## Softimage to Houdini transikion guides

## General concepls

## General concepls - Philosophy

## General concepts - General philosophy

Unlike other 3D applications Houdini's approach is more similar to a compiler than anything else because its extensive use of external files (resembling include files), because how presents itself to the user in the form of a filesystem and because its procedural paradigm compares the task of manipulating an object more to a program description than direct maniputation of elements.

How does this affect a softimage user? Well, the trick is to think you are always working with external models, reading data from the outside and manipulating ik. Indeed very similar to Nuke appraoch (although another level completely)

In XSI you have seen bits of this in action in the form of the render tree and certainly ICE, but in Houdini it is a generalised appraoch, everything is procedural although it is very true you can decide to use it in a more trandional way.

A good example would be to build a city, you could simply scatter points and copy buildings to them, add some random scale transformations and many clever extra bits to get a quick complex looking result, but you could also decide to layout your scenes manually and work procedurally in the buildings. Ultimately it is up to you but these upcoming guides only try to depict how I see some of my previous approaches with XSI translated to Houdini.

Not a disclaimer but a simple declaration of scope, I am not a Houdini expert and therefore I am sure my crude attempts to outline these workflows could be optimized and made more elegant, thankfully the Houdini community will surely be examining the ideas and bringing to the table more advanced appraoches.

In the meantime relax and hope we can learn a lot together.

## General concepts - Layout

## General concepts - Layout 1

I dont particularly like the general layout provided, the reasons are simple, the way panes are broken down feels to me like forcing the way I work in a certain direction, very data centric and node centric while I am interested in working in the viewport and using the panes in a more dynamic way... sounds familiar? exackly right.

One of the typical things you will experience is that the various contexts (modeling, animation, shading, rendering, dynamics) will lead you to keep switching from one layout to another and loosing sight of your scene objects, this I am not that used to and found a great layout that $i$ also consider will help you tinker with Houdini's configuraiton files.

Have a look at this website developed by Phil spicer and make sure you install the desktop following these steps, it will give you a bird's eye view of the system.


Standard Houdini "Build" Layout (dark theme)
hetp://houdinicreationdesk.ipage.com

# Standard Houdini Layout breakdown 



Phil spicer's layout with extra variations for a more softimage like approach


## General concepts - Layout 4

> as a list, this is the closes you will get to XSI explorer window


## General concepts - Layouts



Window split expanded, channel editor is the fcurve editor in XSI

## General concepts - Comparing interfaces

## General concepls - Comparing incerfaces

Everybody that uses Softimage is very aware of the 4 modules you have to build your work, here are the comparative names in Houdini;

Modelling $=$ SOPs (surface operators)
Animation = kind of like CHOPs (Channel operators)
Simulation $=$ DOPS (Dynamics operators including particles) Render $=$ ROPs (Rendering operators)

And although the equivalence could be seen very close the reality is that there are some major differences.

In softimage these modules are collections of tools all under one single context, in Houdini these are different contexts and they can talk to each other but essentially are very different.

## Ceneral concepls - Comparing interfaces

The way I organise the interface in Houdini may be completely alien to a true Houdini artist but it is based on the idea that to me I see each context as a construction area, each viewer a viewing area and the parameters as a parameter area I do organise my layout like this.

You could build a new layout per tasks too, something they recommend, but I would suggest not to as you will end up hoping from one to the other and the view and their status are not retained.


## General concepts - Comparing interfaces

So in the construction pane I try to collect all the construction related contexts; modelling, animation, rendering, compositing, etc... these are the networks you operate with.

At the same time I collect all the viewers in the Viewing pane, these are the Scene view, the Compositing view, the Rendering view, etc...

And the parameters pane is where I put all the pane tabs related to parameters, being object parameters like in XSI PPGs, or the data tree, or the bundles, or takes, or the dopesheet and animation area, the motion view, etc...


Pane tab for object modelling and scene organisation

## General concepts - Comparing interfaces

Color coding for the panes also is critical, for this I tend to reinforce and on my desktop I always crank the luminosity and saturation to make sure they are visible as you syncronise one tab pane with another, hence the color scheme plus the various icons that are there.

Like Softimage you can have multiple windows open, multiple, locked too, etc... so you should feel at home with it.


## General concepts - Comparing interfaces

So to make your existence easier here is a general comparison of the various viewers you have in both worlds.

Softimage explorer is excellent, so good you barely use the schematic anymore, in Houdini it is not that good but at least you will be familiar with the scheme.

You have some extra functionality and that is great, these icons are the flags described before so very useful but drilling down is not so obviousl, you have to right click to get access by selecting Tree View.

Keep an eye on the flags I have activated on the left panel!

## General concepls - Comparing incerfaces

Now what about the operator stack? The operator stack is your SOP network!! and allows branching!!! and much much more!!!

It is here you can follow all your deformations and operations and soon enough you will realise it is better to navigate the SOP network the Houdini way, it's just better.


## General concepts - Comparing interfaces

And what you are use to see as a PPG in Softimage in Houdini is your parameters view, which you can access either as a floating window or directly in the network view by pressing ' $p$ '.

You will notice though the geometry parameters are nowhere to be seen, that is because the geometry Houdini does not show the geometry operator that creates the actual geometry in the same window but the philosophy pretty much the same, only thing is that in Houdini you cant 'freeze' the geometry, the design is strict and consistent.


## General concepts - Comparing interfaces

Here you can see the same table orderend in the same way than you would do in Softimage.

As you can see it is pretty much identical approach.

Now the key element to remember is that leg 1 is the container of the geometry description. Basically the geometry lives inside, the leg1 is NOT the geometry itself.


## General concepts - Comparing interfaces

And what you are use to see as a PPG in Softimage in Houdini is your parameters view, which you can access either as a floating window or directly in the network view by pressing ' $p$ '.

You will notice though the geometry parameters are nowhere to be seen, that is becuase the geometry node is not the geometry itself, but the container for the description. You will have to dive in to access the parameters or alternatively promoto those to its parent, the leg1 node.


## General concepts - Comparing interfaces

So hopefully you can see these are more or less the same concepts, the same views and it is just a matter of organising them in a similar fashion to get comfortable fast.

On the next guides I will be talking about organisation of scenes which is pretty particular along with various different strategies to build this simple table so we can see the interesting approaches and what you will gain from doing it the XSI way or the Houdini way.

## General concepts - Interaction

## General concepts - Interaction

Unlike every other application Houdini current selection model is strange, specially if you come from other applications that are viewport centric.

The idea is that every object container (OBJ) where the description of the object is stored (SOP) has to be accessed to deal with the geometry in question. For example, if you have four columns all of them as separate objects, in order to edit the columns geometry you will have to dive in.

This is how you would organise your scene in XSI and therefore manipulating the scene will be more painful due to the manipulation model.


## General concepts - Interaction

If we were about to do the same thing the Houdini way you would create one single object (OBJ) that constructs all four columns.
like this...


## General concepts - Interaction

... and inside, the recipe (SOP), will construct one column and copy it 4 times adding the distance in one single node.

This way is not only procedural and more efficient but allows you to retain a similar-ish approach to editing the geometry which will be another node between the trunk node and the copy1 node.
(remember we are inside the object on the image on the right)


## Ceneral concepts - Interaction

The conclusion is that although strange, somehow the whole approach is geared towards a procedural approach, by architecting your scenes the Houdini way you will suffer less and build a more procedural world.

Said that, I dont like the selection model I just described, but I know they guys at sideEffects are aware of the need to modernise these parts of the software. in the meantime remember that you have to dive in, do your stuff, go back out.

If you encapsulate things inside things using subnetworks (like models inside models) you will have to dive in multiple levels, if you further add vopsop and other dive-ble nodes, again your scene will becomes more obscure so there is something for you to study here.

Like in Softimage we dont tend to put models inside models I try to keep my networks as flat as possible and build the scenes as steps on a recipe rather than a subnetwork does everything. More of this later on.

# General concepls - Nodes flags 

## General concepts - Nodes flags OBJ context

As you have in XSI a great system of colors, divots, links and what not to indicate state of a tool, in Houdini you have a completely different one. Knowing these will allow you to read what is going on in a quick way so I am going to try to depct a few things.

One of the most important things to remember is that in OBJ context the input and output is your hierarchy.

The OBJ context reads like this


## General concepts - Nodes flags SOP context

You will notice that once you are working inside the objects (SOP context) the icon changes its setup and the color coding is also different.

At this stage you should look at those nodes as the operators in the stack of an object but with branching and merging capabilities.

Furthermore, these nodes can be VOP networks (think of it as ICE trees as nodes passing data from one to another)

At this stage nodes should be seen as tools modify your object and that store data.

The SOP context reads like this


## General concepts - Nodes flags ROP context

Regarding the rendering context (ROP) again the flags change a bit and the nodes connections here depict flow order.

So if you connect a chain of render nodes Houdini will execute task after task in the correct order which allows you for massively complex setups to be run in the correct order. Something you cant do in any other package.

A great example I keep using all the time is using a network to force a full recache of my simulations and the a different one for my renders, but I can merge these and make sure I can force a full sim and render refresh.

The SOP context reads like this


## General concepts - Nodes flags COP context

With regards with the compositing system, very much like FXTree in the sense that the other parts of the software can interact with the compositor and although very underused (like FXTree) it provides amazing opportunities.

Like with all the other nodes if you lock the node the data upstream wont be computed and effectively would work like a cached node.

The SOP context reads like this


## General concepts - Node flags

So just keep in mind that each of the various contexts described and others I will be explaining later on imply you to undestand if you are talking about hierarchies, data flow (tasks passing data and altering things on your model/scene) or control flow.

It is a crucial part of working with Houdini to understand that the state of the scene and the rendering state of the scene is only one so being tidy is very important.

A similar thing happens in Softimage where the Default.. Pass is your working environment but when you come to render you use the passes and partitions to force things. A typical case is using the Background partition in combination with other partitions together with overrides to force visibility and more... well, you can do this in ROPs by forcing objects, lights and more too.

That said, a good working habit and ettiquete is even more important in Houdini given the software does not promote any particular behaviour like Maya or Softimage.

# General concepls - Organising Scenes 

## General concepls - Organizing Scenes

One of the things I find fundamental all throughout my career is how to organize complex projects, every package seems to want to promote certain behavior, Houdini is very light touched and though, it can become extremely complex to follow.

This is one of the reasons I keep comparing it with a compiter, you can put all your code into one single monster size file that is simply too big to maintain, you can put packages together and make it modular, you can work with objects and make your code as abstract and reusable as possible, you can take it a step further and do test units, etc... And like everybody writing code you end up having a very strict coding standard, for names, for organisation, even for how parenthesis need to be put together.

For example:

```
Yes:
print ("This is much nicer.\n"
            Do it this way.\n")
    No:
    print """This is pretty ugly.
Don't do this.
```

(see it all at http://google-styleguide.googlecode.com/sun/trunk/pyguide.html)
Because of this I have put together basic strategies to organise your objects in a similar fashion to what you probably have been doing one way or another in Softimage. Very few things I am going to describe will have any impact on the proceduralism of your scene and the goal is simple, PORTABILITY.

## General concepls - Organizing Scenes

I have put together a tiny example on how to build an object in Houdini, mostly the same questions that arise when you do in Softimage but I want to make sure you structure things in a Houdini friendly way if possible. Sometimes the effort pays of, specially when constructing tools to be used by others, most of the time it does not matter but for me, it is specially critical everybody in the team does it the same way. Either green or blue, does not matter, but everybody green or everybody blue.

So I am going to explain how to build a simple table with four legs and how to architect your object in many fashions.

On all examples I have wrapped the model on a subnetwork to mimic the approach I use so much in Softimage encapsulating these into models.


## General concepts - Organizing Scenes 1

ONE BIG BLOB APPROACH
In this approach everything is contained in one single geometry node and you output the final thing as one mesh, it is particularly useful for static objects but for moving objects (let's say a car's wheels) this is clearly not a good idea.


## General concepts - Organizing Scenes 1

ONE BIG BLOB APPROACH

Diving in you can see a geometry node that contains the whole table's description.

Also you will notice that I have connected the input 1 node to the geometry node so the
transformations you apply to the subnetwork (model) are passed downstream to the geometry itself.


## General concepts - Organizing Scenes 1

ONE BIG BLOB APPROACH
And here you can see that tabletop and all four legs are single elements, placed by hand where they should and configured out of the first node with the desired scale (not very good but efficient)

Then all merged into one single object and connected to a null that I am calling OUT.

This is a wide spread technique of all Houdini artists and the point is that as other parts of the scene reference the OUT node, if I rewire things before I wont have to touch anything else anywhere else.

Just do it this way always please.


## General concepts - Organizing Scenes 2

## AS INDIVIDUAL OBJECTS

In this approach we put every piece of geometry as an object, this is my preferred method when dealing with animated objects (wheels of a car) as transformations at OBJ level are way faster than at SOP level. Think transforming a mesh versus moving the vertices...

Also will allow you to approach material assignement as drag $\$ d r o p$ in a simple way so something to consider there too.


## General concepts - Organizing Scenes 2

## AS INDIVIDUAL OBJECTS

Once inside the subnetwork (model) you can see that I also plugged it onto the tabletop to inherit transformaitons, then plug these also to the legs.

As you can see this is nicer to look at as it resembles the schematic approach and it is very modular and clear.

Obviously when you have tons of meshes your network will explode but there are ways to havigate through that.


## General concepts - Organizing Scenes 2

## AS INDIVIDUAL OBJECTS

And finally in SOP level you can see the leg, placed in the correct place and connected to an OUT node.

This is for explanation purposes good enough although has some major caviats, specially since we are repeating the geometry in memory in full and will be less than ideal.

Also managing so many objects will be painful but illustrates a good approach.


## General concepls - Organizing Scenes 3

AS TRANSFORMED OBJECTS

In this approach I try to use the previous technique but also try to optimize memory and make sure management of multiple objects is easier.


## General concepts - Organizing Scenes 3

## AS TRANSFORMED OBJECTS

Again, connected to the input but now the legs are just one object that contains the four legs.

This technique also provides a good way to attach materials to the legs in one go, also is modular enough and you could have variations of the legs in a very obvious way to maintain.


## General concepls - Organizing Scenes z

## AS TRANSFORMED OBJECTS

Here is an elegant way of building the legs.

First I construct one in the middle of the world (that is the right way in Houdini) then move it arround to the various positions, then merge it all and push it to the OUT node for referencing.

The great thing is that I only have one mesh, it is easy and selfcontained plus I still can put different materials to different legs if I want to.

The only drawback is that if I have to export an alembic file I will have to convert to mesh before exporting.


## General concepts - Organizing Scenes 4

AS COPY OBJECTS
In this approach I use probably the most important node in Houdini, the copy node which I use in its most simplest form.


## General concepts - Organizing Scenes 4

AS COPY OBJECTS

Again I am relying on grouping tabletop and legs as just two objects.


## General concepts - Organizing Scenes 4

AS COPY OBJECTS
Here you have a more linear way in which I construct the leg, then move it to the first position and with the copy node I build a second copy1 negating one of the axis.

Then both legs are passed to the copy2 node and repeat the process in the other axis.

All that now passed to the OUT node.
Beware you wont be able to easily pass a different material per object.

This method is fantastic for big number of objects, effective and very easy to maintain.


## General concepts - Organizing Scenes s

AS A PROCEDURAL APPROACH
This is just one way of doing a procedural object that would be very easy to encapsulate in an asset, add controls and would be very easy to maintain.


## General concepts - Organizing Scenes s

AS A PROCEDURAL APPROACH
This time I construct it all in one big block.


## General concepts - Organizing Scenes $s$

AS A PROCEDURAL APPROACH
A bit more complex but if you look at it is quite obvious.

Here everything is constructed by copying the legs to an element I use as a template and then merge it all.

In this example I colour the nodes blue when I am expecting the user to tweak values on the tool, with this visual aid later will be easy to collect those variables and put them in the tool interface.


## General concepts - Organizing Scenes $s$

AS A PROCEDURAL APPROACH
As you can see the leg has been defined in the leg node in the center of the world.


## General concepts - Organizing Scenes s

AS A PROCEDURAL APPROACH
Then I move the leg down in $y$ using a variable that has been automatically generated by houdini.

Houdini has provided me for free with \$YMIN that gives me the bounding box minimum $Y$ value and I use it to move the leg down.
of course if you change the leg height this will be updated providing a true procedural approach.


## General concepts - Organizing Scenes s

## AS A PROCEDURAL APPROACH

Then I copy the leg to the vertices of the object I am using as a reference which is simply a scaled down version of the grid.


## General concepts - Organizing Scenes s

AS A PROCEDURAL APPROACH

Then I merge the legs with the tabletop that has been created with a simple extrusion, make sure you tick the Output back option so you get a closed box out of the polyextruede node.


## General concepts - Organizing Scenes $s$

## AS A PROCEDURAL APPROACH

Now I move the table to the floor with the help of a reference expression to another channel.

Here you can see I am reading the value of the leg's height and moving the table up that same distance.

Again, this will be procedurally correct and easy to maintain.


## General concepts - Organizing Scenes $s$

## AS A PROCEDURAL APPROACH

Finally this is what we output to the scene.

Remember to put your view flag and render flag on the objects you intend to render and display.

This is a very loose Houdini procedure but very common and worth keeping an eye.

In Softimage we use to force visibility and render status of objects using Passes and Partitions (still the best feature ever) but in Houdini you cant do it the same way so dont assume that is the case.


## General concepls - Organizing Scenes

As you can see there are at least as many ways to do it in Houdini as in softimage if not more and you have to find your own way, simple as that.

For illustration purposes I have not done any referencing or expressions on any example except the last one but keep in mind all the various examples could be more procedural.

Forcing visibility and render flags can be done in three main ways in Houdini.

- Using object merge nodes to collect all things visible or renderable. This would be a very particular technique that is not that common from what I have seen but I really like although implies quite a bit of manual maintenance so careful.
- Using bundles to force render flag in combination with your pass (ROP) settings (basically instead of refering to an object as a candidate you tell everything should be hidden except whatever is in the bundle $X$ ) More on this Later.
- Using your pass (ROP) settings directly in an explicit way. This in my opinion is the mother of all errors, just avoid it. I know every single Houdini artist I know uses this approach but it is awful. I will explain more later on this rendering strategy subject.


## General concepks - Organising Materials

## General concepts - Organizing Materials

Houdini materials and how to use them is so dependant on how the scene objects are organised that before jumping into how to put together a material I am going to review various strategies that you may want to consider.

First of all in softimage we tend to simply apply a new material to the object and done, we just tweak the parameters that pop up on a PPE window.

Normally you use either an architectural material (mia_material) or some constant/lambert and other utility materials for either previz or generation of matttes.


## General concepts - Organizing Materials

In Houdini is different, you create your material in your SHOPs context by (for example) dragging and dropping from the Material Palette the "mantrasurface" material to your SHOP network or the object directly and a new material node is put.

The material definition (its network) is inside, so again, like with objects you will have to dive in to access all the goods. And there are many becuase SideEffects has opened basically everything and something that is impossible to do in Softimage, which is inspecting how the Architectural material has been put together is perfectly possible in Houdini.

Of course the core building blocks are compiled and those are off limits but you can get a very very good picture of what is under the hood and change its behaviour, interface... pretty much you drive.


## General concepts - Organizing Materials

So we can create a copy of the material and set one to red color, the other to blue.

Now we have in our library of materials two nodes containing inside the network that describes the material (think a rendertree inside each)


## Ceneral concepts - Organizing Materials

And here is where there is a critical decision to be made because you can apply the material to the object containing the network that describes the geometry (OBJ context)...
....or inside the network itself on any part you want (SOP context).


## General concepts - Organizing Materials

So if you remember the Organizing Objects guide you will be able to go through various situations and questions, should I put the materials at OBJ level or SOP level, in some cases is obvious, in others it is not and the general consensus is that there is no right or wrong but this is a very different proposition to softimage... is it?

Well, in Softimage you can apply your materials in branch mode and node mode, you can do it via Groups or even via partitions so we have already recognised there is value in any of these situations...

My rule of thumb is to do it at OBJ level if possible as drag and drop works better this way.


## General concepts - Organizing Materials

Now, navigating a huge scene may be tricky if you have to dive in and out countless times, something we have managed extremely well in softimage is not so obvious in Hoduini, well, let's see how to mimic the behaviour.

You could use the tree view so you have something similar to an explorer but in Houdini materials attached to at OBJ level dont appear, so you may want to evaluate if this is for you, in any case I am not bothered because the networks can be huge (and normally are) so the explorer is pretty much useless for these tasks.

BUT...


## Ceneral concepts - Organizing Materials

You can use the new Data Tree which is in the process os becoming the hub for managing complex scenes and although in its infancy already shows amazing potential when dealing with Alembic files.

Here you can see it the interface, you can control also teh level of display which is very convenient.


## Ceneral concepts - Organizing Materials

You may want to use the Material Palette as if it was your material manager in XSI. and yes, you will notice is way inferior but will give you a key feature, find all the SHOP networks and their materials an list them.


## General concepts - Organizing Materials

As you can see it is not so coherent and slick as XSI in the management side but these are things I expect sideEffects to improve soon.

Now that we have the idea of applying materials either at OBJ level or SOP level I think is time to see the concept of SHOP networks.

One of the most important design factors when building an asset that is going to be deployed for a whole studio is for the asset to be selfcontained, Houdini takes this very seriously (more than Softimage even) and you can not only embed your logic and scripts, help and icon definitons and countless thing, you can also embed other context subnetworks.

So a car asset will be able to not only carry the geometry inside (or as a reference) but also a SHOP subnetwork with all the material definitions inside and thus, making possible all the geometry referes to an internal componnent.

Furthermore, you can add your render configuration too, and your compositing network and anything you want.

This is truly mind-bending because you could set your asset to have an interface that runs a script that renders a turntable of the asset and puts a composite of the various passes on top of a background, brands your asset with your company logo and creates a quicktime movie. And EVERYTHING is inside the asset making ultra-complex setups really possible.

But let's start with the SHOP subnetwork.

## Ceneral concepts - Organizing Materials

If we go to our Table built as one single objects with one single network that describes everything...


## General concepts - Organizing Materials

If we go to our Table built as one single objects with one single network that describes everything...

And inside I press TAB and type SHOP Network and select it...


## General concepts - Organizing Materials

... I will get a SHOP network inside the element I am in, and inside I can build my materials.


## General concepts - Organizing Materials

So now I build a green and yellow material and will proceed to rewire them in the object nodes.


## General concepts - Organizing Materials

Now I will open the explorer tree to find the right operator which in this case may be the green material and also I tick the "export relative path" as I want to do relative references rather than absolute.


## General concepts - Organizing Materials

And inside the network that describes the object construction (SOP context) you do the same, and presto, now our table is referencing all the materials locally instead of globally!!!

Wait a minute... what have you just said?

Well, you should look at it like each subnetwork is a material library but instead of living at scene level like in softimage, they live INSIDE the object, therefore this is a LOCAL library instead of a GLOBAL Library.


## General concepts - Organizing Materials

And funny enough I have found myself using these approaches to build a workflow that I find greast.

I put all my material library I have been curating for quite a long time in the scene SHOP tab that gives me extremely fast access when doing the look dev, then I copy those material definitions into the local library (SHOP network inside an object) and once I am happy I can simply export the asset I am building as a HDA (houdini digital asset) and all the definitions are inside.


# General concepls - Organising Renders 

## General concepts - Organizing Renders

This is one of the areas that is wildly different to XSI, mostly due to the inherent design of the application and thus I recommend you to take a deep breath, as usual, you are going to loose lots of functionality in exchange for lots of functionality, in some cases you will like it, in some cases you will not but hopefully like me the compromise is not too bad given I get something really good in exchange of something really good.

