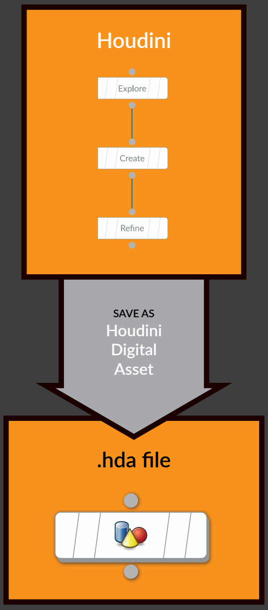
There is a panel in Houdini where all of the Orbolt assets you download or purchase can be stored and made available as you work. These assets can be deployed right away to help you in your projects. You can also post assets to the site and monetize them if you want to make some extra money.

CREATING DIGITAL ASSETS

1 Create nodes and networks in Houdini

2 Package up the networks to be saved out as a Houdini Digital Asset [.hda] file that can be shared with other artists.

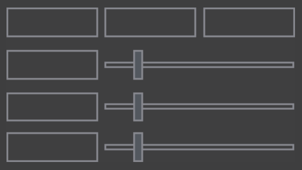
3 Build an interface for your asset by promoting parameters and handles to the top level of the asset.



4 Load the .hda file back into Houdini to use the asset.

Only those parameters promoted to the asset level can be used. All others are locked.

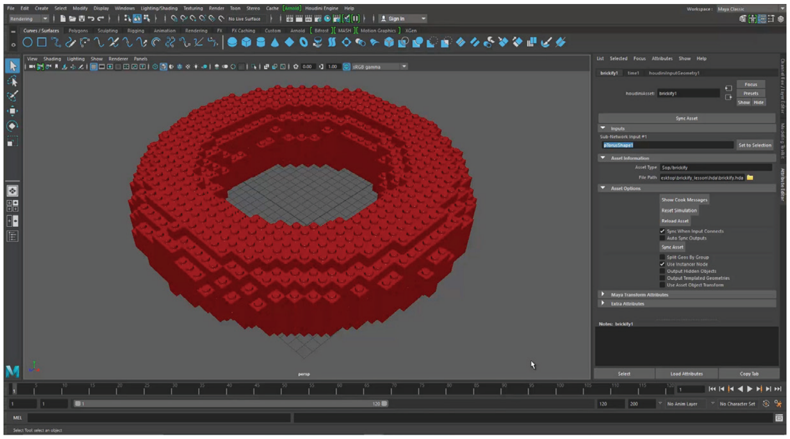
You can use the asset in any number of Houdini scenes. If you make changes to the HDA file then all other assets can be easily synced to the changes.



HOUDINI ENGINE
Sharing with other Apps

Houdini Engine brings a procedural node-based approach to your favorite app. This technology lets you share Houdini Digital Assets with colleagues who can load them directly into 3D apps such as Autodesk® Maya® or Cinema 4D® or into game editors such as Unity® or Unreal Engine.®

The benefits of Houdini Digital Assets can now be enjoyed by artists using other applications thanks to the Houdini Engine. Using plug-ins created using the Houdini Engine API, host applications can load an .hda file and all of the handles and controls will be available. When parameters are set on the asset, Houdini works "under-the-hood" to cook the nodes and networks then deliver the results back to the host.



A Houdini Digital Asset loaded into Autodesk Maya using the Houdini Engine

HOUDINI ENGINE API

This is made possible by an API created by SideFX to allow plug-ins to be created for host applications. Houdini Engine is a flat and small API that is easy to learn and is available on **github** for developers that want to create their own plug-in.

OFFICIAL HOUDINI ENGINE PLUG-INS

There are a number of official Houdini Engine plug-ins that artists can access either through the Houdini installer or online. These have been production tested and can be used confidently by artists and studios.

Each of the plug-ins are designed to create a bridge between the features in a typical Houdini asset and the nature of the host application. For instance a cloud asset that use volumes would work fine in Maya but would not make sense in Unity or Unreal where volumes are not supported.

Here is a list of the official plug-ins:

Unreal Engine - Bring procedural assets into the Unreal editor to set up game art and design levels.

Unity - This plug-in has the same features, works in the game editor and bakes out the results when the game is compiled.

Autodesk Maya - Bring your procedural assets including assets that incorporate fluids, volumes and other effects into Maya.

Autodesk 3DS Max - Coming in 2018!

Cinema 4D - Developed by Maxon, this plug-in is available on their website and brings procedural assets into C4D.

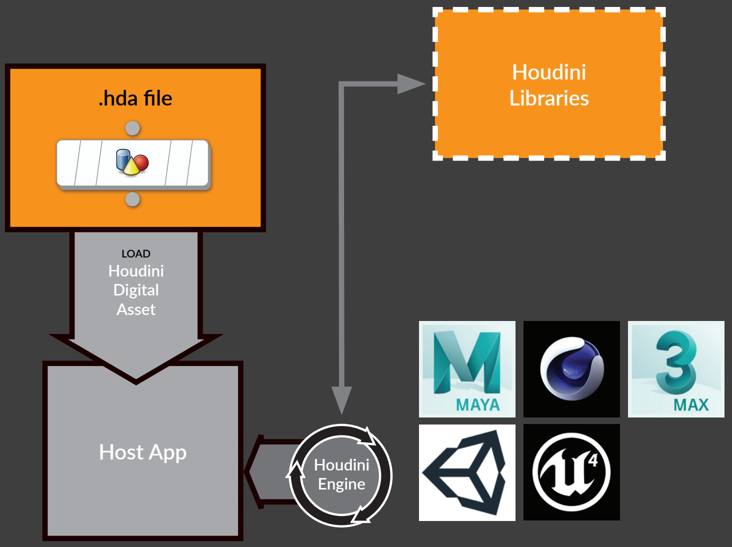
HOUDINI ENGINE PIPELINE

1 Load the .hda file into a host application using the Houdini Engine plug-in.

2 The Host Application accepts the asset and interfaces with the Houdini Engine.

3 The Houdini Engine calls on the Houdini library files to "cook" the nodes and network inside the asset.

4 When an asset is loaded or a parameter is changed then the Engine grabs the Houdini libraries, cooks the nodes. Then delivers the results back to the Host.



Host applications with active Houdini Engine plug-ins include **Autodesk Maya**, **Unity**, **Cinema 4D**, and **Unreal**.

FILM & TV PIPELINE
Animation and VFX

Whether we are talking about live action plates enhanced with visual effects or full CG shots, the ultimate goal of Film and TV projects is to create moving pictures. These pictures are created using pipelines where different kinds of assets come together to create the final result.

Houdini is a full featured package that contributes to all stages of a Film & TV pipeline. From modeling to rendering to animation and final compositing, Houdini has procedural tools that help you get your work completed. Over the years VFX is one area where Houdini is known as the industry standard. SideFX has been honored with several Scientific and Technical Achievement awards including an Award of Merit Oscar for their VFX tools.

Other areas such as procedural modeling, lighting or character work continue to get stronger to the point where a growing request from studios is more skilled Houdini artists.

HOUDINI CORE / HOUDINI FX

There are two commercial versions of Houdini that you use in your pipeline. Houdini Core covers all of Houdini's tools except for dynamics, and Houdini FX has a full toolset. Scenes and VFX created in Houdini FX can be staged, animated, lit and rendered in Houdini Core. This gives you a robust pipeline with Houdini FX licenses for your FX artists and Houdini Core licenses for everyone else.

One solution is for senior technical directors to use Houdini FX to solve a particular production challenge then wrap up the resulting nodes and networks into Houdini Digital Assets. An artist-friendly UI is then built to support the animators and VFX artists who can then use the more cost effective Houdini Core to execute shots.

INTEROPERABILITY

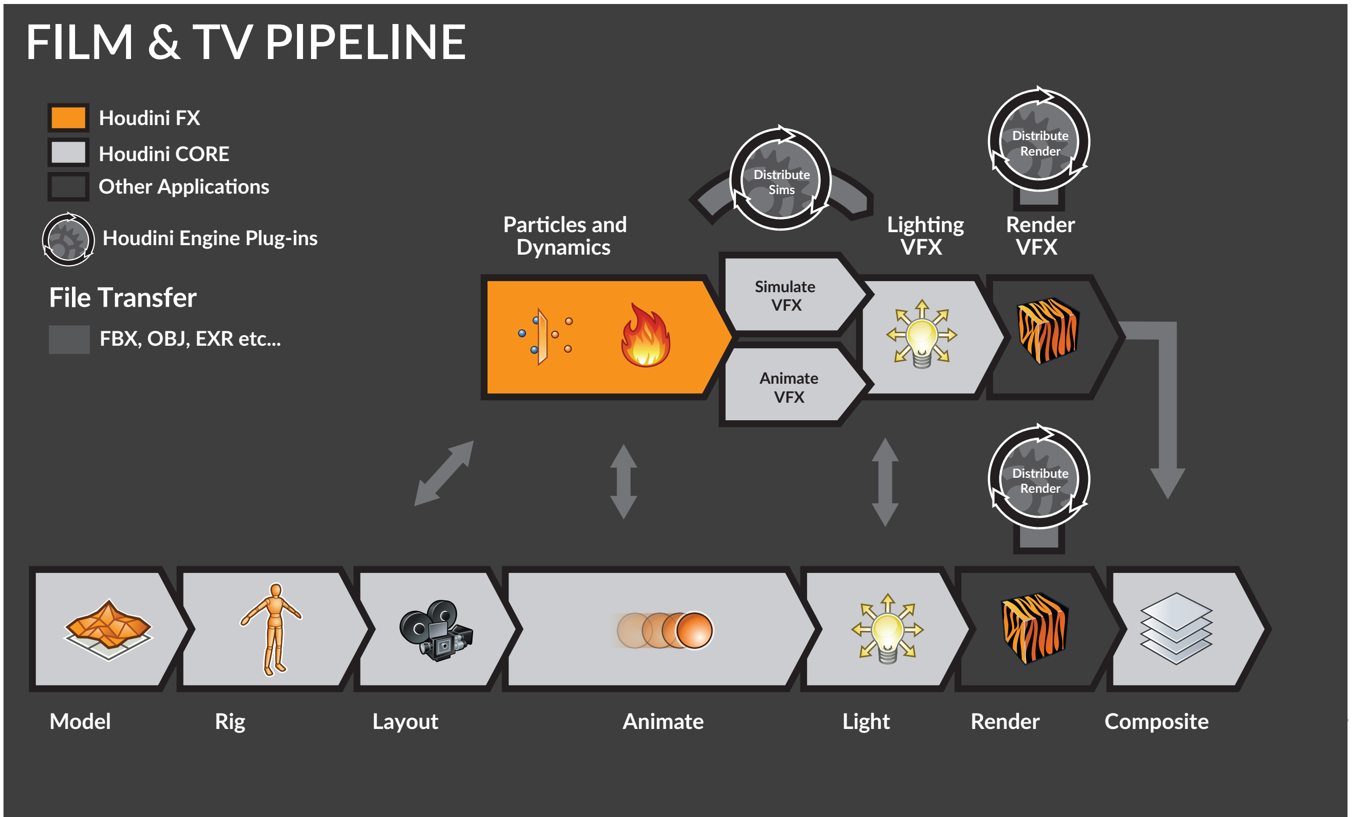
Most studios are equipped with a variety of 3D applications to each handle a different part of the pipeline. Houdini has a lot of strong interoperability tools to allow for this interchange of data. Whether they are using Alembic, FBX or EXR, your artists can easily work back and forth with a wide variety of DCC applications. They can also use the Houdini Engine plug-ins to bring the Houdini Digital Assets into other apps such as Maya or C4D while maintaining the asset's procedural controls.

Smaller studios may want to avoid too much file exchange, especially with tight deadlines, therefore Houdini provides a full featured procedural "pipeline-in-a-box" that can take you through all of the stages under one roof.

DISTRIBUTED RENDERS AND SIMS

Both rendering images and simulating VFX can be time consuming, especially as you aspire towards photorealistic results. For this reason, Houdini lets you distribute both rendering tasks and simulation tasks to a farm where many computers can get the job done much faster than a single computer.

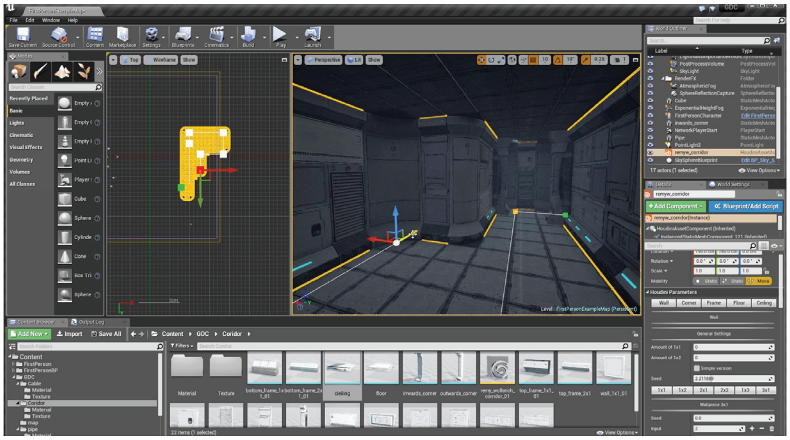
Distributed simulations can also allow you to handle complex effects that would max out the memory on any one computer. By slicing the sim and distributing it, memory is managed without compromising the final result. Studios should definitely consider adding simulation nodes to their farm.



GAMEDEV & VR PIPELINE
Interactive Experiences

In Video Game and Virtual Reality projects, the main focus is creating interactive 3D worlds built using content that is highly optimized for a smooth gameplay experience. This creates a different kind of pipeline compared to rendered out game cinematics which are more like film.

At the core of a games pipeline is a game engine like Unreal Engine or Unity. The engine is where the game art and the game interactions are put together to create a playable experience. Houdini can be used by game artists to create terrain, design and populate levels, build procedural models, build and animate characters and create in-game FX such as fire, fluids and destruction.



A Houdini Digital Asset loaded into UE4 using the Houdini Engine

EXPORTING TO GAME ENGINES

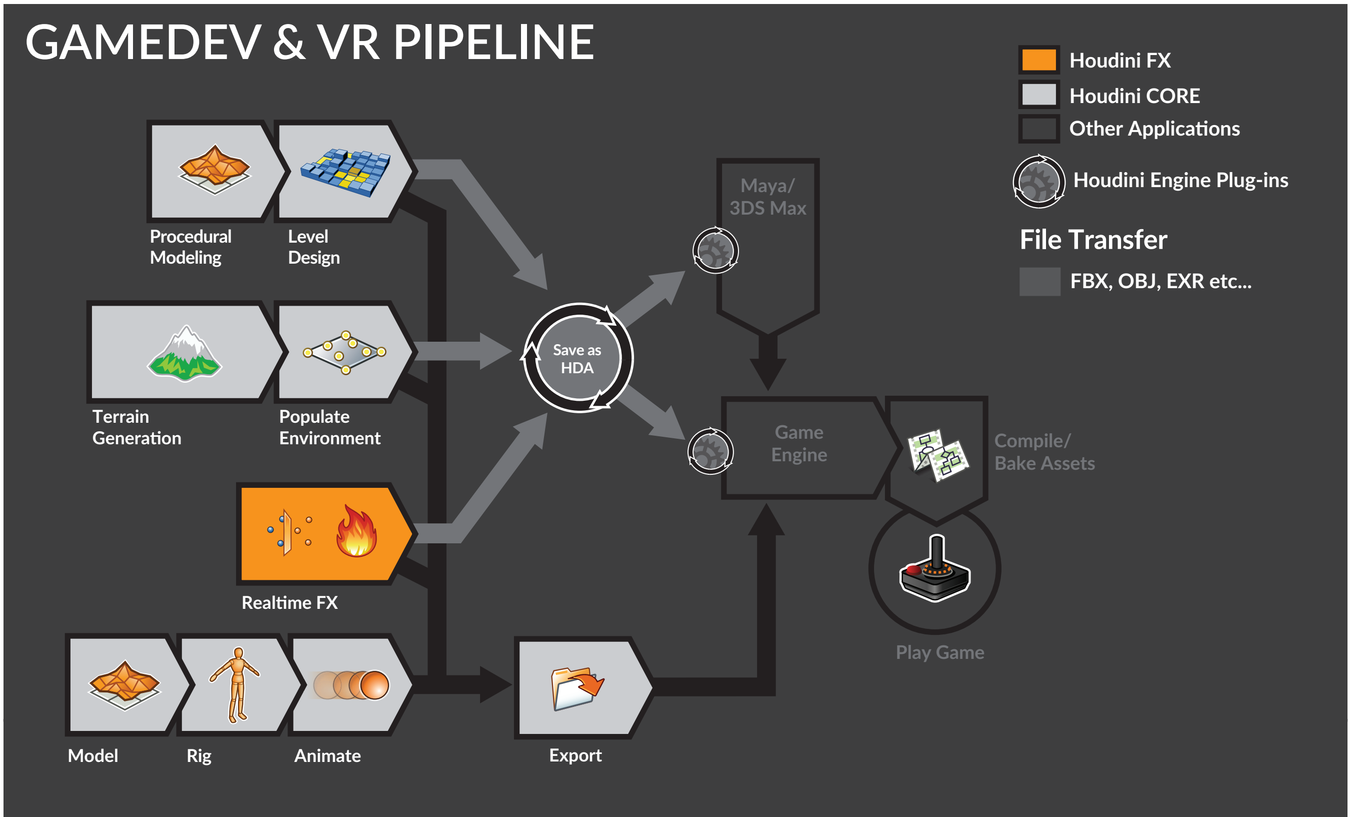
There are two ways of getting content from Houdini to a game engine. The traditional approach is to export out to a format like FBX or OBJ and import this into the engine. You would create procedural systems in Houdini then flatten out the results.

The second approach is to create Houdini Digital Assets and load these into the game engines using the Houdini Engine plug-ins for UE4 and Unity. These assets import into the game editor with their parameters and controls intact. You can therefore make changes inside the game editor and the Houdini Engine works in the background to update the artwork.

This proceduralism is available to game artists inside the editor then when the game is compiled the artwork is baked down. The Houdini Engine is not a runtime solution and you cannot access it as part of the gameplay.

REALTIME FX

Houdini is known for VFX and it is a great tool for creating FX for games. But these FX need to be optimized using techniques such as texture sheets, flowmaps and vertex animation textures. This way the footprint for the effect is as light as possible and does not take away from the frames per second of the game. The Game Development Tools mentioned earlier in this document have been designed to support these kinds of workflows.



Products and Licensing

As you begin working with Houdini, it is useful to understand what Houdini products are available for you to work with. Whether you are a large studio, small studio or a team of indies just getting started, there are different Houdini products to suit your needs. There are also versions of Houdini for school labs and for students who want to learn for free.

COMMERCIAL LICENSES

Houdini Core – Designed for modelers, lighters, character riggers, animators and game artists, Houdini Core also includes features such as compositing and motion editing. Scenes created in Houdini FX can be opened and rendered in Houdini Core which makes it an ideal lighting tool for your VFX.

Houdini FX – Houdini FX combines superior performance and dramatic, ease-of-use to deliver a powerful and accessible 3D experience. With its procedural node-based workflow, Houdini FX lets you create more content faster to reduce time lines and enjoy enhanced flexibility in all your creative tasks.

Houdini Engine – Houdini Engine lets you load Houdini Digital Assets into other digital content creation applications, such as Autodesk® Maya,® and Cinema 4D, or game editors such as Unity® and UE4.® Houdini Engine also gives you command-line access to run in batch mode to batch process renderings and distributed dynamic simulations.

INDIE LICENSES

Houdini Indie – Houdini Indie makes all of Houdini's animation and VFX tools available under a limited commercial [less than $100K USD] license to animators and game makers who want to use Houdini during the incubation stage of their business.

Houdini Engine Indie – Your Houdini Engine Indie license can be used to run Houdini Indie in batch mode or to load Houdini Digital Assets into other content creation apps such as Autodesk® Maya,® and Cinema 4D, or game editors such as Unity® and UE4® under a limited commercial license.

LEARNING LICENSES

Houdini Education – Houdini Education is a full-featured version of Houdini FX designed for use by schools and training centres. Designed to be used in labs and classrooms, Houdini Education will open files created by students using Houdini Apprentice.

Houdini Apprentice – Houdini Apprentice is a free version of Houdini FX which can be used by students, artists and hobbyists to create personal non-commercial projects. With Houdini Apprentice, you have access to virtually all of the features of the award-winning Houdini FX to develop your skills and work on personal projects. Apprentice lets you save to disk and render out with a word mark.

If a student wants to work without the word mark and they don't have access to Houdini Education at their school then we recommend working with Houdini Indie.

LICENSE TYPES

Workstation [Node-Locked] – This license type can be used on a single computer and can only be moved a couple of times if you are setting up on a new computer.

Local/Global Access [Floating] – These licenses can be set up on a server and shared with a team of artists. When an artist starts Houdini a license is checked out of the server as long as there is one available. Local licenses are designed for a single studio and global licenses are for sharing between studios in different locations.

INSTALLING LICENSES

Once you have acquired a license, you will install it by opening up the **Houdini License Administrator [hkey]** application. From there you can choose **File > Install Licenses**. You will be asked for a log in and a password that will match the ones you set up on the SideFX.com website.

Indie and Education licenses actually involve three parts that must all be installed for it to work correctly. **Apprentice licenses** can be set up during the installation process.

Local and Global Access licenses can be installed using this method on a central server. You will then need to make that server available to anyone who needs access to the licenses.

You can also view your licenses on the SideFX.com website by clicking on your avatar in the top right and choosing **Services**. You can then click on the **Manage Licenses** link.

ANNUAL UPGRADE PLAN

For visual effects studios, games studios and 3D artists who want to maximize their investment in Houdini, the Annual Upgrade Plan provides key advantages such as production-level technical support and access to full and dot releases containing the latest software enhancements as well as daily builds containing bug fixes.

SIDEFX SUPPORT

All customers including Apprentice customers can contact SideFX using our email support system to discuss installation and licensing issues. After that, only Annual Upgrade Plan and Commercial Rental Customers may contact our support team to discuss more in depth production issues.

Our Support Specialists can be contacted directly via **support@sidefx.com** . Be sure to include the following information in your email:

- Your Operating system [Windows XP, etc.]
- Version and Build Number of Houdini
- Summary of the installation issue and a diagnostic file if you are having a licensing issue.

To learn more about SideFX support visit **SideFX.com/support**.

Comparison Chart

	LEARNING		INDIE	COMMERCIAL	
PRODUCT	EDUCATION	APPRENTICE	HOUDINI INDIE	HOUDINI FX	HOUDINI CORE
INTENDED USER	Schools	Students | Hobbyists	Indies | Freelancers	Studios | Commercial Artists	
PRICING	$75 USD PER YEAR	FREE	$269 USD PER YEAR	Visit SideFX.com	
OS	MAC OSX, Windows and LINUX				
FEATURES					
Modeling	✓	✓	✓	✓	✓
Character	✓	✓	✓	✓	✓
Animation	✓	✓	✓	✓	✓
Lighting	✓	✓	✓	✓	✓
Rendering	✓	✓	✓	✓	✓
Terrain	✓	✓	✓	✓	✓
Compositing	✓	✓	✓	✓	✓
Volumes	✓	✓	✓	✓	✓
Pyro FX	✓	✓	✓	✓	
Fluids	✓	✓	✓	✓	
Rigid Bodies	✓	✓	✓	✓	
Particles	✓	✓	✓	✓	
Cloth Dynamics	✓	✓	✓	✓	
Wire Dynamics	✓	✓	✓	✓	
Crowds	✓	✓	✓	✓	
LICENSING	Non-Commercial		Limited Commercial	Commercial	
Workstation [Node-Locked]	-	✓	✓	✓	✓
Local/Global Access [Floating]	✓	-	-	✓	✓
USER INTERFACE					
Houdini GUI Access	✓	✓	✓	✓	✓
Command-Line Access	✓	✓	✓	✓	✓
GUI Watermark	Unobtrusive	Unobtrusive	Unobtrusive	No	No
Plug-in Support	✓	✓	✓	✓	✓
HOUDINI ENGINE					
Houdini Engine Plug-ins	✓	No	✓	✓	✓
Create Assets for Engine	✓	No	✓	✓	✓
Create Assets for Orbolt	✓	✓	✓	✓	✓
RENDERING					
Mantra Tokens	Unlimited	1	1	Unlimited	Unlimited
3rd Party Rendering	✓	No	No	✓	✓
Render Watermark	No	✓	No	No	No
Resolution: Animation	Unlimited	1280x720	4096 x 4096	Unlimited	Unlimited
Resolution: Stills	Unlimited	1280x720	Unlimited	Unlimited	Unlimited
FILE FORMATS*					
SCENE					
.hip	.hipnc/.hiplc	.hipnc	.hiplc	✓	✓
.hda	.hdanc/.hdalc	.hdanc	.hdalc	✓	✓
GEOMETRY					
FBX	✓	IMPORT	✓	✓	✓
Alembic	✓	IMPORT	✓	✓	✓
.bgeo	✓	✓	✓	✓	✓
.obj	✓	✓	✓	✓	✓
IMAGES					
.pic	✓	.picnc	.piclc	✓	✓
.openexr	✓	watermarked	✓	✓	✓
.tif	✓	watermarked	✓	✓	✓
.jpg	✓	watermarked	✓	✓	✓
.mov	✓	watermarked	✓	✓	✓

HOUDINI FOUNDATIONS

MODEL, ANIMATE, RENDER

Welcome to Houdini. In this lesson you will start from scratch to model, animate, simulate and render a soccer ball (also known as a football in many parts of the world). You will create a classic bouncing ball animation using the principles of squash and stretch, apply textures and materials, add lights and cameras, and explore the use of dynamics to simulate a group of bouncing soccer balls.

These tasks will introduce you to many different parts of Houdini as you create your first Houdini scene, explore the interface and discover some of its most important tools. You will learn how to work interactively in the **Scene View** and how to use the **Network View** to manage your nodes as you refine your model and build your animation rig. You will also set up materials and textures and render using Houdini's built-in renderer **Mantra,** and finally create a **rigid body simulation**. For **Game Artists**, you will also learn how to **bake out** the hi-res model into texture maps on a low-res model and then how to **export your simulation** for use in the UE4 game engine.

LESSON GOAL

Model, Animate and Render a soccer ball using Houdini's procedural node-based workflow

WHAT YOU WILL LEARN

- How to work with the **View Tools**
- How to use **Shelves**, **Radial Menus** and the **Tab** key
- How to create **Geometry**
- How to work with **Nodes and Networks**
- How to set up **Custom Attributes** and a **For-Each Loop**
- How to set up **Materials** and **Texture UVs**
- How to **Set Keyframes** and add **Motion FX**
- How to use **Rigid Body Dynamics**
- How to **Render** with Mantra

LESSON COMPATIBILITY

Written for the features in Houdini 16.5.378+

The steps in this lesson can be completed using the following Houdini Products:

Houdini Core	Parts 1-11
Houdini FX	✔
Houdini Indie	✔
Houdini Apprentice	✔
Houdini Education	✔

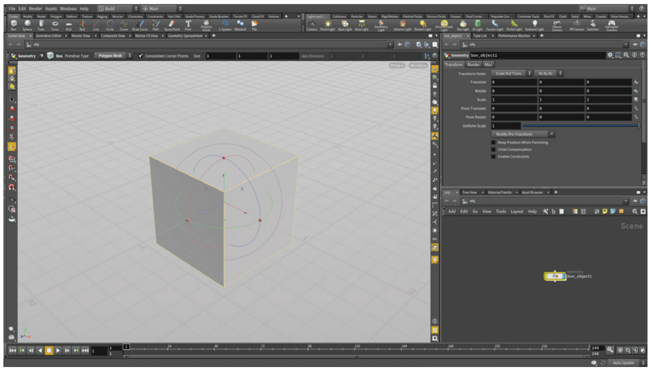
Document Version 1.0 | February 2018

PART ONE:
Explore the Houdini UI

Learning how to work with the Houdini workspace is an important way to get started. The **Viewport** lets you create objects interactively, the **Parameter Pane** lets you review and edit node properties and the **Network Editor** lets you work directly with the nodes.

PROJECT FILES

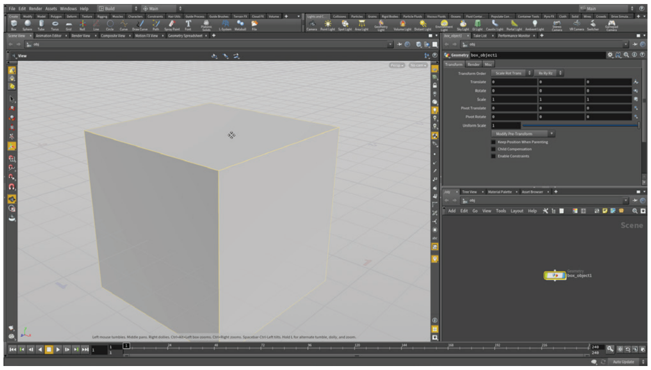
Go to the **soccerball tutorial** page on **SideFX.com**, where you likely got this document, to download the ***intro_lesson*** directory. Put it into the **Houdini Projects** directory which you can find in either the **home** directory or the **documents** directory.



01

In the viewport, **press c** to bring up a radial menu. From this menu, choose **Create > Geometry > Box**. Your cursor now shows the outline of a box waiting to be placed in the scene. **Press Enter** to place it at the origin.

This creates a box in the Scene view, adds a node in the Network editor and shows the object parameters in the Parameter pane. As you work through this project, you will touch on all of these interface elements.



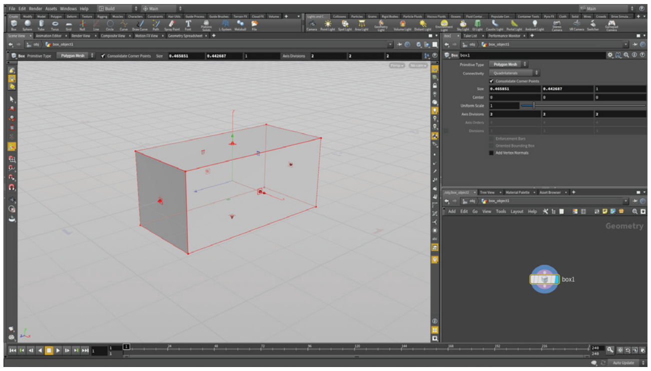
02

You can now explore the **View** tool in Houdini. Press the following hotkeys:

Tumble	Spacebar or Alt[Opt] - LMB click-drag
Pan	Spacebar or Alt[Opt] - MMB click-drag
Dolly	Spacebar or Alt[Opt] - RMB click-drag

In some cases, you will want to home in to get your bearings. There are some hotkeys for that as well:

Home Grid	Spacebar -H
Home All	Spacebar - A
Home Selected	Spacebar - G



03

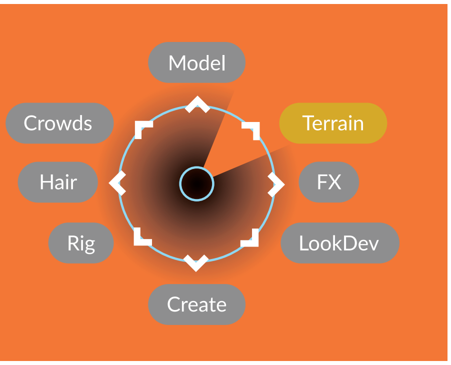
With the object selected, **press i** to go to its geometry level. Use the **Shift** key to drag on handles to make it longer along z axis around the origin.

When an object is created in Houdini, there is an **Object level** which is where you manage the transformations of the object and a **Geometry level** where you define its shape. Pressing **i** brought you down into the geometry level of this object. You can also get there by double-clicking on the object node in the Network editor. To get back to the Object level you can **press U**.

RADIAL MENUS

One way to access tools in Houdini is radial menus which you can access using the X, C and V hotkeys. Each of these brings up a radial menu with lots of options for you to choose from. The main focus of each menu is as follows:

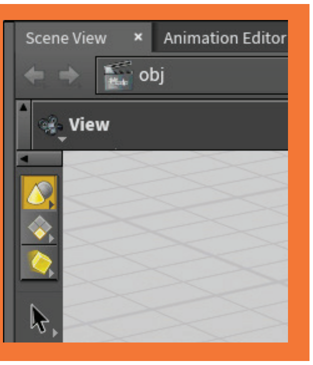
Snapping	**X**
Main (or Custom)	**C**
View	**V**

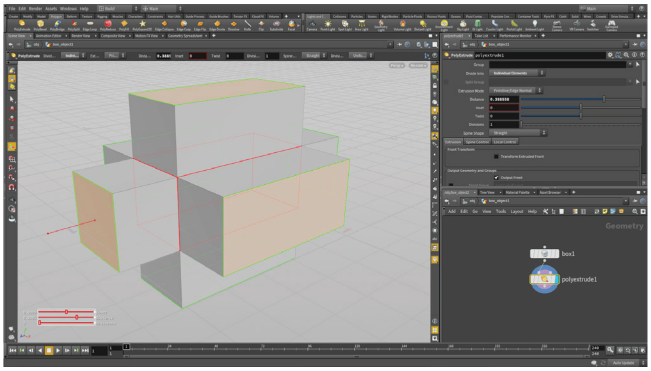


SELECTION HOTKEYS

If you are using the **Select, Move, Rotate, Scale** or **Handles** tools, the following hotkeys will determine your selection mode as well as which level you will be working at. The tools in the toolbar give you access to options that aren't available with hotkeys:

Objects	Object Level	1
Points	Geometry Level	2
Edges	Geometry Level	3
Primitives/Faces	Geometry Level	4
Vertices	Geometry Level	5



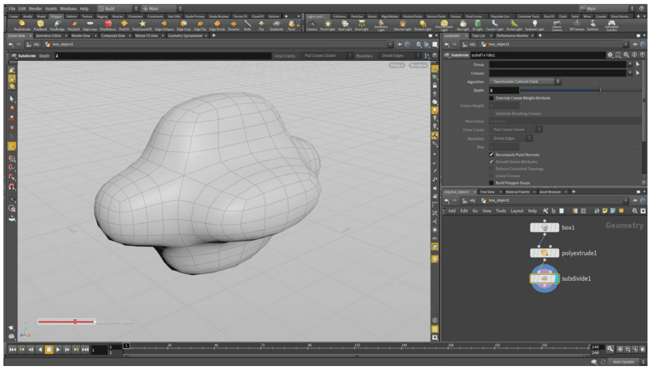


04

Press 4 to access Primitive selection then **n** to select all the faces then click on the **Polygon** shelf and click on the **Poly Extrude** tool.

In the Parameter pane, set **Divide Into** to **Individual Elements** and use the handle to set the **Distance** to around **0.4**. This extrudes all the faces of the box along the normals of each primitive.

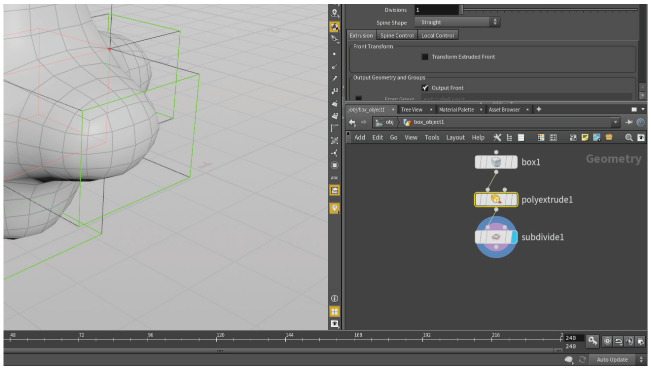
You can see that there are now two nodes in the Network view. Each step you take in Houdini creates a node that you can work with to refine your scene.



05

Press **n** to select all of the new faces and press **Tab** and begin typing **sub...** then select **Subdivide** from the list. The Tab key is another way to access tools in Houdini. Typing the tool name lets you focus the list making it easier to find what you want without navigating the submenus.

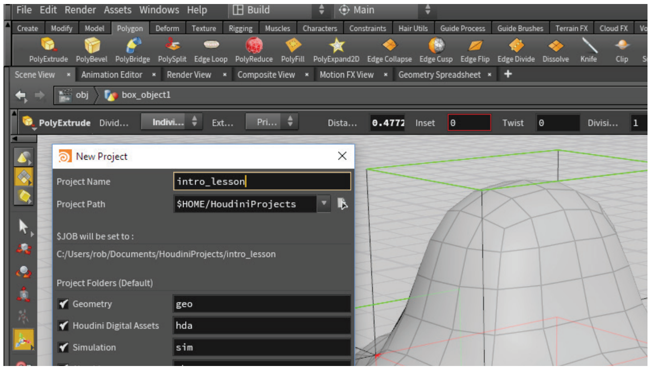
In the Parameter pane, set **Depth** to **2**. This subdivides the geometry to create more polygons. Houdini also has a subdivision display option at the Object level which you can use to see subdivisions without actually adding any geometry, but in this case we do want to create more polygons.