The most important thing you will notice and cry is that fact that Houdini does not have the concept of Partitions nor Overrides that made XSI so powerful when it comes to render... hopefully the nice guys at Side Effects will investigate how to do it???? ;-)

## General concepts - Organizing Renders

So let's refresh our memory first to make sure we talk about the same thing.

Passes are a way to collect objects and manipulate them with the help of Partitions.

In this example you can see I have the Default Pass which we XSI users normally use to develop the scene and work around everything as we go atong and then the THUMBNAIL, VALIDATION and WORKBENCH passes to produce the final images.

It is very very typical when using XSI to force the visibility and render flag of an object along with the materials attached to it.


## General concepts - Organizing Renders

Lets review what is happening here in detial.
First of all the 'Background Object Partition' is where any new object is created, and because it is hidden in its entirety any new object will be hidden too.

In the 'Objects' we have an override to force a particular set of attributes of the partition content to be redefined. So if these objects are hidden because someone was working on something else on the Default, Pass, everything inside this partition will be turned on by force protecting the user from accidents. (I know the override has visibility and render flags on even if I am not showing it in this view)

Furthermore, the various partitions have various different setups so for example the 'Validators' partition are objects that I am forcing to see even if you dont want to, but I am also forcing NOT to be rendered protecting the user from accidental moves.


## General concepts - Organizing Renders

This is a convention all XSI users have used and established as the right way, the most flexible and safe and the outcome is a very important (acutally critical) concept.

## We do separate production objects status from

 development object status by using this convention.What do I mean? Well, no matter how I work with my objects, if I hide things because I am focusing on a character but not a car, I will be rendering the right objects.

Once everybody in the team plays this game the reduction in mistakes derived from having an object 'temporarily hidden' because you wanted a faster responsiveness from the application is pretty much null.

I am fully aware this is totally alien to Houdini artists and Maya artists but this fundamental concept is one of the key ingredients in the magic recipe of how it is possible to be so quick and efficient as a well seasoned XSI team.

Sorry (Houdini aritsts) if it sounds patronizing but you can actually mimic this behaviour in Houdini (with Limitations though)

## General concepts - Organizing Renders

Let's consider this scene and try to replicate the behavior I jut described.

I have put together three objects, ball, grid and a cylinder to build this minimalistic scene.

Also I have put a distant light and a camera and line it up as you can see.


## General concepts - Organizing Renders

Plus I have created four materials, assigned the white to the floor by draging it into the object and as you can see in the parameter's pop up window it has been put in OBJ level by default.

I have coloured light and camera for easier identification but that is just cosmetic.


## General concepts - Organizing Renders

Now what I am incerested is in inspecting how to force things in Houdini like we do in XSI.

So I go to my ROPs (/ouk) and put a mantra ipr node to test what is happening and as you can see the camera is referneced by a string to the object context.

Also I have gone to the render view and do a little render using the manera _ipr and the ROP camera referenced.

Meaning, I am using as a render camera the camera that has been specified in the manera ipr.


## General concepts - Organizing Renders

And if I navigate to the Objects tab I can see a bunch of fields where to specify the objects to manipulate. Here is the interesting thing, you can actually force objects to be rendered, matte objects (holdouts or cut-outs) and phantom objects (primary ray off but everything else on) and also you can rule out others.

The same for lights so it seems quite familiar right?

Well, the only thing is that you will end up likely Eyping by name the objects that you want to force to be rendered, etc... and this is very error prone, specially if you have tons of ROPs nodes in your scene...


We need something better.

## General concepts - Organizing Renders

The answer are bundles which is a type of grouping you will, like me, love.

Open your Bundle List view and create four bundles (in capital letters better to be recognized well) and drag and drop your objects into the respective sections.

I chose to put the cylinder on the PHANTOMS so I can render its shadow but not its geometry (primary rays off in Softimage)


## General concepts - Organizing Renders

Now in my ROPs context I modify the parameter so there are no candidate objects (these are objects to be rendered without forcing them, therefore if they are visible, they will be rendered, if they are not, they will not appear)

And add the bundles names into each respective field.

Now render and voila, we have managed to separate production rendering from development rendering.

Of course you can create more mantra ROPs with candidates on if you prefer so but this is the best way.


Now, some extra goodies...

## General concepts - Organizing Renders

Now if I create a smart bundle and type a string to match, it will find all the nodes with that string and automatically add them to the bundle

So if I were to copy the cilinder three times I have already collect them for free!!!

Furthermore, you can convert a smart bundle into a normal one and play with visibility, selectability from there.

The final point is that you will want to architect your scene based on your project so it is up to you if you want to add different bundle for different cars, you can refer more than one bundle in your ROPs.


## General concepts - Organizing Renders

There are some considerations to be made, you cant truly override any parameter you wish with this method, for that you will have to approach each project in a paticular way but it is beyond this documents how to do that as it is a bit more advanced. I will try to offer a solution or two of course.

Also, as in XSI you will have to explicitly put each object in the correct bundle and we dont have any way to 'match partitions' in Houdini although it is a trivial tool to build.

Furthermore, if you end up with a project that evolves into a much better beast chances are you will have to reestructure the whole render approach, but again, this is something you would do in XSI too so it is simply a suggestion to think about things first.

The key thing to rememer is that a partition is a part of your pass, a bundle it not and this has some benefits becuase you can refer to the bundle pretty much wherever you want (light linking for example? he he he).

The bad news is that you cant export your bundle recipe by default, which means the best approach for a big project is either create a script to deploy to everybody that takes care of this setup or have an empty scene that is not empty at all.

In my case is the Later, becuase the complex configuration of my ROPs with arbitrary outputs (aka AOV, aka framebuffers) is something I want to have contained in one scene.

## General concepts - Organising Overrides

## General concepls - Organizing Overrides

Overrides are something of a godsend and the way XSI has implemented their rendering appraoch it is truly elegant so please bear in mind this is mostly an interpretation of a way to get a similar-ish result rather than a one to one appraoch.

For those who dont know about Overrides in XSI it is simply a tool that, as its name indicates, overrides values, both strings or connections to other objects. They work on groups, partitions and allow you to easily create complex combinations of parameters to arrange your scene in a particular way.

The most common use are for both performance related tasks (like displaying the geometry in its basic form but forcing the render to use subdivision 2 for example) but even more important is used on a daily basis to configure complex passes setups. (like configuring phantom objects on one pass but not another, like forcing the render flag of objects, or lights, or provide variations of materials this includes injecting whole shader trees too-).

Of course if you have never used them it is easy to overtook them but the fact is that this is one of the key reasons softimage was so effective and flourish in small setups, specially advertising, where ultra-fast turnarounds are paramount.

In Houdini you can achieve pretty much all of it but the issue is that it is not a generalised and elegant solution, it is a custom solution per problem and in many cases may involve some serious planning beforehand so I will only show two of the many many methods you could use that hopefully will give you a starting point.

## General concepts - Organizing Overrides 1

The first method is old school but still I like it a lot, mostly because it is very obvious to see but also becuase it forces you to be very organized and strict with your setup, something I particularly like.

The idea is that you have your object, let's say a chair made out of a metal frame, leather pieces and also bolts and I want to override the material.

Like this one;


## General concepts - Organizing Overrides 1

If you inspect by hovering and middle mouse click this particular node you will see the mesh I have imported does not have any groups, just a position (P) attribute and a texture coordinates (uv) attribute.

You will see the basic stats of the object and the amount of memory it is using.


## General concepts - Organizing Overrides 1

What I do so I can visually understand what is happening and for education purposes is to altach a shader I have already pepared before that lives in the /shop area (global library).


## General concepts - Organizing Overrides 1

So now I select the meshes and delete them creating a blast node which does exactly that. I revert the result so I delete what I have not selected and repeat the operation for each of the elements.

Then I export put down a null node and rename it to OUT. Frame, OUT._Leather and OUT_Bolts as I am planning to refer to these objects from somewhere else. Also colour them green to highlight these are the output results of this network.


## General concepts - Organizing Overrides 1

Finally I create another geometry node where I will collect whatever has been preprocessed in the ingest node and assign whatever material I want.


## General concepts - Organizing Overrides 1

And using an object merge I refer to the corresponding OUT node, one per object for the sake of being thorough.

Then I assigh a different color to each part, effectively overriding the material I had previously created.

Then I merge it all back together and add a null node but this time I call this RENDER so I can do an automatic search on the bundle to collect the things I need to render.


## General concepts - Organizing Overrides 1

As you can see on the material I am refering to the global library/shop although I would rather do it on a local library and use a relative reference.

This will make it modular as the intention of this process will be to encapsulate it all into a HDA I can use wherever we want without requiring any particular configuration on the scene.


## General concepts - Organizing Overrides 1

So, this method is not simply an overriding method but could be used for tons of things, truly up to you.

A good example of the paradigm would be a car mesh coming from softimage for example.

You will first prep the object in an ingest node and exported as I just did. The benefit is that now the wheels will be objects in their own right and their axis will be something you can manipulate at object level which is much faster than SOP level.

Doing these kind of manipulations may seem over the top but Houdini caching mechanism is acutally amazing, it 'cooks' the nodes and keeps data stored so it can prevent from cooking things unnecessarily. If you lock the final node that you want to make sure the cache is reading from Houdini wont attempt to recook upstream.

Many Houdini users (in fact all of them) tend to actually cache things to disk and generate files with all the data (bgeo holds lots of data).

I will discuss the merits of these approaches, but coming form softimage I rather not do that as file management becomes very complex if you are not careful.

Regarding this method to override you can of course do other things like change attributes, deform the mesh, clean up the geomery, etc...

## General concepts - Organizing Overrides 2

The second method (of many) is to use takes, a tool that I feel it has not got the attention it deserves as it really brings amazing value if you are careful.

Funny enough in softimage we are very used to it by means of dealing with partitions and passes plus all the hierarchy of priorities (object, group, layer, partition) so for us is acutally the natural way to go. Just be other Houdini artists around you will not expect you to use it. Weird.

The idea behind takes is simply to truly override parameters, of course you will have to take care of refering in your render node (ROP) to the correct take but you can rename them tom make them more easy to read.

Here is my little example.

## General concepts - Organizing Overrides 2

In this case instead of going for a blast approach in which I branch out and then regroup them in a separate geometry node I have use the grouping tool to create the groups I am going to manipulate.

I simply go into component selection and in the tool menu click on Select Connected Geometry so once I touch a polygon it gives me all the connected ones.


## General concepts - Organizing Overrides 2

then with the selection completed I press TAB and type group so a new node is dropped in the network and the selection is added to it.

I repeat the operation for the rest of objects until I am happy.

Of course the little bolts are hard to pick, the solution is to use a 'pseudo-Isolate' trick by dropping temporarily a visibility node that allows me to hide what I want so I can pick things correctly.

After that I can delete the visibility node and move on.


## General concepts - Organizing Overrides 2

The group node is specially important and although I wont go into details I wanted to show you a couple of things.

First of all I am using the \$0S variable (operator name) to create a group with the name of the node itself so whenever I copy the group, because you cant have two nodes with the same name Houdini will add a number to the node which will indeed create automatically a new group.

You certainly will exploit this a lot in the future.
Regarding the Pattern, this is the polygon indexes you are grouping, indeed it is a string and as you suspect you can read it somewhere else to do whatever you want, you can override the values... well, you get it.

## General concepts - Organizing Overrides 2

Now when I go to the material node you will see there are three little tabs $(1,2,3)$ in which I am assigning a different shader to each group. All in one go.

Doing this is less readable from the outside so I tend to rename the node to something like material multi to indicate that there are various things going on in this node.


## General concepts - Organizing Overrides 2

So considering we have this model now prepped, let's move into the overriding with takes.

Open a new 'take list' Pane tab, this is your take manage and where we will operate.


## General concepts - Organizing Overrides 2

Then create a new take and called RED . VERSION.

Now to add the overrides what you have to do is drag and drop the parameters you want to override to the take pane and voila, it will give you the path to the object.

In my case I added the three materials and you can see here they refer to /shop/gree, /shop/white and/shop/blue.


## General concepts - Organizing Overrides 2

So now is just a matter to go to the parameter that is suddenly unlocked and you can point it to whatever you want.

Of course after you point all these materials parameters to the red material you will have the Main version of the scene and the RED .VERSION of the scene and by selecting one or the other you will see the materials are changing.

My only suggestion is to turn off Autotakes, which is the way softimage people work anyway. You basically need to explicitly say which parameter to override rather than dumping the whole thing.

## General concepts - Organizing Overrides 2

And that is it, now is time to configure your ROP to use the RED . VERSSION instead of the Main one.

This is the modern approach to Overrides, again much closer to Softimage.

Now you must be wondering why have I shown you the first method? well, in some places they will tell you now to use takes for whatever historical reason, like in Softimage people tell you not to use Layers... I use Takes and love them.

The point is that the other method is very useful for other things and combined together is extremely powerful.


## General concepls - Camera

## Ceneral concepes - Camera

Like every other package, Houdini has a camera that is different, the good news is that its better. In this small chapter I am going to explain a few things about it.

First of all, like everything else in houdini, the camera is a just a camera, without a rig like softimage gives you, which let's be honest, we learned to use and love but was not great at all. Also all the mess with FOV vs. Film aperture is gone and the camera works and is presented in a very coherenet way.

If you drop a camera in your network using the TAB menu or the Shelf toolse you will notice when browsing the parameters it has some very weird values, like the resolution which is set by default to $640 \times 480$ and something I hope SideEffects fix soon.

There are some names that are very misleading so I will try to outline the various names of the camera comparing them with Softimage.


## General concepts - Camera



## General concepts - Camera

Also keep in mind the sampling tab relates to the motion blur, Houdini does not have a physically correct model of the world like Maxwell or Vray have, therefore these values are not going to affect the luminosity of the scene as they should, but only control the motion blur and depth of field effect.

In Houdini 13 SideEffects added a camera shader functionality if you want to emulate that behaviour, something I am about to start but so far it is not standard.

NOTE: If you come from Mental Ray you are going to have a good surprise, motion blur, displacement, depth of field are truly fast to the point I do render now pretty much everything with true 3D motion blur, no need for motion vectors anymore.

## General concepts - Camera

Name equivalence between packages

| Houdini | XSI | Maya | SyMbol |
| :---: | :---: | :---: | :---: |
| Focal length (mm) | Focal length (inches) | Focal length (inches) | f |
| Aperture (mm) | Film aperture (inches) | Camera Aperture (inches) | $d$ |
| $-n / a-$ | Field of view (degrees) | Angle of view (degrees) | $a$ |

Calculate the Field of View in Houdini using this formula

$$
a=2 \arctan (d / 2 f)
$$

And if you need to calculate the Focal Length from the field of view

$$
f=(d / 2) *(\cot (a / 2))
$$

## General concepls - Camera

Camera controls in Houdini are a bit hidden, let's face it, this seems too simple but wait a minute, the actual tool is freaking good.

Visual transformations of the camera are accesible if you press enter and dive in, a position + interest handle the ability to pivot form both sides and correctly lined up to the camera is available.

This is extremely useful to finesse complex moves without the need of rigs.

As there are three different modes you will be swapping back and forth, the shortcut for this mode is ' 0 '.

## Ceneral concepts - Camera

Visual focus control appears if you press ' $z$ ', and three handles appear so you can control the focus point and the out of focus zone.

The little cube is your focus, the arrows are the near/far controls for focus which will change the values on the sampling page of the camera.


## Ceneral concepts - Camera

To control the camera fustrum cropping you can press ' $F$ ' and the visual aids make obvious how to change the near/far plane.

You will notice there is a planar scale tool on the near position, this will change the "camera" aperture. (aka sensor size)


## General concepts - Camera

One thing you will like to keep in mind is that like every single application, scaling camera is never a good idea, for this i will show you a tiny trick that will help you "clean" the camera transformations.

As so often happens that the scene and camera are scaled up/down after importing a camera track from 3DEqualizer the net result is that the camera inherits the scaling... if you want to reset it to 1 here is how.


## General concepts - Camera

Simply add a blend node after the camera, deactivate the scaling flags and create a new camera and parent it to the blend.

Of course the lens data, resolution etc.. will require you to copy the parameters by either typing them or if they are animated adding an expression but now you have a camera to render that is clean.


## General concepts - Camera

One of the last things you will certainly want to do is to have some black flags to help you frame your action.

For example you will be rendering HD $1920 \times 1080$ but there is a 'creative crop' to mimic the feel of film (2,36 aspect).

The way to do this is to add a viewport mask and activate it editing the camera interface itself.


## Ceneral concepts - Camera

And when the configuration control (yes, it is huge and there are tons of extras you can add but I will let you explore it by yourself)

You are after the View Mask under Viemport Display. Once you find it move it to the interface pressing the right arrow (in white here) and accept the changes.


## General concepts - Camera

Finally over the camera let's activate the View mask by hovering on the camera view and pressing ' $d$ ', now the options appear and as shown in the image activate the 'Camera view mask'

Now you have a wonderful creative crop (white) that you can turn black to help you out.

Other more advanced things like foreground planes with masks etc... is something I will talk in the future.


# Ceneral concepls - Color Management 

## General concepts - Color Management

Since Houdini 13, the approach to color management is prelly similar to Softimage; - )

You just need to set a general selting for color in the preferences menu and compensate for your screen gamma selting which can be assumed to be 2.2 by defaulk. Also loggle on all the options to make sure it is applied accordingly on every viewport and ikem (picker included).


## General concepts - Color Management

The only difference is that in XSI there are three modes for the textures; Linear, sRGB or Automatic, in Houdini is always automatic and the assumption is trivial.

- 8 and 16 bits images are assumed sRC-B.
- 16 bit EXR are assumed Linear
- 32 bit images are assumed Linear.

The idea is that whaver you refer to is automatically linearised, the render engine that its computations and saves an image in the format you are after putting the color space correctly. (Like XSI)
of course if you want a very particular setup and to have full control of every color conversion, something I would discourage strongly unless you really know what you are doing, there is always the possibility of setting the environment variables;

HOUDINI_AUTOCONVERT_IMAGE_FILES to O HOUDINI_IMAGE_DISPLAY_GAMMA to 1.0.


## General concepts - Color Management

The last thing to mention in this very short guide is that you want to watch out the various views as they have stowed parameters, sometimes they control the color and you can use them to explore 'what if' situations.

For example, playing with the gamma and brightness (without re-rendering) to see if an increase in Luminosity would be better.

Interestingly you can use also your own LUTs all across the appliction as you would do with softimage although you can also differenciate between gamma for the interface itself and the viewport display, something really useful if you are working in a complex setup.


# General concepls - Expressions 

## General concepts - Expressions

Houdini as has been designed in a very particular way and my impression is that Renderman compatibility and philosophy were embraced in the very design of the software since the early days.

I have this opinion, may be I am wrong, since it looks more a compiler of to generate screen descriptions (RIB files) than a normal 3D package.

Therefore we are going to go through a few concepts you will be using a lok, from environment variables, internal variables to expressions and string evaluations so you can get a feeling of how to manipulate various things and also take advantage of them to do more work faster.

## Ceneral concepls - Expressions - Variables

Houdini provides a lot of variables that will make your life easier plus will help you build fully procedural scenes, here are just a few based on how to use them.

## \$0s

This variable will return the name of the node, and it is specially useful to try to marry the name of the node as a parameter inside that node, avoiding having to dive in every single time you tweak it.

For example, in this version of the chair I have created some groups for each section, but I dont like my group node and the name of the group I create being different, so rather than typing like crazy...


## Ceneral concepls - Expressions - Variables

I use the \$0S variable to make sure they are in sync always.

And if you inspect the state of the sop network at the group point (hey, this reminds me of a breakpoint), you will see that now I have a group with the same name than the node that is creating it.

Becuase when you copy nodes they change name automatically (you cant have two nodes with the same name) I am protecting myself from my own mistakes too, which is one of the fundamental things I try to do always.


## General concepts - Expressions - Variables

You can exploit this also when rendering, this way if you assume each ROP is a pass (let's talk about framebuffers -aka AOVs- Later on) this will protect you from writing over your own files and maka very easy to establish a workflow without having to write a single line of code.



## Ceneral concepls - Expressions - Variables

Houdini also gives you simple string management to deal with small things.

For example, there the name is builg from the path to the HIP file, plus the filename but by adding the :r modifier it strips out everything else and returns the root of the name;
chair.hip $\rightarrow$ chair
and then get's the ROP node to construct the name;
/users/jordi/Desktop/mychair_beauty_vo1.exr
all without writting a single script.


And the good news is that Houdini you can read your environment varibles using exactly this method so suddenly your pipelin will be a pleasure to work with.


## General concepts - Expressions - Variables

Another varible you are going to use every single day is;
$\$ F$
This returns the frame you are, a typical use of this is for a see to a function like SIN to produce oscillations, etc... but more
importantly you will want to use it for loading and writting sequences too.

For example, here I am adding the \$F to write every frame, otherwise I would be writting the same frame over and over.
(sorry if it is too obvious but I need to go step by step here)


## General concepts - Expressions - Variables

Now, what happens if you want to use padded zeroes?

Well, you can produce them just by adding the number of digits.

At frame 1, $\$ F 4=0001$


## General concepts - Expressions - Variables

The thing that may throw you will be when to force an evaluation on the screen.

My approach is, Houdini evaluates the field and resolves the variables with the content of them.

But if for example I want to calculate a frame offset the parser does not know if +10 is simply a string or if there is a need to evaluate $1+10$ as an expression resulting in the error you see here.


## General concepts - Expressions - Variables

To fix this you will require to force an evaluation, this is achieved by using the backticks character like this;

Assuming you are in frame 1
$\$ 54+10$
will resolve correctly as


## General concepts - Expression Functions

of course environment variables are not enough, you want to run fuctions which give you huge amount of extra firepower.

For example I want to create a camera slate as Houdini does not provide truly a real one so why not do il ourselves?


## Ceneral concepes - Expression Funclions

of course environment variables are not enough, you want to run fuctions which give you huge amount of extra firepower.

For example I want to create a camera slate as Houdini does not provide truly a real one so why not do it ourselves?

I just moved the text in $z$ so it fits inside the camera and the only thing left is to add an expression function to bring the name of whatever this node (fonl) has as input 1.

In this case the expression would be 'opinput(". .",0)
with backticks to execute it.


## Ceneral concepes - Expression Functions

Now when I change the name of the camera or connect the font node to another camera or object it will resolve it by itself.

This is a way also to create a tool to label objects for example.

I wanted to show this to bring one idea to the table, Houdini gives you building blocks (like lego) and you do whatever you need to do with them... onion skinning in the viewport? easy.

Of course these tools can later be deployed to the studio so everybody benefits from them and yet again, you wont need to code at all in a huge number of cases.

## General concepts - Expression Functions

Another very useful tool you will be using across your projects is the 'opdigits' expression function.

In this example I am using the node suffix name as a seed for random number generation which I can later use for texture selection, scaling, etc...

I dive inside and add a transform node to illustrate the operation.

As you can see the opdigits("..") command really is referring to the parent object (think of it as folders) and grabbing the number in there.

This will return 1 as the name of my object is "geo1"


## General concepts - Expression Functions

Then I am wrapping the result and passing it to a randomization function that will give us a value between 0 and 1


## General concepts - Expression Functions

And now I remap it to the values I want using the 'fit' function.

I am in this case I map from the original range o to 1 to a
destination rage of 10 to 20 and the result on this object happens to be 18.0046

Now if I copy the object the network will be recomputed and given the new name will be 2, the end results surely will be different.


## Ceneral concepts - Expressions - Functions

Houdini provides also two scripting languages to script whatever you want. On of the (the oldest) is Hscript which I will show how to use a few of these tools in your string evaluation.