06

Select the different nodes in the chain. The handles for each of the nodes appear as you select them but the display remains on the final shape. Set the **Display Flag** on each of the nodes to change which node is the display node. You can also try some of the other flags such as **Bypass** or **Template. Wiggle** the *polyextrude* node out of the network then drop it back in.

At the end, return everything to normal and set the **Display flag** on the *subdivide* node. This is very important. The Display flag determines what you will see at the object level. **Always check to make sure you have the right display flag set!**


File Edit Render Assets Windows Help
Build
Main
Create Modify Model Polygon Deform Texture Rigging Muscles Characters Constraints Hair Utils Guide Process Guide Brushes Terrain FX Cloud FX
PolyExtrude PolyBevel PolyBridge PolySplit Edge Loop PolyReduce PolyFill PolyExpand2D Edge Collapse Edge Cusp Edge Flip Edge Divide Dissolve Knife Clip
Scene View Animation Editor Render View Composite View Motion FX View Geometry Spreadsheet
obj box_object1

PolyExtrude Divid... Indivi... Ext... Pri... Dista... 0.4772 Inset 0 Twist 0 Divisi... 1

New Project

Project Name intro_lesson

Project Path $HOME/HoudiniProjects

$JOB will be set to :
C:/Users/rob/Documents/HoudiniProjects/intro_lesson

Project Folders (Default)

☑ Geometry geo

☑ Houdini Digital Assets hda

☑ Simulation sim

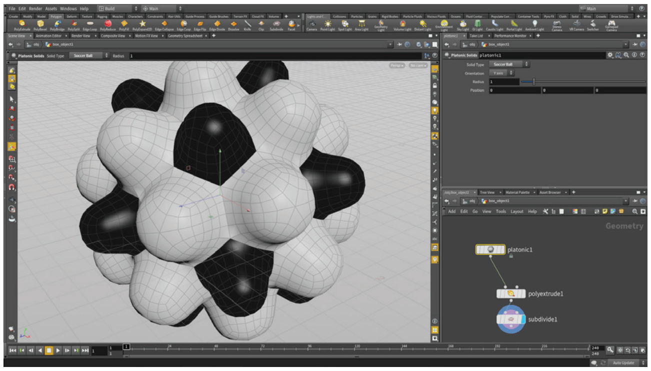
07

Select **File > Set Project.** Find the *intro_lesson* directory that you downloaded earlier and press **Accept**. This makes this project directory and its sub folders the place for all the files associated with this shot.

Select **File > Save As...** You should be looking into the new *intro_lesson* directory. Set the file name to *soccerball_01.hip* (or *football_01.hip* if you would prefer) and click **Accept** to save.

PART TWO: Adding the Soccerball Geometry

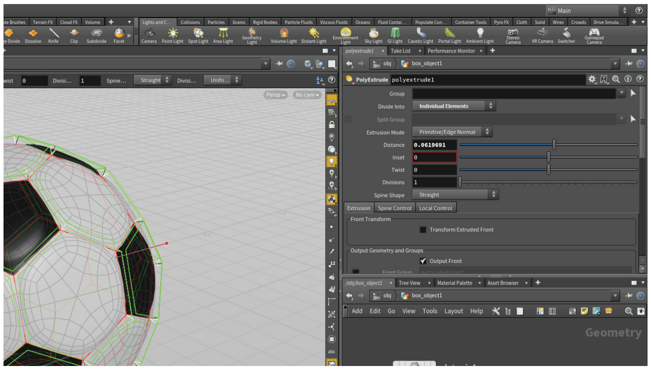
The extruded box we have here doesn't look like a soccer ball but not to worry. Using Houdini's procedural approach, you can replace the box node with soccerball geometry. From there you will add some more Houdini nodes to make it look more like a proper soccerball. This ability to swap out input nodes gives you lots of flexibility as you develop your projects.



01

In the **Network editor**, use the **Tab** key to add a **Platonic solids** node to the network. Click to place it down near the top of the chain. Wire the *platonic* node into the *polyextrude* node. In the parameter pane, set **Solid Type** to **Soccer Ball**. Select and delete the *box* node.

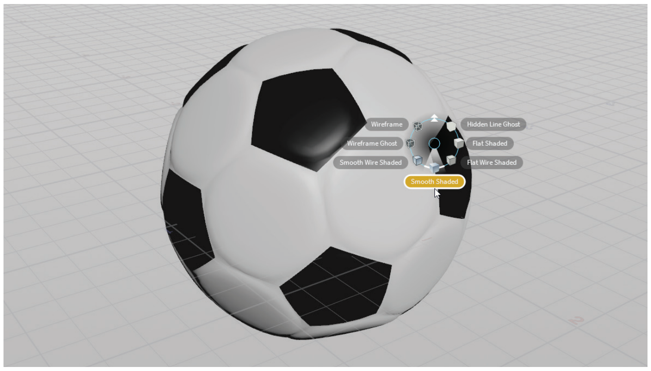
Because of Houdini's procedural nature, it is often possible to replace an input node and have the whole network function properly. This gives you flexibility as you work and if you don't like the results after the change then you can always wire back the original shape.



02

Select the *polyextrude* node and use the handle in the viewport to set a smaller **Depth**. You can also set the parameter value in the Parameter pane. This creates a better look for the soccerball. Remember that even though we are viewing the *subdivide* node, selecting the *polyextrude* node gives us access to its handles and parameters.

You might think that with this primitive type we are all set but it is really just a truncated icosahedron with flat faces. We need a round soccerball so we will have to put a little more work into it.



03

Press **V** in the viewport and from the **Radial Menu,** select **Shading > Smooth Shaded**. You can also use the menu in the top right of the viewport to change your shading.

This soccerball looks like a cheap plastic ball rather than a proper leathery soccerball. You are now going to branch off and add more nodes to get a better look.

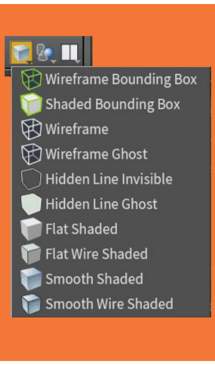
After analyzing it, set the shading back to **Smooth Wire Shaded**.

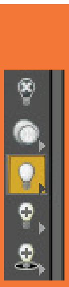
SHADING OPTIONS

There are a number of **Shading Options** available from either the **View** radial menu or the **Shading** menu in the top right of the Viewport.

For the shading of your objects, the lighting is determined by the **Display Options** on the right edge of the Viewport. You can choose from a headlight, normal lighting or high quality lighting with shadows.

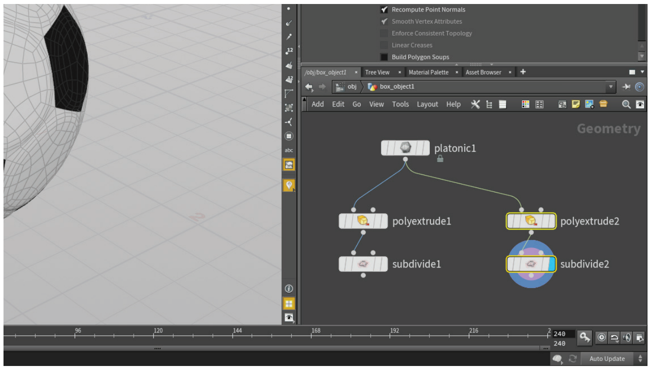
To quickly toggle from your shaded view to wireframe press the **W** key.





PART THREE: Create a Realistic Soccerball

There are often many ways to solve a problem using Houdini's nodes and networks. To encourage exploration, you can branch off and try new things without losing your original idea. You are now going to add more detail to your soccerball by copying some of the nodes to create a new chain. The display flag determines which of the chains you are seeing in the Viewport.

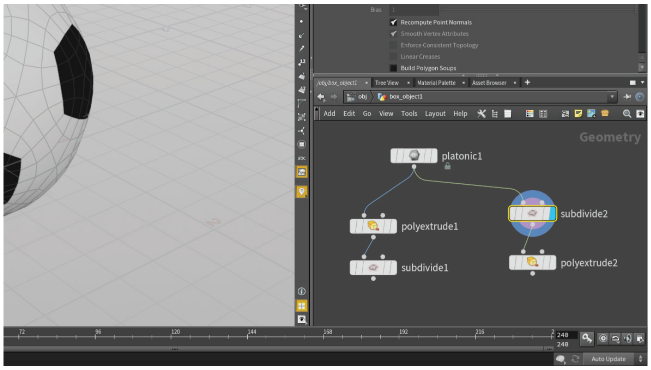


☑ Recompute Point Normals
☑ Smooth Vertex Attributes
□ Enforce Consistent Topology
□ Linear Creases
□ Build Polygon Soups

01

Select the *polyextrude* and *subdivide* nodes then press the **Alt** key and **click-drag** to make copies. These new nodes will still be wired into the *platonic* node but will create a separate branch to the network.

You can decide which of these you want to view using the display flag. It has automatically shifted to our new chain when we made the copies. You will now rewire these nodes to add detail to the soccer ball.

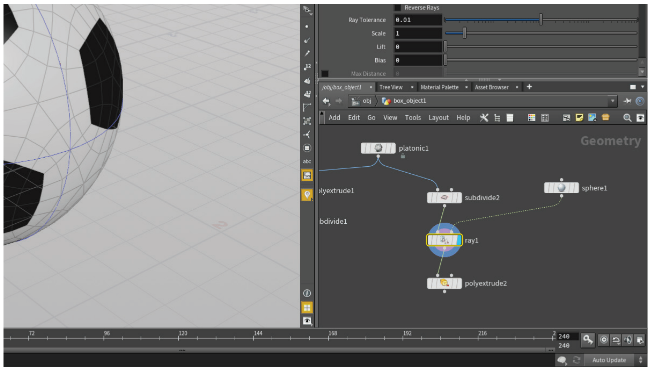


☑ Recompute Point Normals
☑ Smooth Vertex Attributes
□ Enforce Consistent Topology
□ Linear Creases
□ Build Polygon Soups

02

Press the **Y** key and drag across the line connecting the *subdivide* node and the *polyextrude* node to break the connection. Now move the *subdivide* node in between the *platonic* solid node and the *polyextrude* node. You can drop it on the connecting wire and it will insert itself in automatically. If not then jiggle it a little until it finds the connection.

This will give more detail to the sphere before it is extruded. If you set the display on the *polyextrude* node you will see that it doesn't look right because we have lots of small polygons being extruded but you will fix that in the following steps.



□ Reverse Rays
Ray Tolerance 0.01
Scale 1
Lift 0
Bias 0
Max Distance 0

03

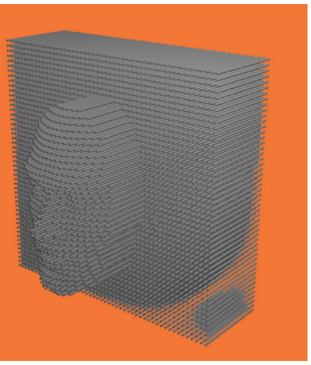
You can see that the sphere isn't perfectly round. Use the **tab key** in the network editor to add a **Ray** node and wire it in after the *subdivide*. Now add a **sphere** node to the network and wire it into the second input on the *ray* node. This will project the subdivided ball onto a perfect sphere.

This is a very powerful node in Houdini that lets you project points from one piece of geometry onto another. It is the perfect solution to our problem of a subdivided soccerball that wasn't truly round.

THE RAY NODE

The ray node is a tool that projects points out to another piece of geometry. This is similar to the pinboard toy you played with as a kid. In fact, this is the node you would use to set up a pinboard in Houdini.

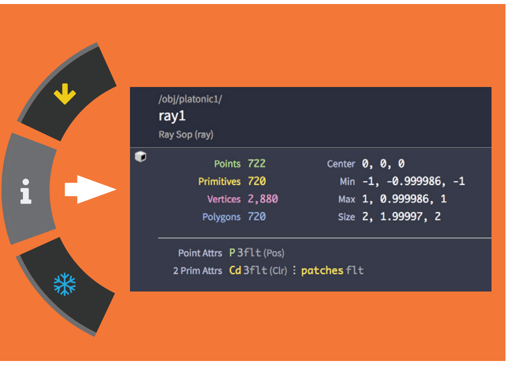
GETTING HELP | To learn more about each node, you can click on the ? help button in the top right of the Parameter pane to open up the node's online documentation. You can also hover over the tool in the shelf and **press F1**. In many cases, there are sample files that you can open in Houdini to learn what the node can do.



WORKING WITH ATTRIBUTES

Attributes can be assigned to Points, Primitives or Vertices. Some typical types of attributes include color (Cd) or UVs. You can see the attributes at any point in your chain by mousing over a node and choosing the **i** from the radial menu. You can also review the attribute values in the **Geometry Spreadsheet** panel.

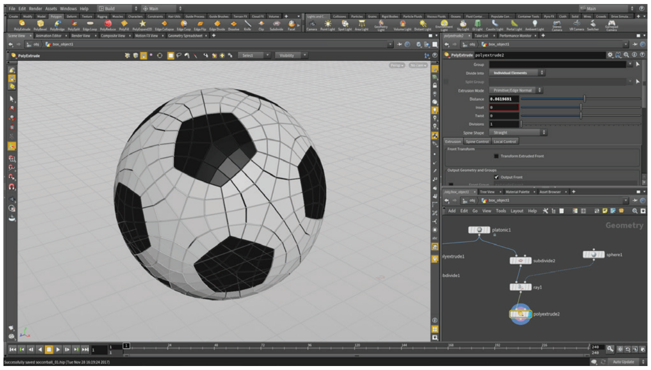
In this lesson you will be creating a custom attribute called ***patches*** which will help you later on in the network.



/obj/platonic1/
ray1
Ray Sop (ray)

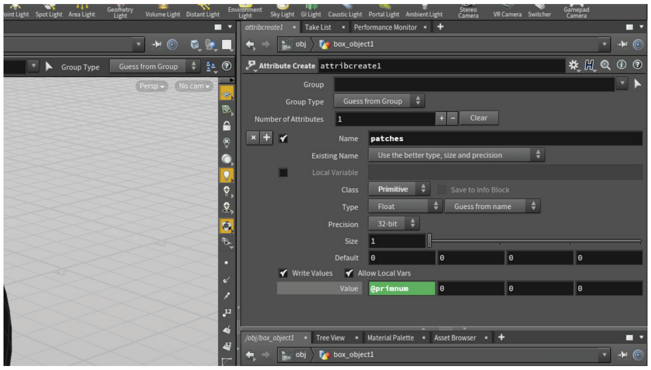
```
Points      722        Center  0, 0, 0
Primitives  720        Min     -1, -0.999986, -1
Vertices    2,880      Max     1, 0.999986, 1
Polygons    720        Size    2, 1.99997, 2

Point Attrs    P 3flt (Pos)
2 Prim Attrs   Cd 3flt (Clr) ⋮ patches flt
```



04 Set the **Display flag** on the *polyextrude* node. With **Divide into** set to **Individual elements** all the small polygons are extruded but we don't want that. But if we change it to **Connected Components** then all the polygons are extruded which is also not what we want yet.

Set it to **Connected Components**. We need a way for this network to let us extrude the original patches of the soccerball but after the ball has been subdivided. We can do this using the primitive numbers on the original geometry.

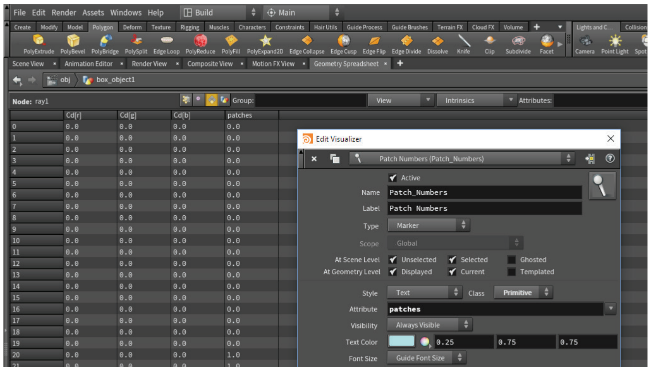


Attribute Create attribcreate1

- Group:
- Group Type: Guess from Group
- Number of Attributes: 1
- ☑ Name: patches
- Existing Name: Use the better type, size and precision
- ■ Local Variable
- Class: Primitive □ Save to Info Block
- Type: Float, Guess from name
- Precision: 32-bit
- Size: 1
- Default: 0 0 0 0
- ☑ Write Values ☑ Allow Local Vars
- Value: @primnum 0 0 0

05 Add an **AttributeCreate** node after the *platonic* node. – Set its **Name** to *patches* and its **Class** to **Primitive.** Now set the first of the **Value** fields to **@primnum.** This expression takes the primitive number attribute and turns it into a new attribute called *patches*.

With the *attributecreate* node selected, click on the **Geometry Spreadsheet** tab next to the main viewport. Click on the **Primitive** button and you can see the primitive numbers on the left, three color attributes which show the color of the patches and the *patches* attribute which matches the primitive numbers.



Node: ray1

	Cd[r]	Cd[g]	Cd[b]	patches
0	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0
8	0.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0
10	0.0	0.0	0.0	0.0
11	0.0	0.0	0.0	0.0
12	0.0	0.0	0.0	0.0
13	0.0	0.0	0.0	0.0
14	0.0	0.0	0.0	0.0
15	0.0	0.0	0.0	0.0
16	0.0	0.0	0.0	0.0
17	0.0	0.0	0.0	0.0
18	0.0	0.0	0.0	0.0
19	0.0	0.0	0.0	0.0
20	0.0	0.0	0.0	1.0

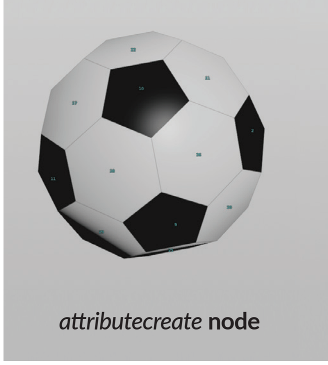
Edit Visualizer

Patch Numbers (Patch_Numbers)

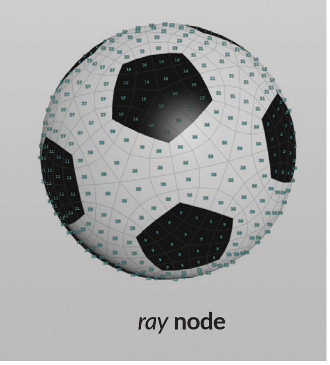
- ☑ Active
- Name: Patch_Numbers
- Label: Patch Numbers
- Type: Marker
- Scope: Global
- At Scene Level: ☑ Unselected ☑ Selected ■ Ghosted
- At Geometry Level: ☑ Displayed ☑ Current ■ Templated
- Style: Text Class: Primitive
- Attribute: patches
- Visibility: Always Visible
- Text Color: 0.25 0.75 0.75
- Font Size: Guide Font Size

06 Now click on the *ray* node. This attribute will be carried forward when the shape is subdivided. You can now see there are a lot more primitives but the *patches* attribute only goes as high as 31 and then it goes back to 0.

Too see this in the viewport, **RMB-click** on the **Visualizer display** button on the **Display Options** bar and click on the **+ Plus sign** next to **Scene** and choose **Marker.** In the Edit Visualizer panel, set **Name** and **Label** to *Patch_Numbers*, set **Type** to **Marker**, **Class** to **Primitive** and **Attribute** to *patches*.



attributecreate node



ray node

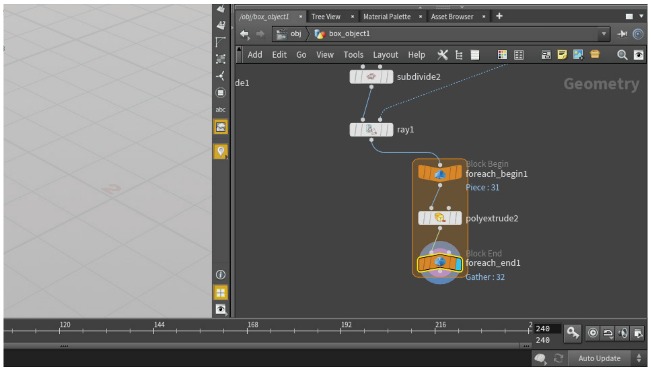
07 Now make sure the **Visualizer display** button is on and you will see the patch values on the soccerball in the viewport. You can see that the prim number from the original platonic solid have been transferred to the subdivided faces.

Change the display flag to different nodes to see the relationship. This information will be used to polyextrude the patches properly using a for-each loop.

Save your work. These steps have been a bit abstract and probably feel a bit too technical, but don't worry the payoff is coming.

PART FOUR:
The For-Each Node

Now you get to see the magic as the attributes you just created are fed into a for-each loop that will extrude each of the original patches using the subdivided geometry. This will give thickness to the patches that provide a better look for the soccerball once you subdivide it once more.

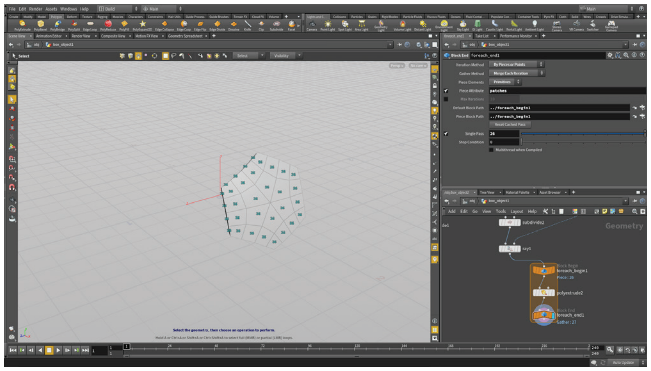


subdivide2
ray1
Block Begin
foreach_begin1
Piece : 31
polyextrude2
Block End
foreach_end1
Gather : 32

01

In the Network editor, press **tab** and start typing **For-each Named Primitive** to access two nodes that you can then place into the scene. Wire the *foreach_begin* between the *ray* node and the *polyextrude* node and them the *foreach_end* after the *polyextrude* node. Select the *foreach_end* node and in the parameter pane leave **Piece Elements** set to **Primitives** and set **Piece Attribute** to *patches*. Set display on the *foreach_end* node.

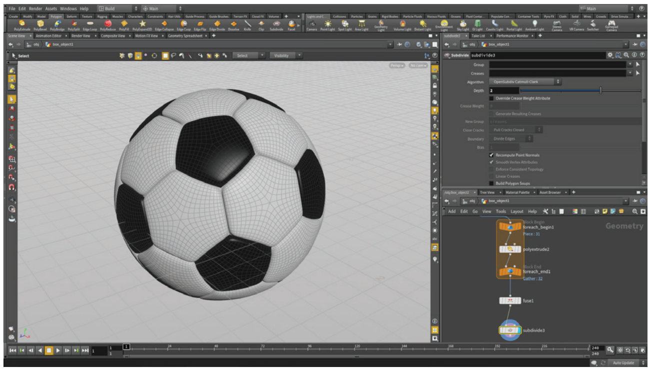
You should now see the original patches being extruded together based on the *patches* attribute. If they are not make sure **Divide to** is set to **Connected Components** on the *polyextrude* node.



02

Click the checkbox next to **Single Pass** to explore what is happening. Drag on the slider to watch as each of the patches is polyextruded individually. You can also set values higher than 10 to see more of the patches.

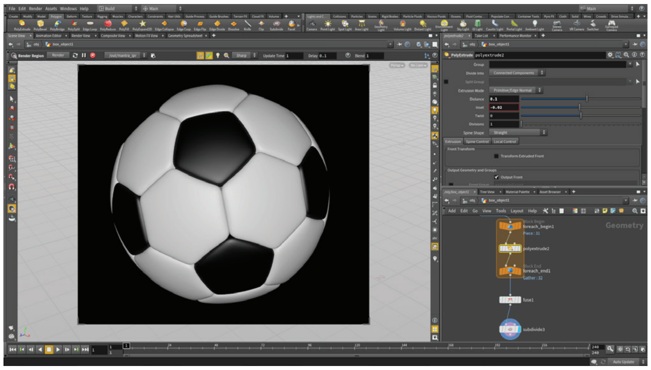
Turn off **Single Pass** to see the full shape. The *for-each* nodes create all of the patches then return the final geometry. The for-each loop is a powerful set of nodes that you will use often with Houdini.



03

Add a **Fuse** node after the *foreach_end* node and set its **Display flag**. This connects the pieces into a single topology. The *for-each* nodes broke them into the different patches but didn't fuse them back together.

Add a **Subdivide** node after the Fuse. Set the **depth** to **2**. This will give you more detail in the viewport that you can use to evaluate your model. This adds more polygons but will not yet render as a true subdivision surface. Houdini also lets you set Subdivision display in the viewport without adding geometry but this Subdivide node is needed for later in the lesson.



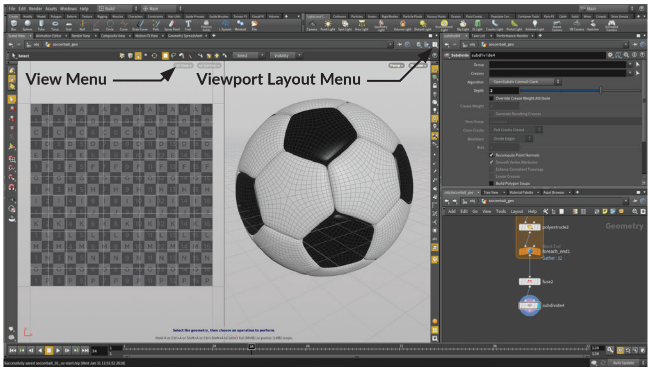
04

Play with *polyextrude* values to get a nice leathery soccer ball. Here we set a **Distance** of **0.1** and an **Inset** of **-0.02**. This gives us nice rounded patches that look much better than the original attempt.

Go to the **Object level** and rename the object *soccerball_geo* in the Parameter pane. Click on the **Render** tab and turn on **Render Polygons as Subdivisions (Mantra)** to set up true subdivisions at render time. Select the **Render Region** tool, then draw a box around the soccer ball in the viewport create a preview rendering. To cancel, click on the **x button** in the top right of the region.

PART FIVE:
Setting up UVs

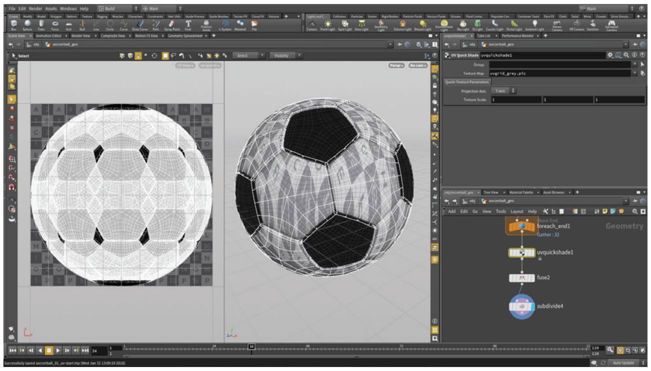
In order to set up materials and textures, it is important to make sure that there are proper UVs set up on your object. Geometry in Houdini does not come with UVs, therefore you must create them yourself. This means adding some extra nodes to the network, which in this case means adding a UV Flatten node.



01

In the top right of the Scene View, use the **Viewport Layout menu** to choose the **Two Views Side by Side** option. You could also us the hotkeys **Ctrl - 4** to get these views.

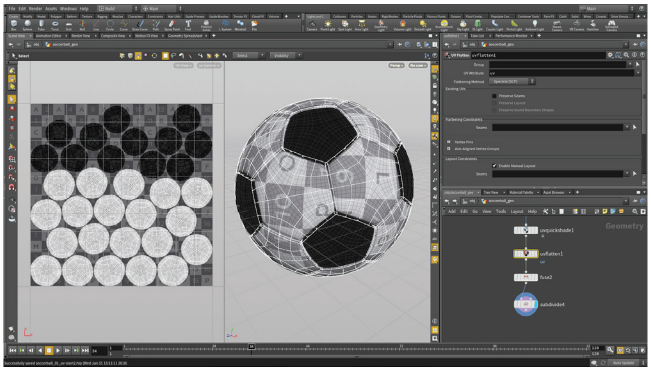
In the left panel, click the **View menu** Top and choose **UV View** to change this panel to a UV view. You can also hover your mouse over the panel and press **spacebar-5**.



02

To set up UVs for the soccerball, you can choose from a number of different techniques. The fastest is to add a **UV Quickshade** node into the soccerball chain. Place it right after the *foreach_end* node.

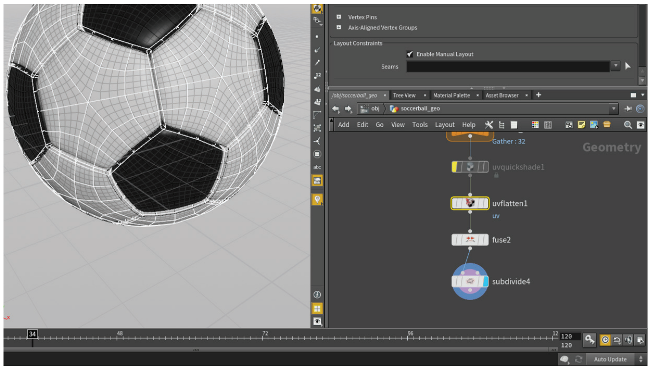
Now you will see that this node created UVs by projecting down along the Y axis which is not creating useful results. This is fine because you are only using this node to assign a UV grid for previewing purposes and will replace the UV layout with a better solution in the next step.



03

Add a **UV Flatten** node after the *uvquickshade* node but before the *Fuse* node. This will flatten the soccerball geometry using the patch boundaries to lay out the UVs. In the perspective view you can see how this affects the look of the uvgrid on the ball.

If we had placed this node after the *fuse* or after the *subdivide* then there wouldn't have been any boundaries to work with. Houdini's ability to let you set up UVs in the middle of the network can be extremely useful when working with different kinds of geometry.



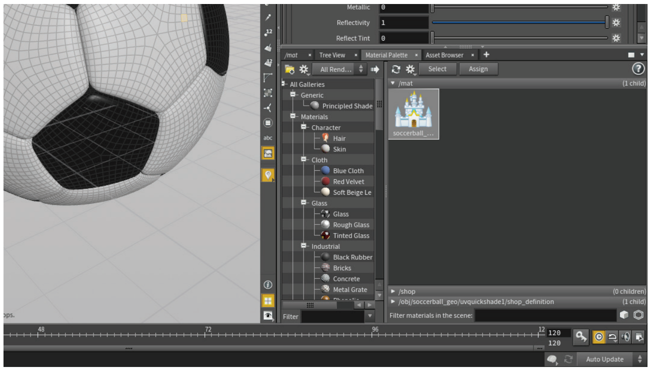
04

Set the *quickshade* node to **Bypass** to hide the assignment of the UV grid. This node was only needed when you were evaluating UVs. It could be deleted but using bypass means that we can turn it back on later if we want to tweak the UVs. The UV Flatten node is now creating the UVs for the model.

The **Bypass flag** is a useful way of blocking the influence of a node to either test out a design idea or to get rid of nodes that aren't needed. You could also delete the node but then you won't be able to go back and retrace your steps if it is needed again.

PART SIX: Materials & Shaders

Shading properties are created in Houdini using Materials which you can find in the /mat context. Materials include the ability to refine, texture and even animate shader properties. For the soccer ball, you are going to use the Principled Material which has all the properties you will need to define the look of this object.

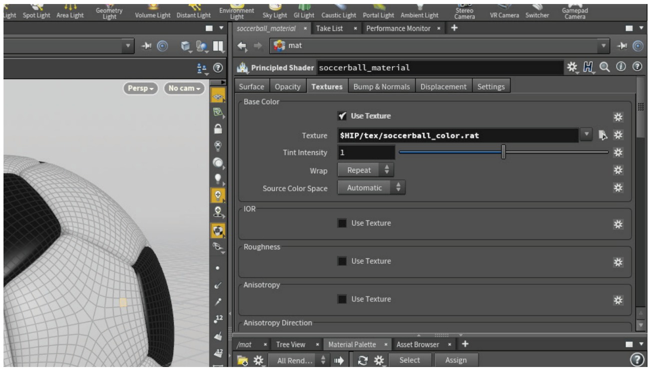


Metallic 0
Reflectivity 1
Reflect Tint 0

/obj/soccerball_geo/uvquickshade1/shop_definition (1 child)

01 Click on the **Material Palette** tab in the lower right pane. In the gallery listing to the left, find the **Principled Shader** and drag it into the area to the right to add it to the scene. In the parameter pane, rename it *soccerball_material*.

The **Principled Shader** has been pre-built to include the ability to assign textures directly and set up other features such as bumps, normals, displacements and more. The principled shader is a material based on the Disney "principled" BRDF by Brent Burley.



Principled Shader soccerball_material

Base Color
☑ Use Texture
Texture $HIP/tex/soccerball_color.rat
Tint Intensity 1
Wrap Repeat
Source Color Space Automatic

IOR
□ Use Texture

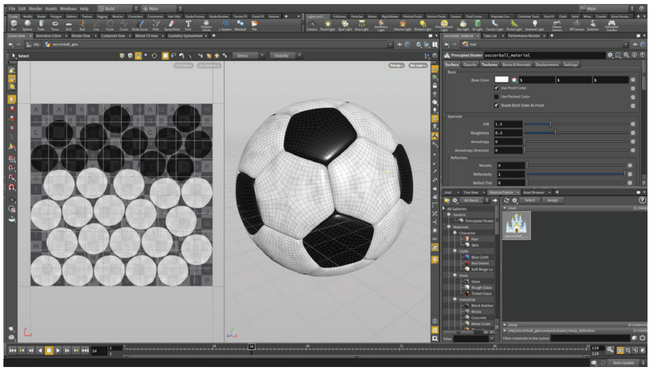
Roughness
□ Use Texture

Anisotropy
□ Use Texture

Anisotropy Direction

02 Click on the **Textures** tab and under **Base Color** click on **Use Texture** then use the button next to **Texture** to call up the file window. Click on **$HIP** in the side list then click on the *tex* folder to open it and then click once on *soccerball_color.rat* to select it. Click **Accept** to assign the texture to the material.

The $HIP reference makes sure that the reference is relative to the location of your scene file. That way if you were to move your project directories to another computer the reference will still work.

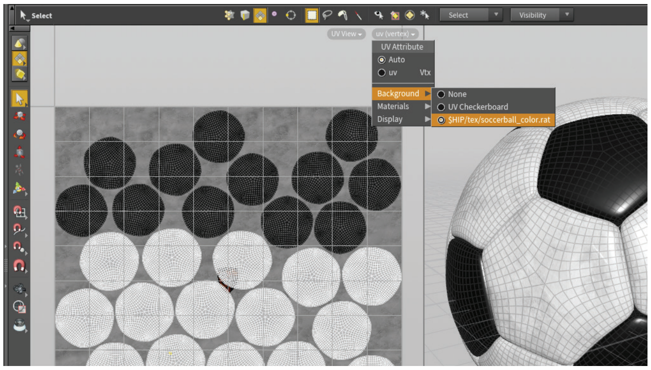


03 Drag this new material from the palette to the *soccerball* object in the Viewport. This will assign it. This adds the texture to the soccer ball but it feels a bit dark.

Go to the **Surface** tab and set **Base Color** to **1, 1, 1**. The Base Color gets multiplied with your texture map so it should be white if you want to let the texture define the color on its own.

MATERIALS IN HOUDINI

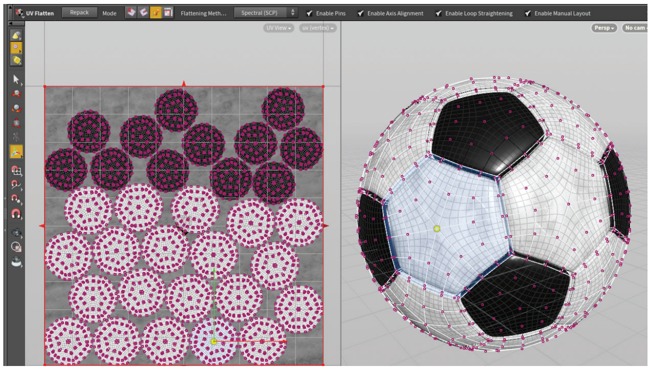
Materials in Houdini live in the **/mat** context. Once you drag them into your scene that is where they live. A material is made up of VOP nodes that define all of its material qualities. The **Principled Shader** is an uber material that can be used on its own to assign texture maps and achieve a large variety of looks. You can also build your own shaders and materials but that is not necessary in this lesson.



04

In the UV view, click on the UV menu in the top right and choose **Background > *soccerball_color.rat***. Now you can see this texture in the background of the UV panel which will allow you to make some adjustments to how this view works.

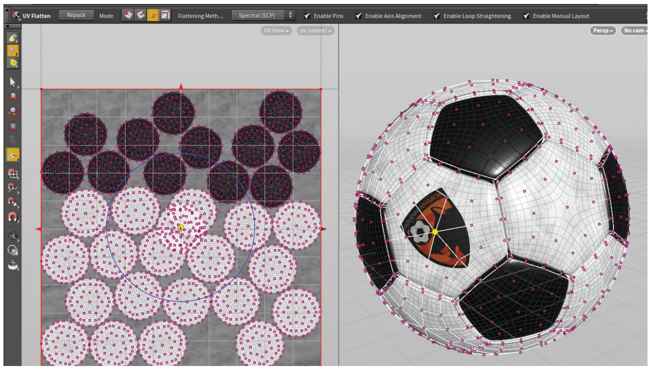
This texture has a team insignia at the center of the image. You want to get one of the visible patches lined up with that insignia so that it is visible. To do this you will go back to the UV Flatten node and adjust the patches.



05

Bring the **Network view** forward by clicking on its tab. You can see that it is currently focused on the material network. Click the **Jump Back** arrow to go back to the object level. Double click on the *soccerball_geo* object to dive into it. Select the *uvflatten* node and click on the **Handle** tool.

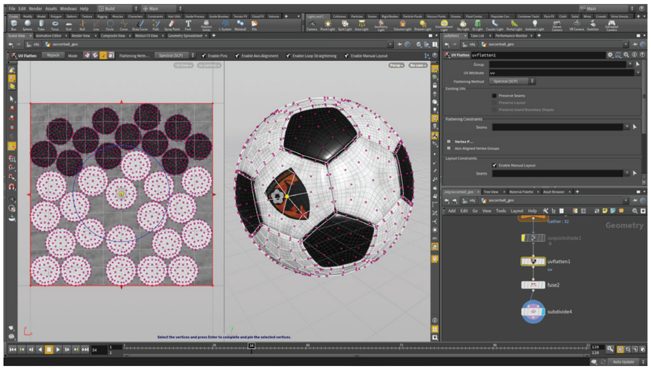
Mouse over the geometry you can see the patches highlight. Click on the **Pin Vertices** button in the **Operation Controls** bar at the top of the viewport. Mouse over the patch shown in this image and in the 3D view, click on the center vertex of the patch shown to select it. Press **Enter** to create the Pin.



06

Now go to the UV view and **move** the pin and the patch so that it is centered on the logo. **Press R** to get a rotate handle and rotate the patch until you line up the logo the way you want.

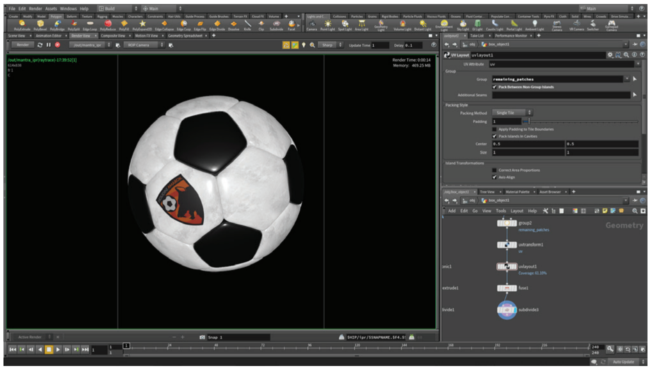
By pinning vertices on patches in the UV view, you can lock down their position. At first there is some overlap of the neighboring patches but you easily can fix this by repacking.


UV Flatten
Repack
uvflatten1
Geometry
uvflatten1
fuse2
subdivide4
Select the vertices and press Enter to complete and pin the selected vertices.

07

Click on the **Repack** button in the Operation Controls bar to reorganize the other patches around the new one. You can continue moving the patch around using the pin but may need another **Repack** if you create overlapping UVs.

Note: *Setting up UVs to match an existing texture is not the normal order of operations. In practice, the UVs are generated then passed on to a texture artist who builds the texture. We are taking this approach in the tutorial so that you can use an existing texture and don't need to paint it yourself.*



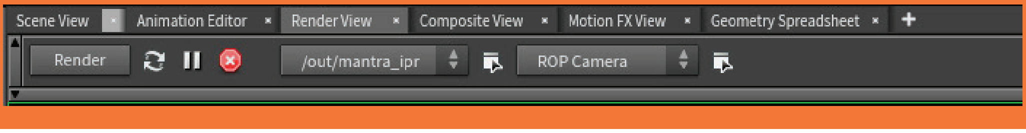
08

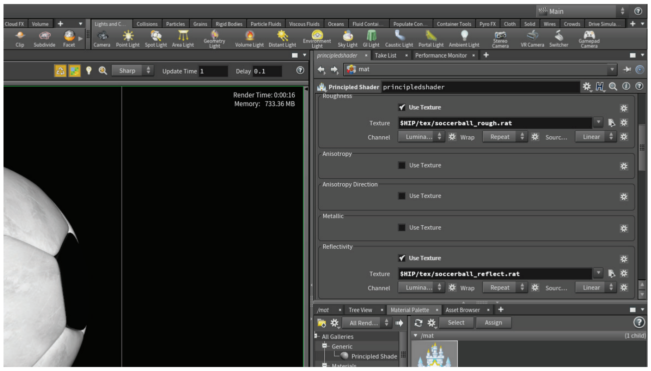
In Viewport , mouse over the perspective view and press **Spacebar-B** to expand it. Use your view tools to center the soccerball in the view.

Click on the **Render View** panel and click on the **Render** button to see what the soccerball is looking like. You will get a Mantra rendered version of the soccerball to review. You can keep this panel up as you make adjustments to the properties of the material.

RENDER VIEW

The Render View offers interactive photorealistic rendering [IPR]. This lets you tweak and explore the parameters in your scene and this view will update the rendering to show you the results as fast as possible then refining them to get the final look. You can stop and start the rendering process at any time or make changes to your lights, cameras or materials.




Render Time: 0:00:16
Memory: 733.36 MB

Principled Shader principledshader

Roughness: ☑ Use Texture; Texture $HIP/tex/soccerball_rough.rat; Channel Lumina...; Wrap Repeat; Linear

Anisotropy: □ Use Texture

Anisotropy Direction: □ Use Texture

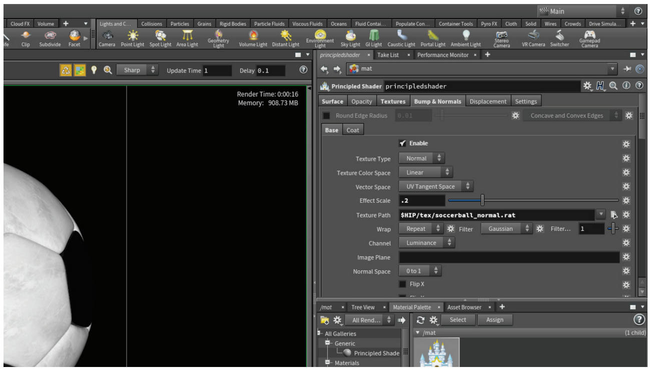
Metallic: □ Use Texture

Reflectivity: ☑ Use Texture; Texture $HIP/tex/soccerball_reflect.rat; Channel Lumina...; Wrap Repeat; Linear

09 Now that your UV's are working with the texture map, you can add more detail to the material. Go to the **Material Palette** and click on the *soccerball_material*.

In the Parameter pane, go to the **Textures** tab and use the technique you learned earlier to assign textures to **Roughness** and **Reflectivity**. You will find the appropriate textures in the *tex* folder.

Each of these changes will update the image in the Render View. You can use this to evaluate the textures on the object.

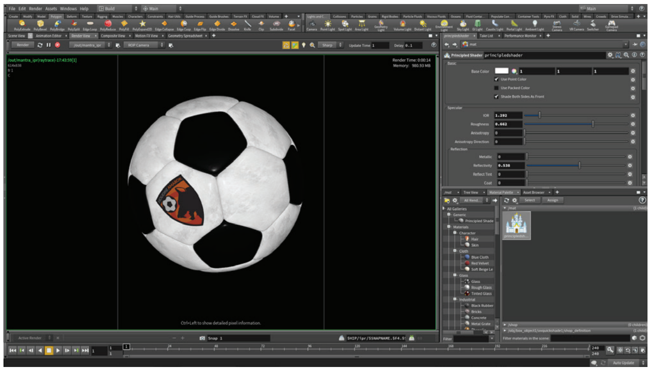

Render Time: 0:00:16
Memory: 908.73 MB

Principled Shader principledshader

Surface | Opacity | Textures | Bump & Normals | Displacement | Settings

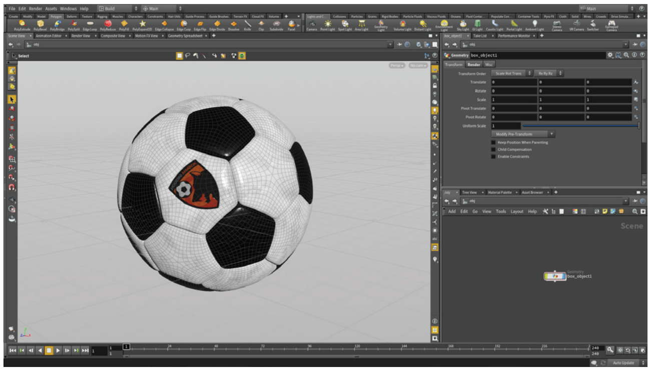
☑ Enable
Texture Type: Normal
Texture Color Space: Linear
Vector Space: UV Tangent Space
Effect Scale: .2
Texture Path: $HIP/tex/soccerball_normal.rat
Wrap: Repeat; Filter: Gaussian
Channel: Luminance
Normal Space: 0 to 1
□ Flip X

10 Go to the **Bumps & Normals** tab on the material and click the **Enable** button. Click on the arrow next to **Texture Path** and from the tex directory choose the *soccerball_normal.rat* file. Set the **Effect Scale** to around **0.2** and see how it looks in the Render View.



11 Go to the **Surface** tab on the material and explore some of the parameters to see how they affect the look of your soccerball. The **Roughness** and **Reflectivity** parameters are still affecting those texture maps so don't be afraid to tinker with them to get the look you want.

Once you are finished, press **Stop** in Render View so that it is not updating in the background as you work.

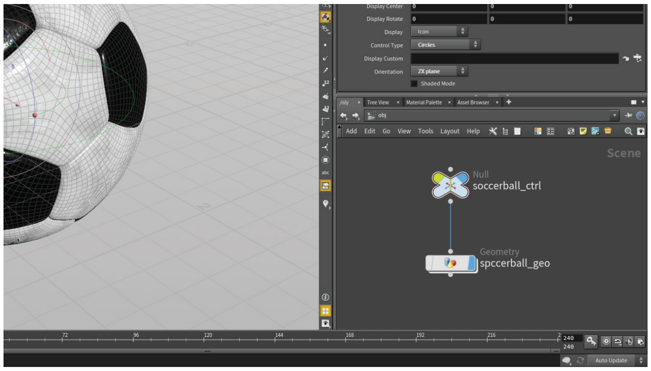


12 Go back to the Scene View, click on **OBJ** in the navigation bar to go back to the object level. Move your cursor over the perspective view and press **Spacebar-B** to expand that view. Now that you know the ball will render properly, it is time to rig it up and then animate it bouncing.

Save your work.

PART SEVEN: Rig the Soccerball

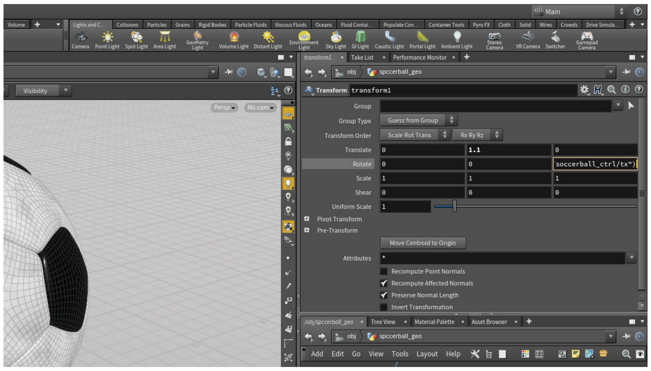
In order to create an animation of the ball bouncing, you will start by building a simple rig that will make the keyframing process easier. This will involve setting up some null objects so that you can work interactively in the viewport and adding nodes to the soccerball geometry network to accommodate the ball rotation and add some squash and stretch.


Display Center 0 0 0
Display Rotate 0 0 0
Display: Icon
Control Type: Circles
Display Custom
Orientation: ZX plane
□ Shaded Mode

01

On the **Create** shelf, click on the **Null** tool then press **Enter** to place it at the origin. Name it *soccerball_ctrl*. Go to the Misc tab and set **Control Type** to **Circles** and **Orientation** to **ZX Plane**. Set the **Display Uniform Scale** to **4**. This creates a handle for the rig that is easy to select that won't render later on.

In the network editor, connect the input of the *soccerball_geo* object to the output of the *soccerball_ctrl* null to create a child/parent relationship. Moving the null will move the ball. Turn off the **selection flag** on the *soccerball_geo* so that you don't select it by accident in the viewport while animating. You will use *soccerball_ctrl* instead.


Transform transform1
Group Type: Guess from Group
Transform Order: Scale Rot Trans, Rx Ry Rz
Translate 0 1.1 0
Rotate 0 0 soccerball_ctrl/tx")
Scale 1 1 1
Shear 0 0 0
Uniform Scale 1
□ Recompute Point Normals
☑ Recompute Affected Normals
☑ Preserve Normal Length
□ Invert Transformation

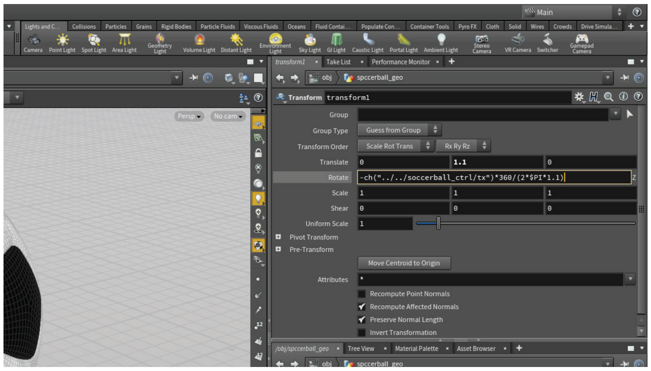
02

Select the *soccerball_ctrl* node. In the Parameter pane, click on the **Transform** tab then **RMB-click** on the **Translate X** parameter. Choose **Copy Parameter**.

Dive into the *soccerball_geo* object. Add a **Transform** node and **RMB-click** on the **Rotate Z** and choose **Paste Relative References**. This places a channel reference expression in this parameter.

```
ch("../../soccerball_ctrl/tx")
```

Set **Translate Y** to **1.1** so that the soccer ball geometry sits on the ground surface.


Transform transform1
Group Type: Guess from Group
Transform Order: Scale Rot Trans, Rx Ry Rz
Translate 0 1.1 0
Rotate -ch("../../soccerball_ctrl/tx")*360/(2*$PI*1.1)
Scale 1 1 1
Shear 0 0 0
Uniform Scale 1
Attributes *
□ Recompute Point Normals
☑ Recompute Affected Normals
☑ Preserve Normal Length
□ Invert Transformation

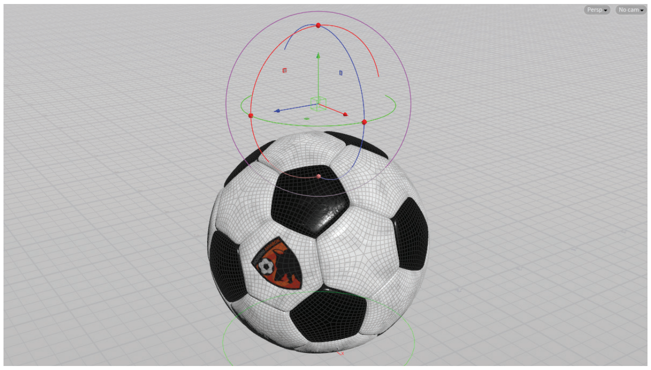
03

Click on the parameter to expand the channel. You are now going to use the **ball's circumference (2πr)** to determine the ball's rotation as it moves forward.

Edit the expression to read:

```
-ch("../../soccerball_ctrl/tx")*360/(2*$PI*1.1)
```

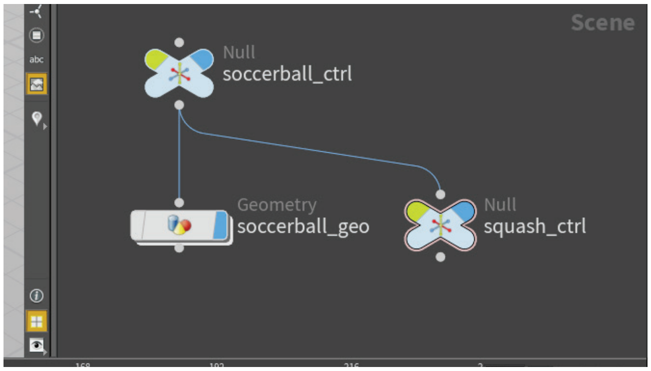
First you add a negative **(-)** at the front. You then multiply the position of the ball by **360** degrees and divide by **2πr**. At the object level, move the *soccerball_ctrl* along the **X axis**. The expression will rotate the ball to match the motion. Put it back at the origin when you are finished.



04

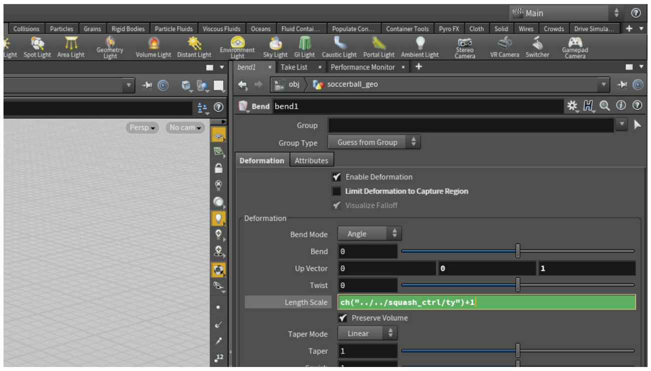
In the Viewport, create another Null object at the origin. Call it *squash_ctrl*. Go to the Misc tab and set **Control Type** to **Box** and the **Display Uniform Scale** to **0.2**.

Move the null up just above the ball. **Translate Y** should be about **2.5**. In the Parameter pane, choose the **Modify Pre-Transforms** menu and select **Clean Translates**. This sets the Translate Y value of the null to **0** even though it is above the ground. In order for this null to drive the squash and stretch, it needs a default value of 0.



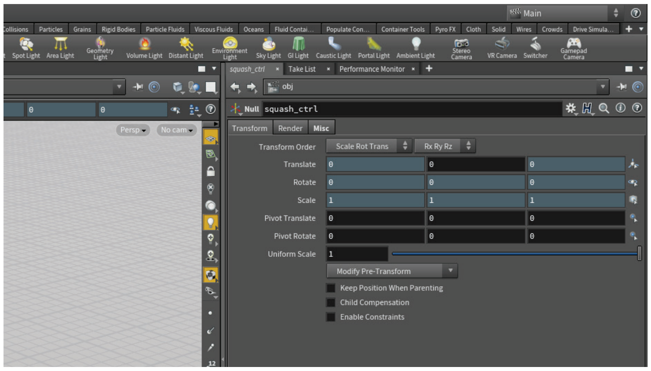
05 Parent the *squash_ctrl* null to the *soccerball_ctrl* null. This will ensure that this secondary null moves when you animate the control null.

RMB-click on the *squash_ctrl* node's **Translate Y** parameter. Choose **Copy Parameter**. You will use this parameter to drive the squash and stretch of the ball. This will allow you to control the squash and stretch from the viewport interactively.


Bend bend1
Group Type: Guess from Group
Deformation | Attributes
☑ Enable Deformation
□ Limit Deformation to Capture Region
☑ Visualize Falloff
Deformation
Bend Mode: Angle
Bend: 0
Up Vector: 0 0 1
Twist: 0
Length Scale: ch("../../squash_ctrl/ty")+1
☑ Preserve Volume
Taper Mode: Linear
Taper: 1

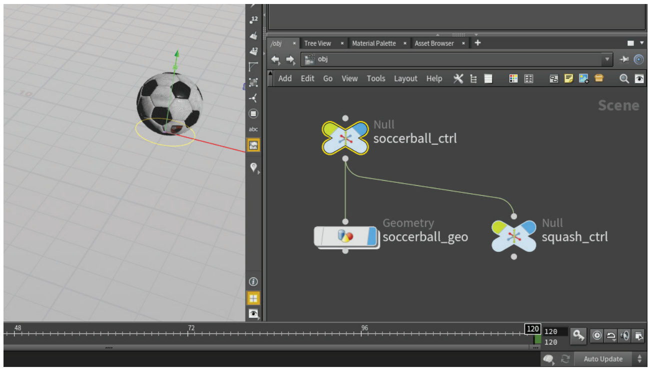
06 Dive into the *soccerball_geo* object. Add a **Bend** node to the End of the chain and set its display flag. Turn off the **Limit Deformation to Capture Region** then set **Up Vector** to **0, 0, 1**, **Capture Direction** to **0, 1, 0** and **Capture Length** to **2.2**.

RMB-click on the **Length Scale** parameter and choose **Paste Relative References** to create the channel reference. The ball is flattened therefore click to edit the expression and add a **+1** at the end. Now go to the object level, select the *squash_ctrl* null and **Move** it up and down along the Y axis to see the ball squash and stretch. When you are finished set the **Translate Y** value to **0**.


Null squash_ctrl
Transform | Render | Misc
Transform Order: Scale Rot Trans, Rx Ry Rz
Translate: 0 0 0
Rotate: 0 0 0
Scale: 1 1 1
Pivot Translate: 0 0 0
Pivot Rotate: 0 0 0
Uniform Scale: 1
Modify Pre-Transform
□ Keep Position When Parenting
□ Child Compensation
□ Enable Constraints

07 Select the *soccerball_ctrl* object. **RMB-click** on the **Scale** and **Rotate** parameters and choose **Lock Parameter** to lock all three channels then **RMB-click** on **Translate Z** and choose **Lock Parameter.** Now when you select the object you will only see handles for Translate X and Y. This is a good way to make it easier to work with a rig because then the animator can only manipulate parameters you have made available to them.

Lock all the transform parameters on *squash_ctrl* except for **Translate Y.** Now when you select either of the null objects, you only see handles for the parameters you want to animate.



08 Now test out the rig by moving it around in X and Y and using the second handle to squash and stretch it. Once you are sure that all the parts are working, return all the values to **0** and get ready to animate.

Make sure that the select flag on the *soccerball_geo* object is off so that you don't select that node by accident. You won't be animating any parameters on that node. Only on the two control nulls.

Save your scene file before proceeding.

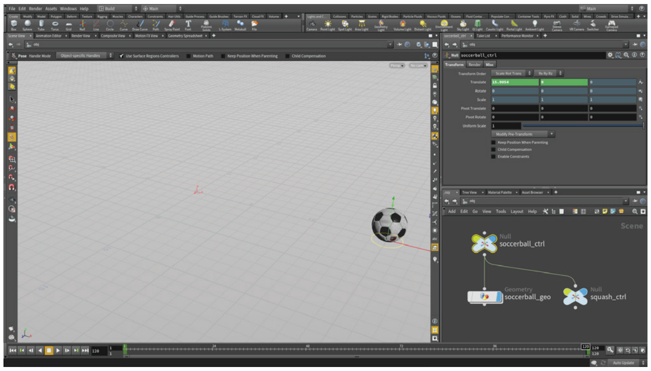
WHY RIG THE BOUNCING BALL?

By rigging the ball, you reduce clicks for the animator which will speed up their creative process. Taking extra time to create a rig will make it easier for them to focus on the viewport and work exclusively at the object level.

It would certainly be possible to animate the soccerball geometry directly and to set keyframes on the bend and transform nodes at the geometry level but this would involve jumping in and out of network types during the process of keyframing.

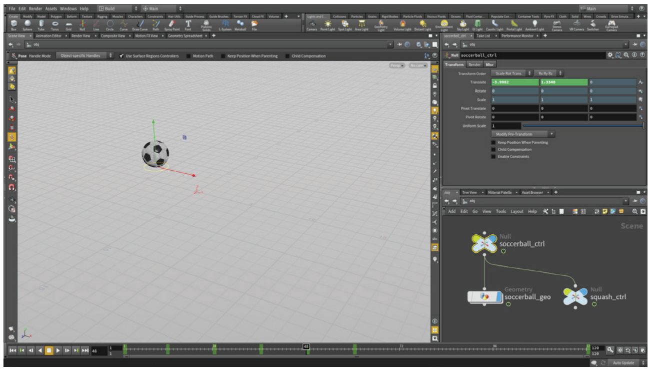
PART EIGHT:
Animate a Bouncing Ball

You can now take the soccer ball rig and use it to animate the ball bouncing. You will learn how to set keyframes, adjust animation curves and work with time-space handles in the viewport. The bouncing ball is a classic animation exercise that offers a great opportunity to learning the basics of animating in Houdini.



01 At the bottom right edge of the **Timeline**, click on the **Global Animation Options** button. Set the *End* to **120** and click **Close**. This will set the timeline range to 120 frames.

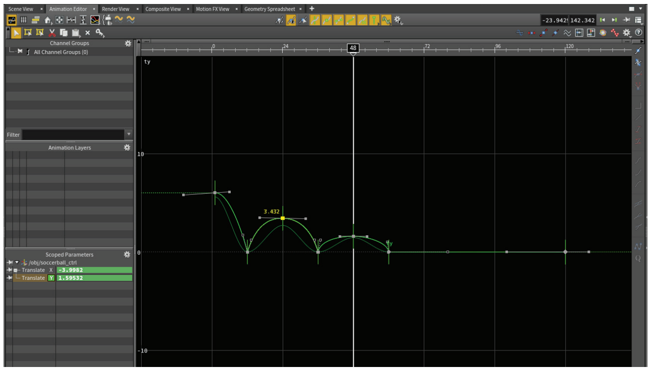
Make sure you are on **frame 1**. Click on the **Pose** tool on the toolbar to your left and select the *soccerball_ctrl*. Move the ball to around **-15** in X and **press K** to keyframe it. Move the timeline to **frame 120.** Move the ball to the around **15** in X and **press K** to set a second keyframe. Scrub through the timeline to make sure that the ball is animating. It should be moving and rotating based on the rig design.



02 Move the timeline to frame **12** and **press K** to set an intermediate key. Repeat at frames **36** and **60**. All of these keyframes are sitting on the ground.

Now go to **frame 1** and lift the ball up in the Y direction. You don't need to set another key because this move simply updates the keyframe you already set at frame 1 .

Move to **frame 24** and lift up the ball in the Y direction a little less than you did at frame 1. **Press K** to set a keyframe. Move to **frame 48** and lift the ball up even less. **Press K** to set another keyframe.

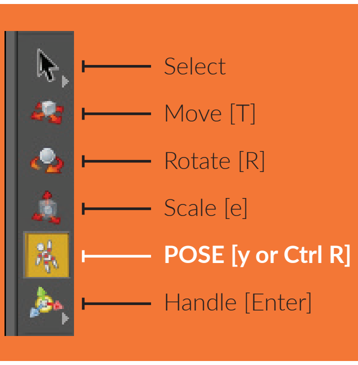


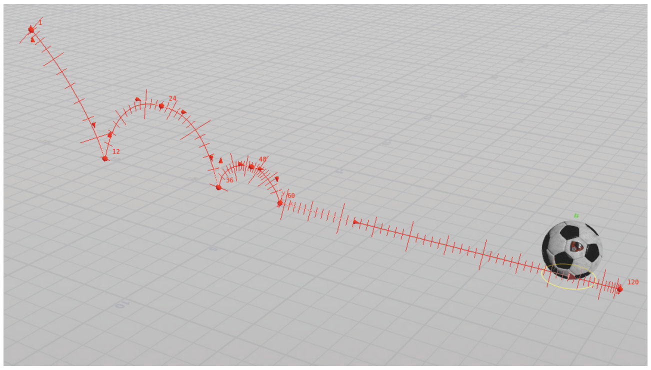
03 **Scrub** through the timeline to see that the ball appears to float whereas you want hard hits when the ball contacts the ground. Click on the **Animation Editor** pane tab.

From the Scoped parameter list, click on the *Translate Y* channel. **Press H** to home the view of the curve. Select the three keyframes where the ball contacts the ground and press the **Untie Handles** button on the right side **Functions** bar. Now click in empty space to deselect then start tweaking the tangent handles to create a sharp bounce at each point. You can also stretch out the handles at the top to slow the ball down at the peak.

THE POSE TOOL

You can easily manipulate the two control objects in the soccerball rig using the **Move** or **Handle** tool. The advantage of using the **Pose** tool is that it gives you access to the **Motion Path** handle and if you were working with Inverse kinematics, there are special handles that you can use to control the system. Therefore be sure to remember this tool when you start setting keyframes on your rigs.

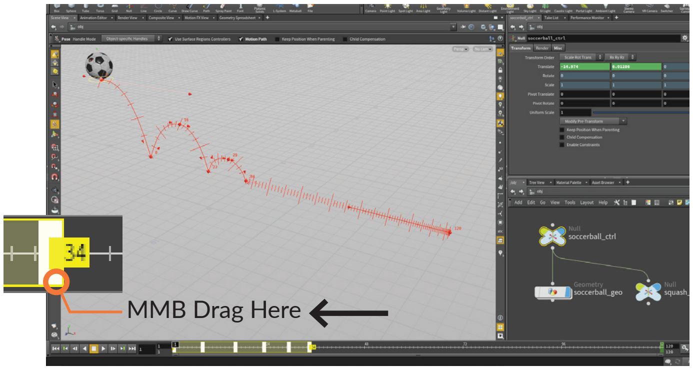




04

Go back to the **Scene view** and preview the results. You should now have a sharper hit each time the ball hits the ground. Make sure you have the **Pose** tool active and the *soccerball_ctrl* node selected. In the top bar, turn on the **Motion Path** handle. This shows a path outlined where the ball is bouncing.

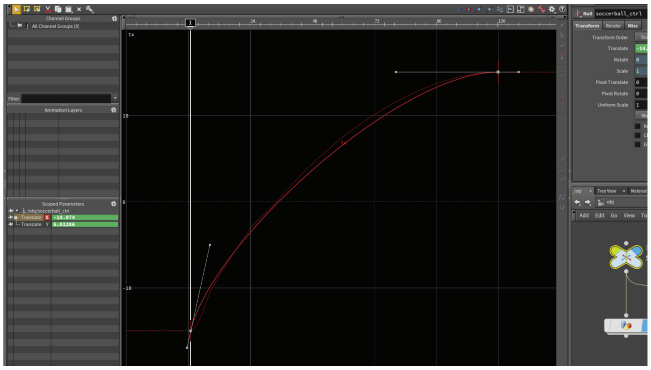
You can now tweak the position and height of the bounce using this handle. Click on each keyframe marker and manipulate the handle. You can also **RMB-click** on the handle and display the tangents to further manipulate the shape of the curve.



05

To adjust the timing, you can also use the timeline. Press **Shift** and drag a bounding box from frame 1 to the last key in the timeline **to select** all the keys. Next, **MMB drag** on the end of the box underneath the handle to scale the timing of the bounces to speed them up. You can also select each key using **MMB** and then drag with **LMB** to time each keyframe the way you want.

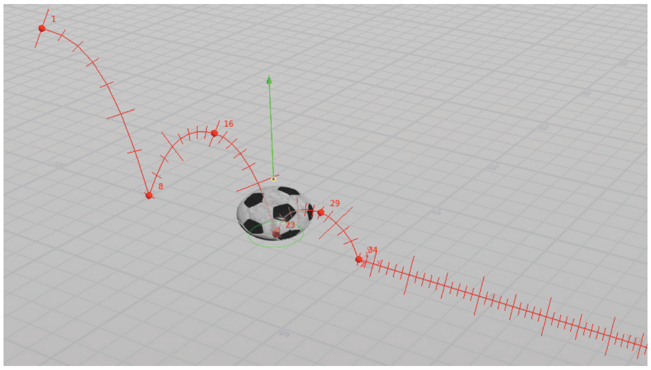
This is where you will determine the timing of the bounces. Keep exploring until you get the look that you want. Note that there may be some awkwardness in the bouncing because of the translate X values. You will fix that in the next step.



06

Click on the **Animation Editor** pane tab and you will see two curves. From the Scoped Parameters list, click on **Translate X** for *soccerball_ctrl*. Now select all the keys except for the first and the last. Press **Delete**. Now use the curve handles to go from a high slope to a low slope. This will have the ball moving faster at the beginning and slower at the end.

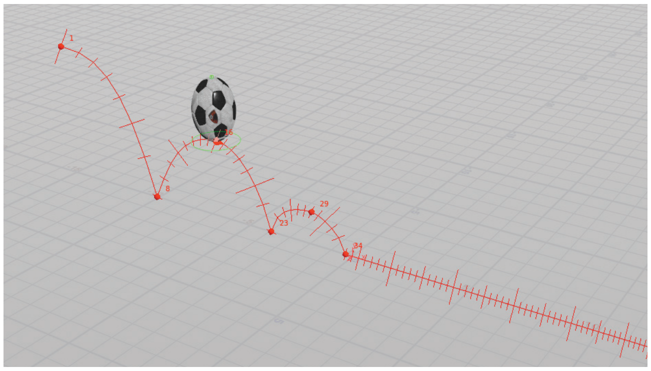
Note that if you go back to tweak the points in X on the motion path handle, you will get strange results because there are no longer intermediate keys in that direction - only use it to tweak in Y from this point on.



07

Go back to the Scene view. **RMB-click** on the **Motion Path** handle and choose **Persistent**. This will keep it around as a guide as you set keyframes on the squash and stretch.

Select the ***squash_ctrl*** and turn off its **Motion Path**. Go to the first bounce and move back one frame. Select the ***squash_ctrl*** handle and stretch out the ball a bit. **Set a Keyframe** using the **K** key. Now go to the bounce frame and move the handle down to create squash. **Set another key**. Go two frames forward and stretch the ball out again. **Set another key**. Repeat for all the bounces.



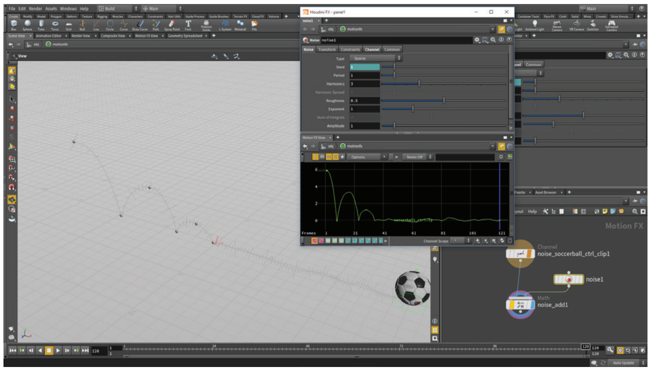
08

Now go to the peaks of the bounces and add in a bit more stretching. **Set keyframes**. When you are finished, scrub and playback the motion to preview the results. Make sure the ⊕ **Real Time Toggle** is **On** in the Timeline to properly evaluate the motion.

You can now use the **Animation editor** to make tweaks to the squash and stretch. Make sure you keep the keyframes aligned with the bouncing of the soccer balls.

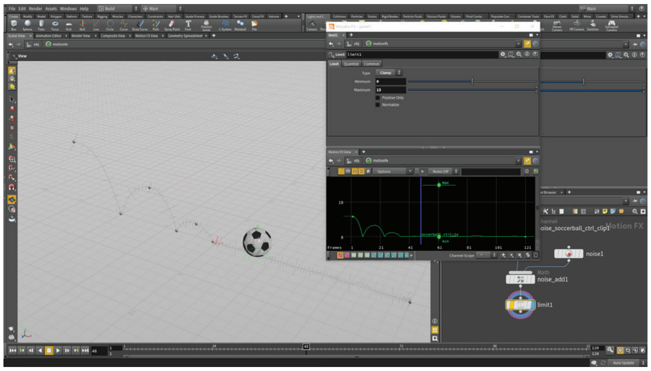
PART NINE:
Add Motion FX

In addition to keyframing everything by hand, you can also use the Motion FX menu to create procedural animation nodes that make it easier to add detail to the motion. These nodes are Channel Operators or CHOPs and they offer a whole new way for animators to create procedural motion. You can add effects that act on all of the motion at the same time or focus on one channel.