Python is very well implemented (better than Maya and XSI certainly) and you will notice Side Effects is putting a lot of effort in moving towards Python.

So let's explore the scene from a completely different angle for a second, let's dive as if it was a filesystem.

Open a textport and type 'Is -l'... exackly, you are navigating a 3D scene as if it was a filesystem.

## Ceneral concepts - Expressions - Functions

And you can move into the subfolders (your objects - OBJ) to find at the end your files (your operators

- SOP) and of course you can run normal (kind of) shell commands like deleting, linking, eti...

The text completion is giving you names of objects and also the properties of the object you are in, like the display flag for example.

So you could build a script to run a bunch of commands to perform a particular task.


## General concepts - Expressions - Functions

To know the list of functions just type help and as you would do in a normal shell you cand ask help for a particular command.


## General concepts - Expression Functions

Just to recap, you can leverage the evaluation of functions, variable resolution and scripting on your project or company, just make sure you keep these references at hand.

EXPRESSIONS IN FILENAMES
hetp://www,sidefx.com/docs/houdini13.0/render/expressions
EXPRESSIONS
hetp:// wnww,side $f x$.com/docs/houdini13.0/expressions/
COMMANDS
http://www.sidefx.com/docs/houdini13.0/commands/

## Shading

## Shading - Introduction

## Shading - Introduction

As you can imagine the principles of shading are the same so I wont bore you with shading 101.

Let's start saying Houdini provides four render engines under Mantra, but I use exclusively the PBR render engine which in my humble opinion is probably the best production PBR renderer in the market.

You have an equivalent to the 'Rendertree' in the form of a 'VOP network' where you can describe your shader.

The concept is a bit more open than in softimage. While in Softimage you throw a few prebuilt nodes that are not accessible (meaning these are black boxes and you dont have acces to the internals) in Houdini these are more open and you an dive inside these black boxes to go to the core of the computations and change them.
of course being more granular means you may have to do some more work at the beginning to switch to the new approach but ultimately is very similar.

The one thing you have to remember is that there are prebuilt networks for you that you can use in a very similar way to Softimage. A good example is the 'mia_material' case that encompasses a ton of things into one big shader that although less efficient than a very optimized shader allows you to get on with the look quickly and in a physically correct way.

In Houdini I use pretty much exclusively 'mantrasurface' which is the equivalent to 'mia material' and then I plug my extra bits if necessary.

A good example of this extra flexibility would be to substitute how 'mantrasurface' calculates SSS, we simply plugged out own shader inside the standard shader and suddenly the 'mantrasurface' shader was on steroids.

## Shading - Introduction

Conceptually, like with objects, you have a container that stores the shader network, this container can have an interface exposed for accessibility and you may build it to your taste but it is not necessary so I will start with a tiny example to move later on to 'mantrasurface'

Let's open SHOPs and create a 'material_shader_builder' which is an empty shader definition (in softimage we call this a material)

Careful with this, in softimage, for the sake of making things simpler when you create a new material the software does create a material but also adds a shader network inside with your lambert shader model applied. In Houdini, and this is across the board, they dont try to nanny you so their approach is more purist and truer to what really is happening.

## Shading - Introduction

You will notice that the parameter window shows a Compiler command, this is because every VOP network needs to be compiled to VEX (Vector Expressions) prior to use by the render engine.

VEX is a very powerful language that is used across the board for shading, geometry manipulation, compositing... you name it.

Does it mean you could write your shaders in VEX? well, yes, that is how started but since the introduction of VOPs long time ago you dont have to.


## Shading - Introduction

When you dive in you will see a few elements laid out for you, these are your inputs and outputs, and now the task of defining the shader (very much like the rendertree) starts.

As with Softimage you will notice color coded nodes, and yes, these colors tell you the types of inputs and outputs on the network.
for example, light green represents vectors, dark green represents floak, blue represents integer, yellow represents a bsdf, white represents undefined...

displacement_globals



## Shading - Introduction

So I added a lamber shader in the BSDF port as I am using PBR engine, change the color to blue, then assigned the shader to a torus and go to the render view to see the result.

As you can see it is pretty much the same than the rendertree.

Well, almost, there is automatic conversion but highlighted as a dashed line instead of adding a conversion node like in XSI.

It is just a malter of spending a fair amount of time to acquire the new vocabulary and idiosyncrasies.


## Shading - Introduction

To add the hightight I just add a specular node to the lambert node result and pipe that to the BSDF.

And voila, you get a shiny donut.
Now if you go back to the material (the container) there is no interface there but of course you want to make the life of your team easier... the solution is to promote these parameters you are interested into the material interface so nobody needs to dive in the network which can be daunting some times.


## Shading - Introduction

By clicking in the strange looking icon on the right side of the parameter (in the parameter window) you have access to a pop up that will allow you to pipe nodes onto that particular input.

In our case we go for the Promote Parameter.


## Shading - Introduction

Which will be represented by a little green dot in the connector.


## Shading - Introduction

Now the Diffuse color is easily accessible and you can reorganise it using the gear icon (parameter interface editor), add logic, scripts to control things your way if you require so and so on and so forth.

Finally you could save your new shader to the gallery too.

## Shading - Introduction

If you fancty to see the internals of what you just did you can right click and go to the 'View VEX code' menu which will open the actual source generated. The one that will be compiled before rendering, remember?

Furthermore, this is VEX so you can learn the language and if you know how to use Renderman you will notice how similar this is to RSL (Renderman Shading Language)

Of course you could decide to add modify the shader and add any extra things you may want to tweak too, this is not just a view, is the actual thing.


## Shading - Introduction

Ok so we are now ready to start developing shaders like we did with softimage so I will go back and forth between both applications to compare things as well as I can.

Have a look at these links for more information.

WORKING WITH VOPS<br>hetp://www,sidefx.com/docs/houdini13.0/shade/vops<br>VOPS NODES REFERNECE<br>hetp://www, sidefx.com/docs/houdini13.0/nodes/vop/

## Shading - Mia_Material vs MantraSurface

## Shading - Mia_Material vs MantraSurface

During the first period using XSI we had to build our shaders pretty much using the technologies and illumination models from the 80 s and 90 (lamber, phong, etc...). Then mental images started to deliver a set of advanced shaders that took the mental ray community by storm and the key shader was mia_material.

An advanced all rounded and complex to master power horse that brought back the idea of having one single big shader that although expensive to compute was the swiss army we needed. Physically correct and production tested we started to see a new wave of high quality images coming out from XSi, Maya and Max that before were the realm of Vray and very few others

Lots of issues ot deal with, from flickering of the final gathering technique to photo methods, it is still today used across the board and although Arnold has taken its place by storm since mental images was sold to nvidia the reality is that the concepts are the same more or less.

In Houdini, having probably the best render engine out there (extremely fast and extremely scalable, is the same story, you have a bunch of old methods and 'legacy' shaders you may want to explore buy my recommendaion if you come from XSI is to focus for a big period of time in mastring 'Mantrasurface' which is in my humble opinion better and easier than 'Mia_Material'.

The concepts are similar so I will try to relate the XSI version against the Houdini version but you will notice that it is easier to pick up than you might expect... plus you can look under the hood and rewire it as required!!!

## Shading - Mia_material vs MantraSurface

First of all, let's go through a little project in XSI to showcase the workflow which will be later mimicked in Houdini.

For those of you XSI users, bear with me, it is just a technical illustration of workflow even if the image does not look great at all.

So I created a simple torus object and upped the geometry divisions to see it properly.

Also set my explorer view and render tree view so we can start working.


## Shading - Mia_Material vs MantraSurface

Then I refresh the render tree and we can see the borus has inherited the default scene material so I altach an Architectural shader straight away.


## Shading - Mia_material vs MantraSurface

The render tree changes to reflect I have now defined a new malrial and alkached an architectural shader all in one go.

The PPG opens to show all the parameters so you can start describing the surface.


## Shading - Mia_material vs MantraSurface

I click on the connector to bring an image so I can test my UV's (nonexistent at the moment)


## Shading - Mia_material vs MantraSurface

The image PPG opens and I see already a texture support put in place for me, the image node connected to the image clip.

Let's remember an image clip in softimage is a wrapper to the image that adds processing abilities like image filtering, tiling...

In Houdini it is different and simpler, the image is the image and it is referred directly so if you want to do any tiling it is done by hand by you.

Again, remember Houdini does not nanny you, it is up to you to do things how you want and does not force your workflow in any way.


## Shading - Mia_material vs MantraSurface

I test the texture support object and the UV's by rotating the object which redefines the UV space... everything works.

In Houdini you will find that there is no concept of texture support, you apply the UVs to the object and that is it, if you want to use texture supports is very easy to do so but again, you have to do it yourself.

This is something I will try to show in future guides. How to construct a texture support.


## Shading - Mia_material vs MantraSurface

Because I want to play with a material that will have reflections I also wank to put an environment map so I go to my pass, in this case the Defaulk Pass and creale and add an Environment shader of type Environment where I will be connecking my HDR image.


## Shading - Mia_Material vs MantraSurface

And in a similar fashion to before I click on the image connector, browse for my HDR and bring it to the scene.

The environment PPG is different from the image PPG as you will expect and I will leave it like that but my transformations are there to be tweaked.


## Shading - Mia_Material vs MantraSurface

I now go to the render options and kurn on my 'final gathering' function so now I get global illumination working.

And we are prelly much sel to start working on the materials.


## Shading - Mia_Material vs MantraSurface

Tweak the shader diffuse parameter and suddenly looks a bit more like what I was after.

Also you will notice I removed the image clip and the imge nodes which were just to check the UVs.


## Shading - Mia_Material vs MantraSurface

Change my shader parameters to get some blurry reflections, less glossy, more samples...

And then $i$ insert a checker board on the reflectivity connector.


## Shading - Mia_Material vs MantraSurface

And tweak the paramters and also memorise the interactive render region so I can compare before and after, something essential in my workflow.


## Shading - Mia_Material vs MantraSurface

Now I define the metalic behavior by using the index of reflaction of a metal (25) plus turning on the BRDF flag.

The cut of the 'curtain line' you see is becuase I am looking at two things, on the left I see my previous definition, on the right the new one.


## Shading - Mia_material vs MantraSurface

I now remove the memory as I am happy with my kweaks and then proceed to up the subdivision level of my geomelry so I see a low resolukion un-subdivided mesh when working but I render a second subdivision.

As you can see the borders are very smooth now.

So how do you do this in Houdini?


## Shading - Mia_Material vs MantraSurface

The houdini way.
Start throwing a torus object into your scene.


## Shading - Mia_Material vs MantraSurface

And dive in to change the subdivisions like we did before but also set the primitive type to mesh so we get the same kind of polygon mesh we are used in XSI.


## Shading - Mia_Material vs MantraSurface

Now wire a uvtexture to create your UV properties and switch it to Polar like we did on XSI.

You will see the uvquickshade node is just so I can check the UV/s like I did in XSI.


## Shading - Mia_material vs MantraSurface

So I go back and change the primitive type to mesh so I get the kind of connection and parametrization I am after.

Now works as expected.


## Shading - Mia_Material vs MantraSurface

I disconnect the uvquickshade and leave there but click on the display/ render flag of the texture becuase that is what I want to display and render.
dont worry leaving things left around, they can be handy later and as long as you keep track of what is the things it will be ok.


## Shading - Mia_Material vs MantraSurface

Now I drag and drop the 'mantra surface' shader from the material palette to the viewport and as you can see it is referenced in the node material.

The actual material has been dropped in the global library, so you can edit it later on.

Now let's configure camera and environment.

## Shading - Mia_Material vs MantraSurface

So I get a camera, configure it for $1920 \times 1080$.

Then I bring an environment light as in Houdini that is the way to use HDR, they are lights.

And find the same texture I used before.


## Shading - Mia_Material vs MantraSurface

Because when I render I want to see the HDR (al least for now) I also lick the 'Render Light geometry' so the background is on.


## Shading - Mia_Material vs MantraSurface

Now go to my ROPs..
I configure my ROP (render pass) to use $P$ PR render engine (global illumination).


## Shading - Mia_material vs MantraSurface

Back to my objects..
So I switch to the camera view in the viewport and now we kurn on Houdini render region and voila, that is our render.


## Shading - Mia_material vs MantraSurface

Let's now recap on the shader by comparing both...


## Shading - Mia_Material vs MantraSurface

and continue looking...


## Shading - Mia_Material vs MantraSurface

and more...


## Shading - Mia_Material vs MantraSurface

and more...


## Shading - Mia_material vs MantraSurface

Now after comparing the base parameters between shaders let's dive into the checkerboard setup.

In this case and for illustration purposes I have done it via a premade texture but I will dive into a procedural version in a few seconds.

So this is pretty much identical to the XSI version.


## Shading - Mia_Material vs MantraSurface

In order to implement the multiple resolution (geometry approximation) than in XSI there are two ways, and I want to show you first the easy direct way.

Just go to your object level, and in render tick the 'Render Polygos As Subdivision'

Now the render is smooth...
But there is another approach, much more manual that you will use in this and many other situations.


## Shading - Mia_Material vs MantraSurface

The manual appraoch would mean for you to branch out the Torus tree and converting it to polygons so I can subdivide it (we need polys for this operation)

Then subdividing to get smooth result.

And finally creating two nulls named DISPLAY and another RENDER.

Now it is a matter of activating the Render flag on the RENDER node and the Display flag on the DISPLAY node.


## Shading - Mia_material vs MantraSurface

And voila, we have identical behavior.
By doing this I keep the animators happy as the performance is better but when rendering I make sure we have enough polygon resolution... of course I could set this up based on camera distance, effectively building a Level of Detail setup but I guess that is more advanced.


## Shading - Mia_Material vs MantraSurface

So if I dive inside the 'MantraSurface' shader this is how it is being implemented.

A bit daunting but once you get to know it is a goldmine that you can actually modify it and hack it to your taste.

In many occasions you will end up doing it so dont be affraid, Houdini really wants you to be in full control and all throughout the phylosophy promotes this so just give it a go.


## Shading - Mia_Material vs MantraSurface

So for example, in the diffuse computation you have a very simple setup that pipes lots of parameters into the diffuse calculation, then exports a variable so you can retrieve it on your ROP and generate a frame buffer. (more on this on the next tutorial)


## Shading - Mia_material vs MantraSurface

This is the reflection section and although a bit more complex due to the reflection mask functionality it is easy to pick up.

A key thing is that we are seeing the logic behind the interface!!!


## Shading - Mia_material vs MantraSurface

So let's hack it with a procedural checkerboard.

First we deactivate the 'Use Specular Map' so the shader does not use the jpeg texture I provided.


## Shading - Mia_material vs MantraSurface

Dive in and we proceed to locate where the Base reflection section is.


## Shading - Mia_material vs MantraSurface

Go to the shader that holds the definition, in this case the 'reflectColor' one.

And instead of using the text field which is a different type from the checker I simply plug it into the Base Color of the reflection.


## Shading - Mia_Material vs MantraSurface

This will create the three nodes required to construct the checkerboard.


## Shading - Mia_material vs MantraSurface

In the color mix we change the colors to black and white, indicating maximum reflectivity and minimum reflectivity.


## Shading - Mia_material vs MantraSurface

Like this...


## Shading - Mia_Material vs MantraSurface

And render it again... as you can see.. looks identical.


## Shading - Mia_Material vs MantraSurface

Now you have a glimpse of how materials work in Houdini and hopefulty will make you think that it is way simpler than you expected.

My feeling is that it is extremely familiar and therefore with a bit of practice it gets really quick.
Once you are comfortable you will ask yourself if it wouldnt be better to avoid computing all these things that you are not using, like SSS in a metalic alloy...
...well, that is the next step and you will soon optimize your renders much more than you would expect.

## Shading - AOVs

## Shading - AOVs

A common approach to rendering with mental ray has been to use framebuffers that allows you to render an image and generate extra 'passes' that can either be stored on the disk as a separate file or everything in one single file if you are using EXR.

In Houdini this has been the bread and butter of the whole operation since the roots of the software and the integration with RenderMan has been the best out there since its inception. In the world of Renderman this is called an 'arbitrary output variabe' a.k.a AOV.

The process is very similar and easy to operate so I will be succinct on


## Shading - AOVs

Houdini provides out of the box with a plethora of AOVs (framebuffer channels) you can use without any hassle and they are exposed in your ROP (render operator), like this;

Go to your ROP Propertiessoutput tabs and click on the Extra Image Planes.

Then select one you are after (in this case 'N (normal)' which will give you the object normals.

## Shading - AOVs

Now in your Render view you only need to select your variable to be displayed and that is it.

Now when you render this will be default be stored in the same EXR image you have generated.


## Shading - AOVs

And of course you may want to add as many as you want, for example if you want to recombine your image you can go ticking all the ones you are after.

## Shading - AOVs

Now the question that arises is what if I want an ID pass that I want to control?

For example, if you have a crowd you may want to color them based on costume and that way color correct the crowd's costume...

For this you will have to create your own variable in your shader and export it.

So we will dive into our new blue shader and you will see something scary like this but dont worry, we wont mess with it.

Now we drop a 'Parameter' node in the network.


## Shading - AOVs

And place it very close to the outputcollect so we are reminded of the purpose... just aethetics though.


## Shading - AOVs

And edit both the Label, the Name of the variable and the type to Color. In this case I select a red color...


## Shading - AOVs

## And select ALWAYS on the Export

 parameter.
## As you can see you can program logic into this export and even callback scripts to evaluate it.



## Shading - AOVs

As you can see as this is a parameter automatically has been promoted to the interface of the shader container, you could decide to reorganise the interface so it is more friendly but for the moment I will leave it there in the bottom of the interface.


## Shading - AOVs

And finally in our ROP we add a new image plane and call our variable 'id'.

Also we select the id display in our 'Render View' and you should get something like this.


## Shading - AOVs

We do the same for the other shader and we have our ID pass ready for production.


## Shading - AOVs

As you can see it is pretty trivial, the only thing you should be aware is that overtoading EXRs with tons of extra image planes will have other implications, specially regarding management of these extra framebuffers but also because the turning on and off of these AOVs may prove annoying and you will end up like many of us just rendering it again, which in turn will have huge space implications, etc...

My suggestion is to do all your image planes on a pass (ROP) that is superfast to compute instead of adding it to your beauty pass so if you have to re-render will take little time.

Have a look at these links for more information.

## WORKING WITH IMAGEPLANES

hetp://www,sidefx.com/docs/houdini13.0/render/deeprasters

## Shading - Ambient Occlusion

## Shading - Ambient Occlusion

I encounterd a dilema when using Houdini and this is due to the various approaches everybody uses in Houdini to compute Ambient Occlusion.

Normally, and you will see this in a number of tutorials and even Houdini documentation, it is handled as a light and there are a number of techniques ranging from environment map approaches to point clouds.

From what I have seen it is mostly due to performance but I really like the approach in softimage (mental ray really) and that is exactly what I am describing here.

A bit of history... Ambient Occlusion was first introduced at Siggraph by ILM as a cheap way to fake global illumination qualities into traditional rendering, it was very effective and everybody started to use it, some well, some wrong and since you can find it in many pipelines in incredible number of flavours.

I have been for many years going for a Physically Based Rendering approach and therefore I dont have much need from it other than to use it as a dirt control, or modulate the color and wear of a texture, or control reflections on certain areas on my shader rather than to mimic $\in I$.

Therefore I need the compute it on the shader, not as a light and this is my personal take on it.

## Shading - Ambient Occlusion

So in Softimage you would select your models and apply an Ambient Occlusion material...


## Shading - Ambient Occlusion

A new PPG for the material will appear and you proceed to tweak the parameters.

At the end of the day you can see on the render tree that it is simply a shader.


## Shading - Ambient Occlusion

And with the render region you fine tune all the settings.


## Shading - Ambient Occlusion

In Houdini is pretty much identical... except that there is no preset ready, you can create yours quickly and added to the gallery but that will be explained later.

So we put down a 'Material Shader Builder' which is basically an emply material and rename it to ' $a o^{\prime}$ '.


## Shading - Ambient Occlusion

The content is a template for you to start quickly, first the surface and displacement globals which are provided by the object for free.

Then a surface output and a displacement output too linked to the output collect node.

In this example we dont need the displacement so we can delete it.


## Shading - Ambient Occlusion

So now I just bring a 'occlusion' node and connect it like this.

I connect the surface position to the position in the occlusion shader.

I connect the surface normal to the surface normal of the occlusion shader.

And finally the ambient occlusion result to the surface color.

And that is it.


## Shading - Ambient Occlusion

Add a floor I forgot to add...
Now inspecting the parameters of the occlusion shader I change the Background color to white, find the Maximum Ray Distance and the Number of samples which I want to promote so the users have an easy time.


## Shading - Ambient Occlusion

And proceed to put the names as I want them for both the Label the users sees and the actual name of the parameter.


## Shading - Ambient Occlusion

And finally add the values I am after and turn the Render View window and render it to see how it looks.


## Shading - Ambient Occlusion

The one thing I havent seen documented is the image looks too grey, the white is far from white so I tend to push the Background color to 2 which makes it look very very close to white.


## Shading - Ambient Occlusion

On the documentation there are a few other examples using Lights so I will encourage you to have a look, I am sure you may want to use that approach too.

## LIGHTING AMBIENT OCCUSION

hEtp://www.sidefx.com/docs/houdini12.1/light/ambientocclusion

## Shading - Material Libraries

## Shading - Material Libraries

When facing complex jobs the best strategy is to divide in tiny chunks and a typical tool I have used in the past with XSI is the use of referenced material libraries where one artists 'owns' the library of materials and curates every single one, making sure they are coherent with each other (meaning they play well together under any lighting).

To achieve this in XSI you can structure an external material library to be read as a reference and every single external model (asset) to use these materials so if the 'curator' decides to tweak one of these materials it will be propagated across the job.

Of course this can be very dangerous as you at risk, not only of breaking the renders on your farm, but even more important, of breaking previous states of your scenes.

One of the most important rules I impose myself, the rollback rule, is that I should be able to open an old scene and everything should work without me doing anything at all.
of course we can structure a similiar system in Houdini although it is not pre-structured for us, you have to do it and the best way is to define a digital asset to hold your materials.

Furthermore, because assets can be versioned you have your roll-back rule intact and happy times ahead.

So how do you do all this? The idea is to have an asset that contains your material library, subdivided into various types if you want (metals, plastics, liquids, stones, etc...) and a set of shaders properly named for you to point to.

Because your asset can live locally you can always author and tweak/hack the situation if necessary so you are in an even better situation than you were in XSI.

## Shading - Material Libraries

A Eypical approach in XSI regarding material libraries would be to create these libraries and define a ton of materials in each as you see fit.

Over the years I buitt a big library that can handle $90 \%$ of a job making my life very easy.

As I am now working exclusively with Houdini I have done the same for Houdini and I am in the process of redoing all the materials but as you can imagine this will take me a long time as I had to start from scratch.
(Note the one in italics 'METALS' is our current library so any material we create will be stored in it)


## Shading - Material Libraries

The library interface looks like this for woods for example.


## Shading - Material Libraries

And some materials are a bit more convoluted than others but I try to just define the basic shader, not specific applications, otherwise it would be so big of a task I could not manage it all by my own.


## Shading－Material Libraries

With every iteration I export the library as a dotXSI format（binary） for the project and everybody hooks to it．

Then is just painting by numbers．．． kind of．
©－Images T－Images
Materials
由－GLASS
List
円 ${ }^{( }$ALLOYS
T－CARPAINTS
－CERAMICS
me＿bonechina＿Materials：CERAMICS
mr＿carbon＜N
－mr＿carbon＿fibr
［目 carbon＿fibe－CERAMICS
［1］carbon＿ma，Name
－1 coating
II Mix 2 Col
Scalar＿Cha
Scalar＿Imag
Scalarimag
mr＿ceramic＿sandpaper＜Not used＞
mr＿ceramic＿used＜Not used＞
T．mr＿ceramic＿used＜Not
mr＿plaster＜Not used＞
mr＿porcelain＿glazed＿old＜Not used＞

## Shading - Material Libraries

To translate this concept in Houdini you may want to use it this other way as the concept of library really is not the same but we can easily replicate it.

My suggestion is to assume a shading network 'shop network' is your library and as you can have as many 'shot networks' inside an object you can wrap it in that manner to make it easy to deploy in a big scale situation.

In this image you can see a normal object that will become my portable library.


## Shading - Material Libraries

Then I create my shop networks as my libraries containing families of materials.


## Shading - Material Libraries

And inside I store all my materials ready to be used.
And again, like in XSI I dont try to build the perfect material with layers, dirt, grease, etc... these are all about material characteristics, $i$ leave the finish, damage and weathering for the project as it would be a monumental task.


## Shading - Material Libraries

Now is time to store as a HDA (houdini digital asset) that will be installed studio wide.

First of all we convert the object into a Digital Asset...


## Shading - Material Libraries

Then we define name and label, also where it is going to be stored and the minimum and maximum number of inputs.

Where to save it is important but you can later move it wherever you want.

My approach is to have a new folder added to the search path when you start the software. This way Houdini will scan for new Digital Assets and load them if found.

If you update Houdini version the only thing to do is add the folder path to the environment variable and you will be set in no time.

## Shading - Material Libraries

After accepting a big panel to configure the asset, its behavior, interface, etc... will be in display and I tend to hide the parameters making it a blank interface as this is simply a library.


Make folder invisible

## Shading - Material Libraries

Now that I have defined the asset it is time to save it and as you can see the name is red, this means the definition is local and I should save it.

So to save it I select 'Save Operator Type', now it is on disk.


## Shading - Material Libraries

...and once it is saved I 'Match Current Definition' so the material library name will be blue meaning I am using exactly the same version it is on disk.

The asset is locked and unless I unlock it allowing its editing I will be using the definition from disk.

If I save the scene and someone changes the definition of a material I will inherit that for free.

To manage the needs from production you can version your assets and go up and down so you can retain a level of certainty and safety in case of a roll back.


## Shading - Material Libraries

The details on how to use them (in all its glory), how to configure and add logic and scripts to them is well documented so I wont go into detail and let you read the documentation which is very good.

## OTL.s

hetp://www.sidefx.com/docs/houdini13.0/assets/
OTL.S MANAGEMENT
http://www,sidefx.com/docs/houdini13.0/assets/install
OTL.s NAMESPACES
http://www.sidefx.com/docs/houdini13.0/assets/namespaces
ADVANCED OTLS
http://www.sidefx.com/docs/houdini13.0/assets/advanced_otl

## Lighting

## Ligthing - Types of Lights

## Ligthing - Types of lights

The three key components of any scene are materials (includes shading and textures), cameras and lights.

We have seen a bit of the PBR material set and discussed the houdini camera that although is not a full model of a real world camera is okay (only VRay and Maxwell implement a proper camera model).

In this chapter I will go throught lights from teh poith of view of PBR so please bear in mind these settings may not be adequate if you use the micropolygon engine or others.

Ultimately you have to accept that PBR rendering is stower than other biased methods and therefore it only makes sense if it is used correctly.

Basically I use lights like a director of photography does, no tricks other than the ones he could replicate and this means;

- No light linking
- No point lights
- No lights without shadows
- No lights without physically correct attenuation
- No ramps or hacks of any kind
- No energy compensation

And although Houdini camera model does not include a photographic approach in the sense that there is no conversion from radiometric units to pixels the truth is that we can do a lot in the meantime.

Many of you wont agree with my approach, you are wrong, just happens you dont know yet.

## Ligthing - Types of lights

Lets first of all run through the basic configuration and setup of the scene.

In this scene I have created a surreal forest by using boxes and a grid.

Later I have put down a distant light so you can see one thing.


## Ligthing - Types of lights

If youturn off your light by cliking on the 'light enabled' option inside its parameter box you will see yoru render still works, although looks like we have a headlight.

This is because the ROP has a setting to create a virtual headlight in case there is any light turned on.

I normally turn this off as I am used to this behavior from Softimage and I want to make sure I am judging only the lights I have on the scene, nothing else.


## Ligthing - Types of lights

Just go to your ROP, then to objects and turn off the 'headlight creation'

Now your render view should be full black.


## Ligthing - Types of lights

Now that I have shown you this we can put down a spot light and start to play with some parameters.

In this case I have simply render the default light after lighting it up roughly to the previous light.


## Ligthing - Types of lights

And as you can see there is no attenuation, something I mentioned I am not going to allow.

Attenuation


## Ligthing - Types of lights

But of course now the intensity of the energy arriving to the floor has decreased based on the square distance.


## Ligthing - Types of lights

If you play with various values you will see the net effect is that position AND distance together with intensity are critcial and the decay is very visible.

If you look at the closet boxes you will see they get hotter and hotter while the distant ones dont, this is how tights behave in the real world and thus I want to capture that behavior.


Intensiky $=10$


Intensity $=20$


Intensity $=40$

## Ligthing - Types of lights

The second thing to notice is that the default behaviour is for the spot light to use a cone angle to cover a certain area, in the real world this is done using 'barn doors' but the computer model does not have this featuere so...

Well, that is something that for ilustration purposes I am going to keep on but at the end I will turn it off so you see the difference.


## Ligthing - Types of lights

So as we move into the world of PBR lights let's for a second fix a couple more things in our ROP configuration.

First of all let's up the sampling to see what we are getting into, in this example I move the 'pixel samples' to $6 \times 6$ so I get nice shadows.


## Ligthing - Types of lights

Then let's go to the shading and add one bounce to the difuse light by upping the value of the 'diffuse limit' to 1.

This will give you the effect of a fill light as the light bounces on the floor and columns making the shadow areas way more interesting.


## Ligthing - Types of lights

And now that we have this configured let's work on the area light we have defined and fix one more setting.

Do you notice the little square? well, in the real world the bigger the area, the more energy it contributes to the scene...

## Ligthing - Types of lights

therefore if the light is bigger you should expect more light into the scene, and on the contrary, if the box is reduced, so it is the energy it contributes.

Again, this is how the world works, I am just sticking to it.


## Ligthing - Types of lights

To achieve this make sure you always lick off the 'normalize light intensily to area'


## Ligthing - Types of lights

Like this, on the left you have a source light with an intensity of $s 0$ and a small size of 5 m by sm, while on the right you have a ligth with the same intensity but a 1 m by 1 m .


Intensity $=50$, area size $1 \times 1$


Intensity $=50$, area size $5 \times 5$

## Ligthing - Types of Lights

But of course managing 20 lights like this may be quite a pain? well, the opposite is true given the distance in many cases are known, the sizes too, it is only the intensities that are not and therefore the ideal scenario is to have a way to balance all the lights easily from one single interface.

Welcome the Data tree where you can set it up in 'Light bank' mode and configure, solo, change mode for all the lights in the scene...

A dream come true for a true light designer, this is the ultimate control room.


## Ligthing - Types of Lights

Now, what did happen with the barn doors???
Well, if we now go to the Spot light options tab and turn off the 'enable spot light' you will see the effect of having an emissive object.

In the real world one is called a spot light, the other a diffusor and it is up to you how do you want to use them.

Another option of course is to put a piece of geometry 'black' to act as real barns but of course this will mean longer render times.


## Ligthing - Types of lights

The details on how to use them (in all its glory), how to configure and add logic and scripts to them is well documented so I wont go into detail and let you read the documentation which is very good.

LIGHTING<br>http://www.sidefx.com/docs/houdini13.0/render/lights<br>PBR RENDERING<br>hetp://www.sidefx.com/docs/houdini13.0/render/pbr<br>RENDER QUALITY AND SPEED<br>hetp://www,sidefx.com/docs/houdini13.0/render/quality<br>LIGHT BANK<br>hetp://www.sidefx.com/docs/houdini13.0/render/lightbank

## Ligthing - Real world lights

## Ligthing - Real world lights

One thing that strikes me as curious is that many artists throw lights without diving into how things are done for real, this includes sculpture, stop motion, cel animation and specially lighting.

So in this chapter I just want to reflect on the real world and which things you should keep an eye.

For this we should make a distinction between;

- Interior lighting
- Exterior lighting
- Studio Lighting

The reason is that these categories are massively different and various luminaire manufacturers product information would be very valuable as they release true data from them from which we can learn a lot.

## INTERIOR LIGHTING

These tend to be consumer lights or professional (retail)
lights that are focused on readability and color accuracy.
A manufacturer to keep in mind is PHILLIPS.

## EXTERIOR LICHTING

These tend to be professional lights focusing more on cost, power than color accuracy or readability.
Also you have exterior consumer lights focusing on decorative aspects, these are more color accurate at the expense of cost and energy consumption.
A manufacturer to keep in mind is PHILLIPS.

## STUDIO LIGHTING

These lights are all about color accuracy, control and power so costs may be crazy in our eyes but getting a room like is lit by the sun itself is always going to be expensive. A manufacturer to keep in mind is ARRILIGHT.

## Ligthing - Real world Lights - Luminaires

The first step is to discuss the luminaire, meaning, the device that produces the light; an incandescent light, a fluorescent, a LED Light, a halogen light, etc...

All of these have a signature effect and they are used for certain type of tasks, for example, you dont want to use fluorescents that emit really ugly light for beauty products, instead, you will go for halogen lights or the latest LED lights.

These are the categories from the PHILLIPS website.


## Ligthing - Real world lights - Interior

How we use these different types of Luminaires on interiors is another thing and this insight is the key to get realistic architectural rendering.

The casing and how they are installed respond to artistic and practical decisions, from wall mounted to suspended the knowing the reasons will give you a beliavable setup.

I will let you go and dive into it though.
Light line systems

## Ligthing - Real world Lights - Exterior

And the same goes for exterior lighting, it is the practical and artistic decisions that will drive which Luminaire to use where.


## Ligthing - Real world lights - Studio

With studio lighting you will deal with artistic decisions mostly and these are designed taking in account many other considerations but are very much based on the same principles, only immensely more powerful and accurate.

Make sure you download this PDF and study it.
ARRILIGHT STUDIO LIGHTING PRODUCT LINE
hetp://www,arri.com/?eID=registration $\$$ file_uid $=11326$
On how they use these lights on studio you could write a few books but there are a few key elements you need to remember.

# Ligthing - Real world lights - Studio 

## SOFTBOX

These are lights in which we have placed a piece of translucent material between the light and the subject to make the shadows softer.

DIFFUSERS
These are lights projected to an umbrella with a reflective material making a key light very wide and very diffuse. A very flexible tool.

## REFLECTORS

These are white (silver or gold sometimes) objects to make the light bounce in a natural way and fill areas to reveal the shape.

## FLAGS

These are black objects to suck the light and avoid bounces, also with the intention of reveal the shape.


## Ligthing - Real world lights - Studio

So the question is how to recreate these various objects to play around in our scenes, answer lies on some fundamental elements regarding how lights have been constructed in Houdini.

Let's try to construct a true sofbox which is probably the most complex one and the one you want to save in your SHELF for easy access.

Starting with out light scene I just changed the area size of our light to $3 \times 3$, the intensity to 25 .


## Ligthing - Real world lights - Studio

First the light rig...
Also I create a geometry object, put down a grid operator and configure it to size $1 \times 1$, rows $3 \times 3$ and $X Y$ plane as I know I will have to set this in this manner once I parent it to the light.


## Ligthing - Real world lights - Studio

Now by parenting my geometry matches the position and rotation of my light...


## Ligthing - Real world lights - Studio

And proceed to match the size of my area light to the size of my unit grid.

So I simply copy the parameters for the area light size...


## Ligthing - Real world lights - Studio

And in at object level I apply the expression to the scale of the object so now my piece of geometry will match also the size of the area light.


## Ligthing - Real world lights - Studio

Like this...


## Ligthing - Real world lights - Studio

...so I go back to my $3 \times 3$ size again and proceed to configure my softbox.

First I refer my geometry object parameter in the light to point to my real geometry.

Then I select a constant material I have prepared that has as a texture an HDRI texture of a softbox and wire it all up in the material parameter inside the light.


## Ligthing - Real world lights - Studio

And set the lighting to be single sided and self shadow...


## Ligthing - Real world lights - Studio

But if I render from another angle two strange things are happening, the texture seems repeated $2 \times 2$ and there is a white artifact (band) in there...


## Ligthing - Real world lights - Studio

There are two issues, the artifact it is simply that we have 2 copies of the grid, one for the light object that refers to the geometry and the geometry as an object itself.

The solution is to hide the geometry object so it is not visible nor rendered.


## Ligthing - Real world lights - Studio

The repetition is due to the geometry being subdivided so it is only a matter of going down to the grid and change the rows to $2 \times 2$ instead.


## Ligthing - Real world lights - Studio

Now I finish the light setup by turning off the 'Enable Spot Light' and this is a softbox for you.

Clearly too much work, this is something you should be able to point and click...


## Ligthing - Real world lights - Studio

So let's put this mini-recipe into the shelf, that way next time we open houdini it truly will be a one button thing.

I start by tidying up the names, the light is now called 'softbox' and the geomery is called 'softbox geo'.


## Ligthing - Real world lights - Studio

Then I create a new shelf and name it as 'pbrlighting' and a label 'PER Lighting'

## Ligthing - Real world Lights - Studio

And drag and drop these two nodes into the emply shelf...

And once the popup appears I give them some names for readability but also I give it the icon of another shelf tool that will help tell the story, in this case 'OBJ_light..area'.


## Ligthing - Real world lights - Studio

Delete my lights... (you may want to save the scene first just in case)


## Ligthing - Real world lights - Studio

And now by clicking on the new icon I get my softbox... for free from now on.


## Ligthing - Real world lights - Studio

So as you can see modeling these types of behaviours is possible and once done once you can store it for later and distribute to your friends making your workflow really efficient.

ARRI LICHTINE<br>hktp://www, arri.com/lighting/lighking equipmenk/<br>KINOFLO<br>hekp://www kinoflo.com<br>PHILLIPS<br>hetp://www.ecak.Lighting philips.co.uk/l/<br>ERCO<br>hktp://wuwierco.com/homepage/home/en us/

## Ligthing - Environment Lights

## Ligthing - Environment Lights

One of the tools we give for granted are environment maps (xsi terminology) or environment lights (houdini terminoloty).

Although this little guide may seem unnecessary there are a few things to consider to make sure your life is easier so please bear with me if it seems obvious at first.

So let's start with the scene we were working and remove the softbox and put down an environment light node which will hold our HDRI panorama.


## Ligthing - Environment Lights

If you bother to render it as it is you will notice you will get an occlusion pass straight away and if this is what you are after, great, but as I mentioned on the Occlusion guide I like to use it for other things and therefor tend to do shader based occlusion.

But I digress...


## Ligthing - Environment Lights

So I simply load the texture I am interested, in this case a $4 K$ hdr map I downloaded from HDRILABS.COM.

Render and I turn on the flag to 'Render Light Geometry' so I can see the background on the render view as lining up things is very important.


## Ligthing - Environment Lights

But the problem is that having to render a big scene just to line up the environment is pretty annoying so I would like to have a visual reference.

What I am interested is doing the same thing I was doing in XSI.

Simply load the texture on a sphere, set the Texture Projection to Spherical XY (by clicking on UVs) and as you can see the texture support icon is now shown in dark green surrounding the object so I can move arround until I am happy.


## Ligthing - Environment Lights

Kind of...
If I add an environment map to the pass and select the same image...


## Ligthing - Environment Lights

And inspect the image I normally want to link the 'Transformation' Rotations of the texture support rotations, or viceversa... but having a direct representation is very very handy.


## Ligthing - Environment Lights

If I dare to put a chrome like material to the sphere and render and then compare with the viewport texture I will notice these images do not line up at all... so I will have to compensate for the transformations of the environment vs. the sphere.

First by negating the $X$ scaling on the Image clip plus adding 180 degrees to the transformation.

Now they should match.


## Ligthing - Environment Lights

How to do this in Houdini?

Well, it is pretty much identical although more interesting...

Bring a sphere to the scene (i made it bigger to see it well) and also convert the sphere type to polygons.


## Ligthing - Environment Lights

Added a 'uvproject' node and linked the rotation of the 'environment light' node to the rotation of the 'uvproject' so now if I rotate the light I will be rotating also the texture projection.

Also, in order to visualize $i$ drop a 'uvquickshade' node so I can see it in the viewport without rendering.


## Ligthing - Environment Lights

And point the texture to pick the same image than the light, something I could do with an expression like we have seen before but I will not do it this time.


## Ligthing - Environment Lights

As you can see these dont line up well with the render.

I have rendered the 'uvquickshade' and then applied a chrome material to the sphere to see the reflection and also rendered.

The comparison tool in the viewport (very much like in XSI rende region buffers tool) allows me to do nice evaluations of previous current renders.

Forget about the brightness and scale, the important thing is the features are very different.

Again I have used the DISPLAY/ RENDER trick on the network.


## Ligthing - Environment Lights

What I need to do is negate $X$ as shown in the image and also add a 180 degrees rolalion.

The images now will line up.


## Ligthing - Environment Lights

As you can see both viewport and render are referring to the same features, the dark wood panel, the white table, the windows on the white wall on the left... It is by no means perfect but it is a good guide so we line up things fast.


## Ligthing - Environment Lights

There is plenty more and many more ways to deal with environment maps but I guess these are the things you want to look at to know more about the subject.

## ENVIRONMENT LIGHT

hetp://www.sidefx.com/docs/houdini13.0/Modes/obj/envlight
ENVIRONMENT AND REFLECTION MAPS
htep://www.sidefx.com/docs/houdini13.0/render/envmaps
ISIXPACK UTILITY
http://www.sidefx.com/docs/houdini13.0/ref/utils/isixpack
PORTAL LIGHTS
hetp://www,sidefx.com/docs/houdini13.0/shelf/portallight
LIGHTING
htep://www.sidefx.com/docs/houdini13.0/render/lights

## Ligthing - Photometric Lights

## Ligthing - Photometric Lights

Coming from the world of architecture is only natural I am fascinated by light and accuracy to represent the space and surfaces in a realistic manner.

Not only that, I would like the idea of measuring and talking about meaningful values by means of having units rather than obscure numbers.

In the case of lighting the best solution we have is to use photometric lights data from the manufacturers that try to capture the "signature" of these lights by measuring two stices and building a diagram.

My natural interest is to use the manufacturer data inside Houdini and benefit from the huge potential to create realistic renders.

So the intention with this guide is to show you how to use IES profiles in your scenes and analyse a few aspects relate to them.

But first of all, make sure you downtoad some IES profiles and ideally also a IES viewer.

IES VIEWER
hetp://www, photometricviewer.com
LITHONIA IES DATA DOWNLOAD
hetp://www.Lithonia.com/photometrics.aspx

## Ligthing - Photometric Lights

The anatomy of an IES profile is very simple and most importantly is text based so it is human readable too.

You have the IESNA luminaire reference number, the test applied, the manufacturer (in this case ARRI), the type of luminaire, temperature, type, angle, the lamp characteristics with the lumens produced and Wattage and you may see the manufacturer adding some extra fields.

Then two series of numbers describing the two profiles for that particular luminaire.


## Ligthing - Photometric Lights

That same file represented as a diagram using the IES Viewer will look like this.

The red lobe is the signature of the light and using teh IES Viewer you have a range of options to explore the light profile.


## Ligthing - Photometric Lights

Even a fast profile visualization that will help you get a pictureo of what the file describes.


## Ligthing - Photometric Lights

Let's do this in softimage first as this is one of the examples in which Houdini's way is much easier.

So we bring a point light for example, go to the 'rendertree' and search for the 'mia photometric_light' and connect it to the light shader leaving the default 'soft_light' disconnected.


## Ligthing - Photometric Lights

In the PPG you want to also change the defaults to use the IES data instead so change the 'intensity' and the 'distribution' to IES Profile.

As you can see the profile graph is shown in a tiny icon which is very helpful along with the text file of the description.. something even more handy as you can inspect the manufacturer informaiton there quite quickly.


## Ligthing - Photometric Lights

And the the one thing I love is that the profile lobe is recreated in the icon of the light, giving a good quide of where this point light is pointing to.

As you can see on the render region the result is quite something, this is no longer a boring cg looking light, this has a very particular profile, this light is a decorative interior light so of course is much nicer.


## Ligthing - Photometric Lights

And if I move the light away from the floor and rotate it a bit you can easily tell this is way more interesting than cg lights.

The only drawback I see is that the way it is implemented they are quite heavy on the scene so when you add a few of those the renders can go really high.


## Ligthing - Photometric Lights

Now in Houdini...

Let's put down a spot light as it is easier to manipulate once we have a selup.


## Ligthing - Photometric Lights

Then Let's select our IES profile data file and making sure it does not contain illegal characters in the name. In my case I have simply renamed it to 'profile.ies'.

Then make sure your 'cone angle' is 360 as you dont want to clip the data coming from the manufacturer and no 'cone delta' at all.

You now can see the signature of the light.


## Ligthing - Photometric Lights

If now we go to our forest scene and I change the light position you will see the dramatic improvement in the light being trown to the floor and the effect on the cubes.

It is so much rich it is pretty incredible difference.


## Ligthing - Environment Lights

This is something you may want to explore further, may be even building a custom tool to show the profile of the light to help visualizing it. Here are are few important links to keep around.

ILLUMINATING ENGINEERING SOCIETY
hetp://www.iesha.org/
IES VIEWER (Windows)
hetp://wwwiphotometricviewer.com
VARIOUS IES PROFILES DATA
http://www,usa.lighting,philips.com/connect/tools_literature/photometric_data_1_b.upd
hktp://genet.gelighting.com/LightProducks/Dispatcher?REQUEST=IESCATEGORYPAGE
hetp://www.Lumec.com/downloads/IES..files.html
hetp://www.Lithonia.com/photometrics.aspx
STUDIO LIGHTING IES PROFILE DATA
hetp://www,arri.com/lighting/lighting_equipment/dowhloads/?
suchoption 1 id=Product Information \$suchoption2.id=LED Lampheads\&suchoption 3 _id=L-Series hetp://www,kinoflo.com/Products\%2OButton/Fixtures\%2OBuilt-In/Diva-Lite/Photometrics.html

## Digital Assels

# Digital Assets - Introduction 

## Digital Assets - Introduction

Digital Assets is quite a subject to take on but I dont aspire to write the definitive guide so I will start with some clarifications about the terminology.

What is an asset exactly? an asset can be seen in many lights but it is meant to be a self contained piece of geometry or tool.

With Houdini it takes a new meaning as the architecture allows you to include 'intelligence' and costomization like you could not dream of with other tools, specially since you can embed very easily scripts, logic, controls, documentation and the procedural nature of the asset and its design will allow you to construct extremely sophisticated worlds in a very compartmentalised manner.

Imagine a city where very single building is a variation of one asset that has been defined procedurally, from textures, color correction, shaders, geometry...

Now imagine the street being another asset, the layout, the cars, the characters... ultimately promotes the division of work although it is up to you to define how do you want to structure it of course.

There is no real need for scene assembly toolset and reinventing the wheel with every job although you most certainly can build on top of it to handle extremely large jobs.

So how this compare to Softimage? well the idea of tool as an asset really does not exist, the only 'asset' really is the model and although very powerful you will see a new world out there when using Houdini.