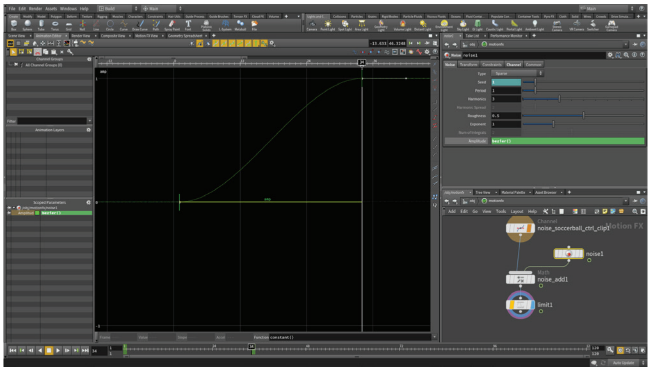
01 Select the *soccerball_ctrl* null object. **RMB-click** on the **Translate Y** and choose **Motion FX > Noise**. A panel pops up with parameters that you can use to control the noise. A new subnetwork was created with the CHOP nodes that send their information back to the *soccerball_ctrl's* **Translate Y** channel.

Set the **Amplitude** to **5** and press **Play** to see how this looks. This adds dramatic up and down motion which make it feel like there is some serious turbulence. Set the **Amplitude** to **1.** This offers a more subtle bump to the motion of the ball.



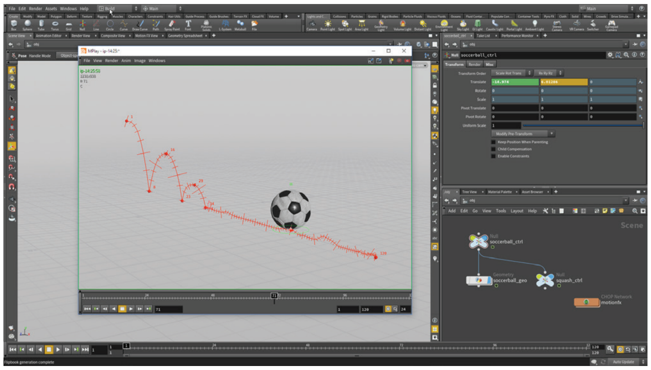
02 Some of this motion goes below the ground. You will want to focus on creating bumps above the ground.

Go back to the object level. Select the *soccerball_ctrl* null object. **RMB-click** on the **Translate Y** and choose **Motion FX > Limit**. Set **Minimum** to 0 and **Maximum** to 15. Now the ball moves flat with some bumps instead of going up and down the whole way.



03 There should not be any noise while the ball is bouncing. It is only needed when the ball is rolling. You can keyframe the **Amplitude** to turn the noise on and off.

Select the *noise1* CHOP node. Go to the frame where the ball stops bouncing and starts rolling. **Alt-click** on **Amplitude** to set a keyframe. Go to **Frame 1** and set **Amplitude** to **0. Alt-click** on **Amplitude** again to set a second keyframe. Go to the **Animation Editor** and select the curve. Click on the **Constant** button in the Functions bar. This creates a sharp cut from no amplitude to an amplitude of 1.



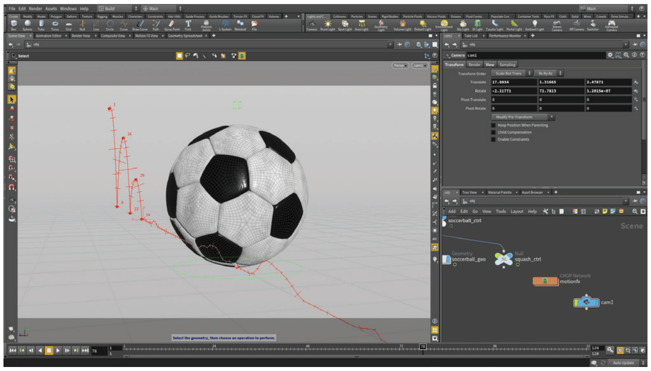
04 In the toolbar at the left side of the Scene view, click on the **Render Flipbook** button. Leave the default settings and click **Start**. Wait while the sequence is captured and then you will see the flipbook in an **Mplay** window. You can **Play** this back and scrub through to evaluate your motion.

You can now go back and adjust the curves on the motion or the channel operators in CHOPs to get the result you are looking for. Then you can re-flipbook to evaluate.

Save your work.

PART TEN:
Lights, Camera, Action!

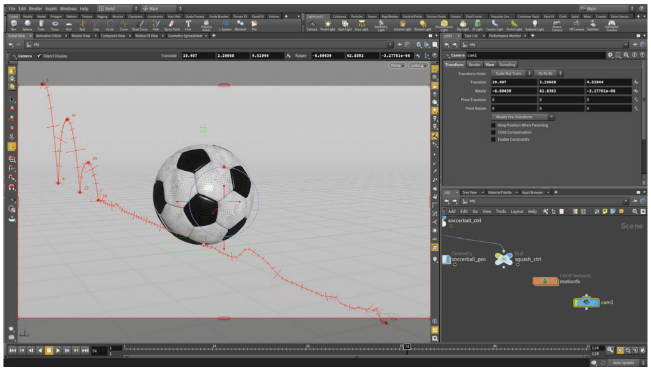
To render out the animated soccer ball, you will need to add lights and a camera. These are essential elements when creating the look of your shot and can be set up in the viewport. These are elements that you will likely want to tweak and adjust to get the perfect rendering. There are different ways to make these adjustments in Houdini.



01

In the Scene view, use your view tools to look at the ball so that it bounces towards you and off the screen. Set up your perspective view to see the whole bounce

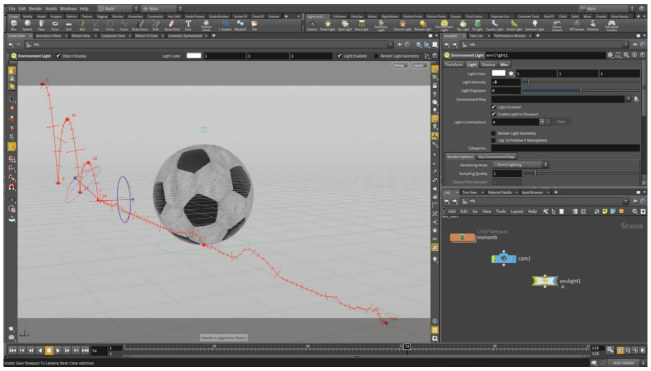
From the top right of the main viewport, click on the **Camera** menu and choose **New Camera**. This creates a camera looking from this point of view. You can see that the camera is framing the view based on the grey bars at the top and the bottom indicating areas that are "off camera."



02

If you want to tweak the view there are two options. You can click on the 🔒 **Lock to View** button. Now any view changes you make in the viewport will move your camera. Once you have the view you like be sure to turn this off otherwise you might lose the camera position you chose.

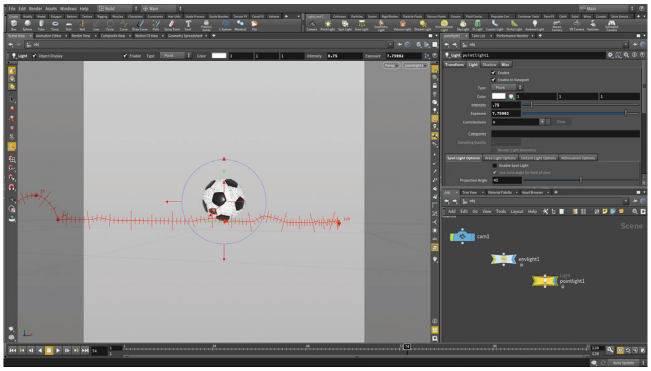
Another approach is to click on the **Camera** menu and choose **Select Camera**. Make sure the **Handle tool** is active and you will get handles in the viewport that you can use to tweak the view.



03

With your mouse in the Scene View, **Press c** to bring up the radial menu and select **LookDev > Environment Light**. Press **Enter** to add an environment light at the origin.

Set **Light intensity** to **0.5**. You could also assign an environment map to get a more subtle look but for now we will use the light color of white.



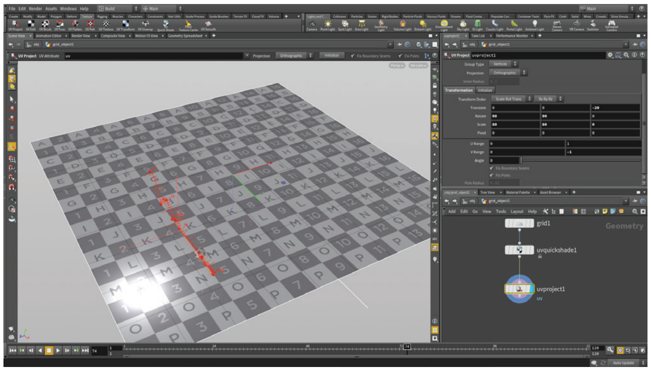
04

Make sure the 🔒 **Lock to View** button is off then tumble the view until you are looking at the ball from the side. Now press the **Ctrl** key and then in the **Lights and Cameras** shelf, click on the **Point Light** tool. This puts a light down at your current viewpoint. You can use view tools to adjust this if needed using the same methods you used for the camera.

Now select *cam1* from the **Camera** menu to see how the light affects the look of you scene.

PART ELEVEN:
Render the Shot

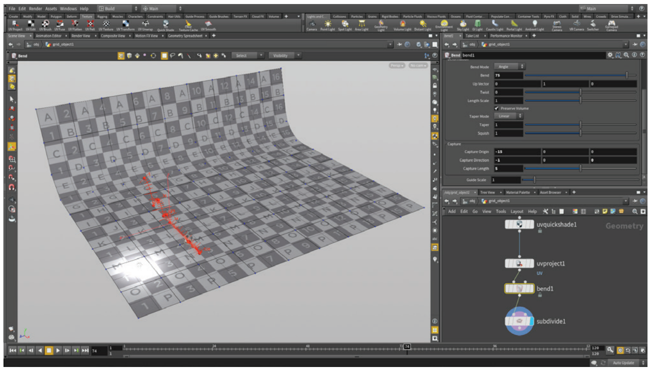
Next you will add a backdrop to act as a ground surface for your bouncing soccer ball. You will quickly model this with UVs and then assign a material with texture maps. You will then test render the complete animation then save out a sequence to disk for final review. It is always a good idea to test render a bit before committing to full render.



01

Go to the Object level. Add a **grid** object to your scene at the origin. Name it *Backdrop*. Dive to the geometry level and set the **Size** to **80x80** and the **Center Z** to **-20.** This is a simple grid that you will reshape into backdrop for your shot.

Add a **uvQuickshade** then a **uvProject**. Go to the **initialize** tab and click the **initialize** button then go back to the **Transformation** tab and set **V Range** to **0** to **-1** and **Rotate Y** to **90**.

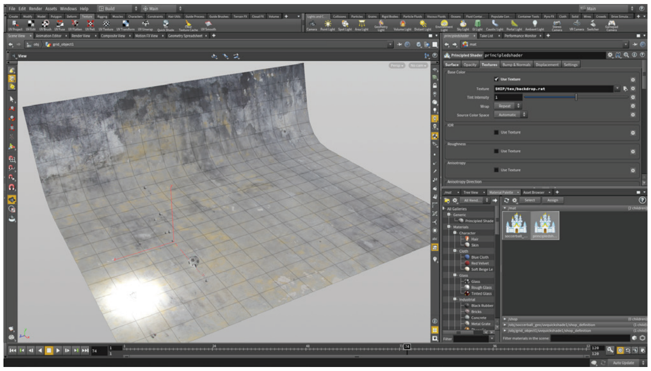


02

Add a **Bend** node. Set the **Bend** to **75**. To orient this properly, change **Capture Origin** to **-15, 0, 0** , **Capture Direction** to **-1, 0, 0** and **Capture Length** to **5.**

Add a **Subdivide** node at the end of the chain to smooth out the geometry.

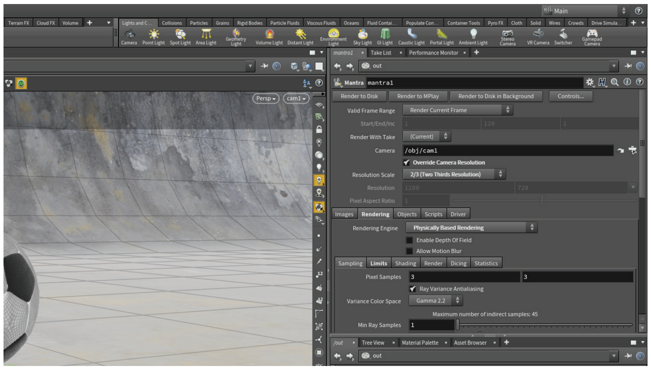
IMPORTANT: Set the **Bypass flag** on the **uvQuickshade** node so that the grid texture is removed. This will allow you to see any texture maps in the Scene view once you assign a material.



03

Click on the **Material Palette** and Drag a **Principled Shader** into the area to the right. Rename it *backdrop_material*. Drag this material to the *backdrop* object in the Viewport.

Go to the **Surface** tab and set **Base Color** to **1, 1, 1**. Click on the **Textures** tab and under **Base Color** click on **Use Texture** then use the button next to **Texture** to call up the file window. Click on **$HIP** in the side list then click on the *tex* folder to open it and then click once on *backdrop.rat* to select it. Click **Accept** to assign the texture to the material. You can also assign *backdrop_reflect.rat* to the **Reflectivity** Texture channel to add more detail.



Mantra mantra1

Render to Disk | Render to MPlay | Render to Disk in Background | Controls...

Valid Frame Range: Render Current Frame
Start/End/Inc: 1 120 1
Render With Take: (Current)
Camera: /obj/cam1
☑ Override Camera Resolution
Resolution Scale: 2/3 (Two Thirds Resolution)
Resolution: 1280 720
Pixel Aspect Ratio: 1

Images | Rendering | Objects | Scripts | Driver

Rendering Engine: Physically Based Rendering
□ Enable Depth Of Field
□ Allow Motion Blur

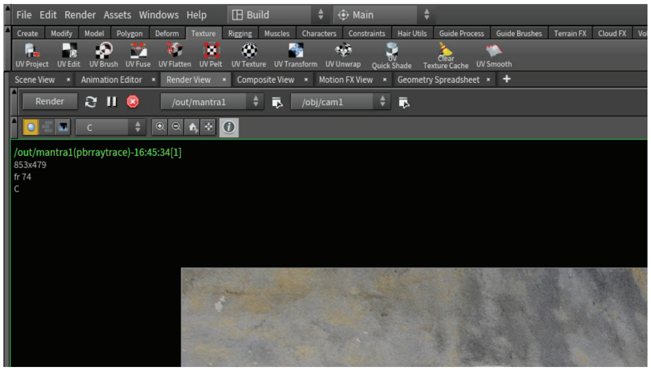
Sampling | Limits | Shading | Render | Dicing | Statistics

Pixel Samples: 3 3
☑ Ray Variance Antialiasing
Variance Color Space: Gamma 2.2
Maximum number of indirect samples: 45
Min Ray Samples: 1

04

From the **Render** menu, select **Create Render node > Mantra PBR**. This creates a Mantra node that is set up for a physically-based render. Make sure the **Camera** parameter is pointed to *cam1*. Turn on **Overide Camera Resolution** and set **Resolution Scale** to **2/3 resolution**.

You can have more than one Mantra node set up with different output options. There is already one there which was used by the Render view window earlier. You will now use the new setup.

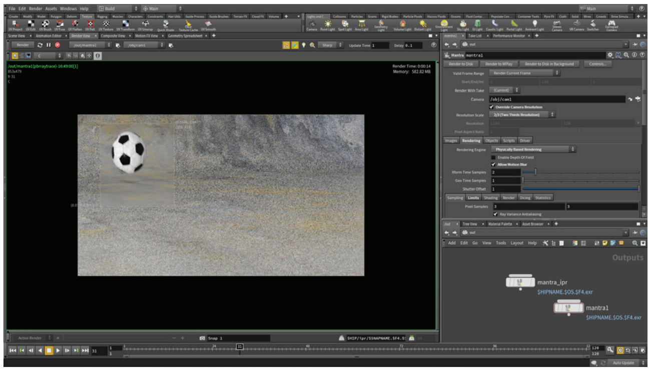


```
/out/mantra1(pbrraytrace)-16:45:34[1]
853x479
fr 74
C
```

05

Go to the **Render View** panel. Set the **ROP camera** menu to *cam1* and the **Output Driver to Render** menu to *Mantra1*. Click **Render** to watch the progress. As the shot renders you can click on any point in the render space to focus the rendering on that area.

You can tweak lights, materials to find the right look. You can also change the timeline if you want to check out a different frame. Go to a frame where there the ball is falling to the ground. You can see what we don't have any motion blur set up yet.

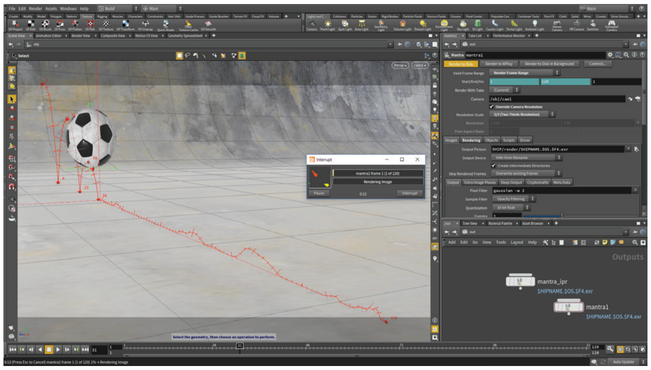


06

In Render View, press **Shift** and draw a box around the ball to focus updates on this area. This is a way to work faster as you explore a parameter change.

From the **Render** menu, select **Edit Render node > Mantra1**. Click on the **Rendering** tab and turn on **Allow Motion Blur**. Now you can see the blur as the ball moves through the shot.

This tab also has various options for increasing the quality of your rendering. Adjust these with care because cranking up these numbers will take longer to render.



07

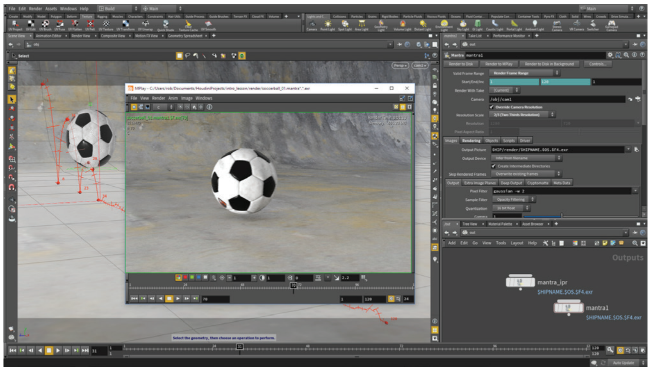
Save your work. Now you can edit the Mantra node to render out an image sequence.

At the top of the parameter pane, set **Valid Frame Range** to **Render Frame Range**. Click on the **Images** tab and set **Output Picture** to:

`$HIP/render/soccerball_animation.$F4.exr*`

This will create a sequence of numbered images of your animation. Click on **Render to Disk**. Watch the progress

* Apprentice users can use *.pic* as their output type to render without a watermark.



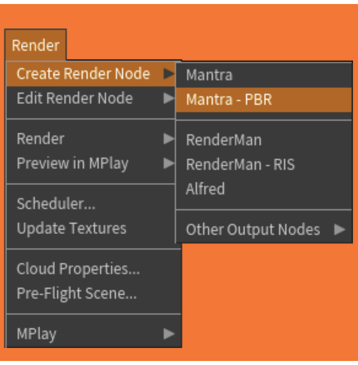
08

From the **Render** menu, select **MPlay > Load Disk Files**. Dive into the **Render** directory and click on the *soccerball_animation_$F.exr* sequence. Click **Load**.

Now you can playback the animation of your soccerball bouncing with motion blur. Note that Mplay has lots of options for tweaking **Brightness**, **Contrast** and **Bright Shift** that you can use to explore settings you might want to use later in the compositor.

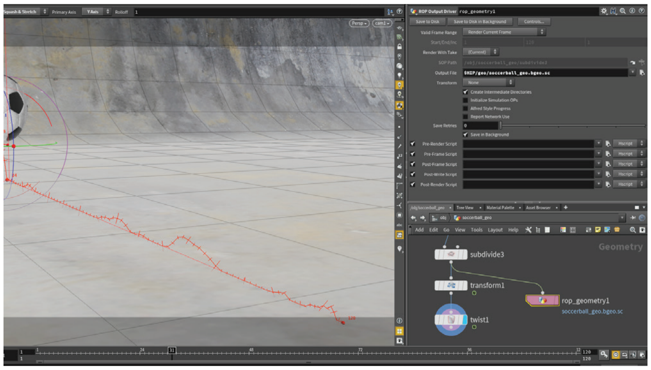
MANTRA PBR

Mantra is the highly advanced renderer included with Houdini. It implements **Scanline**, **Raytracing**, and **Physically-based** rendering in one renderer. You should use the physically based rendering engine unless you have a good reason to use another engine. Mantra has deep integration with Houdini and all of the lights and materials have been designed to work well with it.



PART TWELVE:
Set up a Rigid Body Simulation

While traditional animation is great for animating a single soccer ball, dynamics would be a better option if you want to animate a bunch of soccer balls. Dynamics requires a simulation so that the solver can go frame by frame determining how each of the participating objects interact with each other. You will use instancing to get an efficient result for this simulation.

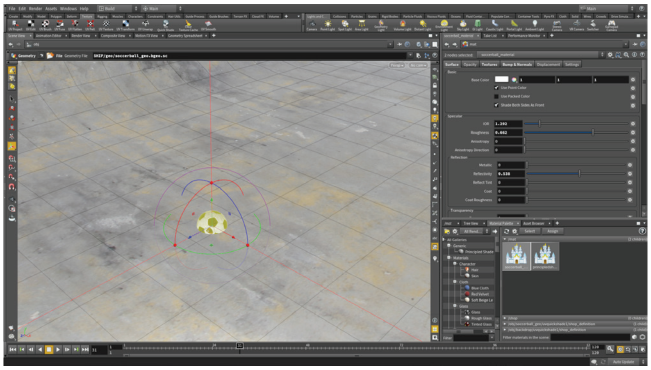


01 Go back to the **Scene View.** You will now save a copy of the soccerball without animation to disk. Navigate into the *soccerball_geo* geometry network. **Press tab** and start typing **ROP**. Choose **ROP Output Driver** from the menu and place the node next to the *transform* node. Wire the output of the *subdivide* node into the *rop_geometry* node.

In the parameter pane, set the **Output file** to:

```
$HIP/geo/soccerball_geo.bgeo.sc
```

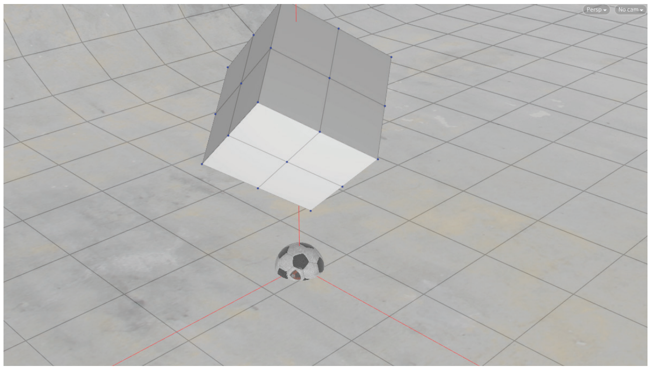
Click **Save to Disk** to save out the file.



02 Go to the **Object** level and turn off the visibility of the *soccerball_geo* node. **RMB-click** on the motion path and turn off Persistent so that it goes away.

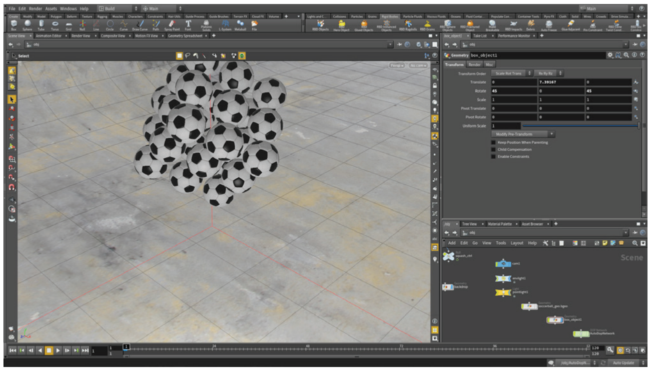
In the viewport, press **tab > File (Create)**. this brings up a file window. Navigate into the `/geo` directory and select the `soccerball_geo.bgeo.sc` file. Click **Accept**. Press **Enter** to place the imported object at the origin.

Go to the **Material palette** and drag the *soccerball_material* onto the new geometry object. Now it looks just like the original soccer ball.



03 **Press c** to bring up the radial menu and choose **Create > Geometry > Box**. Press **Enter** to place it at the origin. You are going to position soccer balls at all of the corners of this box. Let's add more detail.

Dive down to the geometry level and set the box's **Size** to **5, 5, 5** and its **Axis Divisions** to **3, 3, 3**. Add a **Transform** node and **Move** the box up around **7** units and **Rotate** it **45** degrees in both **X** and **Z**.

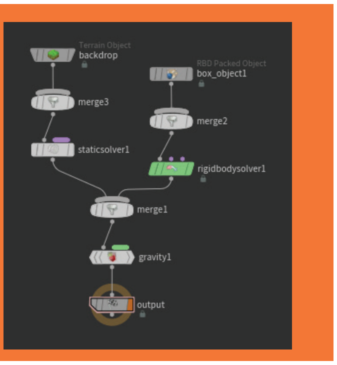


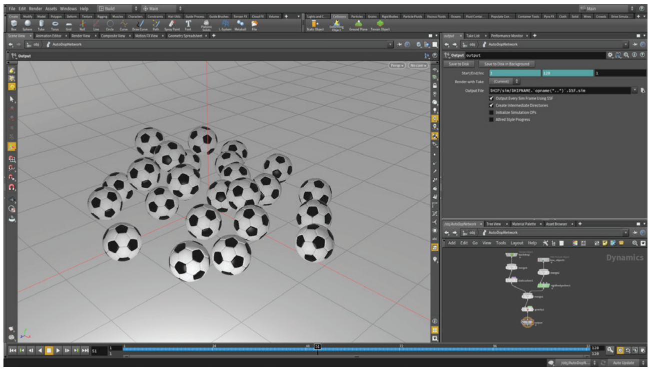
04 Go back to the **Object level** and keep the *box_object* selected. Make sure you are on **Frame 1**. Go to the **Rigid Bodies** shelf tab and click on the **RBD Instanced Objects** tool. The *box_object* will be accepted as the *"object whose points will be used for instancing."* Now click on the *soccerball* and press **Enter** to set the *"object to instance at each point..."*

Now there will be one soccer ball at all the points of the box. These are being instanced which is an efficient way of working. You can now use these to simulate the balls falling to the ground.

DOPS

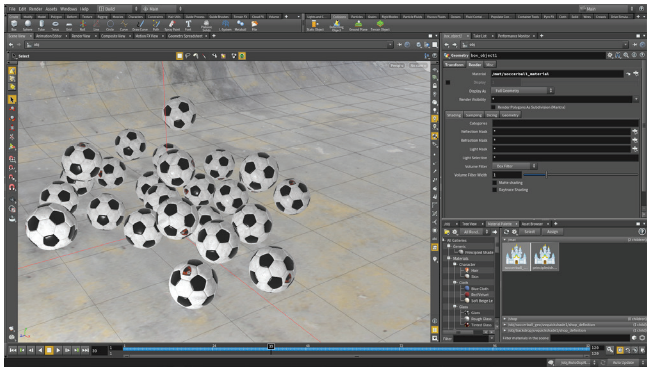
Dynamic operators work a little differently compared to other networks in Houdini. Nodes collect all of the dynamic objects from your scene, feed them into solvers that are then merged and affected by forces. The results are then sent back to the geometry networks of the participating objects. Therefore you can expect to find some nodes at the geometry network level that are used to set up the simulation and receive the finished sim data.





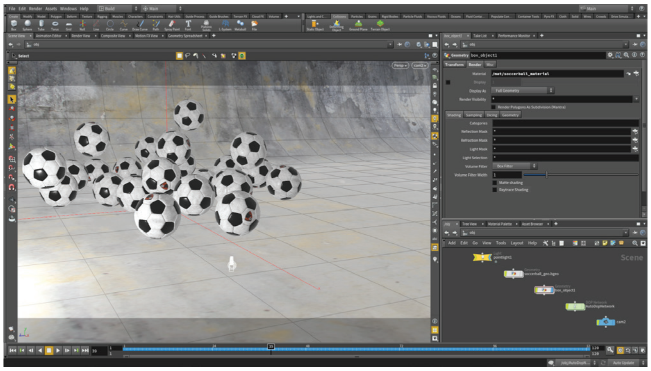
05 Go back to the object level and select the *backdrop* geometry. From the **Collisions** shelf, click on the **Terrain Object** tool. Now the soccer balls with collide with this surface.

In the timeline, press **Play** to begin the simulation. You will watch as the balls fall and hit the ground. You will notice a blue bar that appears in the timeline to indicate how much of the simulation has been cached. You can scrub anywhere within this area at any time to preview the results. Once the whole sequence has been simulated go back to the Object level.



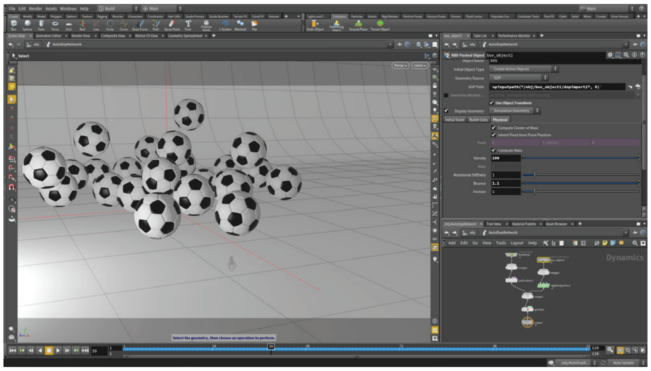
06 While you assigned the *soccerball_material* to the *soccerball_geo.bgeo* object earlier, the material is not assigned to the instanced soccerballs. This is because the **RBD Instanced Objects** tool hid the *soccerball_geo.bgeo* node, while merging it into the *box_object* object so that it could be instanced.

If you go to the **Render** tab of the *box_object* object you will see that there is no material assigned. **Reassign** the material to *box_object* to see the textures on all the instanced soccer balls.



07 **Tumble** the view in the viewport to center around the simulation then from the top right of the main viewport, click on the **No Cam** button and choose **New Camera**.

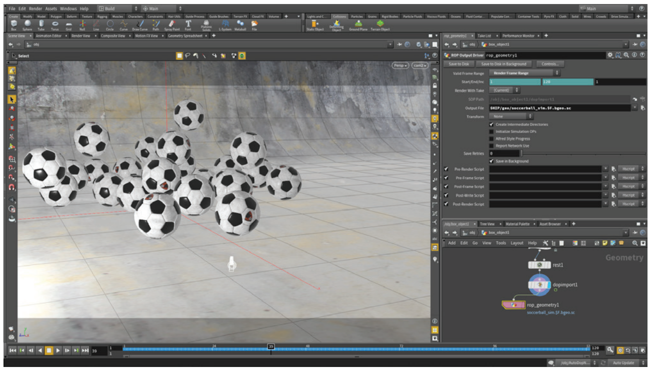
Play back or **scrub** through the simulation and make adjustments to the camera if needed.



08 From the object level, double-click on the *AutoDopNetwork* node to see the DOP nodes. To organize the nodes, you can **press L** in the Network view to clean up the layout.

Select the *box_object* node and click on the **Physical** tab. Set **Density** to **100** and **Bounce** to **1.2**. Press **Play** to re-simulate with the new “bouncier” values.

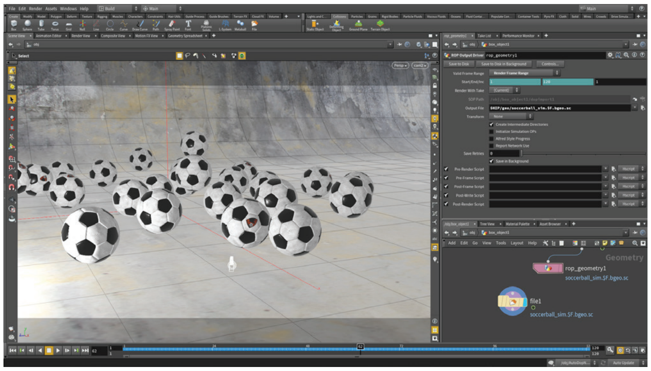
Go back and tweak to get the results that you want.



09 Navigate back up to the object level then into the *box_object* geometry network. **Press tab** and start typing **ROP**. Choose **ROP Output Driver** from the menu and place the node at the end of the chain. Wire the output of the *dopimport* node into the *rop_geometry* node. In the parameter pane, set **Valid Frame Range** to **Render Frame Range** and the **Output file** to:

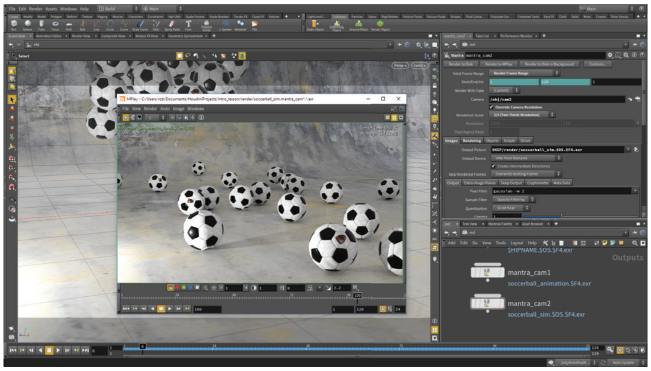
```
$HIP/geo/soccerball_sim.$F4.bgeo.sc
```

Click **Save to Disk**. This will cache out the simulation so that you can work more efficiently when you render.



10 In the Network view, press **tab > File. Click** to place the node next to the output node. In the Parameter pane, click on the button next to **Geometry File** and go into the `/geo` directory and select the `soccerball_sim.$F4.bgeo.sc` file. Click **Accept**. Set the **Display flag** on the new *file* node.

Now this cached sequence will be used to render and you won't need to resimulate each time you render out the shot.



11 Go back to the Object level. From the **Render** menu, select **Create Render node > Mantra PBR**. This creates another Mantra node. Make sure the **Camera** parameter is pointed to *cam2*, turn on **Override Camera Resolution** and **Resolution Scale** is set to **2/3 resolution**. At the top of the parameter pane, set **Valid Frame Range** to **Render Frame Range**. Click on the **Images** tab and set **Output Picture** to:

```
$HIP/render/soccerball_simulation.$F4.exr
```

This will create a sequence of numbered images of your animation. Click on **Render to Disk**. When it is finished, preview in **Mplay**.

CONCLUSION

You have now built a scene from scratch, touching on many different aspects of Houdini. You have modeled, set up textures, animated, rendered and simulated. Along the way you have learned about the different Houdini contexts and how to navigate back and forth between them.

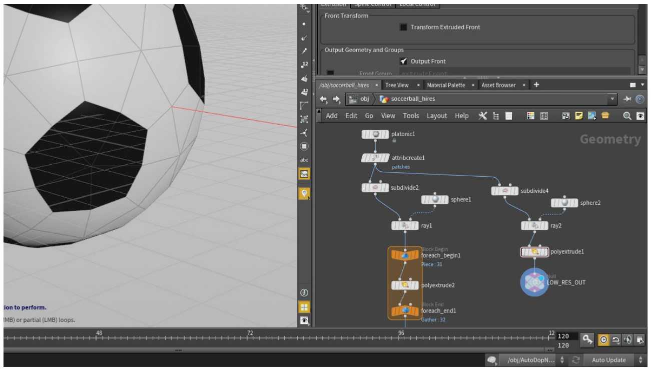
While this lesson doesn't result in blockbuster VFX, it introduces you to fundamental skills which you will carry with you as you dive deeper into Houdini and begin exploring its comprehensive toolset.

There is a wealth of learning material available on the SideFX website to help you take your next steps.

Best of luck on your journey!

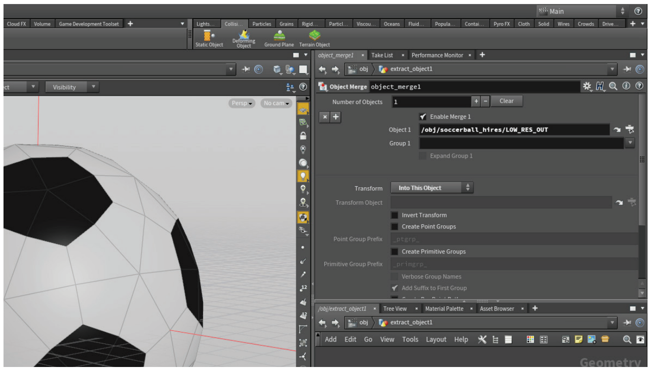
BONUS: Create Game-Ready Art for UE4

If you are a game artist then rendering out a simulation will only help you to create game cinematics. To create game-ready art you will need optimized content that is exported properly to your game engine. To do that with the soccerballs, you will bake out textures for a low resolution model using the high-resolution geometry and the existing texture maps. You will then use this new geometry in the RBD sim and export to FBX for use in UE4.



01 At the object level, select the *soccerball_geo* object and then **Alt-drag** to make a copy. Unparent it from the Ctrl and place it at the origin. Name this new node *soccerball_hires.*

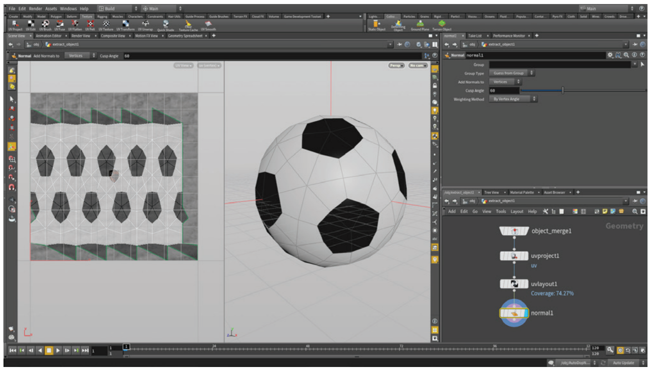
Dive into *soccerball_hires* object. Select the **Subdivide, Ray** and **Sphere** nodes then **Alt-drag** to area on the right to make copies. Leave them connected. On the new **Subdivide** node change **Subdivide Depth** to **1**. Add a **PolyExtrude** node after the *Ray*. Set **Distance** to **0.1**. This adds thickness to better align it with the other soccerball. Add a null object to the end of this chain and name it LOW_RES_OUT.


Object Merge object_merge1
Number of Objects 1
Enable Merge 1
Object 1 /obj/soccerball_hires/LOW_RES_OUT
Transform Into This Object
extract_object1

02 Go to the **Select** tool then with your mouse over the Scene View, **press 4** to go to primitive selection then **press n** to select all of the primitives [faces]. Go to the **Modify** shelf and click on the **Extract** tool. This extracts the low-res version of the soccerball into a new object called *extract-object* with an **Object Merge** node which is pulling the geometry from the other network. The path on the **Object Merge** node will be:

`/obj/soccerball_geo1/LOW_RES_OUT`

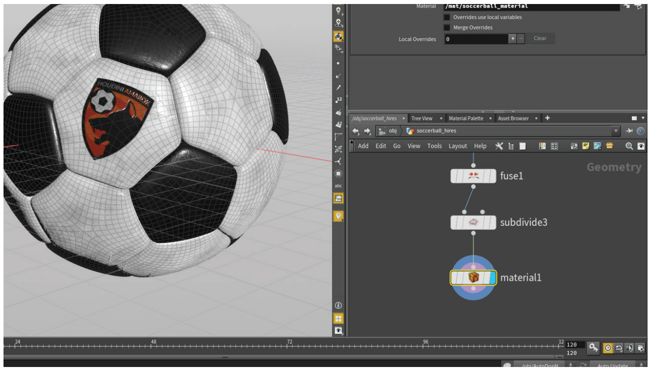
This is a live connection so changes made to the original network will also update this network.



03 Go to the **Texture** shelf and click on **UV Project**. In the Parameter pane for the *uvproject* node, set **Projection** to **Polar**. In the network editor, **RMB-click** on the output of the *uvproject* node then start typing **UV Layout...** Choose **UV Layout** and click to place it down.

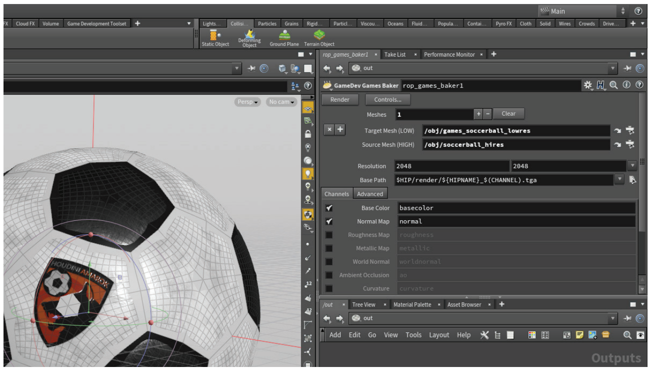
Next, add a **normal** node to the end of the chain then set its **Display Flag**. This will add smoothing normals to the low res soccerball.

Go back to the object level and rename the *extract_object* object to *games_soccerball_lowres.*



04 Go back to the object level then double-click on the *soccerball_hires* object to dive into it. Delete the *Transform* and *Bend* nodes because they were part of the bouncing ball setup that we aren't using in this network.

Add a **Material** node to after the *subdivide* node and set the **Material** parameter on the new node to `/mat/soccerball_material`. Set the **Display Flag** on the *Material* node. In order for texture baking to work, you need to have the Material assigned at the geometry level.

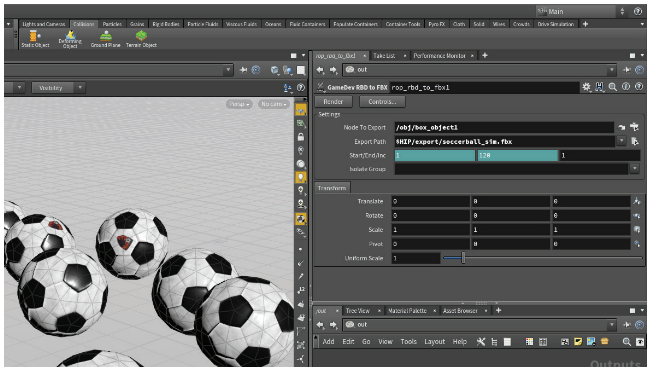


GameDev Games Baker: rop_games_baker1
Meshes: 1
Target Mesh (LOW): /obj/games_soccerball_lowres
Source Mesh (HIGH): /obj/soccerball_hires
Resolution: 2048 2048
Base Path: $HIP/render/${HIPNAME}_$(CHANNEL).tga
☑ Base Color: basecolor
☑ Normal Map: normal
□ Roughness Map: roughness
□ Metallic Map: metallic
□ World Normal: worldnormal
□ Ambient Occlusion: ao
□ Curvature: curvature

05 Click on the plus sign next to the shelf and find the **Game Development Toolset** shelf to expose it. Click on **Update Toolset** to add these tools to Houdini.

Click on the **Games Baker** tool. Set **Target Mesh** to */obj/game_soccerball_lowres* and **Source Mesh** to */obj/soccerball_hires*. Click **Render**. To see the results, you can open up the two textures using **Mplay** to verify that they did what you needed them to.

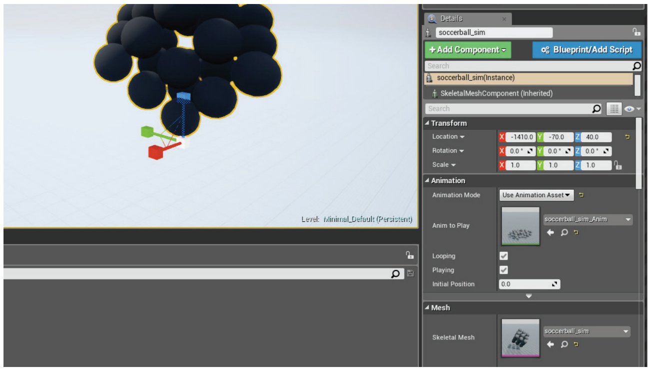
Create a new **Principled Shader** material, assign the *basecolor* and *normal* maps. Turn on **Flip Y** under the normal map then assign it to *games_soccerball_lowres*.



GameDev RBD to FBX: rop_rbd_to_fbx1
Settings
Node To Export: /obj/box_object1
Export Path: $HIP/export/soccerball_sim.fbx
Start/End/Inc: 1 120 1
Isolate Group:
Transform
Translate: 0 0 0
Rotate: 0 0 0
Scale: 1 1 1
Pivot: 0 0 0
Uniform Scale: 1

06 Go to the *box_object* node and set the **Display Flag** on the *dopimport* node. Select the *fetch_instance_Geometry* node. Set **object1** to */obj/ games_soccerball_lowres.* Go to Frame 1 and press **Play** to simulate using the new low resolution geometry.

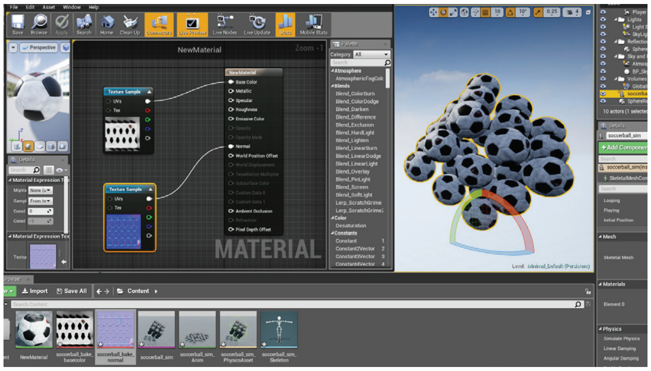
On the **Games** shelf, click on the **RBD to FBX** tool. Select the *rbd_to_fbx* ROP node and set the **Node to Export** to the */obj/ box_object* node and **Export Path** to *$HIP/export/soccerball_sim.fbx*. Set **Uniform Scale** to **100** because Houdini and UE4 have different scale factors – meters compared to centimeters. Click **Render** to export the file to disk.



Details: soccerball_sim
soccerball_sim(Instance)
SkeletalMeshComponent (Inherited)
Transform
Location: X -1410.0 Y -70.0 Z 40.0
Rotation: X 0.0° Y 0.0° Z 0.0°
Scale: X 1.0 Y 1.0 Z 1.0
Animation
Animation Mode: Use Animation Asset
Anim to Play: soccerball_sim_Anim
Looping: ☑
Playing: ☑
Initial Position: 0.0
Mesh
Skeletal Mesh: soccerball_sim

07 IN UE4 – Press **Import** and grab the *soccerball_sim.fbx* file. Check **Skeletal Mesh, Import Mesh, Import Animations** and **Import Meshes in B one.** Expand the **Mesh** section and set **Normal Import Method** to import **Normals and Tangents**. Click **Import**.

Three items appear in the content list. Drag the *geometry* asset into your workspace. In the **Details** pane, set **Animation Mode** to **Use Animation Asset**. Drag the **Anim** asset to the **Anim to Play** parameter.



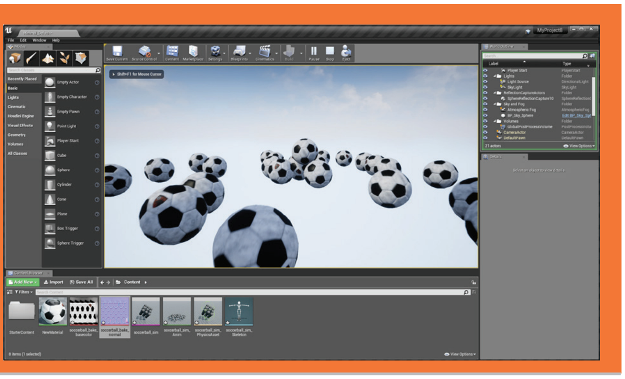
08 IN UE4 – Click **Import** and find the two baked texture maps from the Houdini project directories.

Under **Material**, click on menu and choose **Create New Asset > Material**. Double click on the *NewMaterial* in the **Content Browser** to open up a Material editor. Drag the *basecolor* and *normal map* textures from the content browser to the Material editor and wire them into the **New Material** node. Click **Save** in the Material editor to accept these changes Close the Material editor.

Press **Play** to see the simulation working in game space.

PLAY THE SIM IN UE4

You now have low-res geometry with baked textures animated using bones within the UE4 environment. Houdini is a great tool for creating game-ready art and, in some cases, the same content that you set up for high resolution rendering can be retrofitted for a more interactive experience.



HOUDINI FOUNDATIONS

PROCEDURAL GAME ASSETS FOR UE4

To create game assets using Houdini's node-based workflow it is important to start learning how to think and work procedurally. In this lesson, you will learn how to create these assets using procedural nodes and networks then deploy them directly into Unreal Engine 4 using the Houdini Engine.

Along the way, you will get to use different aspects of Houdini's user interface. You will learn how the different UI elements work together to support you as you build your game assets. You will also learn how to create collision geometry and how to create and export a rigid body simulation for use in UE4.

LESSON GOAL

To create assets that can be imported into UE4 as game art.

WHAT YOU WILL LEARN

- How to copy objects to points.
- How to control the orientation and scale of the objects using attributes.
- How to work with **Houdini's Nodes and Networks** to control the flow of data.
- How **Houdini Digital Assets** can be used to package and share your solution with others.
- How to load a Houdini Digital Assets into the **UE4 Editor**.
- How to create a game asset with **Collision Geometry** for use in UE4.
- How to export a **Rigid Body Simulation** as an FBX file for use in UE4.

LESSON COMPATIBILITY

Written for the features in Houdini 16.5.378+

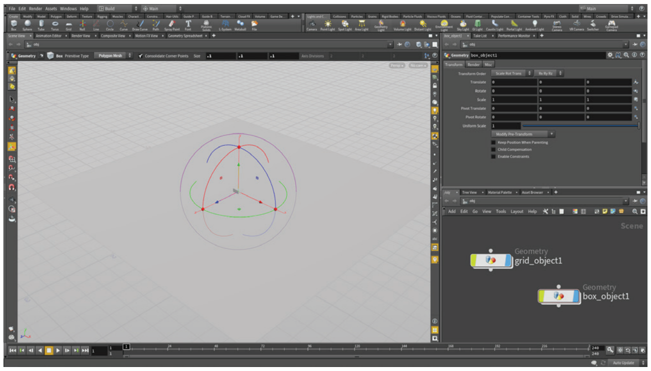
The steps in this lesson can be completed using the following Houdini Products:

Houdini Core	✔
Houdini FX	✔
Houdini Indie	✔
Houdini Apprentice	✘
Houdini Education	✔

Document Version 1.0 | February 2018

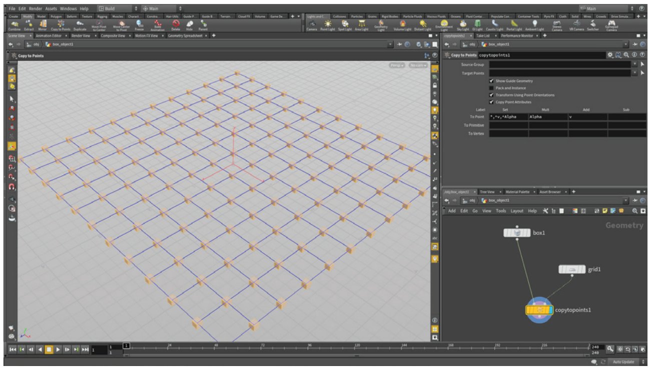
PART ONE:
Copy to Points

In this part of the lesson, you are going to copy some boxes to points of a grid. You will then randomize those points to achieve a more organic look and add random attributes to rotate and scale the boxes to create more variety in the distribution of the shapes. This creates a procedural system that you will later wrap up into a Houdini Digital Asset.



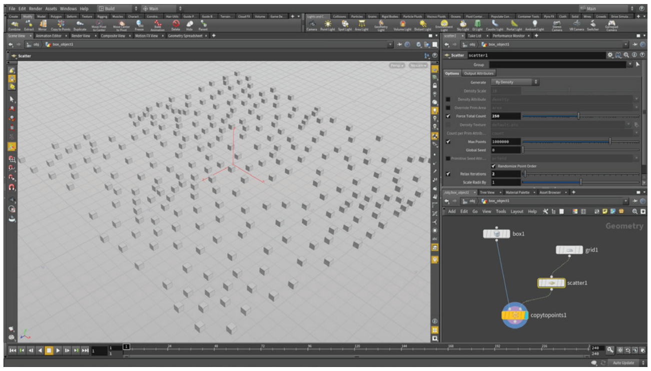
01 In the viewport, press the **c** key to bring up a radial menu and select **Create > Geometry > Grid**. Release and then press **Enter** to place at the origin. You will start by using the points on this grid surface to instance geometry.

Use the same radial menu to **Create > Geometry > Box** and again press **Enter** to place at the origin. In the Operation Control Bar at the top of the viewport, set the **Size** to **0.1 0.1 0.1**. This is the geometry that you will be copying to the points on the grid.



02 With the *box* still selected, go to the **Modify** shelf and click on the **Copy to Points** tool. Select the *grid* and press **Enter**. Now the boxes have been copied to all of the grid points.

You can now adjust the look of the system by editing parameters. Select the *grid* node and change the **Size** to **6, 6** and the **Rows** and **Columns** to **12**. The grid gets bigger and has more points but the boxes aren't being copied to all the points. Click on the *copytopoints* node where you can see that **Target Points** are set to **0-99** to match the points on the original grid. Remove the **0-99** which will copy boxes to the whole grid.

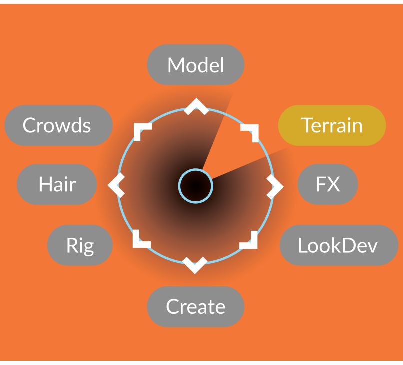


03 In the Network Editor, press tab and begin to type **sca...** then choose the **Scatter** tool. Place the *scatter* node in the Network editor between the *grid* and the *copytopoints* nodes. It will wire itself into the network automatically and the boxes will now be copied to these new points.

Set **Force Total Count** to **250**. Play with the **Relax Iterations** to adjust how the points are being laid out. The scatter node gives you a more organic layout compared to the original grid points.

DIFFERENT WAYS TO CREATE NODES

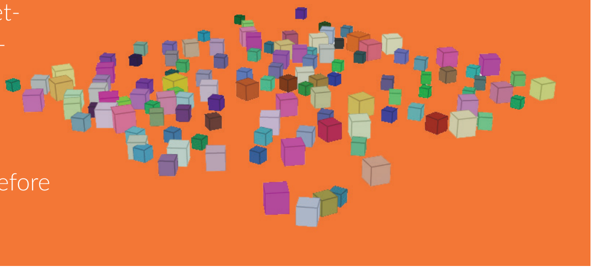
In this lesson, we have create nodes in either the Viewport or the Network Editor. In Houdini, you can use the **Radial menu**, the **Shelf** or the **Tab** key to interact with geometry in the viewport which will create nodes in the Network Editor. The **Tab** key can be used in the Network Editor to place nodes directly and you can wire it into your chain of nodes as you see fit. Both ways are valid ways of working in Houdini and over time you will learn the best time to use each method.

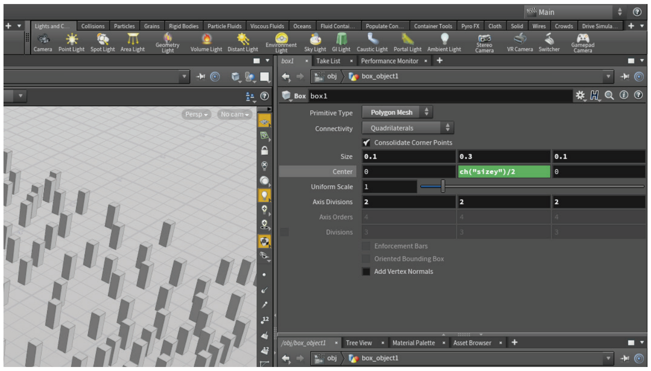


HOW ATTRIBUTES WORK

Attributes that are assigned to geometry are passed down the network chain to different nodes. In the case of this network, the attributes being assigned to the grid geometry are passed on to the scattered points which in turn affect the copied geometry. This is an important way to control the flow of data in Houdini.

The attributes are initially assigned to the points on the grid therefore more points lead to more randomization in the attribute values.



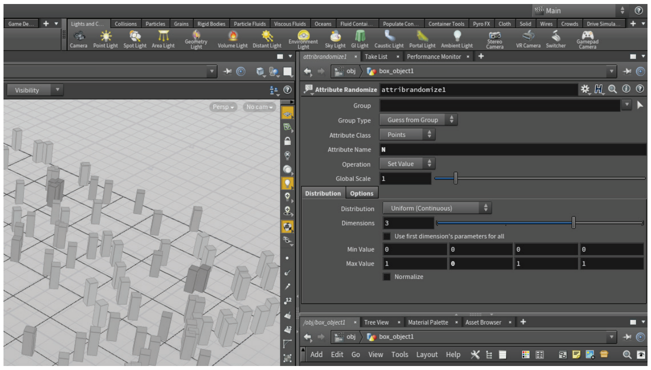


Box box1

Primitive Type: Polygon Mesh
Connectivity: Quadrilaterals
☑ Consolidate Corner Points
Size: 0.1 | 0.3 | 0.1
Center: 0 | ch("sizey")/2 | 0
Uniform Scale: 1
Axis Divisions: 2 | 2 | 2
Axis Orders: 4 | 4 | 4
Divisions: 3 | 3 | 3
□ Enforcement Bars
□ Oriented Bounding Box
□ Add Vertex Normals

04 Select the *box* node. Select the value in **Size Y** and drag it onto **Center Y**. From the menu, choose **Relative Channel Reference** then add "/2" to divide it by two. This expression will make sure that the box sits on the ground even if you make it taller.

Set **Size Y** to **0.3** to test the expression. You can see that all of the boxes are sitting on the ground. This is a useful trick that you may find yourself using with many of your modeling tasks.

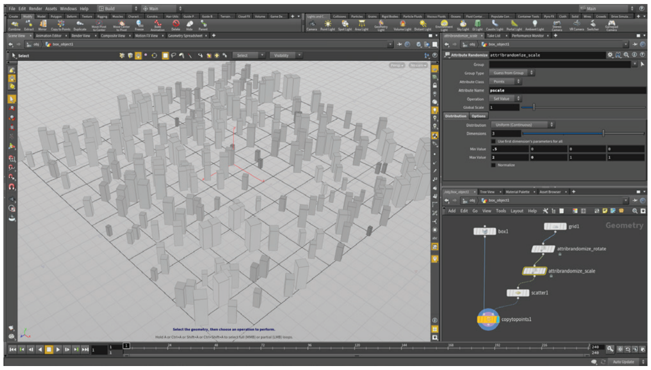


Attribute Randomize attribrandomize1

Group:
Group Type: Guess from Group
Attribute Class: Points
Attribute Name: N
Operation: Set Value
Global Scale: 1
Distribution | Options
Distribution: Uniform (Continuous)
Dimensions: 3
□ Use first dimension's parameters for all
Min Value: 0 | 0 | 0 | 0
Max Value: 1 | 0 | 1 | 1
□ Normalize

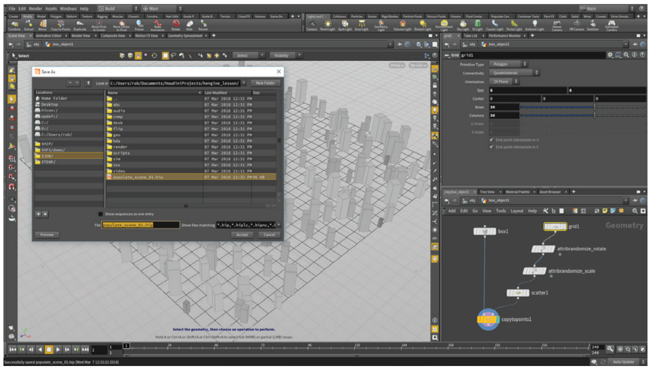
05 In the Network Editor, press tab and begin to type **attr...** then choose the **Attribute Randomize** tool. Place the *attributerandomize* node between the *grid* and the *scatter* nodes. At first you will see random colors on your boxes because the default attribute is color (Cd).

Set the **Attribute Name** to **N**. This changes the attribute to the normal direction and all the boxes are now pointed in different directions. Set **Max Value Y** to **0** which will limit the randomness to the X and Z directions. Name it *attrrandomize_rotate*.



06 Press the **Alt** key and drag on the *attrrandomize_rotate* node to make a copy of it. Wiggle the node to break it from any connections then drop it between the *attrrandomize_rotate* and the *scatter* nodes. Name it *attrrandomize_scale*.

Set the **Attribute Name** to **pscale**. Set **Min Value** to **0.5** and **Max Value** to **2**. This gives you a nice variety of sizes for the boxes on the grid.



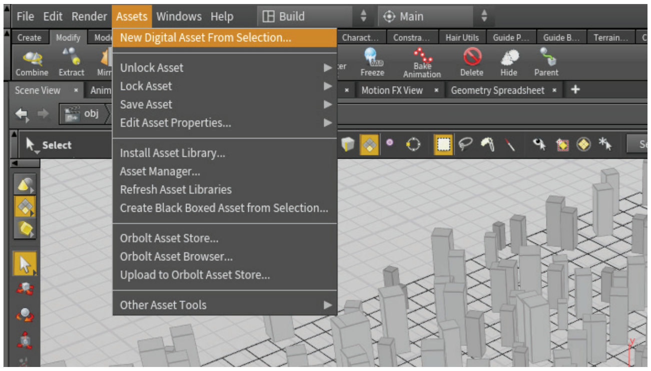
07 Select the *grid* node and increase the **Rows** and **Columns** to **30** – this creates more points on the grid which creates a wider variety of values on the scatter points.

Select **File > New Project.** Change the **Project Name** to *hengine_lesson* and press **Accept**. This creates a project directory with subfolders for all the files associated with this shot.

Select **File > Save As...** You should be looking into the new *hengine_lesson* directory. Set the file name to *populate_scene_01.hip* and click **Accept** to save.

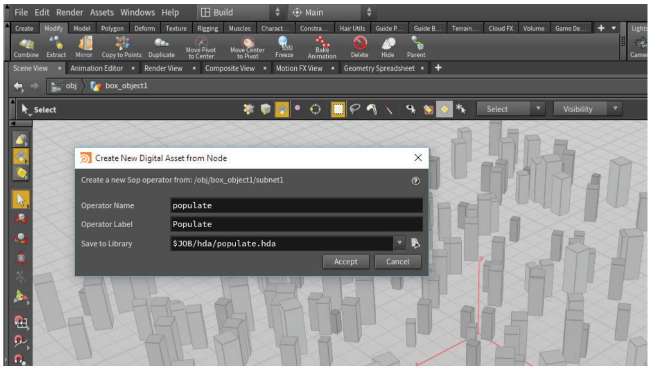
PART TWO:
Create a Houdini Digital Asset

In this part of the lesson, you are going to use a teapot model to define a grid of points then copy cubes to each of the points. You will learn how to work in the Scene view to create objects and the Network view to make adjustments. You will also use expressions to help control the final solution.



01 Select all the nodes in the Network editor. From the **Asset** menu, choose **New Digital Asset from Selection**. This will collapse the network into a subnetwork then use that subnet node to create the Digital Asset.

The nodes you used to build the asset will remain a part of the asset even after you save it. This will allow you to continue making changes even after you have started using it in your UE4 game levels.



Create New Digital Asset from Node

Create a new Sop operator from: /obj/box_object1/subnet1

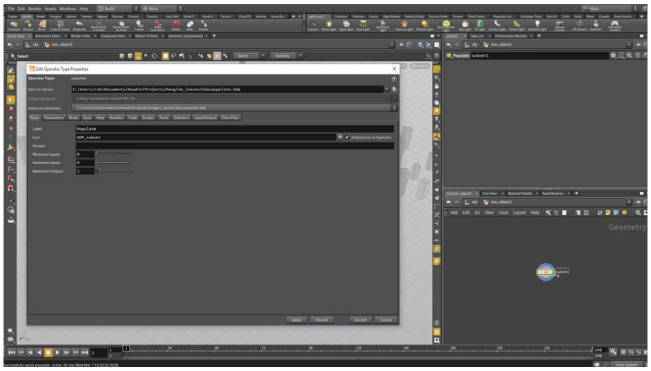
Operator Name: populate

Operator Label: Populate

Save to Library: $JOB/hda/populate.hda

02 Set the **Operator Name** to *populate* which will change the **Operator Label** to *Populate*. Click on the button to the far right of **Save to Library.** In the *Locations* sidebar, click on $JOB/ and then double-click on the *hda* directory. Press **Accept** and then **Accept** again to save the asset to disk.

This creates a Houdini Digital Asset file (.hda) on disk that is now being referenced by this scene. It can also be referenced into other Houdini scenes or into other applications such as UE4 using the Houdini Engine.



Edit Operator Type Properties

Operator Type: populate

Save to Library: C:/Users/rob/Documents/HoudiniProjects/hengine_lesson/hda/populate.hda

Install Library to: Library installed to Current HIP File

Switch to Definition: C:/Users/rob/Documents/HoudiniProjects/hengine_lesson/hda/populate.hda

Label: Populate

Icon: SOP_subnet ☑ Embed Icon in Operator

Version:

Minimum Inputs: 0

Maximum Inputs: 0

Maximum Outputs: 1

03 The Edit Type Properties window opens up. This panel is where you will build the user interface for your asset. You will revisit this window later. Click **Accept** to close this window.

In order for you to maintain the procedural nature of the Houdini Digital Asset, you can build a high level interface that can be used to access the nodes inside the network. You will add to the interface for this asset later on in the lesson.

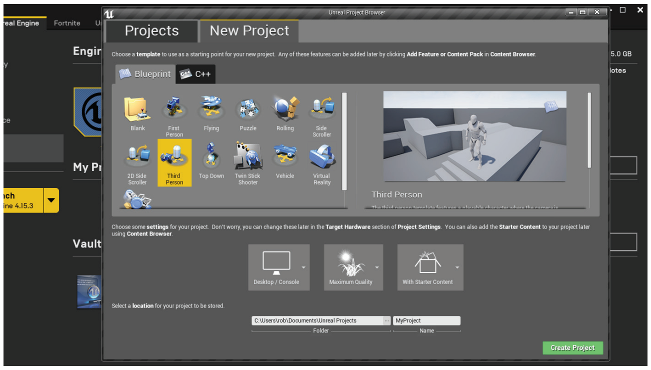
WHAT IS AN HDA FILE?

Houdini nodes and networks can be encapsulated into single nodes called **Houdini Digital Assets** which let you share your techniques with colleagues. These assets are saved to disk inside files known as **.hda** files.

Asset files created in older versions of Houdini may have a different extension **.otl** which means Operator Type Library - both these file types work the same way.

PART THREE:
Import the Asset into UE4

Now that you have a digital asset file on disk, you can import it into Unreal Engine 4. This is possible because of the Houdini Engine plug-in which connects the two applications. Houdini Digital Assets that are loaded into UE4 are cooked using the Houdini under the surface. This lets gives you access to the procedural nature of the node within UE4.


Unreal Project Browser

Choose a **template** to use as a starting point for your new project. Any of these features can be added later by clicking **Add Feature or Content Pack** in **Content Browser**.

Choose some **settings** for your project. Don't worry, you can change these later in the **Target Hardware** section of **Project Settings**. You can also add the **Starter Content** to your project later using **Content Browser**.

Select a **location** for your project to be stored.

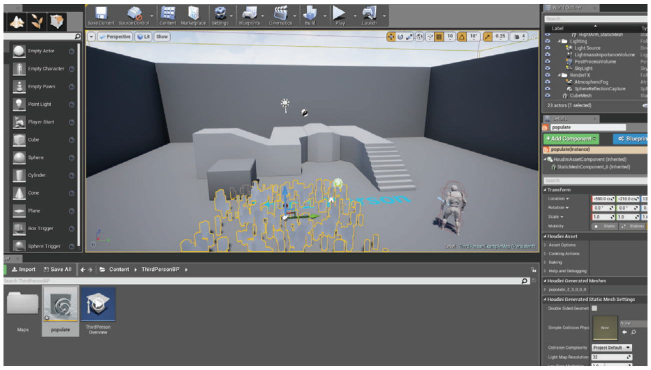
Folder: C:\Users\rob\Documents\Unreal Projects

Name: MyProject

01

IN UE4 – Open Unreal Engine 4 and from the main panel click on the **New Project** tab and choose the **Third Person** template. Click **Create Project.**

This project gives us some context for the asset and a character to walk around once the asset is in place. It will also give us a chance later to explore methods of bringing collision geometry from Houdini into UE4.



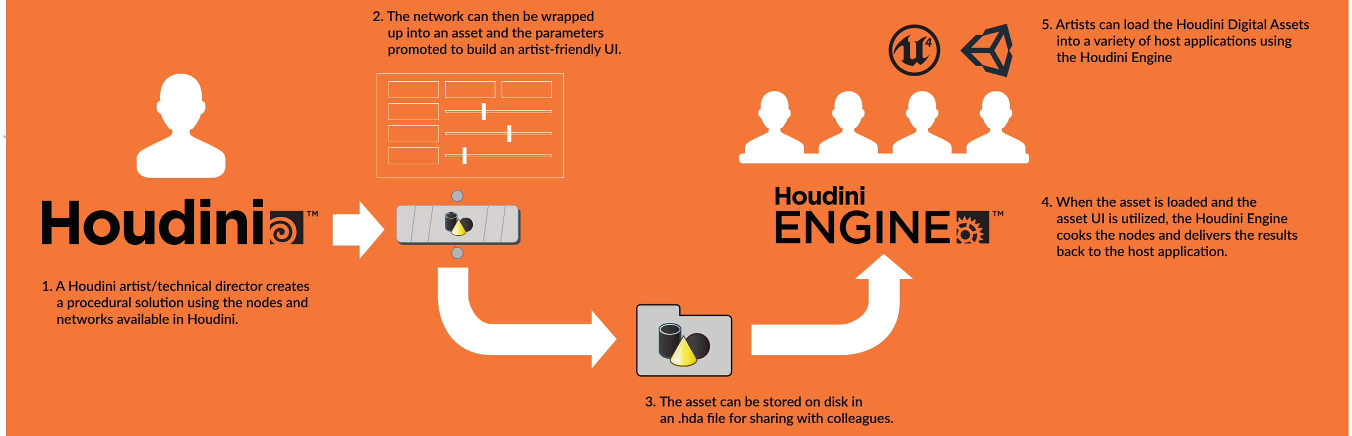
02

From the Content Browser, click **Import** and find the *populate.hda* asset file in the Houdini project directory. Drag the asset from the Content Browser to the 3D workspace. Press **Play** and walk around the asset. It is there but there is nothing special about it at this point. Press **Esc** to return to the asset UI.

NOTE: Importing assets will only work if the Houdini Engine plug-in for UE4 has been installed correctly. If your UE4 installation is on your D: drive instead of your C: drive then the plug-in will not be installed. You will need to move it from the Houdini directories to the UE4 plug-in directory by hand.

HOW DOES THE HOUDINI ENGINE WORK

Houdini Engine lets you load Houdini Digital Assets into your game editor. Any parameters added to the asset are available for editing and handles can be set up in the viewport for interactive manipulation. One asset can be used multiple times in a level or in multiple levels. Changes made to the asset on disk, even deep into production, will update all instantiations of the asset.

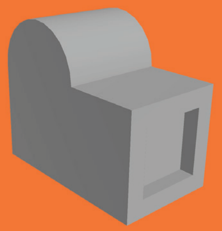


1. A Houdini artist/technical director creates a procedural solution using the nodes and networks available in Houdini.
2. The network can then be wrapped up into an asset and the parameters promoted to build an artist-friendly UI.
3. The asset can be stored on disk in an .hda file for sharing with colleagues.
4. When the asset is loaded and the asset UI is utilized, the Houdini Engine cooks the nodes and delivers the results back to the host application.
5. Artists can load the Houdini Digital Assets into a variety of host applications using the Houdini Engine

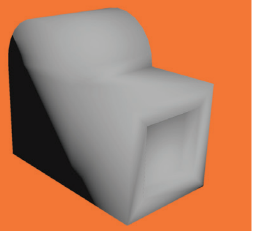
VERTEX NORMALS

By default, Houdini objects like the box have point normals but don't have vertex normals. To make sure that you have proper vertex normals, you can add the normal node which uses a cusp value to decide which edges should appear hard and which ones appear soft.