## Digital Assets - Introduction

So let's recap on how external models work in Softimage first and then expand from there.

In Softimage 'Models' are a way to encapsulate geometry and a few other things as if they were miniscenes.

These can be saved and used as a reference and are specially useful to divide the work of a team and speed up the overall process although as you can imagine coordination becomes much more important between these individuals.


## Digital Assets - Introduction

In this example this character has, not only the mesh but also a series of animation clips where animation curves are stored.

These clips can be put to good use in the animation mixer and edit the motion very much as if you were working on final cut or avid editorial tools, slicing, mixing, layering...

Also these clips can store constraints, shape animation, audio and containers of clips can be built which is extremely useful for motion capture editing.


## Digital Assets - Introduction

Here you can see the same model now loaded as a reference which is able to hold multiple resolutions (one active at a time) and has the ability to run events to automate further some of the aspects of the production.

Having modified the model (Let's say keyframing it) is not lost when the model is offloaded making it possible to have very very lightweight scenes in which may be only two or three models are loaded and the rest is not.

Using this technique I have managed to put together some huge shots and over the time has been very stable.


## Digital Assets - Introduction

As you can imagine coordination is essential when you have a team of 10 people working on various assets that have to be later deployed in a production and altered per shot, for this many studios have built a few tools to manage the export, the import, assemble scenes automatically and what not.. ultimately it is up to you to define how you want to use them but the functionality they give you is very important.

Now in houdini you will face a few quetions too...
The core idea I distilled, and this may be my point of view only but it just makes sense for me, is that 'Houdini Digital Assets' (HDAs) are abstract entities and not just models with a few things inside.

If in Softimage 'Models' get used as way to split the objects without any inherent intelligence, in houdini they are used as tools that have their own interface, logic, geometry, animation, motion analysis, compositing, dynamics, rendering...

This will blow your mind because ultimately you can now divide the scene into pieces but also the intelligence of the scene making your master scene be just a simple container of assets.

Of course, with great power comes great responsibility and you can make everybody's life a pleasure or hell... up to you.

## Digital Assets - Introduction

So for example, you could create a character asset that contains a rig, and references to geometry, textures and animation from disk like you would do in Softimage...

Or you could build an asset that preprocesses the geometry that is procedural in the sense that if you change the topology it can reconfigure itself (for example, polygon cleanup, cluster reassignment, weight reassignment, hair reconfiguration, etc...) but also can expose to the user multi-resolution generated procedurally, motion capture selection, that can render itself isolated of the environment on a turntable and even precomposite itself into a branded image that makes you look like a superpro.

The bad news is that it is easy for you to have too much fun and start over-designing and over-engineering it, this is the only thing I will suggest... start small.

## Digital Assets - Introduction

In this HDA of a character called 'guy' you can see the interface where a bunch of controls are provided to configure the actions.

The idea of this particular asset without diving into a massive amount of detail for an introduction is to be able to select the type of mesh that will be rendered, if it is visible or not, the configuration of props required based on the type of character and on a second part the motion capture to be used.

Instead of having a mixer where you load clips I am selecting the right ones from a library that has been authored before avoiding any confusion to a team.

Furthermore, animation manipulation and texture selection are also procedural and therefor it is trivial to put together a crowd without having to dive inside this asset.


## Digital Assets - Introduction

The authoring scene would load all the assets to be exported in 'bclip' format to form a library (a bunch of files on disk)

The mokion preprocessing, cleanup and gimbal lock corrections happen here so we end up exporting edited data that is both optimized (no more animation curves than necessary) and corrected to feet are in the same place for perfect transitions.


## Digital Assets - Introduction

Once the library is ready the asset with the skeleton and the mesh deformed by it will be animated by loading in chops the data and exporting it to the bones channels, in a case like this one everything except the hips is rotation based so it is later very light.


## Digital Assets - Introduction

The whole structure will be pretty similar to softimage if you look at it from the schematic.


## Digital Assets - Introduction

The real power though comes from the fact that the actual asset logic (what to display, how to do things) is embedded in it and the control is totally independent from the scene.

For example:
I deform the mesh and add a switch to show or not the geometry, driven by the parameter you saw on the interface.

Then I switch the CL color to make sure I can see on the interface which character are Michael, which characters are Dave.

And switch based on the parameter for the actor on the parameter interface.

Finally I pass it to an null node as we discusses so I can pick that data somewhere else later.


## Digital Assets - Introduction

So what we just saw was a tiny glimpse of what you can do, in this case I built the intelligence on the character so layout would be trivial and choreographing loks of characters very simple task.

HOUDINI DIGITAL ASSETS DOCUMENTATION<br>hetp://www, sidefx.com/docs/houdini13.0/assels/<br>hetp://www,sidefx.com/docs/houdini13.0/hom/assetscripts<br>HOUDINI DIGITAL ASSETS FOR HOUDINI ENGINE<br>hktp://www, sidefx.com/index.php?opkion=com_conkenk\&kask=view $\$ i d=2625 \$$ Itemid $=66$<br>SIDE EFFECTS HDA WEBINAR<br>hklp://www,sidefx.com/index.php?option=com conkent\&lask=view \&id=2042\&Ilemid=354<br>SIDE EFFECTS HDA INTRO<br>help://www,sidefx.com/index.php?oplion=com_content\&leask=view \&id=2166 $\ddagger$ Itemid=361<br>SIDE EFFECTS AND GAME AUTHORING<br>hktp://www.sidefx.com/index.php?option=com_conkenk $\$$ lask=view $\$ i d=2017 \$$ Itemid $=364$

# Digital Assets - Creaking an HDA 

## Digital Assets - Creating an HDA

I wanted to do a tiny brief section to discuss how to approach Digital Assets in a way that can grow with you...

The point is that given that HDAs have been designed with the power to include scripts, Logic, etc... it is obvious that your tools, those that you want to share with your coworkers are the perfect candidate to end up as an HDA.

But also when you break up a scene into elements that can be distributed like we just saw these models and all their properties and intelligence are candiates to end up as an HDA as well.

The point I want to make is that Houdini does not make a distinction and is up to you to find out how to, not only architect them, but use them in a way that it is clear to the user what is a tool and what is not.

For example, HDAs allow you to version them and deployment to a team can benefit enormously from it, but there is a limitation that makes using versioning of HDAs much more interesting for tools than for models.

The same way you want do auto-refresh those HDAs to your team for tools but not for models as you may not want to break the ability to roll back to a previous version.

## Digital Assets - Creating an HDA

First let's recap on fundamental terminology as you are going to hear about OTL.s, HDAs, etc...

HOUDINI DIGITAL ASSET (HDA)
This is a definition of a custom type. Basically, this is your custom tool serving whatever purpose you want.

OPERATOR TYPE LIBRARY (OTL)
This is a library of definitions, one file with extension 'otl' can store multiple definitions of multiple assets.

NODES ON YOUR SCENE
These are instances of the definition you have stored in the otl file.

The picture then is that you can store inside one otl multiple tools and its version, hence the term library, and to manage it all you have functionality inside Houdini to organise things the way you want by means of the 'operator type manager.

You can find the properties of a HDAs by using the 'Type properties tool'.

Furthermore, the asset you put on your network is an instance of the asset and as such is locked, if you want to change the internals you will have to first unlock it, then tweak it and if you are happy and want to save it you will also want to save the new definition inside your type library (otl). After that you may want to synchronise the node on your scene (the instance that is still unlocked) with the one you just saved, forcing an automatic reload and lock of the asset.

## Digital Assets - Creating an HDA

So let's run through a small exercise to understand the process better.

First let's define an asset...
Starting with the Wassili Chair from chapter 1 (page 90) we are going to put together an asset of this model that could be deployed on a scene for both rendering and layout.


## Digital Assets - Creating an HDA

Let's remember the ingest node does a preparation of the model for later use, I chose to keep all this network alive and export three branches.


## Digital Assets - Creating an HDA

On the processed model (probably not a nice name but let's carry on) I pick these objects as separate streams, add materials and merge them together in two flavours.

1) One branch is the RENDER object that will be setup to render as I have the rende flag on the null node, a heavy detailed mesh of 60000 polys.
2) Another brand is the DISPLAY object that has been polyreduced so the final mesh for the vieuport is much lighter and makes the layout artist life much easier of around 1300 polys.


## Digital Assets - Creaking an HDA

So we just wrap it under a subnet by selecting both nodes and then clicking on the box icon.
(This produces the same effect you would get in softimage when you select a few objects and creates a new model)

And rename it as you like... in my case as I plan to have many types of chairs on my library I rather categorise them using the name so they appear all neally together.


## Digital Assets - Creating an HDA

The actual scenario is that the shaders are in the global library (/) shops) so I create a local library inside the model and rename it to materials.

Then I copy/paste the materials from one library to another.


## Digital Assets - Creating an HDA

And relink these in the material nodes to the local library as a relative path this time making it bullet-proof to renaming.

Then I delete the materials from the global library so I dont link things by mistake.


## Digital Assets - Creating an HDA

So now the next step is to provide a way for animators and layout artists to control the position of the chair object, for this I parent a null to the chair resulting model and I do it in OBJ context for performance as this could easily go inside the object itself (SOP context)

The positioning and look of the icon is defined in the 'Misc' tab of the node so I just configured as I wanted.


## Digital Assets - Creating an HDA

So now I go the scene level (/obj) and right click and select the 'Create Digital Asset' and will convert my subnetwork onto an asset.


## Digital Assets - Creaking an HDA

So now I go the scene level (/obj) and right click and select the 'Create Digital Asset' and will convert my subnetwork onto an asset.

A pop-up appears and I configure the operator name adding a prefix 'model' in order to group all my model assets and separate them clearly from my tool assets.

Then add version one at the end of the operator name which will allow me to create new versions.

And I will store this chair definition and all the subsequent versions onto one single OTL file named like the chair and also add 'OBJ' at the end to make it obvious that this oft is for the OBJ context.


## Digital Assets - Creating an HDA

A second window appears now that is the definition of the operator, here you can find everything you need, from icon, how to access the node, parameters, custom code, tools... you name it.

For the moment I will do small cosmetic changes.


## Digital Assets - Creating an HDA

First of all I will allow the user editing the control I did create to position the object, but nothing else, so without unlocking the asset you can place it where you want, but nothing else.

This window appear because I pressed on the icon for 'Editable Nodes'


## Digital Assets - Creating an HDA

And I will leave it like this for the moment...


## Digital Assets - Creating an HDA

Now I simply 'Save Operator Type', and once it is saved I 'Match Current Definition'.

The first step save it to the disk, the second synchronises the disk version with the scene making it identical.



## Digital Assets - Creating an HDA

Now if I press TAB and type chair you will notice our HDA is available on the pop up menu.


## Digital Assets - Creaking an HDA

The normal workflow after you create this asset is to installi it to a range of users while you keep working on version two which will add all the functionality and will be stored in the same oll.

Then your users will simply switch the oll to look at definition two and everybody happy.
Untike softimage you wont need to deal with deltas so everything is much more solid to start with.

HDA NAMESPACES<br>hetp://wuw, side fx.com/docs/houdini13.0/assets/namespaces<br>OPERATOR TYPE MANAGER<br>hetp://www.sidefx.com/docs/houdini13.0/ref/windows/optypemanager<br>OPERATOR TYPE PROPERTIES WINDOW<br>hetp://uww.sidefx.com/docs/houdini13.0/ref/windows/optype

## Digital Assets - Smart HDA

## Digital Assets - Smart HDA

Following the previous example it is obvious we should exploit the HDA potential and think about assets that know about how to configure themselves, providing tools for their manipulation is what I tend to call an Smart HDA.

So we are going to dive into the Chair we were playing with and add some functionality that will convert this apparently simple model into a smart asset that will open the door to really important concepts later on.

Contrary to softimage where you may be able to attach scripts to be run, the fact is that it was never designed to do it and you can see it.

The byproduct of that design approach is that if you want to add certain "intelligence" to he asset you end up building scripts that are installed on the software or even the scene by means of 'Custom Property Sets' to store information.

## We can do better than that...

The main idea I follow is to encapsulate all the data, documentation, tools and intelligence onto the asset itself, and all my assets have to have this base configuration so when I deal with a scene with 20 different assets the interface is practially the same.

Let's learn now...

## Digital Assets - Smart HDA

In this particular example I am planning to setup my materials so I can do quick layout rendering and animation blocking, I dont particularly want to carry the extra burden of very heavy shaders on my render when all I want is to evaluate the animation and continuity so every single asset in the production will have a switch to over-ride the materials to a different material.

In this particular case will be a simple lambert shader but you could set it up to separate these in better and more clever ways.


## Digital Assets - Smart HDA

So instead of doing a true override what $i$ will do is apply on the ingest a grey shader for later bypass the shader assignment.

First let's change the red shader to a grey one.


## Digital Assets - Smart HDA

As I may insert many nodes in the middle I will put a null and called 'bypass1' so I remember it does nothing other than splitting the flow in a more convenient way.

Also I drop a switch that I will use in a minute.

## Digital Assets - Smart HDA

Now I drop a switch node that receives as first parameter the material re-assigned, as a second dirckly what comes from the ingest network.


## Digital Assets - Smart HDA

And do it for all of them...


## Digital Assets - Smart HDA

So I edit the interface as I want my asset to have a nice configuration tab.


## Digital Assets - Smart HDA

Now we create a new tab called 'Config' where I tend to put all configuration information for an asset, then add a toggle parameter and rename it appropriately.


## Digital Assets - Smart HDA

## And select the other tabs and hide them by clicking on the

 'Invisible' parameter.This way only my tab will be visible.


## Digital Assets - Smart HDA

And now I copy the parameter I just created and use it as an expression on the switch node, the point is that this is a true/ false node that has value ' $o$ ' but when turned on returns the value ' 1 ', this is then used in the switch to select either o (my material override looking beautiful) or 1 (bypass the material override).

I tend to configure always the parameters in a manner that is consistent from the behaivour point of view, meaning to modify the default asset you need to perform an action.


## Digital Assets - Smart HDA

And do on all the switches so now you can see the chair is grey.


## Digital Assets - Smart HDA

And back on the asset I will save it and synchronise leaving the flag as off which is how I intend the users to see it when they drop the node.


## Digital Assets - Smart HDA

Although I may be a very simple guide the point is that the interface building tool can put all sorts of types of controls, from buttons to dropdowns, from integers to floats, ramps, angles...

Some fo these objects require some basic logic, other will have to have scripts attached that will call functions or directly execute existing functions on a node.

Let's take it a bit further by adding a rendering subnetwork inside...

The idea being the I want the asset to be able to render itself in its own lighting whatever is outside so I can build a simple library of assets in my company.

When I deptoy the asset on a scene this will be a nice way to test the asset is healthy and not textures are missing, etc...

Think of it as unit testing. ;-)

## Digital Assets - Smart HDA

So I create a new camera with a resolution of $500 \times 500$ and place it like in the image.

Also I create a light that I will be using to light this asset making it tokally independent of what is going on in the scene.

I place it to light it from the top left to the bottom right.


## Digital Assets - Smart HDA

Now I create a preview ROP subnetwork inside the asset itself.

If you were to imagine a XSI version of this (which is not possible) imagin the asset has its own render passet setup and scene properties.


## Digital Assets - Smart HDA

Then I create a manira node and refer to the camera inside the assel to be the one to be used.

I may loggle off the Show in Viewport menu' so I dont clubter my scene with kons of cameras in the viewport selection menu.

If I was populaking a stadium with this chair I will end up with thousands of those and you wont be happy... Eurn it off then.


## Digital Assets - Smart HDA

Now I avoid any object by removing the 'Candidate Objects' and referencing the object to render in the asset explicitly.

Also I force a solo light, the one inside the asset so no matter what I do this will be rendered with this configuration if the users chooses to do so.


## Digital Assets - Smart HDA

I drop a new HD camera just to illustrate the fact that the asset uses its own...


## Digital Assets - Smart HDA

And in the parameters window I add a new button that when pressed will execute the mantra render button.

To do this I attach a simple script in the menu to execute the mantra ROP which I renamed to 'mantra' instead of 'mantra1'.

The expression will look like this;

## opparm -c preview/mantra execute

## Digital Assets - Smart HDA

And voila, you will now be able to render it from the asset interface with its very own configuration.

Now we press the button and a new 'mplay' will appear with our image. This is because in the ROP I chose not to save the image, instead I left it by default which pipes the image to 'ip' which in turns behaves this way.

And it is what I wanted anyway, now I can choose to save it or not.


## Digital Assets - Creating an HDA

A few more links to keep you busy...
EDIT PARAMETER INTERFACE WINDOW
hetp://www,sidefx.com/docs/houdini13.0/ref/windows/edit_parameter_interface\#tidm47115664718928
EMPOWERING ASSETS WITH PYTHON
hetps://cmivfx.com/store/218-houdinitempowering+digital+assets+with+python
EXAMPLES OF ASSETS ON ORBOLT
hetp://www.orbolt.com/search/?qEtags:tool

## Digilal Assels - Wrapping an HDA

## Digital Assets - Wrapping an HDA

One of the things you may want to consider is that every single asset will have to be wrapped for usage and thus, the more funky you get with it adding features, the more you are going to test it, the more you should be documenting it, the more you should optimize it, etc...

For this reason I want to wrap a simple asset so you understand what is involved and design yours carefully.

Riggers are very used to this kind of task in which they protect their creation from unwanted approaches, make sure is idiot-proof and provide consistent behaviour for the many rigs they have been working so I am sure if you are into rigging you will find this strangely familiar.

What do I means by 'wrapping'?
It is a way to define that bringing all the aspects of the asset together and publishing for others to use means (or at least should) that you have to invest some time making user is ready to use.

My recommendation, add a day in your quotes per asset as this is really what it takes (being optimistic) for the life of the asset during production.

## Digital Assets - Wrapping an HDA

So, as you can see I dont have any icon on our asset and feels a bit second rate node so let's start by adding a custom icon to our asset.


## Digital Assels - Wrapping an HDA

We render with our in-asset camera and save a $500 \times 500$ image to the desktop...

In Photoshop we make it smaller (Let's say 32x32) in this case as I want the asset to carry this binary data inside and I rather not put a big file in.


## Digital Assets - Wrapping an HDA

Now we open the type properties and we are going to load an icon internally by going to the 'Extra files' tab, add the image you converted to $32 \times 32$ pixels.

## Digital Assets - Wrapping an HDA

At this stage we have succesfully loaded a binary which will be later referred to.


## Digital Assets - Wrapping an HDA

Now we open go to the 'Basic' tab and point to that file we just stored inside the asset.

As I used prefixes and a version my path to the icon is not that easy to get but I will try to explain.

My asset on creation is named
model::chair_wassili::1
but the 'Operation Type Manager' it resolves to
model::Object/chair_wassili::1
therefore that is the real path.
opdef:/model::Object/chair_wassili::1?icon-32.png


## Digital Assets - Wrapping an HDA

But we are not finished yet... we have to tell the asset where to install itself.

So in the 'Toolsroptions' menu we refer to the same icon using the same name approach (copy/paste is your friend)


## Digital Assets - Wrapping an HDA

And now we go to the
'ContextiNetwork Pane' and tick the OBJ context as we want this to appear to the user only in the right places.

So if we are doing compositing I wont want to see my models coming up.

Also, I add a new Tab submeny by setting 'My Models' as the name so when we tab I will see a nice subfolder with my stuff.

Now we save the asset, delete it from the scene (exiting Houdini is a paranoia from mine but I do it as I have found a few moments where it has solved some of my issues).

## Digital Assets - Wrapping an HDA

So now is time to recall our asset with its shinny new icon, on the right side you can see I sent the asset to a linux workstation and everything loaded fine (As the OTL does not have the geometry and textures included you have to make sure your paths resolve well and the files are where they should be)


## Digital Assets - Wrapping an HDA

Now is time to document the asset so let's also include the help files inside, something I particularly like from Houdini.

You define your help using WIKI commands so it is trivial to setup and the layout is beautifully handled by Houdini. (more on the last page for references to the syntax)

I use this simple template from the documentation to help me with the structure of the document so now is just time to fill the areas.


## Digital Assets - Wrapping an HDA

Becuase is a simple asset a few things need to be documented but you can build quite sophisticated documents and once again this travels with the asset!!!

Although of course you can point to a filesystem document which allows for much more advanced setups like indexing these files of proprietary tools onto a centralised wiki as well as being connected to the asset. Nice.


## Digital Assets - Wrapping an HDA

And here is the result after saving and clicking on the question mark of the asset's parameter panel.


## Digital Assets - Wrapping an HDA

There are a ton more things to assets so I will encourage you to go through the documentation, in particular the whole part of logic, handles, parameters lists, scripting I have not explored but I though that is something beyond the scope of these guides.

Make sure you bookmark the next pages on your browser as these are very important to me when creating assets.

HDA HELP WIKI FORMAT
hetp://www,sidefx.com/docs/houdini13.0/help/format
HDA NAMESPACES
hetp://www, side $f \times$.com/docs/houdini13.0/assets/namespaces
TIPS AND TRICKS
http://www.sidefx.com/docs/houdini13.0/assets/advanced ott
qLib DEVELOPMENT GUIDELINES ARE PURE GOLD
hetps://github.com/qLab/qLib/wiki/Developer-Guide

## Data Management

## Data Management - Introduction

## Data Management - Introduction

One of the strengths of a procedural workflow in Houdini is that the almost everything is a collection of files on disk that are used once the file is processed and like a compiler, spits out a result at the end.

Probably you think it is very similar to XSI approach with textures and external models, ultimately the difference is that Houdini deals with pretty much everything in this manner... to the extreme.

Of course this means there is a very strong natural link between the scene and its processing and its files, such is the relationships that renaming a file means breaking your scenes and potentially your job.

For this we have to step back a bit and discuss a basic strategy to follow by everybody in the team so the project is easy as possible to track.

A realisation of mine was that due to efficiency strategies by means of caching out stuff and then loading back in to minimise cooking time means that it is not only a series of input files to result in an output image/geometry but potentially a series of steps that will end up being written to disk and you can make your life and that of your team very complicated indeed.

I am afraid there is one only possible outcome to be successful when exploiting the full power if Houdini, be annoyingly methodical so we dont trip over later on.

For this I will discuss some workflow approaches and how to structure the whole setup, this will be my take on it and you should take it as a recommendation or possible solution... I am sure you will find a better way though.

## Data Management - Introduction

It is easy to start working your way and find that you should have structured it all in a very different way, we all have been there but the imptications of these changes when working with Houdini are bigger as there is an extreme dependency between the data on disk and the scene.

A typical scenario in Softimage you load the geometry and you start working, right away you are pushing things around until you have what you need, may be a few scenes later (let's say 20 versions Later) and you end up exporting that geometry to someone else to carry on working.

You may export another 0BJ for rigging, and off you go again...

As you can see some data lives outside, other inside, some half in/half out (referenced models and its deltas for example) plus you can only render images.


## Softimage

The data lives in two places, the disk and the scene and although we have learned to use it the fact is that you tend to trust what is inside your scenes, not the data on disk.

## Daka Management - InEroduckion

A Eypical scenario with Houdini is the other way around, you create your model in Zbrush for example and keep saving versions to disk (may be another 20 versions) and from inside your Houdini scene you just load the one that you require and once you are finished you may save another OBJ on the system to be picked by your rigging guy.

Therefore you end up trusting what is in the filesystem and this is fundamental change in mentality.

Now you need to invest a bit of time on the planning of your data, naming, folders, ekc...

Everything outside although you can choose to store it locally and there is a coherent view to data. You can render out geometry, bitmaps, audio and images.