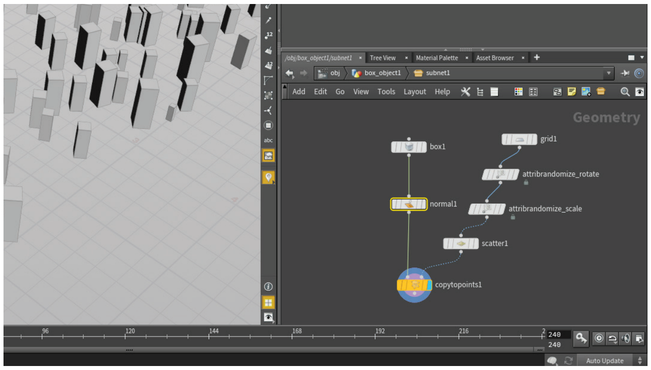
Game editors such as UE4 need vertex normals to display properly which can be set up easily in our existing network.



With Normals

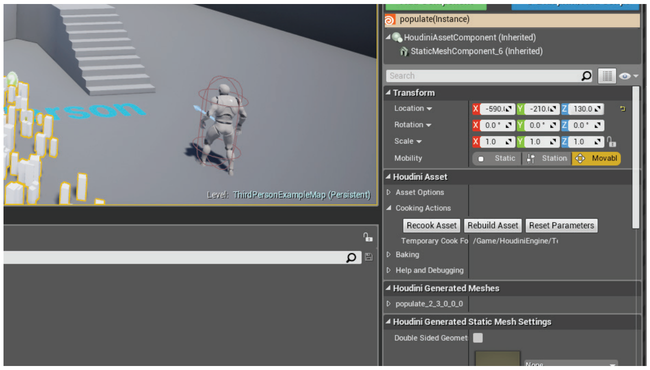


Without Normals



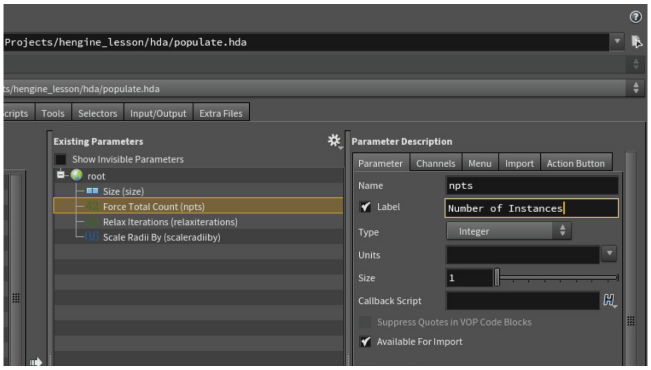
03 **IN HOUDINI** – One thing that you will notice about the asset in UE4 is that the boxes don't have sharp corners. To fix this you should go back to Houdini and in the Network editor, press **tab > "norm..."** then choose the **Normal** tool.

Add a Normal node right after the box node. From the **Asset** menu, choose **Save Asset > Populate**. This saves the change back into the .hda file which makes the change available to anyone who is using the asset. In our case this means that we can update the asset definition inside UE4 and the correct normals will display.



04 **IN UE4** – In the details panel, under Houdini Asset, open the **Cooking Actions** section. Click the **Rebuild Asset** button to accept the changes. This makes sure that the changes you made inside Houdini are properly updated in the UE4 scene.

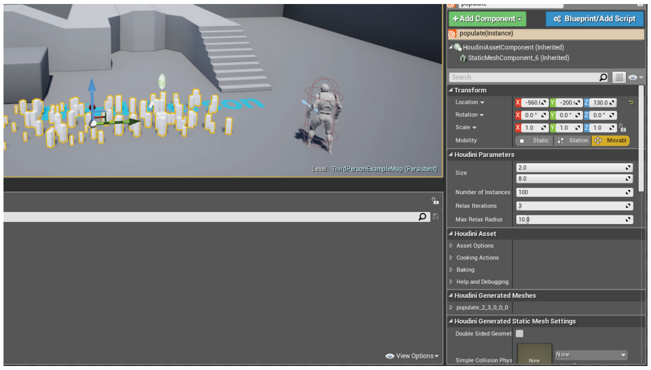
The asset now has proper normals but you don't have any control over the procedural network. To create an interface that you can use in Houdini, you are now going to promote some parameters from inside the asset to the top level.



Name: npts
☑ Label: Number of Instances
Type: Integer
Units:
Size: 1

05 **IN HOUDINI** – Choose **Assets > Edit Asset Properties > Populate**. Click on the **Parameters** tab. In the Network editor, click on the *grid* node. From the Parameter pane, drag the **Size** parameter onto root in the **Existing parameters** list.

In the Network editor, click on the *scatter* node. From the Parameter pane, drag the **Force Total Count** parameter onto root. In the Parameter description, change the **Label** to **Number of Instances**. Drag the **Relax Iterations** and **Max Relax Radius** to add these parameters to the asset. Click **Accept** which saves these new parameters to the asset.



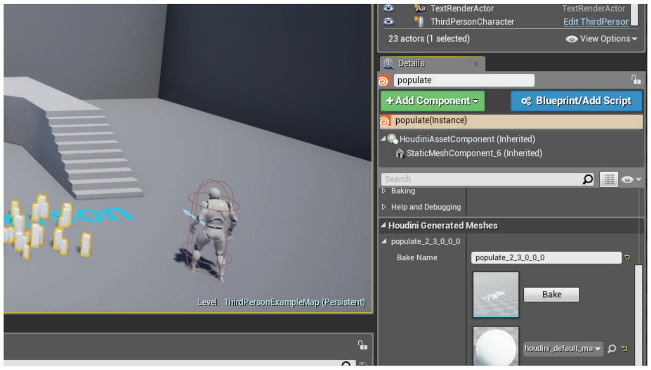
Size: 2.0, 8.0
Number of Instances: 100
Relax Iterations: 3
Max Relax Radius: 10.0

06 **IN UE4** – Click the **Rebuild Asset** button to accept the changes. The promoted parameters show up in the Parameter pane. Change the **Size** and **Number of Instances** to see how they affect the look of the resulting grid of boxes.

Now the procedural nature of the asset has been exposed and you can create unique versions of the asset in different levels. You can add more than one of these populate assets in this level and each of them can have unique settings while referencing the same asset in the .hda file. You can also use this asset in multiple levels being worked on by multiple artists.

PART FOUR: Set up Instancing

When you set up instancing in Houdini properly, you can replace the default shape you have copied to the points with other UE4 props. You can add more than one prop that will be randomly distributed in place of your default shape. You are now going to make sure that instancing is in fact set up properly and then you will use this to add some other props into the system.



23 actors (1 selected)

populate

populate(Instance)

HoudiniAssetComponent (Inherited)

StaticMeshComponent_6 (Inherited)

Houdini Generated Meshes

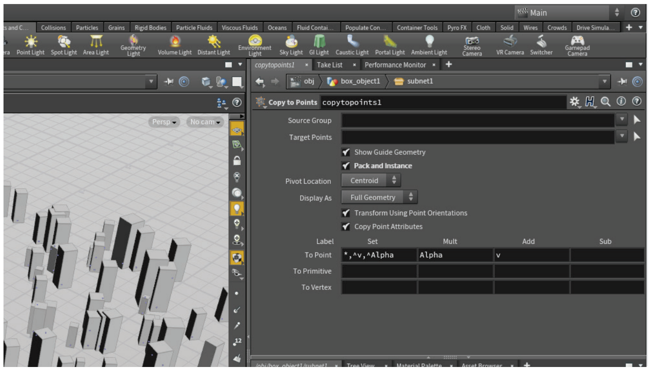
populate_2_3_0_0_0

Bake Name: populate_2_3_0_0_0

houdini_default_ma

01 **IN UE4** – In the Details panel, go to the **Houdini Generated Meshes** section. You can see that the whole collection of boxes is being imported as a single mesh. This means that we are not yet using any instancing. Houdini is copying the boxes to the points then outputting the resulting geometry for UE4.

This would not be very efficient for gameplay. Therefore we need to set things up just a little differently to get the instancing that we need for this tool to work properly.



Copy to Points: copytopoints1

Source Group:

Target Points:

☑ Show Guide Geometry

☑ **Pack and Instance**

Pivot Location: Centroid

Display As: Full Geometry

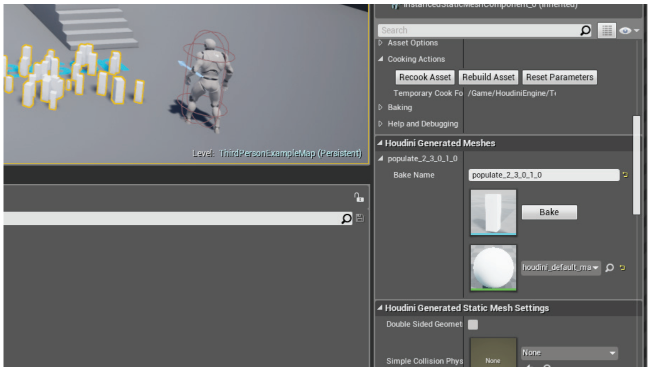
☑ Transform Using Point Orientations

☑ Copy Point Attributes

Label	Set	Mult	Add	Sub
To Point	`*,^v,^Alpha`	Alpha	v	
To Primitive				
To Vertex				

02 **IN HOUDINI** – Select the *copytopoints* node and turn on **Pack and Instance**. From the **Asset** menu, choose **Save Asset > Populate**.

By using packed primitives, the box geometry feeding into the *copytopoints* node is treated as a single primitive. This sets up instancing in Houdini and in turn triggers instancing in UE4 when this asset is loaded into the editor using the Houdini Engine plug-in.



Temporary Cook Fo: /Game/HoudiniEngine/T

Houdini Generated Meshes

populate_2_3_0_1_0

Bake Name: populate_2_3_0_1_0

houdini_default_ma

Houdini Generated Static Mesh Settings

Double Sided Geomet: □

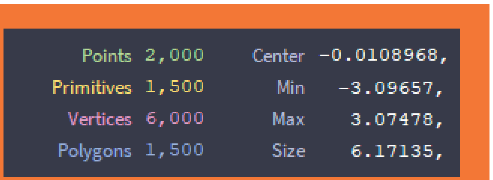
Simple Collision Phys: None

03 **IN UE4** – Click the **Rebuild Asset** button to accept the changes. Go to the **Houdini Generated Meshes** section. You can see that only a single box is being imported and instanced to the points. It is then rotated and scaled based on the attributes we set up earlier.

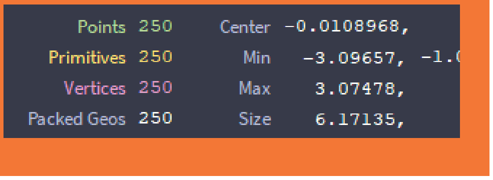
Now you can replace this default box with other geometry in your UE4 environment. This is the ideal way of populating a series of points because you have lots of flexibility in the editor to add different objects to the system.

PACKED PRIMITIVES IN HOUDINI

In Houdini, packed primitives provide an efficient way of managing instances for viewport display and rendering. The geometry feeding the copy node is packed into a single primitive and then treated like an instance. For the copied boxes shown in this lesson, this takes you from 1,500 primitives down to 250 once packing is in place. With the Houdini Engine plug-in for UE4, the packed primitive is recognized as a UE4 instance which will allow for more efficient gameplay.



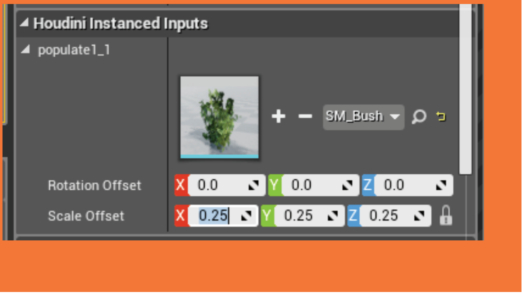
```
    Points  2,000     Center  -0.0108968,
Primitives  1,500        Min    -3.09657,
  Vertices  6,000        Max     3.07478,
  Polygons  1,500       Size     6.17135,
```



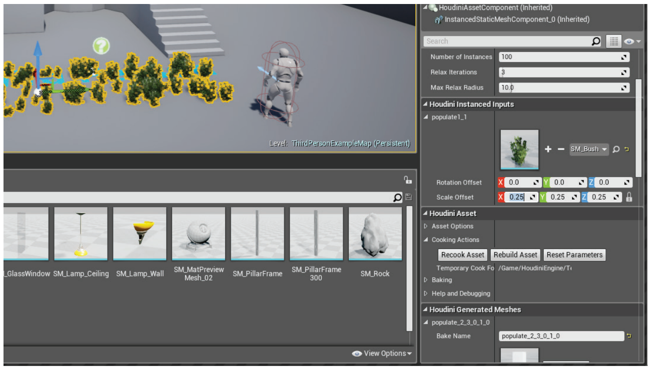
```
     Points  250    Center  -0.0108968,
 Primitives  250       Min    -3.09657,  -1.
   Vertices  250       Max     3.07478,
Packed Geos  250      Size     6.17135,
```

INSTANCES IN UE4

When the Houdini Engine plug-in detects packed primitives, an Instanced Input is created in the Details tab that contains the input geometry and some parameters for Rotating and Scaling the instance. You can replace this with geometry from your UE4 content window and size it accordingly. The Plus sign lets you add more instanced inputs to create more of them in the same system.



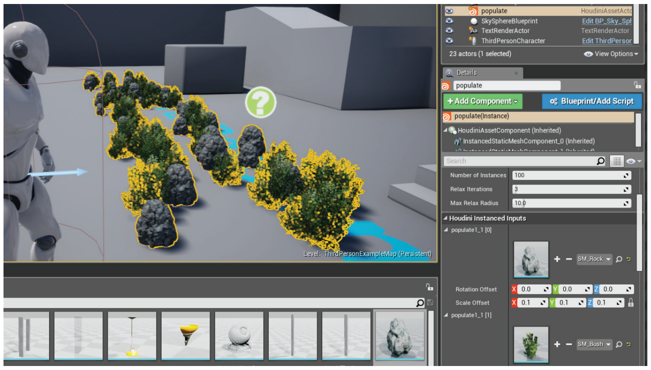
Houdini Instanced Inputs
populate1_1
SM_Bush
Rotation Offset X 0.0 Y 0.0 Z 0.0
Scale Offset X 0.25 Y 0.25 Z 0.25



Number of Instances 100
Relax Iterations 3
Max Relax Radius 10.0
Houdini Instanced Inputs
populate1_1
SM_Bush
Rotation Offset 0.0 0.0 0.0
Scale Offset 0.25 0.25 0.25
Houdini Asset
Recook Asset
Rebuild Asset
Reset Parameters
Houdini Generated Meshes
populate_2_3_0_1_0
Bake Name populate_2_3_0_1_0

04 Go to the **Houdini Instanced Inputs** section and expand *populate1_1*. This is the box instance which you can replace with content from within UE4.

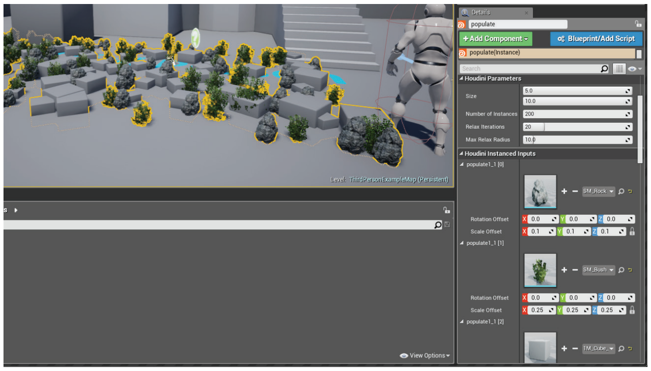
In the Content Browser, open **StarterContent > Props**. Drag the *SM_Bush* prop over to the **Houdini Instanced Input**. Set the **Scale Offset** to **0.25**. The geometry is instanced to the points in the populate asset and rotated and scaled just like the boxes.



Number of Instances 100
Relax Iterations 3
Max Relax Radius 10.0
Houdini Instanced Inputs
populate1_1 [0]
SM_Rock
Rotation Offset 0.0 0.0 0.0
Scale Offset 0.1 0.1 0.1
populate1_1 [1]
SM_Bush

05 Now click the **Plus [+] sign** next to the instance object. This adds a second one. Drag the *SM_Rock* prop over to the new **Houdini Instanced Input**. Set the **Scale Offset** to **0.1**.

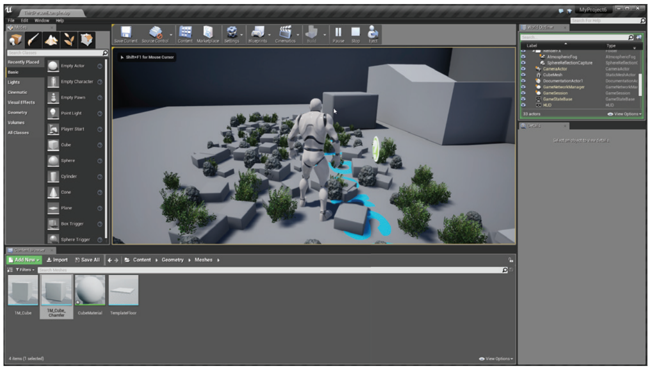
Press **Play** and walk around the scene to see the instances in action. You will notice that you can walk right through these objects. That is because they don't have any collision geometry set up in the instanced geometry.



Houdini Parameters
Size 5.0
10.0
Number of Instances 200
Relax Iterations 20
Max Relax Radius 10.0
Houdini Instanced Inputs
populate1_1 [0]
SM_Rock
Rotation Offset 0.0 0.0 0.0
Scale Offset 0.1 0.1 0.1
populate1_1 [1]
SM_Bush
Rotation Offset 0.0 0.0 0.0
Scale Offset 0.25 0.25 0.25
populate1_1 [2]
1M_Cube

06 Now click the **Plus sign** next to the bush object. This adds a third one. Navigate to the **Content > Geometry > Meshes** folder. Drag the *1m_Cube_Chamfer* prop over to the new **Houdini Instanced Input**. Set the **Scale Offset** to **0.3**.

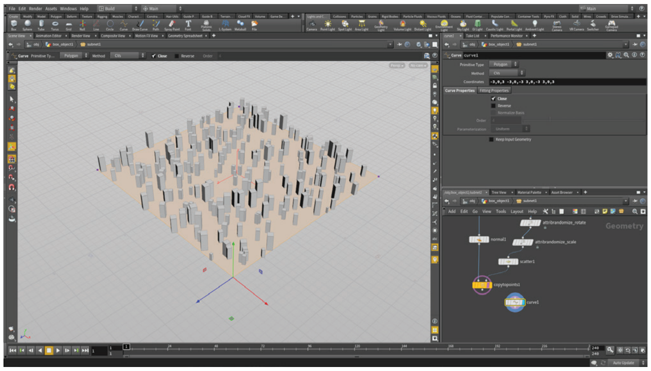
You can continue to add more instanced objects to create more variety. You can use the **Relax Iterations** parameter if you need to separate the instances from each other a bit more.



07 Press **Play** and walk around the scene to see the collision geometry in the new instances in action. Now you can see that we are colliding with the cubes which do have collision geometry set up properly. You should always make sure your instanced objects have collision geometry set up (we will learn how to do that later) for instancing.

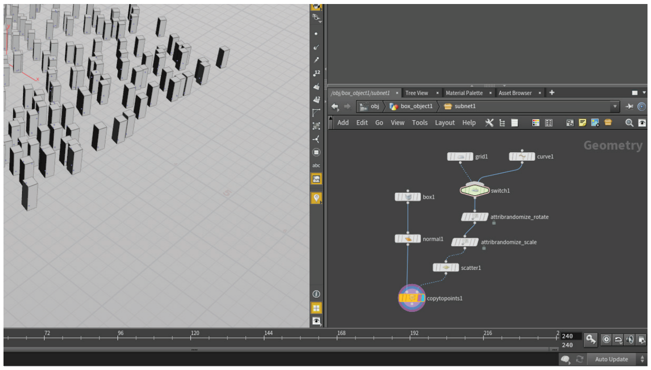
PART FIVE:
Add a Control Curve to the Asset

To add more control over the shape of the populated area, you will add a control curve to the system. At first this curve will only have four points and will have the same shape as the original grid but once it is in UE4, you will be able to add points and create unique shapes.



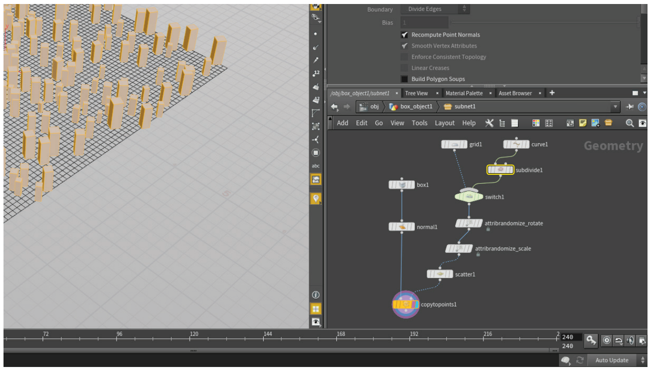
01 **IN HOUDINI** – Make sure you are at the geometry level inside the asset's subnetwork. From the **Tool Options**, choose **Create in Context**. This will make sure the curve you draw goes into the current geometry network instead of creating a new one.

Turn on Grid Snapping in the toolbar on your right then from the **Create** shelf, get the **Curve** tool (not the Draw Curve tool) and click four points to create a curve that matches the grid. Click on the **Close** option in the top bar and press **Enter** to complete the curve.



02 In the Network editor, add a **Switch** node between the *grid* and the *attributerandomize_rotate* nodes. Next, wire the *curve* node into the *switch* node.

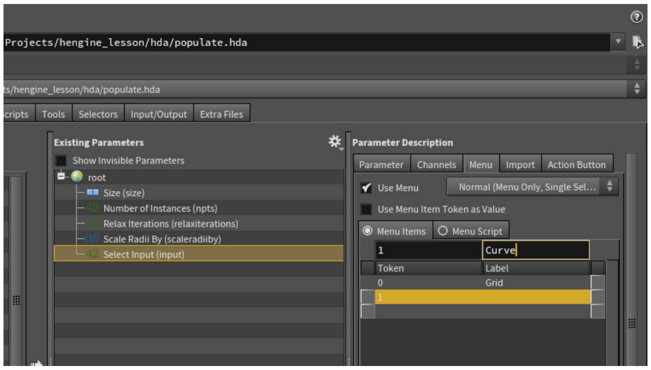
Now if you set the **Select Input** on the switch node to **1**, you can see the curve is being used to distribute the points. Because there is only one polygon created by this, all of the scale and rotate values are no longer being randomized properly.



03 Add a **Subdivide** node after the *curve* and set its **Depth** to **6**. Now there is more detail on the base geometry and the randomization of attributes works properly.

IMPORTANT: Make sure the **Display Flag** is set to the end of the chain otherwise this asset will not display properly at the object level or in UE4. There is a template flag at the end of the chain which makes it look like the whole system is being displayed but that only works at the geometry level.

Set the **Select Input** value to **0** to go back to the *grid* input for now.



Projects/hengine_lesson/hda/populate.hda

ts/hengine_lesson/hda/populate.hda

cripts | Tools | Selectors | Input/Output | Extra Files

Existing Parameters

■ Show Invisible Parameters

- root
  - Size (size)
  - Number of Instances (npts)
  - Relax Iterations (relaxiterations)
  - Scale Radii By (scaleradiiby)
  - Select Input (input)

Parameter Description

Parameter | Channels | Menu | Import | Action Button

☑ Use Menu — Normal (Menu Only, Single Sel...

■ Use Menu Item Token as Value

● Menu Items ○ Menu Script

1 Curve

Token	Label
0	Grid
1	

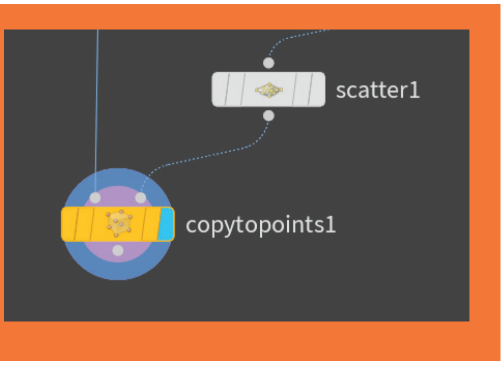
04 Choose **Assets > Edit Asset Properties > Populate**. Click on the **Parameters** tab. In the Network editor, click on the *switch* node. From the Parameter pane, drag the **Select Input** parameter onto root in the Existing parameters list.

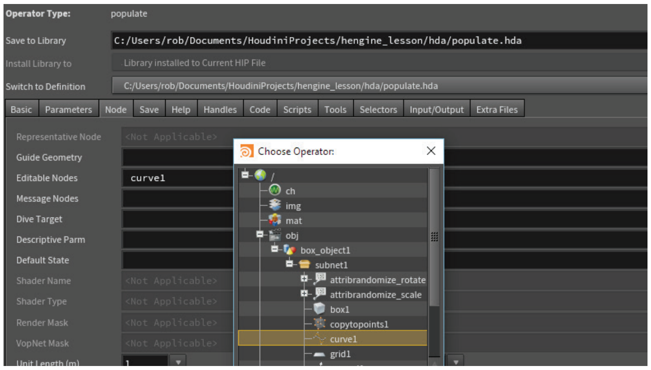
Click on the **Menu** tab then turn on **Use Menu.** Set the first entry as **0, Grid** and the second to **1, Curve**. Now you will be able to make this choice using a menu at the top level of the asset instead of using a number.

Don't click accept yet because we are going to work with another tab in this window to make the curve **editable**.

ALWAYS CHECK THE DISPLAY FLAG

One of the biggest areas of confusion for artists new to Houdini is when the **Display Flag** is not set properly. The network looks good at the geometry level, template flags make everything look good but then you go to the object level or publish your asset to UE4 and something is wrong. Make sure the Display Flag is in the proper place before worrying about anything else. It is often the issue that needs to be addressed.





Operator Type: populate

Save to Library: C:/Users/rob/Documents/HoudiniProjects/hengine_lesson/hda/populate.hda

Install Library to: Library installed to Current HIP File

Switch to Definition: C:/Users/rob/Documents/HoudiniProjects/hengine_lesson/hda/populate.hda

Representative Node: <Not Applicable>

Guide Geometry:

Editable Nodes: curve1

Message Nodes:

Dive Target:

Descriptive Parm:

Default State:

Shader Name: <Not Applicable>

Shader Type: <Not Applicable>

Render Mask: <Not Applicable>

VopNet Mask: <Not Applicable>

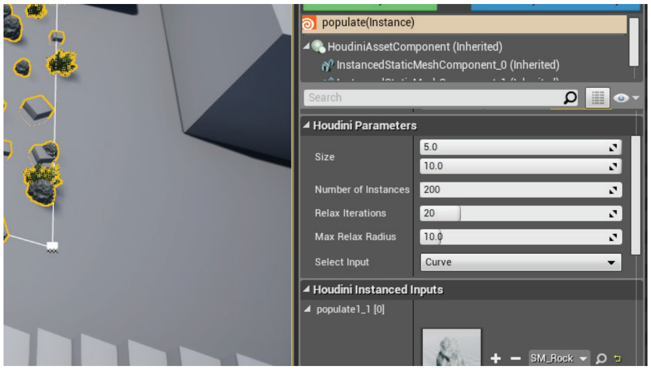
Unit Length (m): 1

Choose Operator:

```
/
  ch
  img
  mat
  obj
    box_object1
      subnet1
        attribrandomize_rotate
        attribrandomize_scale
        box1
        copytopoints1
        curve1
        grid1
```

05 Click on the **Node** tab. Under **Editable Nodes**, click on the **Choose Operator** button and select the *curve* node. Press **Accept**. Click **Accept** on the Type Properties window to save the changes. This will allow you to edit the curve points in UE4.

In Houdini it works a little differently. You will not see these points at the object level but you can use the Network editor to dive down to the *curve* node and once it is selected then you will be able to edit the curve points. In UE4, these curve points will be promoted to the top level of the asset.



populate(Instance)

HoudiniAssetComponent (Inherited)

InstancedStaticMeshComponent_0 (Inherited)

Houdini Parameters

Size: 5.0, 10.0

Number of Instances: 200

Relax Iterations: 20

Max Relax Radius: 10.0

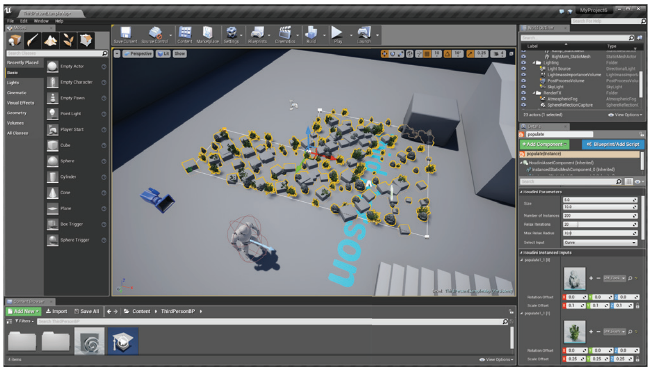
Select Input: Curve

Houdini Instanced Inputs

populate1_1 [0]: SM_Rock

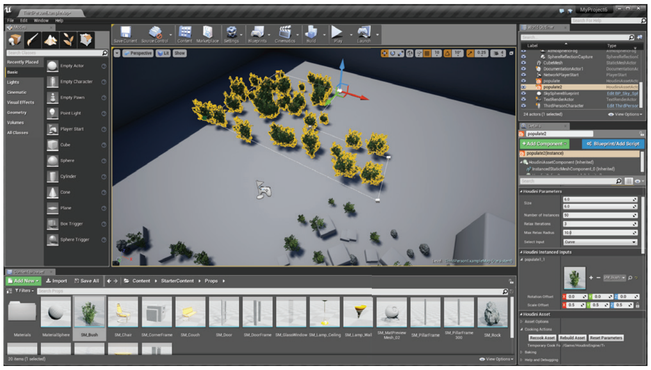
06 **IN UE4** –Click **Rebuild Asset** to accept the changes. You will see four grid points. Click on one of the points, press **W** to get the translate handle and move the point. The instances don't update yet.

From the details panel, use the **Select Input** menu to select **Curve**. Now you can move the points and the instanced area will update. At any point you can choose to use the input curve or the grid to populate your level.



07 Press the **Alt** key as you move a point. Now you are adding a point to the curve and you can shape the area as you see fit. Add a couple more points to shape your garden.

Be sure to keep the points all at the same level. It is easy to drag them off the ground plane and that will create non-planar geometry for the instances. While the asset will still work, it will look better if the points are all coplanar.

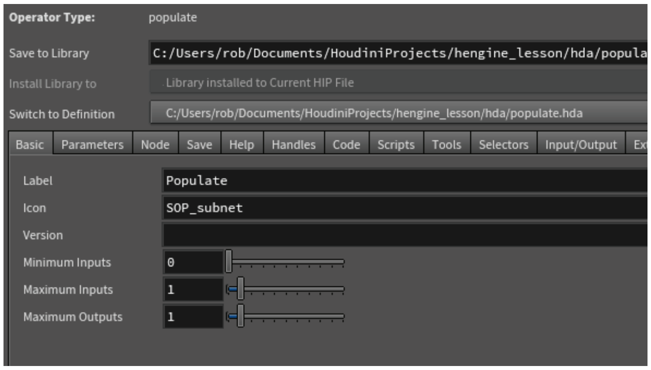


08 In the **Content Browser**, navigate back up to *Content* then down into *Third person BP*. Drag a second version of the *populate* asset into the scene. Use the controls at your disposal to set up an area on top of the platform.

Use the **Curve** input and shape it to get a different shape. Use the **Alt** key to add extra points. Set a small number of instances and add geometry to it. Here we used the bush geometry.

PART SIX:
Use Geometry from UE4 to Drive Asset

So far this asset supplies the input geometry for the populate asset. Another option is to use existing geometry in the UE4 scene to instance the geometry. You will now add an input node to your asset which will accept this UE4 geometry. We will add some nodes in Houdini which won't do anything until later when the asset is in UE4.



Operator Type: populate

Save to Library: C:/Users/rob/Documents/HoudiniProjects/hengine_lesson/hda/popula

Install Library to: Library installed to Current HIP File

Switch to Definition: C:/Users/rob/Documents/HoudiniProjects/hengine_lesson/hda/populate.hda

Label: Populate

Icon: SOP_subnet

Version:

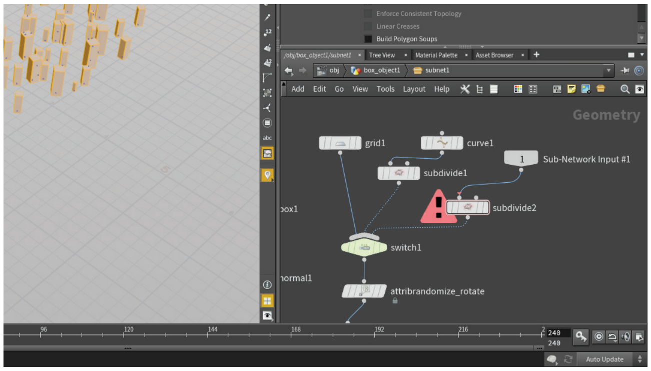
Minimum Inputs: 0

Maximum Inputs: 1

Maximum Outputs: 1

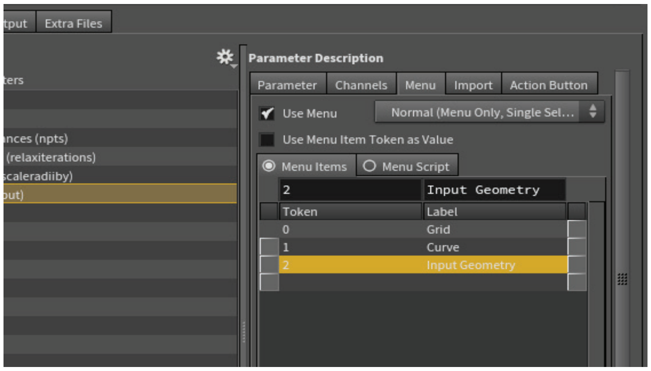
01 **IN HOUDINI** – Choose **Assets > Edit Asset Properties > Populate**. Stay on the Basic tab and set **Maximum Inputs** to **1**. Click **Accept** on the Type Properties window to save the changes. This will create an input node inside the asset that you can wire into the network. This input will let you grab geometry from the UE4 scene later.

Make sure you leave **Minimum Inputs** set to **0**. If it is set higher than that then the asset will ONLY work if the minimum input requirement is met. Otherwise the asset won't cook and you won't see anything.



02 In the Network editor, wire the new Input into Switch node. Use the **Alt** key to copy the **Subdivide** node then wire it after the input node to make sure there is enough detail for the randomizing of attribute values.

NOTE: You are getting an **Error** on the subdivide node because in Houdini there is nothing feeding into the third input. You could go back up one level and feed a box into the asset's input but that isn't necessary for us to get this working in UE4.



Parameter Description

☑ Use Menu — Normal (Menu Only, Single Sel...

□ Use Menu Item Token as Value

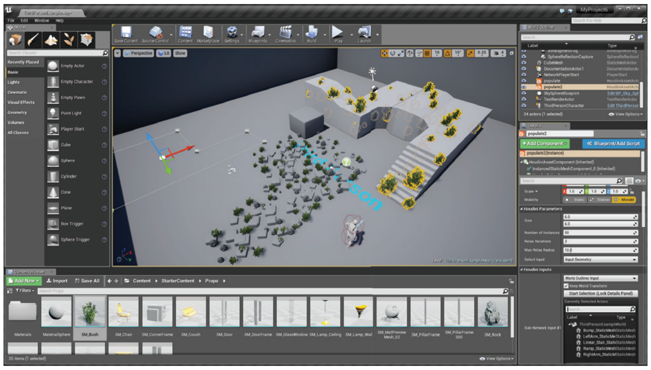
● Menu Items ○ Menu Script

2 | Input Geometry

Token	Label
0	Grid
1	Curve
2	Input Geometry

03 Choose **Assets > Edit Asset Properties > Populate**. Select the **Select Input** parameter. Click on the Menu tab then add a third entry as **2, Input Geometry**. Click **Accept** on the **Type Properties** window to save the changes.

This will add a third menu item to the UI of our asset which you can use in UE4.

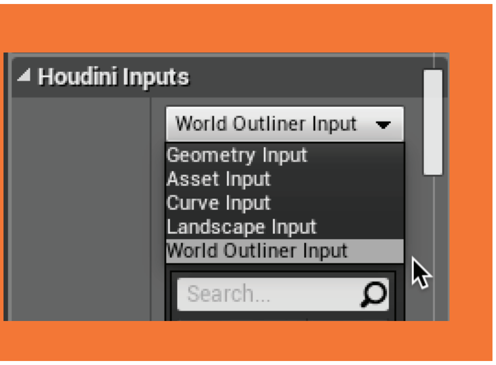


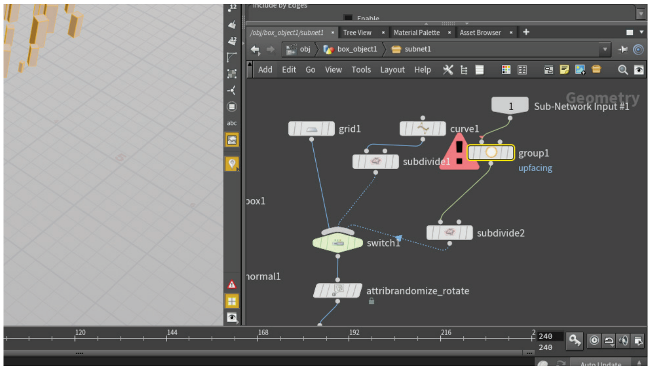
04 **IN UE4** – Click the **Rebuild Asset** button to accept the changes. Use the second *populate* asset you placed off in the corner. Go to the **Houdini parameters** section and from the **Select Input** menu choose **Input Geometry**. Set **Number of Instances** to **50**.

In the **Details** panel, go to the **Houdini Inputs** section and from the menu, choose **World Outliner Input**. Click on the **Start Selection** button and in the 3D scene, select all the parts of the staircase and platform. Click **Use Current Selection** and now the instances are being scattered onto the platform.

USING THE INPUT NODE IN UE4

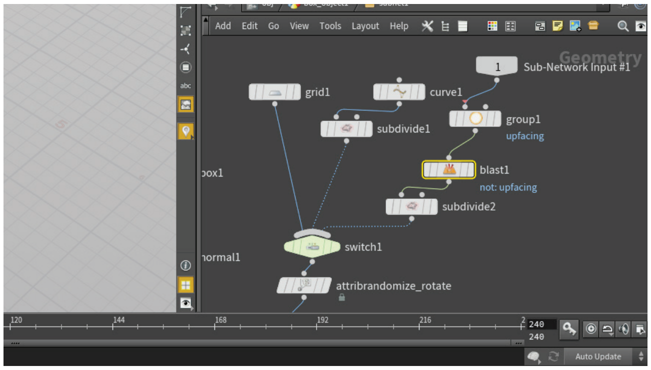
When you set up an input node in your asset, there are a number of options for accessing input geometry. You can use Geometry from the content browser. You can also set up curve input where you draw a curve in UE4. This is an alternative to drawing the curve in Houdini and promoting it to the asset. You can also grab content from the World outliner or input UE4 landscapes at heightfields for use with Houdini's new terrain toolset.





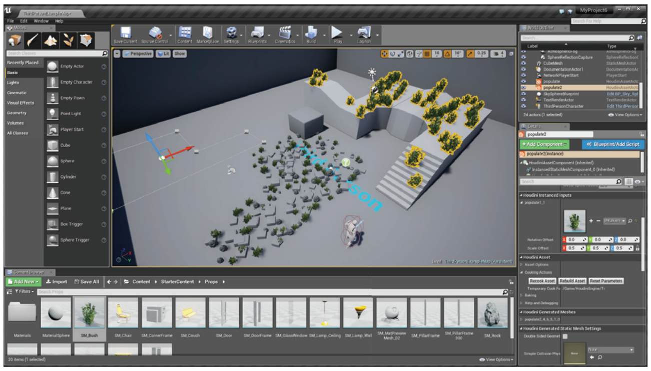
05 **IN HOUDINI** –We are now going to isolate the top faces of the box to limit where the points are being copied. Insert a **Group** node after the *input* node in the network editor. Set the **Group Name** to **upfacing**. Under **Base Group**, turn **Enable** to **Off**. Under **Keep by Normals**, turn **Enable** to **On**. Set the **Direction** to **0, 1, 0** and **Spread Angle** to around **5**.

Instead of scattering points to all the faces of the geometry in UE4, you are going to isolate the top faces and use them instead. Now that you have a group identifying these faces.



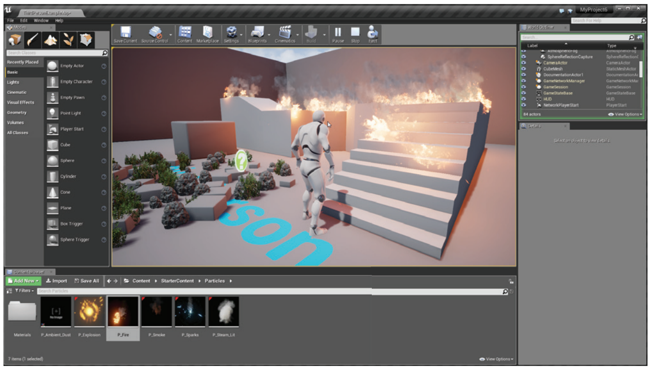
06 Insert a **Blast** node after the *group* node and set the **Group** to *upfacing* and turn **Delete Non Selected** to **On**. Now only those primitives facing up will be kept - the others will be deleted. Choose **Assets > Save Asset > Populate**.

The key here is that because you used to a group to identify the top faces, it doesn't matter what geometry is input into the asset the solution will work. This is how a procedural asset works because it provides a generalized solution instead of a specific one-off solution.



07 **IN UE4** – Click the **Rebuild Asset** button to accept the changes. Now only the top faces of the geometry have boxes on them.

This asset can now use a default grid, a surface created by a curve or geometry sourced from the level to work. There are always lots of options available when building Houdini Digital Assets for use in UE4.



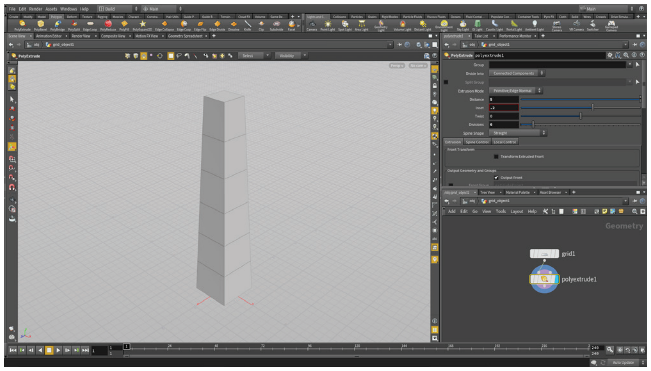
08 Set the **Number of Instances** to **25**. Go to the **Houdini Instanced Inputs** section and expand *populate1_1*.

In the Content Browser, open **StarterContent > Particles**. Drag the *P_Fire* prop over to the **Houdini Instanced Input**. Set the **Scale Offset** to **0.5**.

The instanced points in this asset can be used for more than just geometry. They are now a part of this UE4 level and can be used to solve different problems.

PART SEVEN:
Create an Asset with Collision Geometry

You are now going to create a new asset for the game level and set it up for collisions. This is accomplished in Houdini by setting up a group with the proper name that will be recognized by UE4 as collision geo.

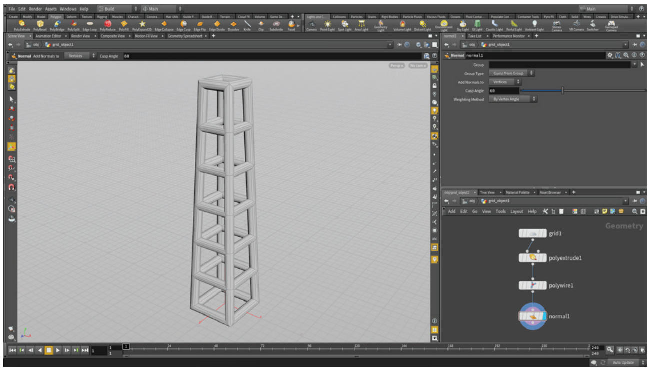


01

IN HOUDINI – Go to the Object level. Get the **Grid** tool and place a grid at the origin. Set its **Size** to **1, 1** and **Rows** and **Columns** to **2** each.

Select the *grid* and from the **Polygon** shelf choose **Polyextrude**.

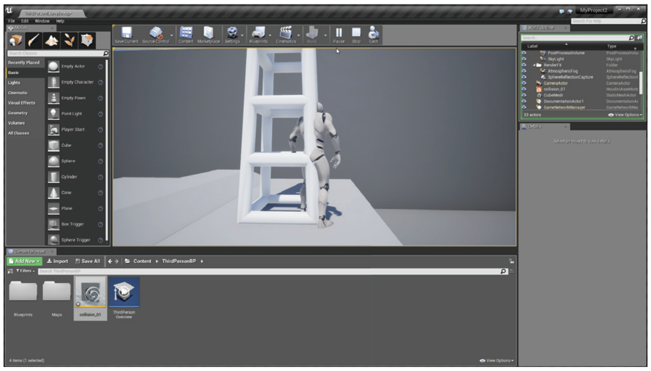
Drag the polygon a **Distance** of **5** units. Set the **Inset** to **0.2** and the **Divisions** to **6**. This creates a tower object that you will use to test out the collision geometry.



02

In the Network editor, add a **Polywire** node after the *polyextrude*. Set **Divisions** to **8**. This creates a simple truss. Add a **Normal** node at the end of the chain to add vertex normals to the geometry.

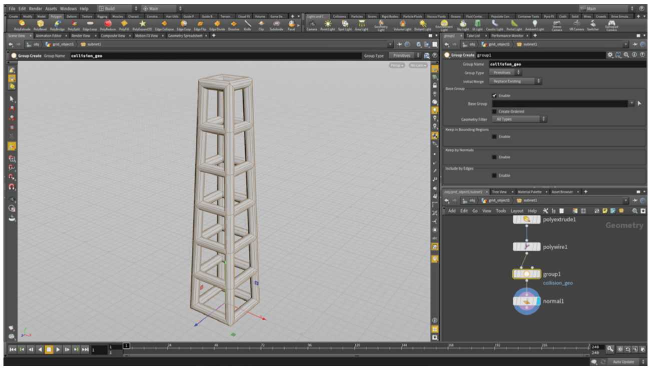
Select all the nodes in the network and turn it into a Digital Asset. Call it *collision_test* and save it into the **HDA** directory of your project.



03

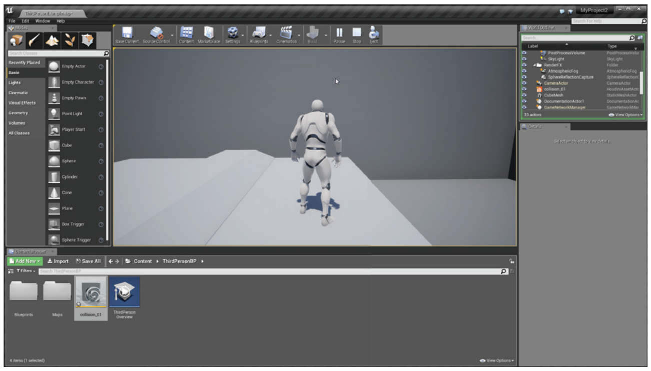
IN UE4 - Click **Import** and find the *collision_test.hda* asset file on disk. Drag the asset from the **Content Browser** to the 3D workspace.

Press **Play** and walk into the asset. You can see that there is no collision geometry in place and you can walk right through the truss.

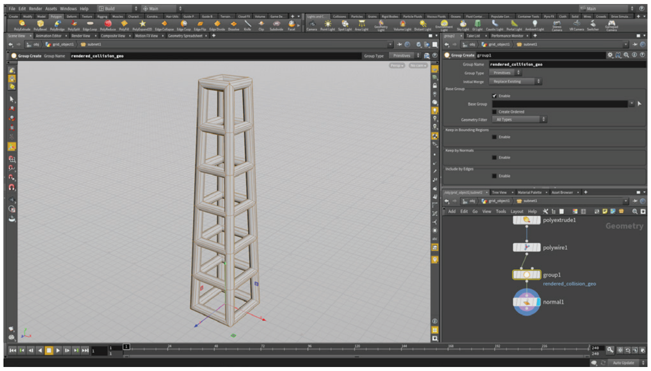


04

IN HOUDINI – Add a **Group** node just before the *normal* node. Set the Group Name to *collision_geo*. Choose **Assets > Save Asset > Collision Test**.

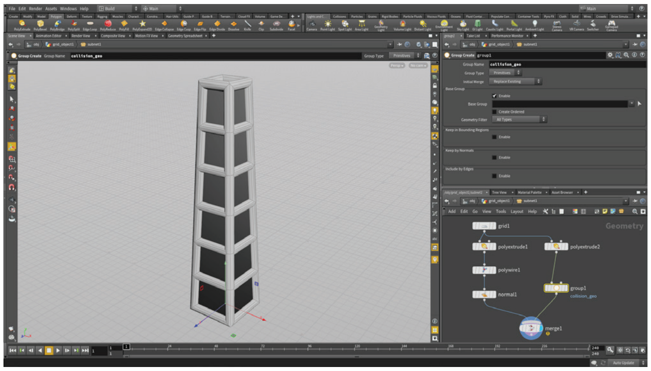


05 **IN UE4** – Click the **Rebuild** Asset button to accept the changes. The trusswork will disappear from the scene but if you press **Play** and walk to where it is you will see that there is collision geometry blocking your way.

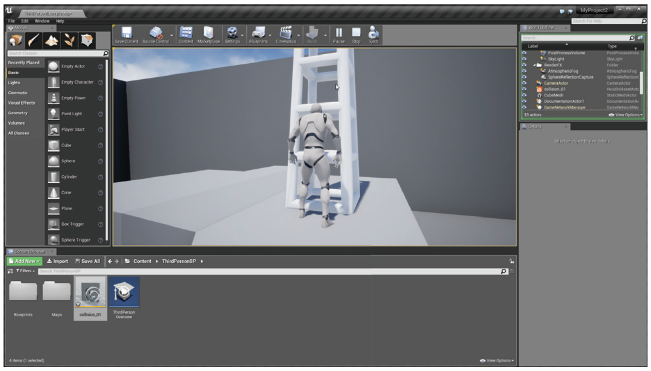


06 **IN HOUDINI** – On the group node, change the Group Name to rendered_collision_geo. Choose **Assets > Save Asset > Collision Test.**

IN UE4 –Click the **Rebuild Asset** button to accept the changes. The trusswork is now visible and works as collision geometry. If you press **Play** and walk to the truss then you will collide with it. The one issue is that you are currently colliding with all of the geometry. It would be better to use different geometry for the collision geo.



07 **IN HOUDINI** – Copy the *polyextrude* node. Set **Divisions** to **1**. Wire it into the *group* node. Change the **Group Name** back to *collision_geo*. Add a **Merge** node just before the *normal* node. Make sure the display flag is set to the end of the chain. Choose **Assets > Save Asset > Collision Test**.

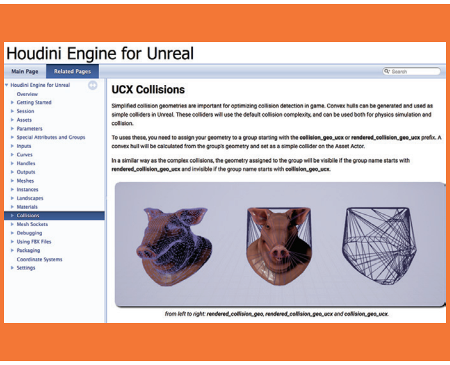


08 IN UE4 –Click the **Rebuild Asset** button to accept the changes. The trusswork is now visible and the lower res geometry is being used as collision geometry.

Press **Play** to test this and make sure that you are colliding properly.

COLLISION GEOMETRY OPTIONS

Open the SideFX website and go to **Support > Documentation > Unreal Plugin Documentation**. Open it up and go to the **Collisions** chapter. Here you can find all of the collision geometry group names and what they create for you.



Houdini Engine for Unreal

Main Page | Related Pages

- Houdini Engine for Unreal
  - Overview
  - Getting Started
  - Session
  - Assets
  - Parameters
  - Special Attributes and Groups
  - Inputs
  - Curves
  - Handles
  - Outputs
  - Meshes
  - Instances
  - Landscapes
  - Materials
  - Collisions
  - Mesh Sockets
  - Debugging
  - Using FBX Files
  - Packaging
  - Coordinate Systems
  - Settings

UCX Collisions

Simplified collision geometries are important for optimizing collision detection in game. Convex hulls can be generated and used as simple colliders in Unreal. These colliders will use the default collision complexity, and can be used both for physics simulation and collision.

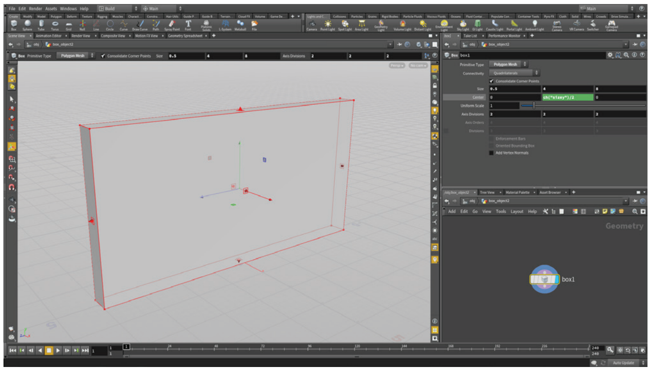
To uses these, you need to assign your geometry to a group starting with the **collision_geo_ucx** or **rendered_collision_geo_ucx** prefix. A convex hull will be calculated from the group's geometry and set as a simple collider on the Asset Actor.

In a similar way as the complex collisions, the geometry assigned to the group will be visibile if the group name starts with **rendered_collision_geo_ucx** and invisible if the group name starts with **collision_geo_ucx**.

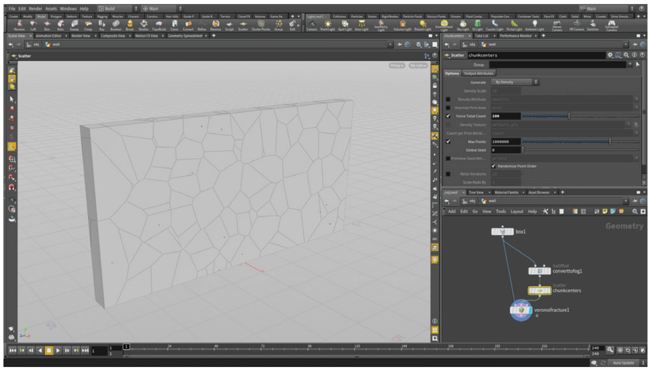
from left to right: ***rendered_collision_geo, rendered_collision_geo_ucx*** *and* ***collision_geo_ucx.***

PART EIGHT:
Import a RBD Simulation into UE4 using FBX

In this part of the lesson, you are going to use a teapot model to define a grid of points then copy cubes to each of the points. You will learn how to work in the Scene view to create objects and the Network view to make adjustments. You will also use expressions to help control the final solution.

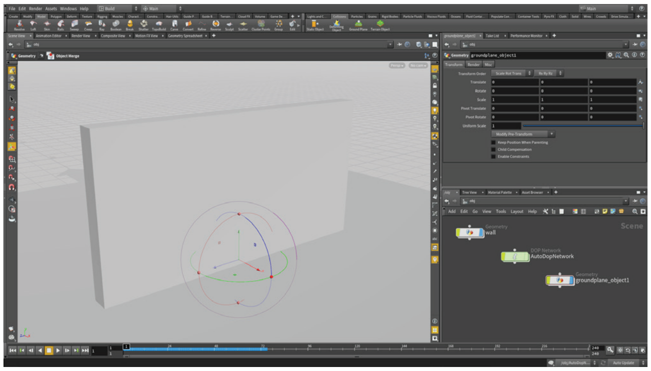


01 **IN HOUDINI** - Create a **Box**. Go to the geometry level and set its **Size** to **0.5, 4, 8** and set the **Center Y** to be half of the height.



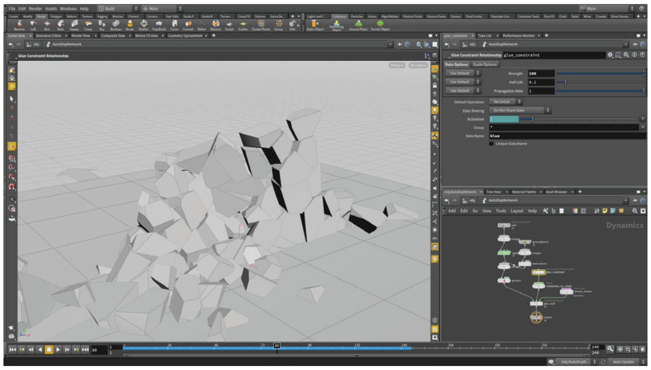
02 Go to the Object level. Rename the node *wall*. Select the wall node and from the **Model** shelf, select **Shatter**.

Dive down to the geometry level, select the *chunkcenters* node and set the **Force Total Count** to **100**.



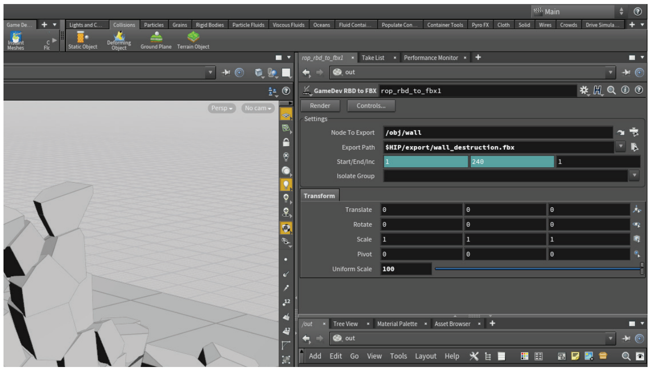
03 Go to the Object level. Select the wall and using the Radial menu (hotkey **c**) select **FX > Rigid Bodies > RBD Glued Objects**. Again, using the Radial menu (hotkey c) select **FX > Collisions > Ground Plane**.

Press **Play** to run the simulation. Nothing happens because the glue setting is too high.



04 Open the *AutoDopNetwork* node and select the *glue_constraint* node. Set **Strength** to **100**.

Press **Play** to run the simulation. The wall falls.



rop_rbd_to_fbx1

GameDev RBD to FBX rop_rbd_to_fbx1

Render | Controls...

Settings

Node To Export: /obj/wall

Export Path: $HIP/export/wall_destruction.fbx

Start/End/Inc: 1 240 1

Isolate Group:

Transform

Translate	0	0	0
Rotate	0	0	0
Scale	1	1	1
Pivot	0	0	0
Uniform Scale	100		

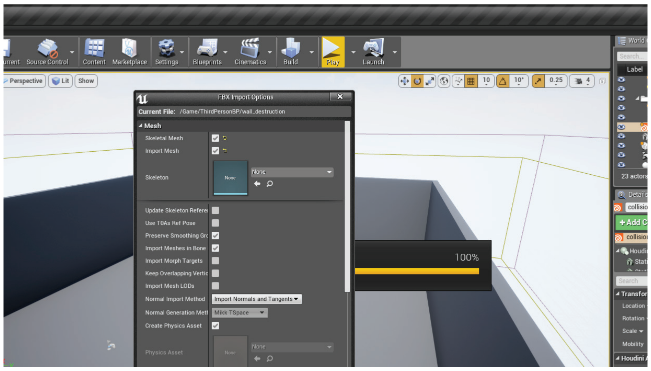
05

Click on the plus sign next to the shelf and find the **Game Development Toolset** shelf to expose it. Click on **Update Toolset** to add them to Houdini. Click on the **RBD to FBX** tool.

In the *rbd_to_fbx* ROP node, set the **Node to Export** to the *wall* node. Set the **export path** to the desktop. Create a new folder and call it *fbxexport* then set the name to *wall_destruction.fbx*. Click **Accept**.

Set **Uniform Scale** to **100** because Houdini and UE4 have different scale factors – meters compared to centimeters.

Click **Render** to save to disk.



FBX Import Options

Current File: /Game/ThirdPersonBP/wall_destruction

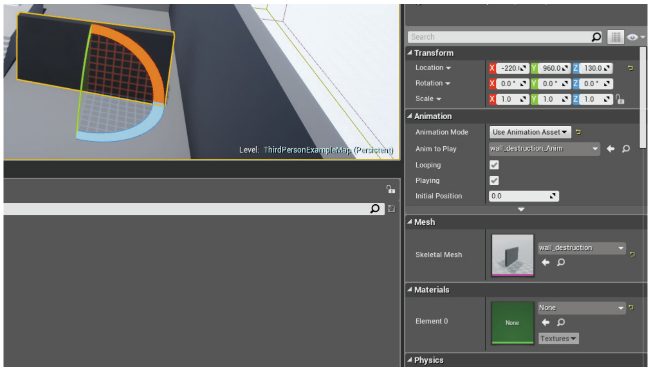
Mesh

- Skeletal Mesh ☑
- Import Mesh ☑
- Skeleton: None
- Update Skeleton Referer ☐
- Use T0As Ref Pose ☐
- Preserve Smoothing Grc ☑
- Import Meshes in Bone ☑
- Import Morph Targets ☐
- Keep Overlapping Vertic ☐
- Import Mesh LODs ☐
- Normal Import Method: Import Normals and Tangents
- Normal Generation Meth: Mikk TSpace
- Create Physics Asset ☑
- Physics Asset: None

06

IN UE4 – Press **Import** and grab the *wall_destruction.fbx* file. Check **Skeletal Mesh, Import Mesh, Import Animations** and **Import Meshes in Bone.** Expand the **Mesh** section and set **Normal Import Method** to import **Normals and Tangents**. Click **Import**.

Three items appear in the content list. Drag the *geometry* asset into your workspace.



Transform

	X	Y	Z
Location	-220.(	960.0	130.0
Rotation	0.0 °	0.0 °	0.0 °
Scale	1.0	1.0	1.0

Animation

- Animation Mode: Use Animation Asset
- Anim to Play: wall_destruction_Anim
- Looping ☑
- Playing ☑
- Initial Position: 0.0

Mesh

- Skeletal Mesh: wall_destruction

Materials

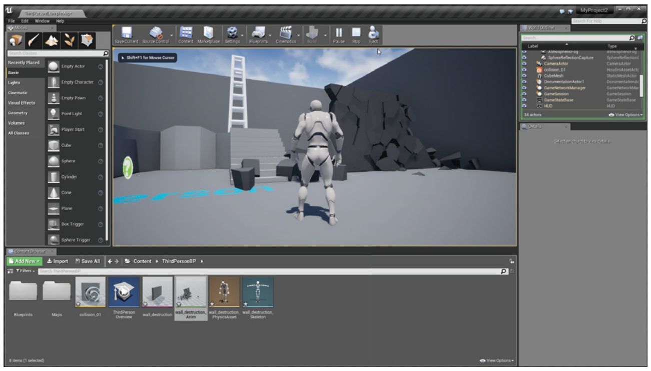
- Element 0: None

Physics

07

In the **Details** pane, set **Animation Mode** to **Use Animation Asset**. This will allow you to assign the animation imported with the fbx file to the wall.

Drag the **Anim** asset from the **Content Browser** to the **Anim to Play** parameter. This contains the animation that was exported from Houdini using the shelf tool.



08

Press **Play > Simulate** to see the simulation working in game. Right now this animation just loops and doesn't interact with anything. You could set up a Blueprint to trigger the animation based on some action on the character's part.

CONCLUSION

You have now created different kinds of game assets for use in UE4. From Houdini Digital Assets to FBX files containing rigid body simulations, you now have a good foundation to build from.

For more lessons on how to use Houdini, go to **SideFX.com** and click on **Learn > Tutorials**. You can then set the Industry filter to Games and there will be lots of material for you to learn from.

HOUDINI FOUNDATIONS

TERRAIN GENERATION

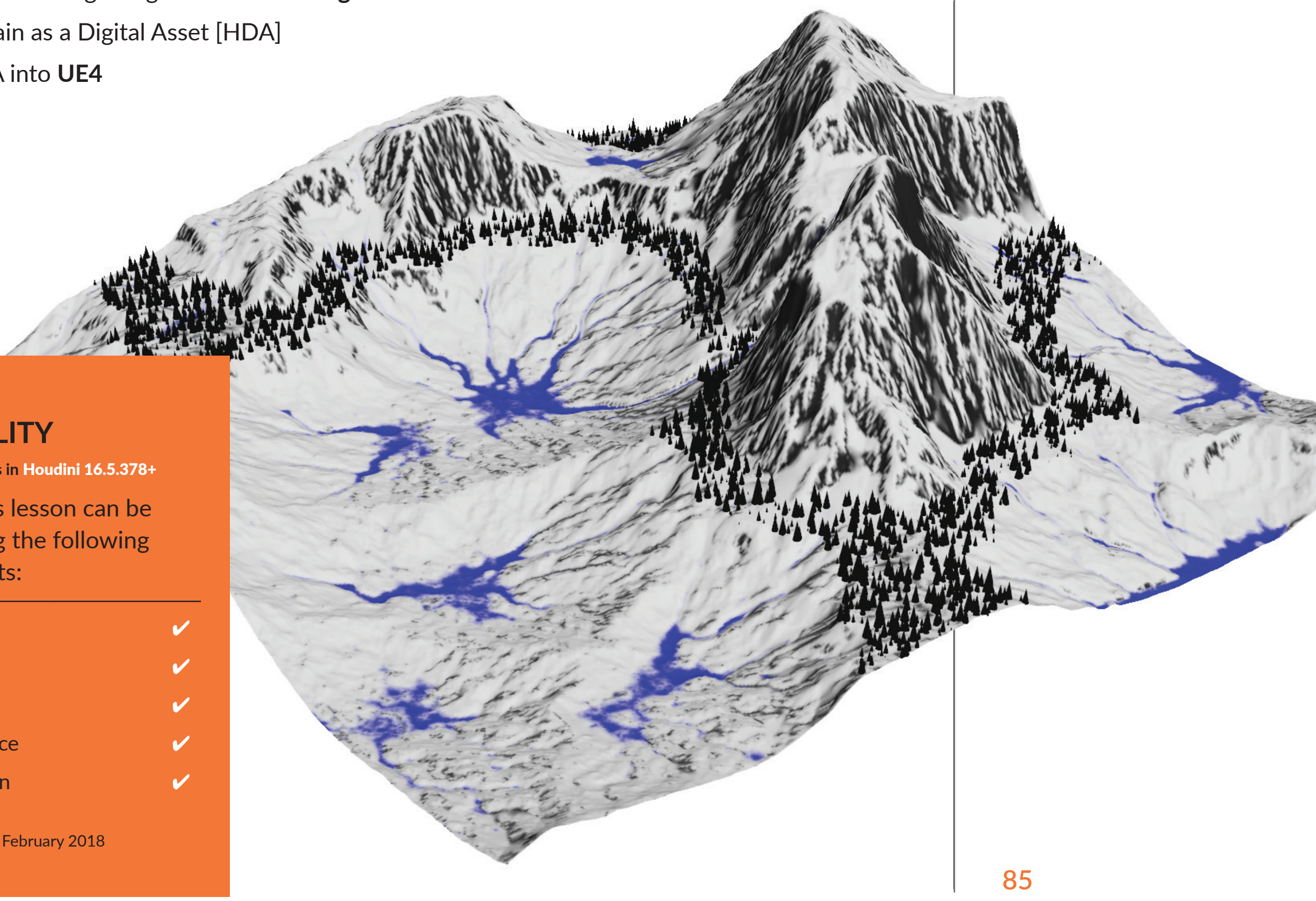
Houdini includes a dedicated toolset for generating and shaping terrains. These tools represent terrain using 2D volumes, called **heightfields,** where each voxel contains the height of the terrain at a particular grid point. The Houdini viewport lets you visualize 2D heightfields as 3D surfaces. You can also set up mask fields that can be used to focus your edits to specific parts of the terrain. In this lesson, you will build up terrains using patterns, noise and erosion then export the results for use in a game engine.

LESSON GOAL

Create a landscape using the Heightfield tools in Houdini and bring it into UE4.

WHAT YOU WILL LEARN

- How to create a Terrain using **heightfields**
- Add **patterns**, **noise** and **distortions**
- Create **Masks** using terrain features
- How to create **Scatter Points** on heightfield
- How to set up Instancing using **Terrain Scattering**
- **Export** the Terrain as a Digital Asset [HDA]
- Import the HDA into **UE4**



LESSON COMPATIBILITY

Written for the features in Houdini 16.5.378+

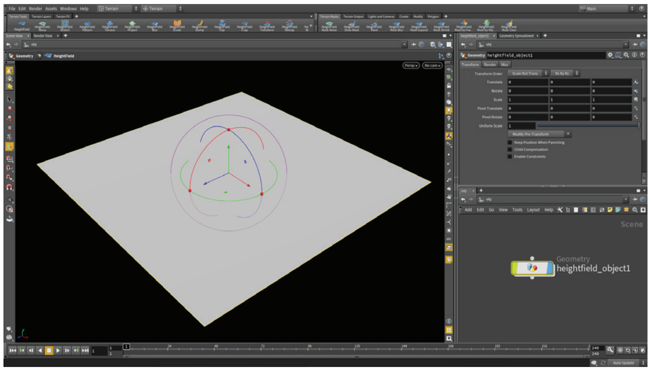
The steps in this lesson can be completed using the following Houdini Products:

Houdini Core	✔
Houdini FX	✔
Houdini Indie	✔
Houdini Apprentice	✔
Houdini Education	✔

Document Version 1.0 | February 2018

PART ONE:
Shape the Terrain using Heightfields

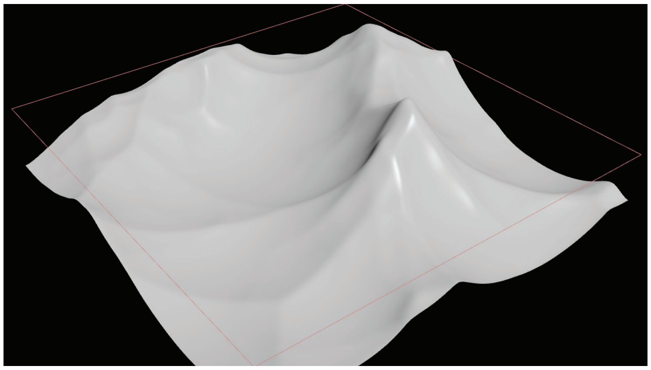
To create terrains in Houdini, you will work with heightfields. You will start with a blank heightfield then add some noise and distortion to define the basic look of the landscape. As you work, you can tweak parameter values on the nodes while layering in details.



01 **IN HOUDINI** - Create a new Houdini scene file. Go to the **Desktop** selector and choose **Terrain** [**Desktop > Terrain** on OSX]. This will give you shelf tools and radial menus that focus on the creation of Terrain.

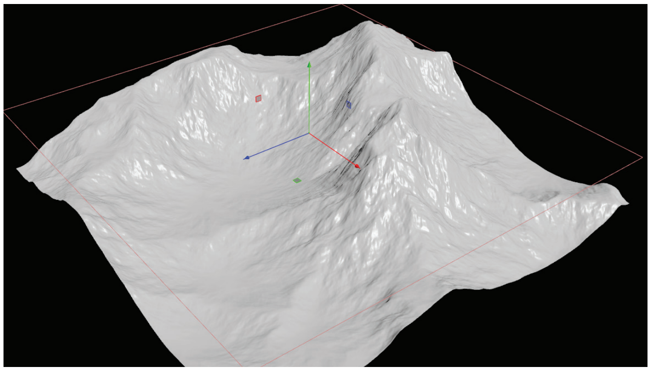
From the **Terrain Tools** shelf, click on the **HeightField** tool. Press **Enter** to place it at the origin. Press **Spacebar H** to show the whole heightfield.

Right now this looks like a simple grid but it is actually a heightfield with volume properties that you will tap into as you layer in effects.



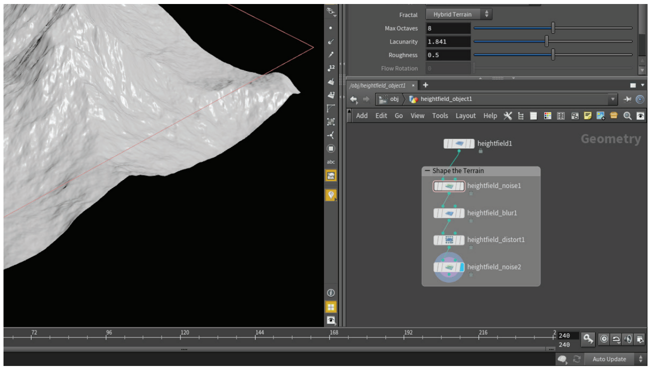
02 Use the main radial menu (hotkey **c**) to select **Deform > Noise**. Set the **Noise Type** to Worley F1. Raise the **Amplitude** to around **360** and the **Offset** to **20, 0, 300**. This kind of noise gives you a good starting point for your terrain.