## Daka Management - Daka Passing

## Daka Management - Daka passing

So looking at Houdini as a full on visual programming environment should point us in the right direction, we want to exploit automation and procedural workflows, sometimes it is not easy or appropriate but we may want to think in terms of designing everything to be reused later on.

So how do you manage the data that goes in side and outside? how do you version it? Furthermore, how do you deal with data that is generated and then used as a source?

Well, the paradigm I follow is that always someone takes my output data as their input data so in reality everything is always fed somewhere else.

You can architect these stage based behaviour as;

- One single fat scene with everything inside, hard to debug, hard to understand, potentially not very modular and can only be worked on by one artist.
- One single fat scene divided into processes or stages in which data is manipulated and passed to the next stage and as the previous one is a one artist mentality.
- Multiple scenes with one or more processes or stages that can be developed in parallel.


## Data Management - Data passing

## ONE BIG FAT SCENE

In this scenario the manipulation, passing data from one place to another is done either by simple referencing or by caching data out and then bringing it back in with the help of the various tools Houdini provides.

ONE SCENE


## Dala Management - Dala passing

ONE BIG FAT MODULAR SCENE

In this case data is processed and the "module" probably will write the data to disk for the next stage to pick up.

These subnetworks may be assets in the future that are "black boxes" that produce the result and as such can be treated as little programs.

ONE SCENE
SUENETWORK 1


## Dala Managemenl - Dala passing

MULTIPLE SMALL FILES

Although it is not very common in the various companies I have seen using Houdini it is actually a very good idea to split these big projects into multiple processing as different scenes if you are aiming to have multiple artists working in parallel.

An animator may pass a cache to the FX artist and this to the the lighting artist making a contained structure of the shot.


## Daka Management - Daka passing

As you can see you can find yourself pretty easily in a daisy chain situation and to manage this properly you ware going to have to come to terms with being thoughtful about your naming convention and folder structure.

So how do you structure your project? how do you manage the huge amount of data you are going to generate on every iteration of your production?

That is the part I cant help you other than start small, learn the issues by yourself and then decide if you want to invest more on developing tools to minimise the friction of the approach.

My suggestion will be to use a strict naming convention for caches in/out too that can be backtracked to the actual scene, time, artist and full name of the node that has generated it.

More on this later.

Dala Management - Dala Freeze

## Data Management - Data freeze

Sorry for inventing words, I need to put a word to it so we can discuss it properlly.

I call data freeze to those situations in which lots of complex to reconstruct processes (like weighting a character) can be stored on disk as a new starting point for the next process.

Imagine for a moment you can skin you character and save an OBJ file with the weights and bones relationships along with any kind of attribute you want on disk... wouldnt that be amazing? Well, in Houdini you have this possibility using BGEO file format.

So the approach is simple, do your big long task and save a BGEO of the final result that is later used on another part of the scene.

ONE SCENE

Data Freeze

Data Load

Process 2

## Process 1



## Data Management - Data freeze

Of course the issue is that you may have many versions of the mesh after the director has finally commited to the design, on the way you may have skinned these different meshes various times and thus you can end up with a complex to follow situation.

Once again, strict naming and folder structure will be your saviour so you can follow these assets live.

Let's discuss that in depth...

Dala Management - Dala Source

## Data Management - Data Source

These are the source geometry/audio/bitmaps we are loading as a starting point and are those coming from another program.

Geometry coming from Zbrush or Topogun, Bitmaps coming from Photoshop or Mari, Audio coming from Protools, etc...

The intention I have in separating them is that Data Sources are the starting point of every single manipulation we do and chances are you will want to decided this to be your root name.

Here is an updated image of our one big scene doing it all, the colors are different to clarify the function of these nodes.

If stage 1 is rigging the data out will be a freeze in this example while if Stage 2 is a cloth simulation it will be passing baked data to the next stage (passing data) and the rendering is just that, rendering.


# Data Management - Organisation 

## Data Management - Organisation

So now that we have gotten out of the way a few concepts we can safely move into defining a naming that woutd allow us to keep a project under control.

There are few things to keep track, namely;

- Scenes (hip files)
- Assets (otl files)
- Geometry (obj, abc, bgeo, etc...)
- Bitmaps (exr, hdri, rak, phg, jpeg, ekc...)
- Audio (wav, aiff, ebc...)
- Motion (bclip files)
- Renders (exr files mostly)
- Composites (exr files mostly)

So as you can see there are quite a few living on disk that, if moved or deleted will break the scenes...

Independently where these are stored (more on that later) you want to keep an eye on the names of all these files based on the function they accomplish.

For example, you could cache out a cloth simulation as a series of 'bgeo' files but you could also save the master mesh skinned and with all the attributes as a 'bgeo'. One is a simple 'cache' which is very easy to reconstruct if deleted, the other is a 'data freeze' which may not be obvious at all as it may be part of the preproduction stage of the job.

## Data Management - Organisation

In order to discuss the various options let's say we have a project regarding an ogre and the head mesh and the body mesh have been done separately, different resolution maps, some texture simulations for rain falling on the skin.

In this example we split geometry and maps, then by function, then we build the name with the character name, the part and the version plus the extension.

It has some serious drawbacks, namely all source geometry for every character lives in the same place, all the freezed one too and caching will become very crowded so although valid, it is not very good.

Furthermore, the node loading data wont have on the name any reference to the type or material loaded which could be misleading.

## Data Management - Organisation

In this second example we add the type to the name of the file so it will become very obvious on your network as Houdini now prints the name of the file loaded.

Still the situation is not great...


## Data Management - Organisation

If we were to remove a level, something you may be tempted you could build the category with the type of operation...
...but let's face it, it will be better for parsing files not to, after all, machine compatibility is quite useful to build a solid toolset and speed up the process.


## Data Management - Organisation

Why dont we try to separate by character type? well, again valid but seems from this structure 'maps' are nto part of the character and ultimately self contained units are extremely useful.

Imagine having to zip a directory for the character and missing all the textures... not great.


## Data Management - Organisation

Well, if we wrap by character, then split by category (geo, maps), then by object (head, body), then by type (sources, freezed, cached) we could end up with a more convenient setup.

Let's analysed... everything is inside a character, different categories allow us to group things neakly, by having the object we know all the versions of the element will live inside that folder.

Plus we have the type of object to deal with, making possible different storage and backup strategies as you may not want to backup cached material but you will will want on a very fast drive.

## Data Management - Organisation

The whole point is that if you try to define these structures you better get a sort of mind mapping software to analyse the various good and bad points every single solution will give you and accept the one you feel more comfortable.

My suggestions are based on my taste, there is no right or wrong but there is one critical thing, everybody has to play the same game or it will be a mess... make sure everybody is in it and everybody is heard, give them something they want and make sure their lives are easier, not more complicated and you will have a nice time.

Here is a proposed directory structure and naming for a small company so you can start straight away.

Note the various folders that are meant for global data (scene based).
of course you may want to improve it, add links to avoid duplication, etc...


## Data Management - Organisation

If you want to build a really good coherent system though, start with a set of rules you want to make sure you keep rather than starting with names.

For example, when Apple designed the iPhone the golden rule was that you should be able to do everything with the finger, this indeed drove all the decisions for the interface, interaction, response times, design and this is exactly how I approach these kinds of tasks.

## Define WHY, not HOW.

## Data Management - Data reconstruction

## Data Management - Daka reconstruction

One of the typical situations you will encounter is that a series of operations that will cache out data specifically to disk will at some point need to be reprocessed and the order of these operations is crucial.

Houdini does not provide a management environment and its is up to you how to use and abuse (for good and bad) the toolset.

I say abuse but the connotation should be a possitive one as it will allow you to do things not even the develpers could have imagine, a good example is using irradiance cache in the form of point clouds as an artistic expression you may want to manipulate in an particular way.

There are multiple strategies but I favour one simple approach via ROPs that will allow you to trigger the recache of data and effectively 'reconstruct' your caches if needed.

A good example of the need for data reconstruction is after restoring a job from a backup where the caches have not been stored, at this point you will certainly want to press a button and reconstruct all the data in a single go while you have a coffee to find a scene ready to operate with.

## Dala Management - Dala reconslruckion

The approach I follow is this to work my way on my character, freeze and pass data as required but every time I do so build a ROP counter part tree that can reconstruct the data later on during the project live or even further down the line.

Of course documenting it helps but it is a habit easy enough to pick that will pay off very quickly.

Here is a tiny example for illustration purposes, I normally would split these tasks in various subnetworks as if they were stages but for the sake of clarity I put it all down in one single OBJ node.

I am after generating a working version of a model that will be used across a project and set the basic material configuration with it.


Model courtesy of ARTEC (www.artec3d.com)

## Data Management - Data reconstruction

So let's start with a global overview of what I have done here.

There are three exports to disk (data freeze in my vocabulary) that are referenced on the file input nodes so a change of the name in the output will not break the scene.

The first output writes the geometry normalized in scale an orientation.

The second writes a polygon reduction of the model.

The third one writes a version of the polygon reduction with the attributes for shading.


## Data Management - Data reconstruction

Looking at the Geometry ROP node that saves the data to disk you will notice you can put your own prerender script, pre-frame scripts, postframe script and post-render script which would allow you to do really funky stuff if you need to.

I am forcing to render just the first frame and use the Main take to avoid any unexpected issues on how the scene state is.

And put a nice name to make sure I dont forget what this is.


## Data Management - Data reconstruction

On the file read I add an expression to read from the geometry ROP node so If I decided to change the name, my read node will be updated.

One thing to remember is that the file node will try to cache in memory the geometry just read, you will have to force a refresh by clicking in the Reload geometry when developing your tree to make sure you are in sync.


## Data Management - Data reconstruction

And I do the same for each export/ load so I now I three export nodes and three load nodes.

Next time I open the scene the whole operation will be very quick as I am not computing any of the very heavy operations of the preparation and polyreduce, making my life and everybody in the team's life very pleasant.


## Data Management - Data reconstruction

But of course I dont want to trawl through many character with many export/loads and obscure setups that have been evolving as the job has.

I have to be able to protect myself from those situation so what I do is set a ROP network that will trigger these renders by using the 'fetch ROP'.

Like this, starting with the same order that I have on the object process (SOP context) and executed in a sequential way instead of parallel, and for this the 'batch ROP' is essential as will force the full execution of the first node, then the second, then the third... and when you deal with sequences it is fundamental.

## Data Management - Data reconstruction

The batch parameter is very simple so the only thing to do is may be wrap it all in one single subnetwork once you are publishing your scene... but this is up to you.

You could also consider on your render network to render your passes assuming everything is up to date and give the option to the user to force a full refresh of the whole thing.

I am starting to use this method as a unit test approach (like serious programmers do) to check the scene consistency before sending thing to another artist.

## Data Management - Data reconstruction

As you can see suddenly your geometry can be reprocessed in one clean way, along with it your particle simulations, dynamics, cloth, fluids... you name it.

But it is clear to me you should have a system in place everybody knows and stands by to make the most of these approaches.

In particular you could take very aggressive approach to disk space if you use this well, for example, you could clean your caches when you have a shot approved in an automatic way knowing that if the client has a comment a reconstruction is trivial.

This could be easily a feature inside an asset as we saw in the smart HDA guide (page 369) in the form of a button to reconstruct delete and then reconstruct all the caches or create a fresh new cache with a new name. of course when designed properly this could be a very valuable tool.

# Daka Management - Scene packaging 

## Data Management - Scene packaging

A typical issue with any 3D application is the fact that sometimes you want to pack all the components necessary to build the scenes; models, textures, audio files, etc... and transfer them to someone else for whatever necessary task.

Specially now that we are heading to a more distributed workflow and cloud computing the needs to bundle it all into one big zip file could be make or break situation in a project.

My intention with this little guide is to show you best techniques so you can put together your scenes from the beginning in a distributed friendly manner.

Civen Houdini provides functionality to read environment variables and define your own, along with aliases and also custom expression functions the strategy is to, as you work, reference always things in a relative way and use these environment variables that I deem secure to specify paths that need to be absolute.

## Daka Management - Scene packaging

Here is a list of the key ones you should keep in mind every step of the way;

## \$HIP

This variable will point automatically to the location your scene (hip file)
is stored and is the single most
important one.

## \$HIPNAME

This variable will contain the name of the current scene as was saved.

## \$HOME

Your home folder.

## \$FPS

Frames per second
$\$ F$
Current frame
\$RFSTART and \$RFEND
Playbar in and out frames.

## $\$ 50 B$

A custom variable you define to point to a job... tempting but unless you have a standardised directory structure and a pipeline around to update the variable on the shell, this my not be that interesting.

In any case you may want to fully abstract yourself from the directory structure by defining direct variables to the various elements of your scene may be adding a prefix too like (\$ZZGEO, \$ZZMAPS, \$ZZCACHES, \$ZZJOB, \$ZZHIP...)

## Data Management - Scene packaging

And you have to remember you can actually do basic string manipulations by using modifiers;

If we assume the scene from the last guide in which were reconstructing data these two variables resolve as follows;
\$HIP = /Users/jordi/Jobs/Houdinicuides/hip
\$HIPNAME = caches_reconstruction.hip
NOTE: I have color coded the various operations to see better where these values are coming from.

- H

ROOT - On a filename all params except extension. \$HIPNAME: r = caches_reconstruction

EXTENSION - The extension of a filename.
\$HIPNAME:e = hip
HEAD - The path part of a pathname.
\$HIP:h = /Users/jordi/Jobs/HoudiniGuides
TAIL - The file part of a pathname.
\$HIP:t = hip
CAPITALISE/UPPERCASE
\$HIPNAME:u = Caches_reconstruction.hip

## LOWERCASE

\$HIP:L =/users/jordi/Jobs/Houdinicuides/hip
SUBSTITUTION - Basic string substitution
\$HIP:s/jordi/Paul/ = /Users/paul/Jobs/HoudiniGuides/hip

## Data Management - Scene packaging

Now, armed with these simple tools you should approach your scene external filenames to;

- Scenes (hip files)
- Assets (oll files)
- Ceometry (obj, abc, bgeo, etc...)
- Bitmaps (exr, hdri, rat, phg, jpeg, etc...)
- Audio (wav, aiff, etc...)
- Motion (bclip files)
- Renders (exr files mostly)
- Composites (exr files mostly)

With care and always reference them using variables so we can pack things easily.

On your design you will also want to think about where data lives, you may be tempted to put a geometry cache in a global folder for your whole job but this is a very bad idea, specially if this data is meant to be bundled to someone else or you cant guarantee their directory structure.

On this note, directory structure guarantee there is a key idea to keep in mind, the only folder you sure have and can reference $100 \%$ is the one where your hip file is stored, therefore you will probably see Houdini artists storing their geometry, maps, etc... under the hip file directory, this way is pretty much impossible to break. Not beautiful but it is certainly useful so I will let you think about it.

Now is time to look at a scene for real.

## Daka Management - Scene packaging

Let's make sure we can package our scene easily by looking at our last scene which I have copied to another computer... all my files are in different places, the links are broken and on a real production job this situation would take quite a bit of work to just figure out all the dependencies as what I am just showing is the tip of the iceberg.

In any case, this is not good news so let's rewire it all in a professional way.


## Data Management - Scene packaging

We start referencing the object in the new location (based on the idea that all the contents of the job lives under the \$HIP Location)


## Data Management - Scene packaging

Like this...


## Daka Management - Scene packaging

And here you can see the variables I have by default on my system, notice the $\$ J O B$ variable is not set automaticlaly!

This is the reason I mentioned before is a bad idea to use it unless you have a pipeline in place that sets it up automatically like we do in the office.


## Daka Management - Scene packaging

So let's construct the name for the cache nodes so everything is automated.

So we have the \$HIP pointing to the folder where the hip file is stored, then the caches subfolder and I am using the $\$ 0$ variable to resolve the name of the node so the file name is identical to the node allowing me to easily backtrack it.

Unfortunately although the resolution is valid there are chances a copy of this same asset would overwrite these files so it is not very well designed.


## Data Management - Scene packaging

If I add the name of the hip file to it and filter out the extension chances are this will be more solid although again, two copies of the same asset will overwrite the cache, something it is not acceptable.


## Data Management - Scene packaging

A much better route would be to designed referencing too the container object of this SOP network, like this;

I am referencing the scene name so another scene in the same folder wont overwrite my files.

I am referencing also the containing object (may be an asset?) so two copies of the object wont overwrite my files.

I am referencing the node that is saving making its name part of the filename so this way it is unbreakable.

I also tick 'Alfred Style Progress' for visual reporting on the shell.

## Data Management - Scene packaging

And finally I rewire as refernces too each fetch to the new names making this section work again.

Now my scene is ready to be zipped and will work.


## Data Management - Scene packaging

But now the question is, how can I double check all the scene I have built (or someone else has built) will be ok other than actually zipping it and putting it in another computer?

Well, you could use the cloud tools to run a 'Pre-flight Scene' which will traverse the whole scene in search of all external material.


## Daka Management - Scene packaging

You will be asked if you wank to use the $\$ H I P$ or $\$ J O B$ variable which you would be using to organise your scene... in our case is \$HIP.


## Data Management - Scene packaging

And as you can see there are a few green lines and a few red ones.

The green indicate things are good, it can find the original geometry, the hip file and the folder containing it.

The red ones are the result not being able to resolve the expressions so unfortunately, even knowing they work fine, we are getting an error.


## Daka Management - Scene packaging

So if you click on one of these you will see the parameter path and the expression that is not being
evaluated, of course we may want to check if this is a problem or not.


## Data Management - Scene packaging

So I went to that file node and deleted the expression and substitute it with an absolute path that does not contain the \$HIP variable.

This is what happens... It is red becuase it is oulside the \$HIP variable.


## Daka Management - Scene packaging

But if that absolute path I do convert to a \$HIP path then I will get a proper green light.

So it is obvious it must be the 'opname(".")' is causing the wrong evaluation.


## Data Management - Scene packaging

But there is another way to structure our data freeze by using the 'filecache' SOP.

This nodes encompases the ROP and read in one single node which makes it readable and easy to use.


## Daka Management - Scene packaging

Let's add our name structure in it, configure our rendering the same way.

And wire this to our ROP network.


## Data Management - Scene packaging

But wait a minute, let's dive inside the filecache node...
It is pretty much the same we did before, only tidier of course!!!

Will it work when we 'pre-flight' then?


## Data Management - Scene packaging

Short answer... NO.
We are going to need to remove the 'opname' function expression and see what happens.


## Daka Management - Scene packaging

Let's then remove it from all the 'filecache' nodes and see how it behaves.


## Data Management - Scene packaging

Finally we are all green, as you can see the caches names have been grouped into one were you can see all the uses.

Although now we have still one problem, if I had two instances of this asset I would be overwritting these files!!!

## Data Management - Scene packaging

Unfortunately I have yet to find a way but I believe is way more important a clash free asset/scene than the preflight to be gree (we do know it works though).

## Data Management - Scene packaging

As you can see the architecture of your scene can be critical for the future of the project so make sure you keep an eye on these elements as you build everything.

I know a lot of Houdini artists dont use the pre-flight approach but coming from softimage it makes sense and ultimately I want to know if the scene will be able to be zipped nicely.

HOUDINI IMPORT/EXPORT<br>hetp://www.sidefx.com/docs/houdini13.0/io/<br>HSCRIPT LANGUAGE GUIDE (Modifiers are here)<br>hetp://www,sidefx.com/docs/houdini13.0/commands/_guide<br>HSCRIPT COMMANDS<br>hetp://www,sidefx.com/docs/houdini13.0/commands/<br>HOUDINI EXPRESSION FUNCTIONS<br>hetp://wwiw.sidefx.com/docs/houdini13.0/expressions/

## Daka Management - Alembic

## Daka Management - Alembic

Over the last few years there has been a major drift towards certain open sourced technologies which are now consolidated. One of the most successful is Alembic as an interchange framework by means of baking geometry into a very efficient format.

Originally created by Sony Imageworks and Lucasfilm it has finally standardised the exchange of backed data between different applications and with it has brought the possibility of teams working in one particular plakform to exchange their results to another team working on another platform.

One the benefits as a side effect is that the geometry resulting from an animator will be passed to the rendering artist without all the extra load of rigs making the production more compartmentalised.

The drawback is that these caches tend to bring a big load on the servers, space, network, backups and specially asset management.

My intention with this little guide is just to show you a few things regarding Alembic files in Houdini so you can design a much better, efficient and cheap pipeline than you could with Maya for example.

So let's see how this works.

## Daka Management - Alembic

Let's assume first we are going to ingest baked data from another department in Alembic format, you can either load it as a scene or as a model.

- As a model (SOP approach) you just load the full blob in on object.
- As a scene (OBJ approach) you are replicating the hierarchy in houdini using alembic nodes that read directly from the alembic file.

Let's have a look as a model first by using the SOP approach.

Just drop a geometry node and dive in...

## Data Management - Alembic

Delete the file node and drop an alembic SOP node.

Then load an alembic file with multiple objects and hierarchies inside.

As you can see I have loaded the whole thing in one go, a couple of expressions are put automatically for me and the basic settings try to optimize for performance by using 'Alembic Delayed Load Primitives'.


## Data Management - Alembic

But you can also browse the various elements inside and select them if you with to load just one element.


## Data Management - Alembic

In this case I have separated the windows from the plane and we could further manipulate these elements if we wish...

H Houdinifx File Edit Render Window Takes Help


## Data Management - Alembic

...although given they are Loaded on demand you will be required to convert them to polygons first.

In this case I am applying apoint jitter just to showcase the possible manipulation.


## Dala Management - Alembic

Now let's have a look as an scene import by using the File>Import tool and choose the desired alembic file.


## Daka Management - Alembic

Now you will see an alembic archive loaded which contains all the
hierarchy and geometry as separated elements.

Let's browse through them.


## Data Management - Alembic

If you go the network view and press ' $e$ ' you will change to tree mode which will help illustrate the result of the previous operation.

Now you can see a tree as it is inside the alembic file referencing it.

If you dive down you will end up finding a geometry object and inside an Alembic operator like the one we described in the previous example.

Basically these are the same Alembic SOP nodes all managed for you to replicate the file as a scene object.


## Daka Management - Alembic

And if we open the parameters you will see it is exactly identical only the separation via selection has been done for you.


## Daka Management - Alembic

If for whatever reason the alembic file content changes you may want to rebuild it by using the tool provided in the master Alembic archive, this way the whole hierarchy will be refreshed.


## Data Management - Alembic

Exporting Alembic files is very much in line to the ROP approach so it should be easy.

If we go back to our old table scene and use the table as elements version in which each leg is a different object.


## Data Management - Alembic

Now we go to the ROP context and drop an 'Alembic' node to export.

Then you add a nice name (in this caes test.abc), select the object you want to export (and everything inside)

Now we are ready to render the Alembic ROP node out.


## Daka Management - Alembic

And as usual, we validate our export by checking it is what we are hoping for.

We are going to bring it as a scene to be able to navigate inside the object in a simple way.

## Data Management - Alembic

And as you can see the hierarchy is stightly different from what we had...

Is this an export error? Well, not really...


## Daka Management - Alembic

You should remember than although table top is passing its position to the $\operatorname{leg} 1, \operatorname{leg} 2, \operatorname{leg} 3$ and $\operatorname{leg} 4$ it is strictly not their 'structural' parent when exporting.

This is something you may want to play around as if you compare this with XSI schematic you will realise that they vary in a fundamental way, in the schematic these relationships drawn are always hierarchy relationships, in Houdini these are data flowing relationships.

I imagine this is due to the fact that we can group these branches, something that does not make sense in the convenient yet simple XSI approach.


## Daka Management - Alembic

Notice also you can use the 'Data tree' pane to analyse and modify some aspects of your Alembic file.

If you dig down to the acutal loading node you will see that the display type and a few other parameters are now accessible so you can override the settings.

I believe the 'Data tree' has a promising future to manage not only objects and basic parameters but a lot more so make sure you start using it and feel confortable.


## Daka Management - Alembic

So we have explored quite a few things relating how we manage data in Houdini, hope it has not been too hardcore but for the next chapters we will want to be aware of these workflows and tools.

Regarding Alembic vs. Bgeo file format, I must confess Bgeo is my favourite for many reasons and I only use Alembic when communicating outside as both in terms of flexibility and file size it is better than Alembic on my tests (or marginally worse on some)

Nevertheless it is obvious that with the standardisation of Alembic across products we are going to have to consider it very adventurous. Just think passing data back and forth between Mari and Houdini.

ALEMBIC OFFICIAL WEBPAGES<br>hetp://alembic.io<br>hetp://code.google.com/p/alembic/<br>ALEMBIC DOCS<br>hetp://www.sidefx.com/docs/houdini13.0/io/alembic<br>RENDERING WORKFL.OW TIPS<br>hetp://www.sidefx.com/docs/houdini13.0/render/tips

Rigging

## Rigging - Introduction

## Rigging - Introduction

As I write I struggle to picture an audience that sees Houdini as something more than an FX package but the truth is that it excels in many areas and in this particular I hope I can show you it is way superior to anything else out there.

We are going to visit some core concepts and I will be constructing simple examples to showcase them and in the spirit of this guides, show you how to move to Houdini rather than show you how to do a professional level end to and Houdini rig.

I would love to though... may be in the future.

## Rigging - Introduction

So let's start with my view of the world when it comes to rigging so you dont freak out when you see me doing things in a different way.

I have been doing character animation and rigging for a long long time, I would say that it is my speciality and over these years I have built a view of how to rig based on how I want to animate and the aesthetics of the animation itself.

From pure cel animation, 3d character animation, motion capture and motion editing my approach to rigging and animation forces animators to work in a particular way; specially how they keyframe and how they plan the shot (yes, you have to plan it).

First we will look at some core tools, then we will dive into FK rigs for motion capture and finally we will jump onto IK rigs for more cg character animation. I hope I
have time to work on the facial rigging but let's see how it goes first.

## Rigging - Transforms

## Rigging - Transforms

Let's start XSI approach to parenting and transformations in a bit more detail.

Let's assume we have $a$ cube and a sphere, represented in the schematic as the two black nodes and on the explorer as the two mesh objects.


## Rigging - Transforms

By parenting them you establish a relationship in which the transformations applied to the parent object (cube in this case) are transferred to the children (sphere in this case).

The cube has its center of coordinates and the sphere too, but the sphere are now dependent on the cube.


## Rigging - Transforms

If you rotate the cube, the sphere will 'follow' as the transformations is passed down the hierarchy.


## Rigging - Transforms

If you cut that relationship the transformation of the cube wont affect the sphere any more as there is no hierarchy stabilised.

Notice also that the coordinates of the sphere have changed as now its parent is the center of the world!!!

This is a fundamental concept, even if there no connection the fact is that the sphere is now defined in base to the centre of the world instead of the parent object.


## Rigging - Transforms

Now if you freeze all the transformations of the sphere the result will be that the center of the sphere will be set to zero but the points that conform the sphere will stay in place (the sphere has not moved).


## Rigging - Transforms

What has happened is that the vertices that conform the sphere have been compensated by the 'freeze tool' so the sphere does not move and thus they have now the transform applied.

Effectively the freeze operation is the same than moving the sphere to the centre by resetting the transformation in $X, Y$ and $Z$ and then moving the vertices by hand to were they were before.


## Rigging - Transforms

Lel's see now how to do the same in Houdini.


## Rigging - Transforms

And Let's parent the sphere to the box...
...wait a minute... the sphere has been moved to the box position!!!


## Rigging - Transforms

Well, Houdini does not try to many things behind your back, so it does not compensate things unless you ask for it... For us XSI users this is a must so let's see how to do it.

Pick the sphere and tick the 'Keep position when parenting' flag.

## Rigging - Transforms

Now the behaviour is identical to XSI.


## Rigging - Transforms

And if I rotate the cube the sphere inherits the transformation.


## Rigging - Transforms

And like in XSI the sphere is free to be moved too, the transormations we apply to it will be referencing its parent, which is the cube right now.


## Rigging - Transforms

And again, if we rotate the cube back the sphere still inherits the transforms like XSI and in fact, every single 3D package.

Houdinifx File Edit Render Window Takes Help


## Rigging - Transforms

But what happens if you disconnect the sphere from the box?

The same, it does not move althoug you will notice one very strange thing.

The transformations of the sphere before and after disconnecting have not changed!!!

Is Houdini doing something for me? finally? well yes, it has added the change of base as a pre-
transformation which is very useful but not so obvious at first.


## Rigging - Transforms

If you select the drop down menu 'Pre-transform' and select 'Extract pre-transform' the numbers will change to the truth behind the sphere position.

Basically Houdini does nanny you a bit after all to make sure you dont loose the precious neutral poses you may want to build when rigging.


## Rigging - Transforms

But how do we 'freeze' the
transformations so the center of the object is in the center of the world and my vertices are where they are now?

You just select in the Modify menu the 'Freeze' tool that will do the job for you like in XSI.


## Rigging - Transforms

So now the translate parameters are zeroed out although on the interface you will notice we have a pivot offset.

One second, what is this Pivot set of values?

Let's look inside the sphere object.


## Rigging - Transforms

A new transformation node has been added to the hierarchy with the values we had on the object level.

Effectively this is moving the vertices to the same position of course.

But if you look carefully you will notice the values on the translate are identical to the values of the object level pivot!!!


## Rigging - Transforms

If now we go to the object level and 'Revert to defaults' the 'Pivot' you will see the axis is zeroed out as you were expecting from the XSI approach.

What Houdini's 'Freeze' tool is trying to do zero it out but give you the object axis compensated to match the object position.

Therefore an XSI style freeze is now a two step process in which you 'Freeze' and then 'Revert to defaults' the Pivok.


## Rigging - Transforms

So as you can see the concept is identical but implemented in a stightly different way.
If you wish you can certainly configure Houdini's default to be closer to XSI although I wont go too far on that as I believe there is a lot to be gained in embracing the philosophy of any package instead of bending it backwards in an unnatural position.

HOUDINI OBJECTS<br>hetp://www,sidefx.com/docs/houdini13.0/basics/objects<br>SOP TRANSFORM<br>hetp://www.sidefx.com/docs/houdini13.0/Modes/sop/xform<br>SCRIPTING TRANSFORMS<br>hetp://www.sidefx.com/docs/houdini13.0/hom/cookbook/object .xform/

## Rigging - Constraints

## Rigging - Constraints

Constraints together with bones are the heart and soul of rigging and something and although simple I though I may show you a few little things that are specific to Houdini.

First of all let's start with object level constraints and then we will dive into SOP level constraints and how we can use CHOPs to manipulate channel data in totally incredible ways.

Let's start by looking how we define a few things related with rigging and constraining so we can use these tools on a more general way.

- Position constraint, we normally refer to the case in which a "constrained" object acquires the position values of another object or objects.
- Rotation constraint, we normally refer to the scenario where a "constrained" object acquires the rotation values of another object or objects.
- Scale constraint, where a "constrained" object acquires the scaling values of another object or objects.

But these definitions are rather poor as they refer to one to one relationships and points in space rather than planes or volume constraints.

So I am going to try to re-define this to be more precise.

## Rigging - Constraints

To work on posilion coneraints you have a few bools available, the most common may be a direct channel expression to get the values from another object, like this;

By reading the value of another object channels (in this case the box object) we have literally copied the values onto the grid object. Effectively this examples limils 3 degrees of freedom on Eranslakion of grid, meaning that you cant move the object.

But writing expressions may not be what you are after, specially since it is difficulk ko follow and debug on a really big rig.


## Rigging - Constraints

Luckily you can follow these relationships by configuring how Houdini displays the data.

Press ' $d$ ' and go to 'Network View > Dependency' tab and turn on the 'Show Local Dependency Links' so you will be able to see the expression relationship.

Again this may be great but we probably want a very consistent way of architecting our rigs... therefore we may want to use other forms of constraining objects.


## Rigging - Constraints

Welcome to the 'Blend' node which is really useful for these things.

Let's imagine we have a box, an sphere, a torus and a grid (yz oriented)

Now we drop a 'Blend node' that will allow us to construct our constraint with a node rather than an expression.


## Rigging - Constraints

Welcome to the 'Blend' node which is really a swiss army constraint tool.

Its default behaviour is to pass the data for all transtation, rotation and scale channels from the input object to the child object. This is called in XSI a POSE CONSTRAINT and we will see it later in detail.

If we toggle of the mask flags for the various channels we can change the behaviour and by using multiple inputs we can blend these as we wish.

## Rigging - Constraints

For the educational purposes I have rotated the box and scale it up, like this.


## Rigging - Constraints

## POSITION CONSTRAINT

By only passing translations we are building a simple position constraint.

You can see the grid in the same position than the box but both rotations and scaling have not been Eransferred.

## Rigging - Constraints

ROTATION CONSTRAINT
By only passing rotations we are building a simple rotation constraint.

The grid is on the centre of the world but rotated as the box.



## Rigging - Constraints

SCALING CONSTRAINT
By only passing scaling we are building a simple scaling constraint.

The grid is now scaled like the box but position and rotation is untouched.


## Rigging - Constraints

POSE CONSTRAINT
By passing all the channels we are matching the transformations of the box, what in XSI is called a 'Pose constraint'


## Rigging - Constraints

But one thing you will find useful is that offsetting the constraint can be done on the object level...


## Rigging - Constraints

Or blend level, something I find way more correct as I want to have the objects as clean as possible.
\& Houdinifx File Edit Render Window Takes Help

\#H


## Rigging - Constraints

To construct a 2 Point constraint you just need to connect two objects to the 'blend' node and it will interpolate between them giving the control to weight it towards one object or the other.

If you select the 'blend' node and in the viewport you press 'enter', two heads up display sliders will appear and you will be able to manipulate them nicely from there.

## Rigging - Constraints

And you can build a 3 point constraint by adding a third as you probably guessed.

In this case this equates to a 3 point position given that I am masking out rotations and scaling and as you can see the weight is normalized.


## Rigging - Constraints

But the 'blend' node does a couple more things you will surely want to look at.

If you change the mode to 'Sequence' it transitions from the first input to the second, to the third, etc...


## Rigging - Constraints

Effectively you are building a sequence of constraints with one single stider indexed form zero.


## Rigging - Constraints

Be aware the slider may go from zero to one, but when you have more than two connections you can still select bigger numbers.


## Rigging - Constraints

Also there is one things you may want to exploit on how the 'blend' node behaves.

If you pass a number to the sequence channel out of bounds, meaning it is bigger than the number of connections or smaller than 0 , Houdini interprets no constraint!!!

Notice on the picture the 'grid' object is at $0,0,0$ as the value passed to the 'Sequence' parameter is 3 (remember it is an index starting at 0 ).


## Rigging - Constraints

Now if you select instead of Sequence the 'Constrain' mode it will behave like the 'Sequence' mode but shapping.

As you can see the 'blend' node brings a lot of functionality and chances are you will use it a lot, just make sure you organise yourself to document what is doing by its name for example.


## Rigging - Constraints

At this point I want to confess I follow the strategy of never affecting the objects I am animating, rendering or rigging, but rather use nulls or similar tools as "supports" so in case I have to change the geometry I am rendering for whatever reason I wont break my rig.

It may be not as efficient in terms of performance but the fact that I abstract the functions gives me more flexible and productive, specially when things get very complicated so I tend to use the 'hode' for every single "support" role.

After this confession let's look at a few constraint tools critical for your work.

## Rigging - Constraints

## DIRECTION CONSTRAINT

Also known as an 'Aim' constraint or 'Look At' constraint this is a fundamental building block for rigging.

In this case it is using the 'Transform' tab on each node that I am using, but I use the blend node as "support" instead.

You can see the grid is now lined up to look at the sphere, which I am using as a 'Look at' object.

Also you have an 'Orient Up vector' parameters like you do in Softimage 'Direction Constraint' tool.


## Rigging - Constraints

## PATH CONSTRAINT

In some cases you will want to attach an object onto a path, for this I have created one using the 'path' tool and then simply reference it on the 'Path Object' parameter inside the 'blend' node.

You will notice a few more objects due to the fact the path controls are also objects but you could always wrap them on a subnetwork.

Anyway, a few parameters are now available to control the object position on the curve.


## Rigging - Constraints

## RIVET CONSTRAINT

In Softimage we call this 'Object to Cluster' constraint but in Houdini the same concept needs to be rethink.

The principle is simple, you apply a rivet on any point or group and the transform is extracted so you can use it to pass the data to another object.

By using various rivets you could extract a plane for example (thru a triangle) which will allow you to build quite interesting mechanisms.

I think I will let you play with this very powerful tool.

## ;-)

(- Houdinifx File Edil Render Window Takes Help linn


## Rigging - Constraints

## SYMMETRY CONSTRAINT

This is the one I am still uncomfortable how I have approached but I guess cant get everything in life...

The idea once again is to use a 'blend' node to be my support and scale on the axis I want to produce a symmetry.

Here I have two grids, one has been translated 2.8 units on $X$.

The blend node is not receiving any input, just acts to mirror the data below.


## Rigging - Constraints

Now the 'grid1' node transtations are referencing the 'grid' object via an expression.

I guess given I am going to do this I could do it all in once inside the 'grid1' node but I like the idea that the "support" holds the scaling.


# Rigging - Constraints 

## SURFACE CONSTRAINT

The last one is by using a 'sticky' object to achieve the constraint to the surface and use the surface UV parametrization to position the object.

The surface has to have a UV space and you have a number of options to orient the object or not, wrap on $U$ or V, etc...

Have a play and make sure you check the 'Sticky Blend' tool to blend between two sticky objects.

This is much more precise than the rivet as you can place the object anywhere on the surface so make sure you play with it.



## Rigging - Constraints

Here you can find a few links to some of the nodes you should keep and eye.
HOUDINI BLEND OBJECT
http://www.sidefx.com/docs/houdini13.0/Modes/obj/blend
HOUDINI BLEND STICKY OBJECT
hetp://www,sidefx.com/docs/houdini13.0/Modes/obj/blendsticky
But sometimes expressions may be the thing you want, and thus Houdini provides a few little tools to remember that will help you with that.

VTORIGIN EXPRESSION
http://www,sidefx.com/docs/houdini13.0/expressions/Vtorigin
MLOOKAT EXPRESSION
http://www,sidefx.com/docs/houdini13.0/expressions/mlookat
MLOOKATUP EXPRESSION
hetp://www,side fx.com/docs/houdini13.0/expressions/mlookatup

## Rigging - Mechanisms

## Rigging - Mechanisms

So far we have seen the tools in Houdini and how they compare to the XSI counterparts in the most primitive way. of course these are the building blocks from which all the rest is holding together and at this point it does make sense to separate the two big areas we are abut to cover;

What I call "mechanisms' that as its name are interpretation of mechanic objects (no Rigid Body Dynamics) and can be divided further onto positional mechanisms and rotational mechanisms. From there you can mix and match but I will be outtining only the basic ones.

What I call "organisms" that are an interpretation of natural systems like muscle systems. Once again these are out of the scope of this guide but I will do at least one to put in perspective the tools.

I am a champion of modelling the world as it is in real life, surely this is a side effect of my training as an architect with lots of interests in both structures and simulation, so when we talk about "mechanisms" we should start looking at the real live world to construct the same structures, from hinges to universal joints and beyond.

Terminology wise it is a bit of conundrum as while in any of the 3D applications we refer to constraints as tools, in the real world constraints are "limitations" so in order not to confuse you I will refer to the real world constraints as "mechanisms" although in reality they are not.

Therefore allow me to go through the fundamentals first.

## Rigging - Mechanisms

Position mechanisms, where an object's position in space is limited by another, either limited on all three dimensions (resulting on a point), on two dimensions (resulting on a line) or one dimension (resulting on a plane), if there are no limits then we still outtine it for the sake of a generalised description.

|  | POSITION CONSTRAINTS |  |
| :---: | :---: | :---: |
| Degrees of freedom | Constraint Type | Example |
| O DoF | Locked position constraint | Full locked object |
| 1 DoF | Line constraine | Piston / slider |
| 2 DoF | Plane constraint | Surface slider |
| 3 DoF | No constraint | Free object |

## Rigging - Mechanisms

So let's analise these various types of real world constraints (or mechanisms).

LOCKED POSITION CONSTRAINT
All three axis are constrained to another object's position making it static.

## LINE CONSTRAINT

This type of constraint allows an object to move through a line, typical of pistons and stiders.

## PLANE CONSTRAINT

The constraint frees the object to move along a plane, typical of projections on the floor.

EREE OBJECT
Although in the world of physics there is such thing you have to remember a free object is still represented mathematically by a limited system, that is the computer so it is the numerical accuracy of the software you use and the machine that will allow you to consider it free before runing into troubles.

As you can imagine we are interested in identifying each object on our mechanism (let's say a crane) and construct these in an accurate manner.

## Rigging - Mechanisms

Rotation constraints, where an object rotation in space is limited by another in either three dimensions (resulting on a lock), two dimensions (resulting on a hinge), one dimension (resulting on a ball).

|  | ROTATINN CONSTRAINTS | Constraine Type |
| :---: | :---: | :---: |
| Degrees of freedom | Locked rotations constraint | Example |
| O DoF | Pivotal constraint | Lock |
| 1 DoF | Cone constraint | Hinge |
| 2 DoF | No constraint | Rotula |
| 3 DoF | Universal Joint |  |

## Rigging - Mechanisms

Regarding rotations we found the same variations we found with positions.

LOCKED ROTATIONS CONSTRAINT
All three axis are constrained to another object's rotation making it static.

PIVOTAL CONSTRAINT
This type of axis keeps one of the axis free allowing for moving on a plane, typical of an hinge or lever.

ROTULA CONSTRAINT
Probably is one of the more complex in the sense that although there are two axis free only an effective cone is available as the support restricts the motion.

UNIVERSAL JOINT OBJECT
On an universal joint the axis are pretty much fully free and allow for an almost complete motion.

## Rigging - Mechanisms

But this is not enough... we should dig deeper so I will also throw a few other mechanisms for you to consider and even take as a challenge.

## GEARS

Two one dimensional rotation constraints linked together with potentially different sizes. Here you have lots of very interesting uses and it is worth looking at it, specially if you have a project related to an engine, transmission system or watch mechanics.

SLIDER CRANK
Two linear constraints linked through a rotating object.

## CHAIN AND SPROCKET

Two rotating objects (potentially different sizes)

OCTAHEDRAL HEXAPOD
A combination of one dimensional transtations to produce complex translations and rotations.

## WORM GEAR

A combination of a gear rotating against bar.

## SCISSOR LIFT

A combination of rotation elements to produce a linear move.

So as you can see there are lots of combinations to construct mechanical rigs (and more I have not even commented).

Enjoy.

## GENEVA MECHANISM

A one dimensional rotation drives a fractional rotation on another object.

## Rigging - Bones

## Rigging - Bones

Bones are the fundamental building block to rigging as they are the "deformers" of the mesh in Houdini and although a simple tool its flexibility combined with the IK solvers makes them essential.

Behind how to use the bones on your rig may be personal preferences, I know mine, but ultimately all boils down to having an efficient and predictable, easy to animate rig that does not get in the middle of the animation process.

One thing to remember is that bones are objects so you can do all the things you come to expect from an object and it should be simple to get your head around the philosophy in Houdini of how these things work.

## Rigging - Bones

Let's start reviewing how bones and chains work in Softimage.

A chain is a structure formed by three types of elements;

- The Root
- The Bones
- The Effector

And while this may be natural in Softimage it is a pre-packaged setup to speed up things, for example, the IK functionality is already there even if you dont use it.


## Rigging - Bones

The way the hierarchy is buit is as follows;

The Root node defines the start of the chain so it is the parent, the first bone coordinate system is based on the root node, something extremely to get clean line-ups on your rig (more in a minute), and then a series of bones.

In the past with the old Softimagel3D the effector was always the end of the chain but in XSI this changed to make it the son of the root node.

Let's now have a look at axis.


## Rigging - Bones

As you can see if I select the bone the axis are already oriented for me, this is something I rely constantly and it is the certainty of knowing the orientation follows this simple rule that leverages a lot of easiness on the rigging side.

The $X$ axis traves along the bone while the $Z$ axis defines the main rotation (flexion), then the $Y$ axis defines the rotation outside the IK plane coming from the IK 2D solver.

The point is that softimage bones are rotation based unlike Maya where they are position based and like all religions both parties have huge followings of people that think they are right and everybody else is wrong.

## Rigging - Bones

If we open the parameters for the first bone you will see the PPG has some extra information defining the Kinematic Joint solver, preferred angles, IK blend, Rotation limits, etc...

In XSI the chain comes preconfigured, in Houdini it does not so let's see that later.


## Rigging - Bones

You can move the effector to the last bone position if you like, I tend to do this mostly because I find it neater, still everything works and as you can see the effector affects the chain in IK and gets rotated at the same time.


## Rigging - Bones

Lel's see how this works in Houdini.

Here you use the 'Bones' tool in the shelf that will put you into bone creation mode.

I did switch the viewport to Right as I want the leg to be oriented in Zt and make the drawing process easier.

Now after clicking three times press return and voila, you have now 3 nodes depicting the chain.

You have a Root, two bones and I miss my Effector but will put it all together later.

## Rigging - Bones

First let's have a look at the parameters of the chain.

You will see your position is locked, your rotation is open but it is zeroed out. This is because the data is in the pre-transform.

Like XSI, Houdini is a rotation based approach so for us is extremely easy to pick up.


## Rigging - Bones

If you select the bone and in the viewport press enter you will switch to manipulation mode and the bone rotation manipulator will appear, this will be useful for accurate line ups.


## Rigging - Bones

Now if you press 'y' you will switch between manipulators and bounding volumes that are the ones defined by the bone to automatically 'capture' the geometry and assign weights.

When I say 'capture' I am using Houdini vocabulary, in XSI this would be to skin the mesh. (In Houdini skin is a completely different operator so do not use that terminology or everybody will be confused).


## Rigging - Bones

So now that we have seen the most basic building block, let's set up a little rig to a real life character I bough from AXYZ Design (www,axyz-design.com) a website with a nice collection of characters priced very nicely.

NOTE: As the asset comes in 3DSMax format I had to convert it to an OBJ and did a fair amount of work on the mesh to make it symmetrical so when you use the same character you will have to do the same.


## Rigging - Bones

As we saw before I start importing the geometry and building a sort of ingest system.

## Rigging - Bones

I put colors on every object so I can identify things later on easily, in this case there are very few object but when dealing with geometry that is joined and I want to later separate it this allows me to know if I am dealing with the pieces or the full merged mesh.


## Rigging - Bones

So I create a sort of "template" on how I am going to organise the scene, later I will substitute these objects for subnetworks as the chains are objects themselves for for now let's simply try to drop one of them and then we will tidy up.

Using the 'Bones' shelf button (topleft) I draw on the view a chain (I this this on a full side view)

As you can see from the example there is no effector and by the icon on the bones it is a FK rig.


## Rigging - Bones

I line it up with the leg internal structure and adjust with the bone controls.

Then add a Null to act as an effector and set the naming convention I am planning to use.


## Rigging - Bones

So I now delete the template object called 'rig' and create a new subnetwork with the leg selected. Rename it to 'rig' and dive inside.