Use the main radial menu to choose **Deform > Blur** to access the **Heightfield Blur** tool. Set the **Radius** to **20**. This softens the edges to make them feel worn by time.



03 Choose the **Heightfield Distort** tool. Set the **Amplitude** to **40** and the **Element Size** to **220**. This node moves the existing values around by advecting them through a noise field.

You can use the parameter values proposed here or explore on your own to get a look that you like better. Houdini's procedural approach would even allow you to come back later and change the parameter values to see how different settings affect the outcome.



Fractal: Hybrid Terrain
Max Octaves: 8
Lacunarity: 1.841
Roughness: 0.5
Flow Rotation

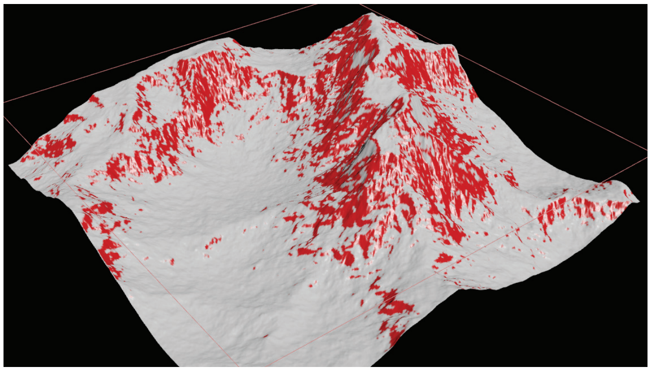
04 Add another **Heightfield Noise** node. Set the **Amplitude** to **10** and the **Element Size** to **20**.

Select the last four nodes in the Network view then click on the Network box icon to add a Network box. Adjust its edges to shape the box then click in the top bar and name it *Shape the Terrain*.

Network boxes make it easier to read the network especially if you share your file with other artists.

PART TWO: Masking by Feature

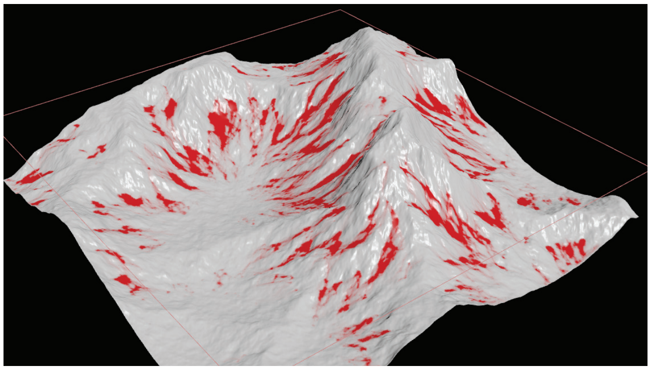
You can use Masks to help you shape your terrain and to later visualize the terrain and to set up texture maps. You can draw or paint masks onto the terrain or derive the mask from the features of the terrain such as slope, height or by finding the peaks and valleys.



01

Use the c radial menu to choose **Mask > Mask by Feature**. Set the **Min Slope Angle** to around **35** and the **Max Slope Angle** to around **60.** This lets you focus on areas on the side of the mountain.

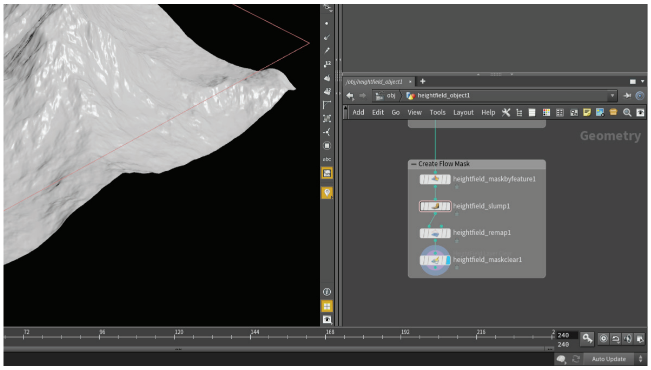
If you were to go back and change the shape of the terrain feeding into this node then the mask would update accordingly.



02

Choose the **Heightfield Slump** tool. Set **Spread Iterations** to **75**. The Slump node creates a type of erosion that moves unstable piles of rubble into a more stable configuration. It affects the Mask layer and also outputs to the **Flow** and **Flow Direction** layers.

If you **MMB-click** on the node, you can see which layers are being created. You can **MMB-click** on an earlier node in the chain for comparison.

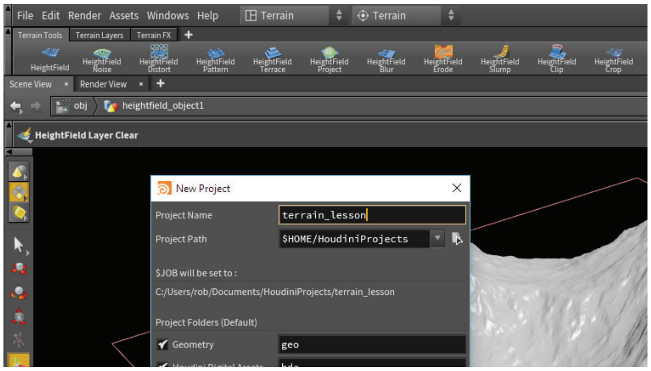


03

Choose the **Heightfield Remap** tool. Set **Layer to Remap** to **Flow** then click on the **Compute Range** button. Set **Output Max** to **1** to normalize these values.

Use the main radial menu to select **Mask > Clear Mask**. This clears the mask layer so that you can use it to create more layers in your setup.

Select the last four nodes in the Network view then click on the Network box icon to add a Network box. Adjust its edges to shape the box then click in the top bar and name it *Create Flow Mask.*



New Project

Project Name: terrain_lesson

Project Path: $HOME/HoudiniProjects

$JOB will be set to :
C:/Users/rob/Documents/HoudiniProjects/terrain_lesson

Project Folders (Default)

☑ Geometry: geo

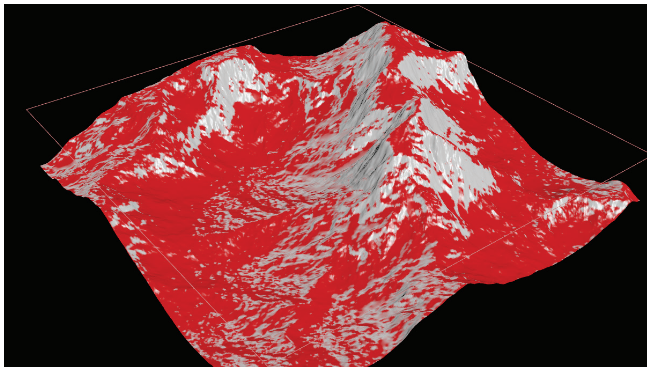
04

Select **File > New Project.** Set the **Project Name** to *terrain_lesson* and press **Accept**. This makes a project directory with sub folders for all the files associated with this shot.

Select **File > Save As...** You should be looking into the new *terrain_lesson* directory. If not then click on $JOB in the sidebar and you will be looking at the directory. Set the file name to *terrain_01.hip* and click **Accept** to save.

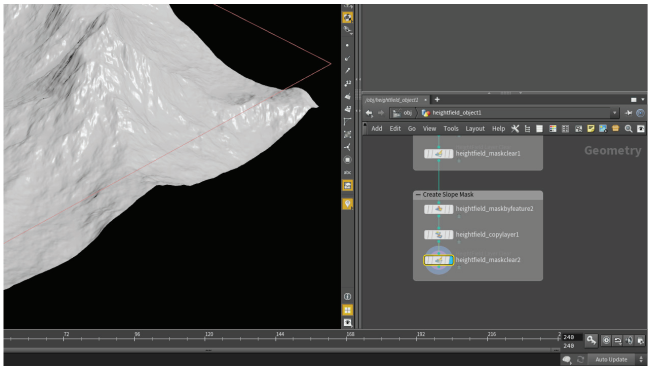
PART THREE: Adding More Layers

You can set up layers on your terrain by first populating the mask then copying that information to a particular layer. You can do this more than once to add more layers. These layers can be used later to visualize key aspects of the terrain.



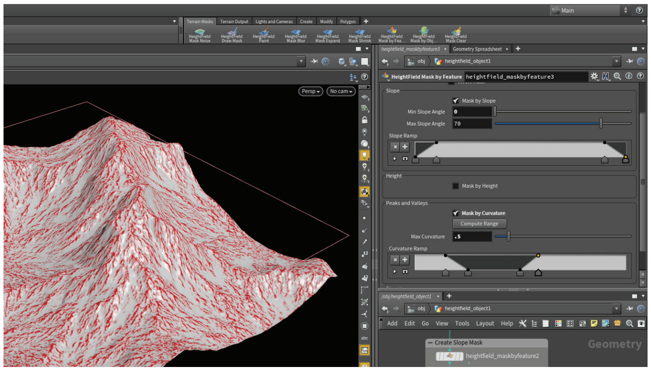
01 Choose **Mask > Mask by Feature**. Turn on **Mask by Slope** and set the **Min Slope Angle** to **0** and the **Max Slope Angle** to around **45.**

Turn on **Mask By Direction** then set **Goal Angle** to around **136** and **Angle Spread** to **180**. These settings cover a larger area of the terrain including the valleys.



02 Choose **Layer > Copy Layer**. Leave the **Source** set to *Mask* and set the **Destination** to *slope*. By copying the mask to a new layer, it leaves you free to clear the mask and use it for other tasks.

Choose **Mask > Clear Mask**. This again clears the mask layer so that you can use it to create more layers in your setup.



HeightField Mask by Feature heightfield_maskbyfeature3

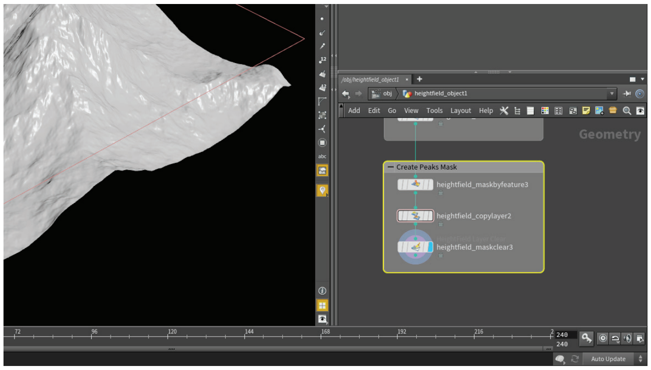
Slope
☑ Mask by Slope
Min Slope Angle 0
Max Slope Angle 70

Height
□ Mask by Height

Peaks and Valleys
☑ Mask by Curvature
Max Curvature .5

03 Use the main radial menu to select **Mask > Mask by Feature**. Turn on **Mask by Slope** and set the **Min Slope Angle** to **0** and the **Max Slope Angle** to around **70.**

Turn on **Mask By Curvature** then set **Max Curvature** to **0.5.** Next move the **Curvature Ramp** points in towards the center to find the peaks of the terrain. This gives you a very detailed masks that you can use to find the peaks of all the key features in the landscape.

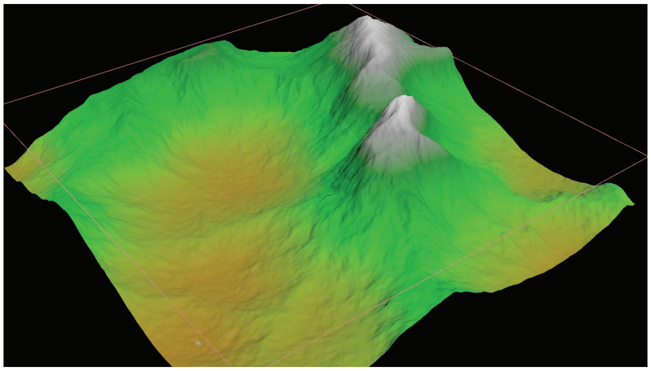


04 Choose **Layer > Copy Layer**. Leave the **Source** set to *Mask* and set the **Destination** to *peaks*. Afterwords, choose **Mask > Clear Mask**. This again clears the mask layer.

You now have three layers that have all been derived from masks. You will use these in the next step to visualize the terrain.

PART FOUR:
Visualizing the Heightfield and its Layers

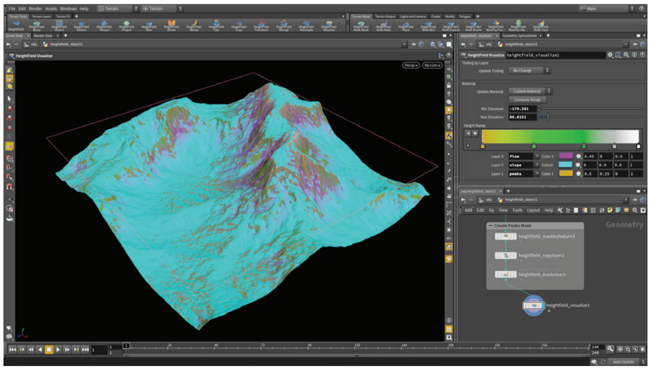
Now that you have a terrain and layers representing flow, slope and peaks, you can visualize them in the viewport using a Heightfield Visualize node. In this case you will use this node to give your terrain a snowy look.



01

Choose **Visualize > Heightfield Visualize**. Click on the **Compute Range** button to align the visualization with the current heightfield range.

This sets the ramp visualizer to go from the base of the terrain to its peaks. You can see how the ramp looks in the 3D view where the mountain tops are highlighted in white.



HeightField Visualize heightfield_visualize1

Tinting by Layer

Update Tinting: No Change

Material

Update Material: Custom Material

Compute Range

Min Elevation: -179.381

Max Elevation: 86.8151

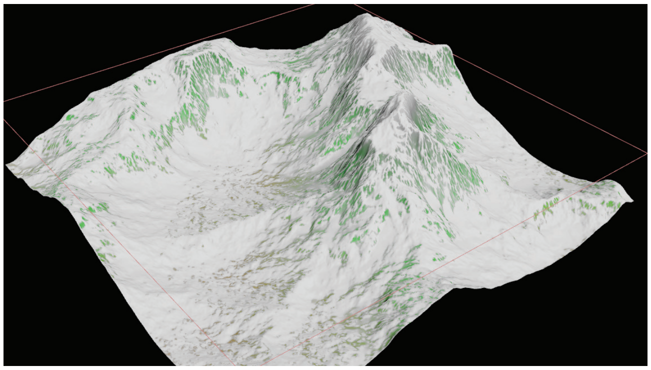
Height Ramp

Layer	Name	Color	R	G	B	A
Layer 3	flow	Color 3	0.45	0	0.9	1
Layer 2	slope	Color2	0	0.9	0.9	1
Layer 1	peaks	Color 1	0.5	0.25	0	1

Create Peaks Mask: heightfield_maskbyfeature3, heightfield_copylayer2, heightfield_maskclear3, heightfield_visualize1

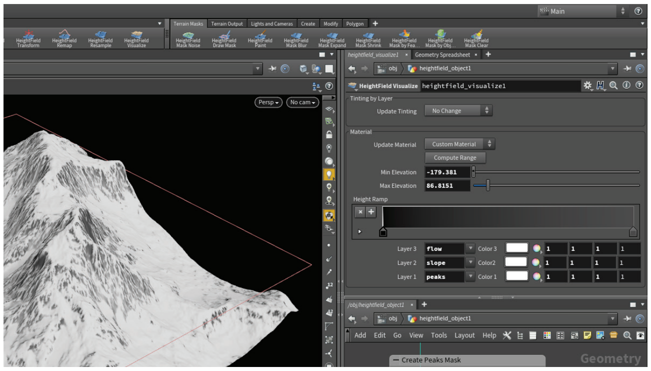
02

Under the ramp widget, set Layer 3 to *flow*, Layer 2 to *Slope* and Layer 1 to *Peaks*. The default colors are now visible in the 3D view using the three layers you built up using masks. You will now use these to define the look of the terrain.



03

Set all three of these layers to **white**. [1, 1, 1] This gives a snowy look to the landscape. You can use these layers for any number of different features, but for this mountain, snow is the look that we want to emphasize.



HeightField Visualize heightfield_visualize1

Tinting by Layer

Update Tinting: No Change

Material

Update Material: Custom Material

Compute Range

Min Elevation: -179.381

Max Elevation: 86.8151

Height Ramp

Layer	Name	Color	R	G	B	A
Layer 3	flow	Color 3	1	1	1	1
Layer 2	slope	Color2	1	1	1	1
Layer 1	peaks	Color 1	1	1	1	1

Create Peaks Mask

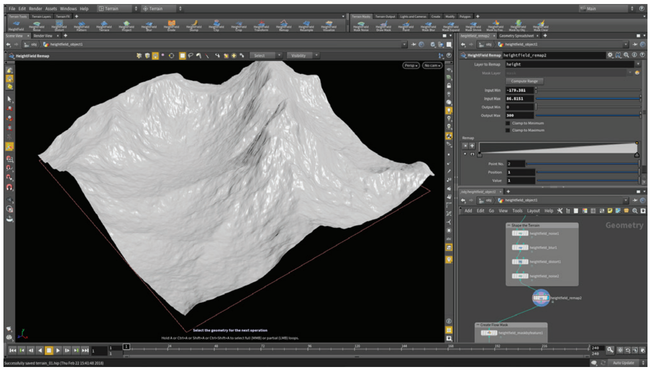
04

In the **Height Ramp**, select and remove all the markers except two. Set the one on the right to black and the one on the left to dark grey. This creates a darker look under the snowy layers which helps the dark areas pop out visually.

Save your work.

PART FIVE: Remap and Erode the Terrain

Right now some of the heightmap is below 0 and some is above. You are going to use a Remap node to change the range and then use the ramp on that node to add a ridge around the mountain. You will then erode the landscape to add new layers to the terrain.



HeightField Remap heightfield_remap2

Layer to Remap: height

Compute Range

Input Min: -179.381

Input Max: 86.8151

Output Min: 0

Output Max: 300

□ Clamp to Minimum

□ Clamp to Maximum

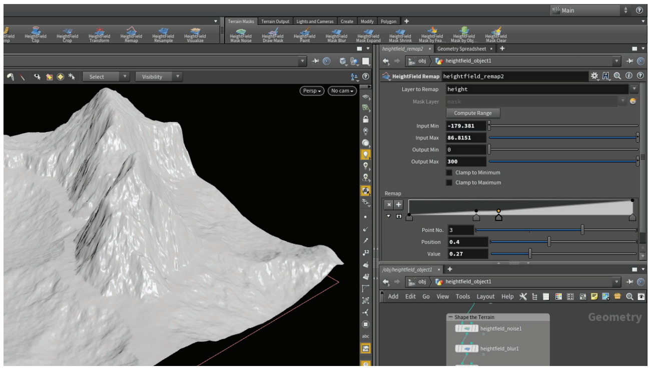
Point No.: 2

Position: 1

Value: 1

01 Find the **Heightfield Noise** node at the end of the Shape by Terrain Network box. Turn on its **Display Flag and click to select his node.**

Choose the **Heightfield Remap** tool. Click on the **Compute Range** button. Set **Output Min** to **0** and **Output Max** to **300** to reframe the heights.



HeightField Remap heightfield_remap2

Layer to Remap: height

Compute Range

Input Min: -179.381

Input Max: 86.8151

Output Min: 0

Output Max: 300

□ Clamp to Minimum

□ Clamp to Maximum

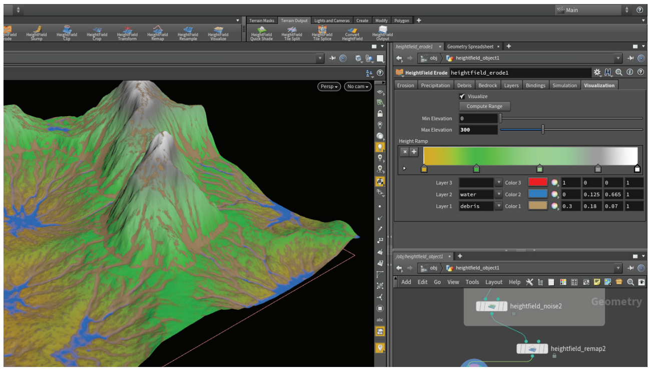
Point No.: 3

Position: 0.4

Value: 0.27

03 Using the **Remap** ramp, add points in the middle then adjust them to so that **Point 2** has a **Position** of **0.3** and a **Value** of **0.25** that **Point 3** has a **Position** of **0.4** and a **Value** of **0.27**.

This creates a ridge running along the edge of the mountain.



HeightField Erode heightfield_erode1

Visualization tab

☑ Visualize

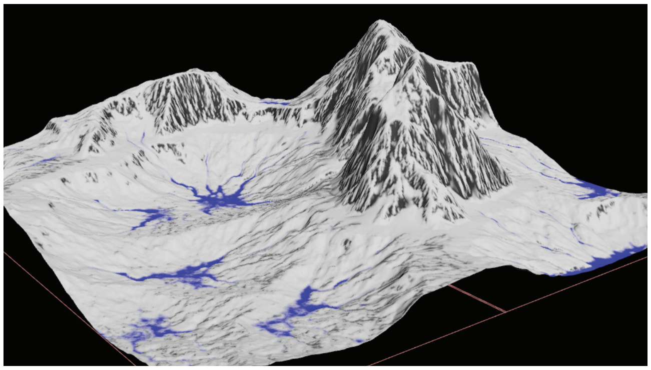
Compute Range

Min Elevation: 0

Max Elevation: 300

Layer 3		Color 3	🟥	1	0	0	1
Layer 2	water	Color 2	🟦	0	0.125	0.665	1
Layer 1	debris	Color 1	🟫	0.3	0.18	0.07	1

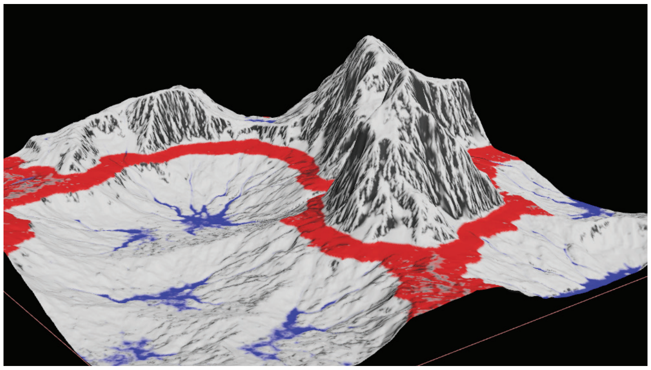
03 Choose **Erode > Erode**. Click on the **Visualization** tab and click the **Compute Range** button. Press **Play** to watch the terrain erode. **Stop** around frame **15**.



04 Set the **Display flag** back to **Heightfield Visualization** node at the end of the chain. Set **Layer 3** to *water* which is a Layer that is coming from the Erode node and change its color to a blue. This will bring out some of these areas in the visualization.

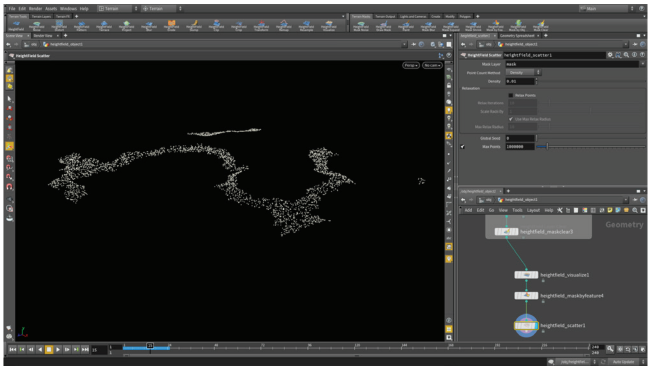
PART SIX: Scatter points on the Terrain

To add trees and rocks, you will mask out the new plateau area then set up a special terrain scatter that will use this mask. These scattered points will then be used to copy instanced cones designed to represent trees. These will be replaced later in UE4.

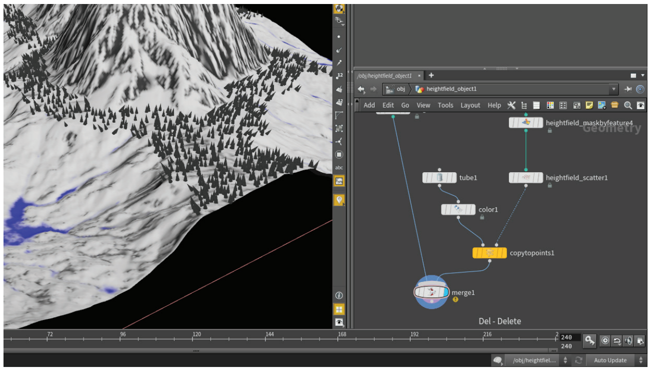


01 Use the **c** radial menu to choose **Mask > Mask by Feature**. Set **Min Slope Angle** to **0** and **Max Slope Angle** to **50**.

Turn on Mask by Height and set Min Height to 70 and Max Height to 85. This should highlight the plateau created with the Remap node. If not then tweak these values until you isolate this area in the Mask.

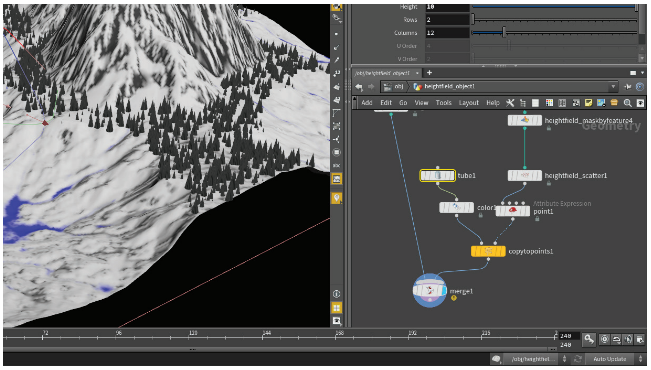


02 In the Network editor **RMB-click** on the output of the **Mask by Feature** node and start typing **scatter...** Place the **Heightfield Scatter** down and set turn on its display flag. The scatter will restrict itself to the area defined by the incoming mask.



03 In the Network Editor, press tab and begin to type **Copy to Points** and click to add this node. Click the **Pack and Instance** checkbox to turn it on. Wire **Heightfield Scatter** into the second input of **Copy** then set its display node.

In the Network Editor, press **Tab > tube** and place a tube node down then feed it into the first input of the Copy node. Set the Orientation to **Z Axis**, **Radius** of the *tube* to **0, 2**, the **Height** to **10** and the **Center** to **0, 0, 5**. You can also add a **color** node after the tube to make the trees **green**. Add a Merge node to the network and feed the *Heightfield Visualize* and the *Copy to Points* nodes into it.

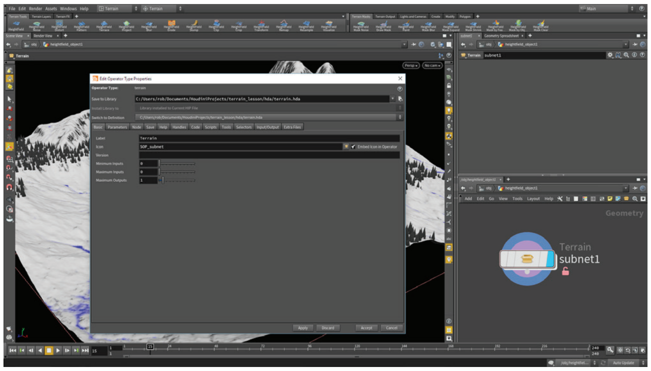


04 Now you can see the “trees” on the landscape all pointing in different directions. In the Network Editor, press tab and begin to type **point...** then choose the **Point** tool. Place the *point*, actually an **Attribute Expression**, node between the *heightfield scatter* and the *copy to points* nodes.

Set the **Attribute** to **Normal (N)**. From the **VEXpression** menu, choose **Constant Value** and set it to **0, 1, 0**. Now the trees are all pointed up. Press the Plus sign to add another attribute and set it to **Scale (pscale)**. From the **VEXpression** menu, choose **Random Scale of Value** and set Constant Value to **2**.

PART SEVEN:
Create a Houdini Digital Asset for UE4

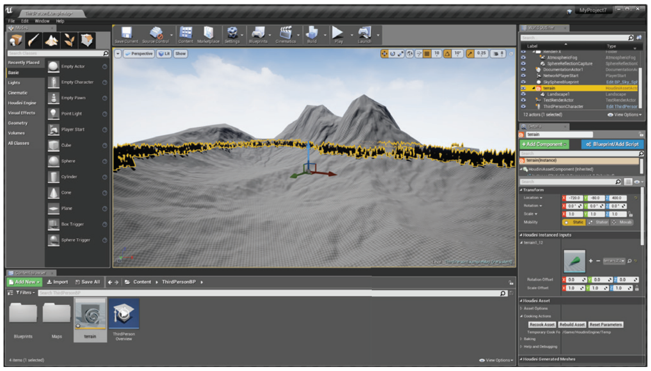
To bring the landscape over to UE4, you will create a Houdini Digital Asset. If you have the Houdini Engine plug-in set up properly for UE4, then this asset can be loaded into the Unreal editor and become a proper UE4 Terrain. Also the copied cones will be instanced objects that you can replace using other content in your level.



01

Select all the nodes in the Network editor. Click on the Create **Subnet from Selected Button**. RMB-click on the new subnet and choose **Create Digital Asset**.

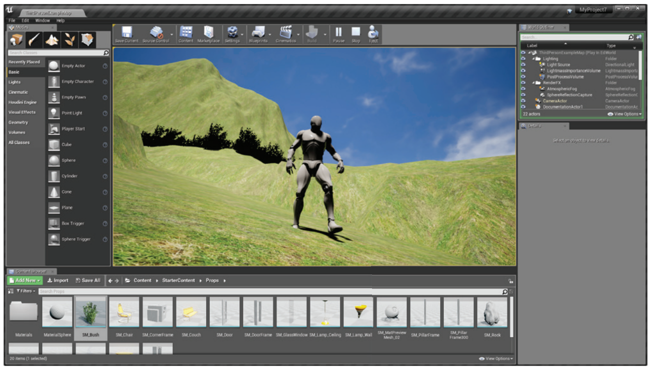
Set the **Operator Name** to *terrain* which will change the **Operator Label** to *Terrain*. Click on the button to the far right of **Save to Library.** In the *Locations* sidebar, click on $JOB/ and then double-click on the *hda* directory. Type *terrain.hda* as the **File** name then press **Accept** and then **Accept** again to save the asset to disk. The **Edit Type Properties** window opens up. Click **Accept** to close this window.



02

Open Unreal Engine 4 and from the main panel click on the **New Project** tab and choose the **Third Person** template. Click **Create Project. When it opens, delete the default geometry so that it doesn't get in the way of your terrain.**

From the Content Browser, click **Import** and find the *terrain.hda* asset file. Drag the asset from the **Content Browser** to the 3D workspace. Move the Press **Play** and walk around the terrain.



03

Go to the **Houdini Instanced Inputs** section and expand *terrain1_1*. This is the cone instance which you can replace with content from within UE4.

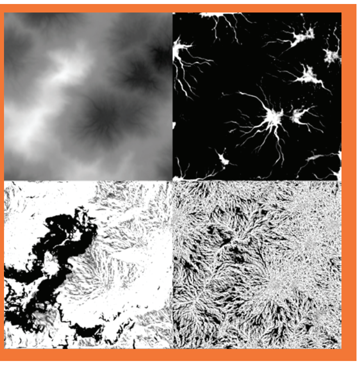
In the Content Browser, open **StarterContent > Props**. Drag the *SM_Bush* prop over to the **Houdini Instanced Input**. Set the **Scale Offset** to an appropriate value. The geometry is instanced to the points randomly scaled just like the cones.

In the outliner, select the **Landscape** node under **terrain.** Beside **Landscape** material, click on the menu and find a grass material. Press **Play** to explore the Terrain.

CONCLUSION

You can also use the terrain layers to create texture maps that you can work with to define the look of your landscape. You can do this using the **Heightfield Output** node. You could then use these channels to build a material in UE4 that references the features of the landscape.

This quick look at the terrain features in Houdini will open up a wealth of possibilities for artists creating landscapes for their games then populating them with details such as rocks and trees.



NOTES

NOTES

SHORTCUTS

VIEWPORT

TOOLS

Select	s
Move	t
Rotate	r
Scale	e
Pose	Ctrl-R
Handle	Enter
View	Esc
Tool Menu	Tab
Custom Radial Menu	c
Repeat Last Tool	q

SELECTION MODES

Objects	1
Points	2
Edges	3
Primitives (Faces)	4
Vertices	5
Select Groups/Connected Geometry	9
Toggle Objects/Geometry	F8

SELECTING

Select	LMB
Add to Selection	Shift + LMB
Remove from Selection	Ctrl + LMB
Toggle Selection	Ctrl + Shift + LMB
Select All	n
Select Nothing	Shift-n

SNAPPING RADIAL MENU

Grid Snap	x ➡
Primitive (Curve) Snap	x ⬆
Point Snap	x ⬅
Multi-Snapping Snap	x ⬇

* OSX | Use Command in place of Ctrl

VIEW

Tumble	Space/Alt + LMB
Pan	Space/Alt + MMB
Dolly	Space/Alt + RMB
Home Grid	Space + h
Home All	Space + a
Home Selected	Space + g

VIEW RADIAL MENU

Selection Tools	v ➡
Selection Options	v ⬆
Viewport	v ⬅
Shading	v ⬇

VIEWPORTS

Expand Viewport	Space + b
Select Viewport	Space + n
Perspective View	Space + 1
Top View	Space + 2
Front View	Space + 3
Right View	Space + 4
UV View	Space + 5
Toggle Wireframe/Shaded	w
Display Options	d

VIEWPORT LAYOUT

Single View	Ctrl + 1
Four Views	Ctrl + 2
Two Views Stacked	Ctrl + 3
Two Views Side by Side	Ctrl + 4
Three Views Split Bottom	Ctrl + 5
Three Views Split Left	Ctrl + 6
Four Views Split Bottom	Ctrl + 7
Four Views Split Left	Ctrl + 8

NETWORK VIEW

VIEW

Pan	Space + LMB or MMB
Zoom	Space + RMB or Scroll Wheel
Show all Nodes	h
Show Selected Nodes	g

CREATE

Node Menu	Tab
Add File Node	=
Create Subnet	Shift + c
Add Background Image	Shift - i

NOTES AND NETWORK BOXES

Add Network Box	Shift + o
Add Sticky	Shift - p
Minimize Selected Notes/Boxes	Shift - j
Expand Selected Notes/Boxes	Shift - k
Shrink box to fit contents	Shift - m

WIRING

Connect Nodes	LMB on Connector
Insert Node	RMB on Connector
Branch	MMB on Connector
Connector List	Alt + MMB on Node
Cut Wire	Y drag across wire
Disconnect from Wires	Shake Node

DOTS

Add Dot	Alt + LMB on wire
Pin/Unpin Dot	Alt + LMB on dot

TOOLS

Toggle Parameter Pane	p
Toggle Tree View	Shift + w
Toggle Network Overview	o
Toggle Color Palette	c
Toggle Shape Palette	s

CLICKS AND DRAGS

Select	LMB
Add to Selection	Shift + LMB
Remove from Selection	Ctrl + LMB
Start Wiring from Node	Alt + LMB
Select Node + Inputs	Alt + Shift + LMB
Select Node + Output	Alt + Ctrl + LMB
Select Inputs + Outputs	Alt + Shift + Ctrl + LMB
Move Node	LMB-Drag
Move Node + Inputs	Shift + LMB-Drag
Move Node + Outputs	Ctrl + LMB-Drag
Copy Selected Nodes	Alt + LMB-Drag
Copy Node + Inputs	Alt + Shift + LMB-Drag
Copy Node + Output	Alt + Ctrl + LMB-Drag
Reference Copy	Alt + Shift + Ctrl + LMB-Drag

NAVIGATION

Enter a Node	Double-click or Enter
Go up a level	u
Create a Quickmark	Ctrl + <# 1-5>
Go to a Quickmark	Shift + <# 1-5>
Go to Previous View	` (Backtick)
Select the Node Upstream	PgUp
Select the Node Downstream	PgDn
Select Previous Sibling	Shift + PgUp
Select Next Sibling	Shift + PgDn

ORGANIZE NODES

Lay out all	L
Align	a + LMB-Drag Down/Across

DISPLAY FLAGS | SOP LEVEL

Render	t + LMB
Display	r + LMB
Template	e + LMB
Footprint	w + LMB
Bypass	q or b + LMB