Also I duplicated the whole Leg chain and add a null on top (center of the world) and scale the null by -1 in $X$ so I have a symmetry.

Before continuing any further let's see how the chain will respond.


## Rigging - Bones

I now check how the animator will perceive the rig by selecting both femur bones and rotating in one axis, I am after these bones behaving like the picture, if one of the legs goes wide, the other should go wide.

Of course there may be implications with this approach when dealing with motion capture, Massive and some other packages but for now let's move on and later you decide how you want to do it.


## Rigging - Bones

Now I create the basic nodes for the placement of the rig and a centre of graviky.


## Rigging - Bones

Now is time for the spine, in this case as my intention is to build a simple rig for far distance shots I dont need much sophistication so I will place a simple 3 bone spine.


## Rigging - Bones

## And the head.

At this point I start grouping things so when I lay out the skeleton I can identify easily these group and more importantly, make sure Houdini does not re-layout it for me.


## Rigging - Bones

Arm, which again I line up extremely carefully to make sure the plane of motion is the correct one.


## Rigging - Bones

Add the clavicles, and do another symmetry to test I am on the right track.

As is my intention to have proper hands I will not spare too much time with the right arm and simply delete it later but I know now that everything is symmetrical.


## Rigging - Bones

And the left hand, carefully naming things and making sure these bones are properly laid out.

A paranoia of mine is to make sure the hand can close convincingly so I position the root and rotate it carefully so that is the case.

As an animator I am specially interested in expression and performance, both face and hands are my tools so I want to make sure they are perfect.


## Rigging - Bones

Now I finish the structure doing the feet, making sure I do a nice symmetry of everything.

Color code green for left and blue for right (left=greenpeace, blue=bankers).

The middle is yellow so I can focus on these elements that are so important.

The black nulls are simply "collectors" so I can identify where I am doing an change of base and grouping things.


## Rigging - Bones

There is one key element to bones that I found on the manual after looking for a while, bone rest positions are not that obvious at first.

You have to remember than when you draw the bone you are defining a rest position which is used for the ik solver to find the right solution out of the many available on ahe solve, think of it as a hint.

Furthermore the ik rest position is critical when capturing the mesh so you should have this correctly setup before or you will have to do some work with your capture node too.

Although the manual tells you to set your rest position by hand, you do not want to do that, instead you want to reset the rest position to the current position of the bones so I will show you how to do that now.

## Rigging - Bones

Here I have created a chain with three bones, and duplicated it so I can illustrate what is going on.


## Rigging - Bones

Now I rotate my right chain to a new pose, let's imagine that is the actual pose of a leg I want to matich.


## Rigging - Bones

Now I can see the IK Rest position by using the Edit menu shown in this picture which will override the viewport settings.

## Rigging - Bones

But as you can see the IK rest position is still the one defined on creation, I want to make sure my bones dont move so we should fix this.


## Rigging - Bones

So let's go to normal view, select the bones and clear the rotations.


## Rigging - Bones

And now on the Bone tab I reset the 'Rest Angle' to the defaults (zeroed)

This is in fact the rest pose;
Pre-transform + Rest $=$ IK Rest


## Rigging - Bones

As you can see in the image, when I override the viewport to IK Rest my bones dont move.

I am now good to skin and solve.


## Rigging - Bones

Until here you can see is very straight forward and extremely similar to XSI although the ability to group things in such a comfortable way makes it much more useful.

## RIGGING APPROACHES

hetp://www,sidefx.com/docs/houdini13.0/character/riggingapproaches
RIGGING TIPS
hetp://www,sidefx.com/docs/houdini13.0/character/riggingtips
DRAWING BONES
hetp://www, side fx.com/docs/houdini13.0/character/drawbones
HOUDINI BONES
http://www,sidefx.com/docs/houdini13.0/Modes/obj/bone

REST ANGLES ON BONES
hEtp://www.sidefx.com/docs/houdini13.0/character/restangles

HOUDINI BONES VS JOINTS
hetp://www.sidefx.com/docs/houdini13.0/character/bonesvsjoints

## Rigging - Kinematics

## Rigging - Kinematics

There is a thing I have tried to avoid, calling the bone structure we saw Forward Kinematics and although I have done it 2 times the fact is that we have to make a very important distinction on how Houdini deals with Kinematics, either forward or inverse.

What we saw was just "No Kinematics" mode, there is no calculations other than basic hierarchical transformations, no solver involved, but Houdini provides also Inverse Kinematics in which a solver is part of the system and provides a "solution" to the effector movement.

On top of that Houdini has a mode called "Forward Kinematics" which in reality could have been called "Forward Kinematics from Eranstations".

Furthermore you have two more modes, the one in which bones follow curves and the one in which we have constraints on the bones (bone limits for example) to consider in the solver for whatever method you choose.

So you could have 'Forward Kinematics' (from transtations) with bone limits so you have to consider this too.

Let's view all these before we continue with out character rig.

## Rigging - Kinematics

Starting with the basic method for a leg;

NO KINEMATICS
In this mode as we have seen we only deal with hierarchical transformations.

No effector or 'goal', just the root of the chain and the two bones I drew.

## Rigging - Kinematics

FORWARD KINEMATICS
In this mode we do have a solver in the form of a CHOP network (channel operations) that does the ackual computations for the kinematics soludion.

Furthermore we have a goal for the knee and another for the heel parented to the hierarchy.

The point is that the bone rotations are derived from the Eranslation of the bones, avoiding the gimbal lock problems and thus, optimal for motion capture processing and many mechanisms described before.


## Rigging - Kinematics

INVERSE KINEMATICS
In this mode we are constructing a classical IK chain with its corresponding solver inside the CHOPS nekwork.

You also have an effector or 'goal' in Houdini kerminology and we could add a knee up vector or 'kwist goal' in Houdini terms co solve for a given plane.


## Rigging - Kinematics

IK WITH CONSTRAINTS
In this mode the chain joint limits, dampening and other "constraints" are kaken in account in the solver.

What I mean is that if you dont activate the constraints mode when you set a bone limit it wont have any effect whatsoever.

But for all these modes you can configure them easily by hand after the fact so building an IK chain and adding the constrints on option is trivial. Just dive into the CHOP network and select the 'Solver type'.


## Rigging - Kinematics

FOLLOW CURVE
In this mode the chain bones are fit to the shape of a curve by using the 'Follow curve' solver.

If you manipulate the curve you will update the bones and this can be of great help for advanced systems.

Not to be confused with the tool in the shelf that creates bones following a curve as a guide but dont build any relationship between these two elements.

A Eypical thing Houdini artists do is transform the curve points and promote the handles as persistent so you get a similar effect to 'cluster to object' constraint in XSI.


## Rigging - Kinematics

As you can see Houdini provides a very convenient toolset for bones that is optimized and will cover all the bases easily.

The one thing I really love is that by having the solver in CHOPs you can also apply all sorts of signal analysis filtering to the motion, for example, you could bring your motion capture from the mocap provider as a group of points and build your rig to follow these but at the same time provide functionality to filter the data, lag, add springs, remove spikes, loop... totally mind bending tools here.

Furthermore, you can easily encapsulate these into an asset to distribute to your team of motion editors and they will edit the motion as required so suddenly what was a hand made process and a very painful one will be come a highly efficient one.

Let's learn a bit more about some of the things you can do with these systems.

## Rigging - Bones redux

## Rigging - Bones redux

So let's remember where we left it and tweak a few things now that we know the IK solvers can handle solutions on any section of a chain, suddenly there is no need for split chains.

Also, I like the idea of having having a simpler to maintain rig.

Furthermore, I have not taken advantage of Houdini and I should so I will change strategy also on the naming so you get the right experience.


## Rigging - Bones redux

See there are two chains on the arm?

Let's make it one, change the naming convention so we can use a much more efficient way to mirror things.


## Rigging - Bones redux

## So I delete my right side...



## Rigging - Bones redux

And connet the humerus bone directly to the clavicle, then I leave the effector there as I am going to use it for other things.

At this point I should point I keep calling these nulls Effectors although I only use them as nults, there is no IK implication on them so I will rename them later.

Ultimately what we called in XSI an Effector in Houdini is called a 'coal' so let's make sure we use the Houdini terminology when I use the IK approach.


# Rigging - Bones redux 

In this case I may re-do the leg to show you a trick.


Rigging - Bones redux
Split your viewport in a front/back approach and you will be able to draw on both, exactly as you do in XSI.

## Rigging - Bones redux

Let's delete the whole leg chain, as I said I am going to redo it for education purposes.

Bring the geometry again as I need a position reference.


## Rigging - Bones redux

Draw the whole leg as one single chain.


# Rigging - Bones redux 

Now we just position it, add the "effectors" (soon to be renamed) and use the new naming approach.


## Rigging - Bones redux

Now we just position it, add the "effectors" (soon to be renamed) and use the new naming approach.


## Rigging - Bones redux

For the spine, well... it is in the centre right? I will prefix it as 'C.' so these objects are sorted correctly when I have to weight the character which is always something I am keen.


# Rigging - Bones redux 

And this is how it looks before the symmetry.

As you can see we have shaved a few nodes.


# Rigging - Bones redux 

I select the bunch of nodes I want to symmetrize (watch out I have not selected the black node!!!).

And in the viewport I tab 'mirror' and press enter.


## Rigging - Bones redux

Now the bones preview appears (you can see my green effector there but not my bones due to the geometry being visible)

I change the prefix options on the mirror telling the tool to change everything called 'L_.' to 'R.' so names are managed for me in an automated way.

## Rigging - Bones redux

So I press enter and the nodes are created after a template review.

Withouk selecting anything I move these nodes to the side so I can organise my scene easily.


## Rigging - Bones redux

As you can see the names have been changed so I avoid any possible mistake with the naming, something that otherwise will haunt me during the job.


## Rigging - Bones redux

Do the same for the leg and I take a bit of time to color my nodes, put them inside the box, ebc...


# Rigging - Bones redux 

And do the same with the whole arms, just make sure you dont pick the top black box and again, a big tidy up operation but at least the names are ok.


# Rigging - Bones redux 

And this is how it looks? but let's configure the rig a bit.


## Rigging - Bones redux

Let's create some smart bundles to find the objects for me.

And I construct 3 bundles, one for roots, another for effectors (to be renamed too) and another for bones.

The point is that I want to see the bones, but not select them.

I want to see the effectors but not select them.

I dont really want to see nor select the roots.


## Rigging - Bones redux

So I get all the roots found and select the object nodes on the right pane, then click on the arrow and the eye to turn all of them off all at once.


# Rigging - Bones redux 

And the same for the bones...

As you can see there are many so it is good we can manipulate them in this way as again, eliminates issues.


## Rigging - Bones redux

Now I type a filter on the node editor to find all the "*. Effector" named nodes and voila... I can edit and see the results of the smart bundle automatically.

Filter scene by name


## Rigging - Bones redux

Finally I have now redone the rig to be a true 'No Kinematics' mode taking advantage of the native mirroring tool, naming mangling and show you how to manage it.

I must confess something I have been unhappy with is the naming of the phalanx on my XSI naming approach as it lead to things like;

Bone.-Thumbl. 1
Bone_Thumbli 2
Bone. Thumbl. 3

So filtering was annoying and lead to errors as it was hard to visualize.
L.-Thumb1_Bone

...is much better.

## Rigging - Capturing

## Rigging - Capturing

Okay so now we know how to build bones, we know how IK works and we should starting putting it all together but first let's see how to skin a character.

The main goal is to "bind" a mesh to the skeleton, in Softimage terminology this called "skinning", in Houdin is "Capturing" so we are going to do just that.

We are going to use the current dumb bone structure (meaning it does not use any solver, just plain hierarchical relationships) to drive the mesh.

At this point I must foreshadow what is going to happen, I have a rig that deforms the mesh, and a set of controls that drive the bones, my approach to rigging and for lack of a better name is more like onion layers so it is the control rig that will have all the actual logic.

My ultimate intention is to be able to change the rig without loosing the animation and on the way have a level of abstraction.

If we were talking databases this would be the same as calling a function that that touches the database instead of the very ugly calling the database directly.

So first of all let's duplicate our rig, and save it for later as we are currently dealing with the rig that will drive the mesh.

## Rigging - Capkuring

Now that we have the motion capture rig as a backup I intend to add extra functionality on the deforming rig (the one that will drive the mesh) so deforming wrists and ankles produces a better result, for this I will add two extra bones on the wrist and one extra on the leg.



## Rigging - Capturing

Because of the nature of Houdini this process is trivial, just copy the bone and parent it, reset the transoforms and change the length of all the three bones so they add to produce the same length than we had before.

Make sure the Reference null (our old Effector) expression is updated to be pointing to its new parent. (this is something we could automate with the 'opinput' expression function.


## Rigging - Capturing

And now on the wrist so we dont have a forearm pinching in the future.

Also let's change the expression for the Reference null.


## Rigging - Capkuring

So now that we have the skeleton ready we can bring the geometry here and "skin" the character (Capture the mesh in Houdini Lerminology)

Let's get a simple geometry object (remember we are still in object contex )


## Rigging - Capturing

Now let's bring the geometry we ingested with an 'object merge' operator and force the transformation to be that of the object we are in by typing a dot (.) in the 'Transform Object' parameter.


## Rigging - Capkuring

We now capture the geometry and for convenience I split the view so on one I see just bones, on the other the mesh.

Doing this will make the selection process very simple.

Then on the shelf we click 'Capture Geometry' tool and select first the mesh, press enter, then select the root of the hierarchy (in this case I picked my null C.coc although it is not a bone it does not matter... Houdini will traverse the structure and find the bones for me)


## Rigging - Capturing

What has been created I could have done by hand but using the shelf tools is a great workflow idea as it shows how certain tools could be put together.

Now the result is a node that reads the mesh and assigns the vertices to the bones using a given algorithm, then that data is passed to the 'deform' node that transforms the vertices taking in account the bone transformation and the vertex weight information.

The color coding is ugly because I have coloured the mesh but that is not important now so dont worry.


## Rigging - Capturing

If you look in the 'Capture' node you will notice the list of objects assigned is expressed as a simple string so you could actually take advantage of this if you want.


## Rigging - Capturing

As you can see the bones define "regions" of influence, you can edit those to have a quick character weighted so you can later refine it but will be a very good starting point.


## Rigging - Capkuring

If I rotate the humerus you will notice some weights are not working and these vertices will be left behind.

No problem, that is the case as we have not touched the bounding object to encompass all these vertices but it is easy enough to do it now.


## Rigging - Capturing

In the viemport type "Edit Capture Regions" and you will be able not to tweak with some very handy visual handles these regions that will change the weights.

## Rigging - Capkuring

All the parameters driven by those adjustments live inside the bone so feel free to explore it.

You will see the anatomy of the region is a capsule with two extremes formed by circles and a decay region that forms the "dome".


## Rigging - Capturing

Here now I can keyframe a few positions on the arm, back, head, legs, etc... and create a Range of Motion animation so I can evaluate the weights in all sorts of critical conditions.

Still some vertices left behind but let's fix them.



## Rigging - Capturing

Like this...


## Rigging - Capturing

And because I know I am going to be doing tons of adjustmens I go through just the main objects and link them together with expressions so my regions are mirrored automatically.

This way an adjustment on the left arm will be propagated to the right side.

It takes me 30 minutes but the weighting will be much easier as I know for sure the mesh is going to change in this case.


## Rigging - Capturing

So I remove the base color of the mesh so I can see things easily and test a few keyframes to see how the deformation behaves.


## Rigging - Capturing

Here are some key poses for the arms... imagine the same for all joints.


## Rigging - Capturing

So now is time to refine my weights and on the network I drop down a 'Capture Layer Paint' which will allow me to paint weights in a very similar fashion to XSI.

You have a brush, opacity, radius and various modes like paint, smooth, pick, erase, reduce, etc...

It is now to stick as XSI in terms of painting tools (for example you have only one brush rather than a brush per tool) but it does the job pretty well.


## Rigging - Capturing

You can also select your bones from the drop down menu or use the tool menues if you wish.


## Rigging - Capturing

So this is the result of barely 10 minutes painting.


## Rigging - Capturing

And I do the same for the rest of the body, after another 30 minutes I have a first pass that will be more than adequate for blocking.


## Rigging - Capturing

Now is time to symmertrize the weights using the 'Capture mirror' tool that will pass all the information from one side to the other.


## Rigging - Capturing

But of course we dont want Houdini to reprocess the whole tree we just created (or bigger) every time we load, we want an efficient, safe and reliable method.

The best is to freeze the data and save the weights and mesh with all the attributes you need onto a file.

Then give the author the opportunity to change from "Development mode" in which you re-evaluate the full network or "Production mode" in which you just pick the version on disk.

Something like this... Have a look the dotted line means it is not the branch evaluated in the switch, only the solid line is.


## Rigging - Capkuring

Now I change the switch to the second input and you can see the result is identical but faster.


## Rigging - Capturing

As you can see there is something truly powerful about the asset having its own history and definition, together with logic in it.

Having on every of your characters various modes of operation is trivial and wont require any scripting, extra plugins nor anything, all the description is inside.

Furthermore, if by accident these files have been deleted or need to be recreated it could be fairly trivial provided you have planned for that (and you should)

So this wraps the capture section, now is time to setup the whole rig as one single character.

Keep in mind what I described here is the tip of the iceberg, this is just one method and I have not even described Muscle based rigging as it is way beyond the scope of this transition guide.

## Rigging - Capturing

CAPTURE
hetp://www,sidefx.com/docs/houdini13.0/nodes/sop/capture
CAPTURE REGIONS
http://www.sidefx.com/docs/houdini13.0/character/captureregions
CAPTURE PROXIMITY
hetp://www.sidefx.com/docs/houdini13.0/character/proximity

CAPTURE LAYER PAINT
hetp://www,side fx.com/docs/houdini13.0/Modes/sop/capturelayerpaint
CAPTURE CORRECT
hetp://www,sidefx.com/docs/houdini13.0/nodes/sop/capturecorrect
DEFORM
hetp://www,sidefx.com/docs/houdini13.0/nodes/sop/deform

## MUSCLES

http://www,sidefx.com/docs/houdini13.0/hodes/obj/muscle http://www,sidefx.com/docs/houdini13.0/character/muscles

## Rigging - Setup

## Rigging - Setup

I guess is time to put things together, let's add some controls to manage the rig and to make sure we are in the same page I am going to explain the theory of how I favour rigging setup.

My approach is to favour an environment for animators so they dont worry about technicalities, about direct manipulation and making sure the rig is not intrusive so for this I avoid adding controls in the form of curves like you can see pretty much everywhere, I avoid any automatism like you can see everywhere, and make sure the "tactile" experience is consistent and "calibrated". By this I mean rotating arms should have a symmetrical setup and the ratio of rotation of an arm an the spine should be based on the range of motion that particular element has.

I this example I wont go that far as to calibrate the rig but I will try to at least discuss my base approach.

So the main concept is that I have always a set of bones deforming the geometry, they dont store keyframes but act only as a vehicle for transformation of geometry and the intelligence on how these bones rotate is handed by the deformation rig.

In its simplest form the control rig is there driving the deformation rig and has all the logic for IK and what not.

Let's see this in action.

## Rigging - Setup

Here you see my "control" rig which in this case is another skeleton rig is for illustration purposes separated from the deformation rig we captured before.

The idea is that we will keyframe the skeleton rig that will be representing the mesh via proxy geometry I have pre-cut on an earlier stage (more on that later).

But my question is why dont allow the animator to manipulate the geometry rather than the bones? well since a long time I have done just that, provide controls that are the segments of these limbs I want to animate.

## Rigging - Setup

The good new is that like in many cases Houdini nodes are made out of other nodes and in the case of the 'bone' node it is made out of two nodes inside (just dive in to see), the 'cregion' node (capture region), and the 'bone link' node that draws the representation of the bone as we see it.

But this means we can change it by adding a new node and displaying it, like this.

Here you can see the left femur bone is now displayed as geometry, I can pick it as if it was geometry but it is in reality a bone.


## Rigging - Setup

The way to do this is by diving inside and adding the geometry that has been "zeroed out" prior to the use.

This involves extracting the wrold transformation the bone has and invert it to make sure there are not doubling of transformations. I will suggest you download the qhib tools for Houdini which in my opinion are indespensible nowadays and use the 'Apply transform qL' node.

Then I just bring the geometry in and select the visibility and render flag for my file object.


## Rigging - Setup

I can do that with the rest and manipulate these bones to illustrate what is happening.

As you can see the animator's feeling will be a more direct and less intrusive control structure as they are indeed manipulating the geomery, or at least that is what they think.


## Rigging - Setup

here I have just edited all the bones, make sure the flag on the 'Misc' tab that says 'Viewport Selecting Enabled' is on so I can select these bones and we are ready.

I have not done the fingers as my approach is also different, I just dont want to distract the animator with finger positioning so I use a siplified yet very effective setup approach described later.


## Rigging - Setup

So now I have it if I change my viewport to shade you will notice the display seems broken (or weird at least) but dont worry just yet.

The point is that I want the animator to rotate these libs but seeing the final deformation mesh, this will give the impression that they are manipulating the deformed geometry. For this illusion to work I need these bones that look weird to be invisible in the vieuport, not render either but I still want to manipulate them, keyframe them too.
:-)


## Rigging - Setup

So let's apply for the moment a solid constant material to all these bones (yes, you can add materials to the bones themselves).

For examples you can see here the leg whike.


## Rigging - Setup

Once finished you will notice the finger bones are still there, I may choose to actually hide these bones as again, this will be manipulated completely differenkly through a panel.


## Rigging - Setup

And now let's change the shader so it is invisibible, for this you will have to change the alpha parameter on the shader's Surface tab and the Openct. tab.



## Rigging - Sekup

Then as you select you will notice the wireframe appears, to solve this you want to tweak the default for the viemport so the wireframe is also invisible.

For this set the 'Wire Blend' to o so they dont appear.

Now these controls are invisible although when you hover the highlighting still works which is a great thing as the animator will have a response that something can be selected.


## Rigging - Setup

Now in our capture geometry lek's remove the color coding attribute that has been created during the process that I wont need to see.

For this we drop a 'altribute' node that I rename and on the 'Point' tab I bype Cd (meaning color) into the 'Delete Altributes' parameker.

Now we are ready to use the character.



## Rigging - Setup

So now we have our "invisible" control rig, we also have the deform rig and need to connect them so the transformations done on the control rig is transferred to the deform rig.

For this we coutd rely on simple constraints with expressions but we there are two drawbacks on that approach;

- We will build a 1 to 1 hard relationship and I am after something more flexible.
- Evaluation of data will also be static meaning that there is no possibility of filtering the data passed from one to the other and given we have an amazing toolset at our disposal we should exploit it.

Therefore I am going to use CHOPs (channel operators) to do the transfer of data and as you will see will open the door to an amazing new world.

Regarding the "invisible controls" approach there is one drawback, the current implementation of Houdini on Mac OSX does not support these two tricks, the fully invisible trick and the wireframe blend so like me, you wont be able to use it for the time being on OSX although I hope this is fixed soon with the new revamped openct support.

## Rigging - Setup

So I simply put a keyframe on the femur bone (control rig) and drop also a CHOP subnetwork that will be in charge of processing the data and exporting it to the deform rig.


## Rigging - Setup

Inside the chop network I drop a 'fetch' node to retrieve the channels from the 'L..Femur..Bone' and analyse what is arriving.

As you can see there are tons of curves in the 'Motion view' so seems like we are getting just too much.


## Rigging - Setup

By selecting on the 'Channels' parameter the rotations using the;
$r ?$
syntax we get $r x, r y$ and $r z$.
Now you can see in the 'Motion view' just those curves.


## Rigging - Setup

And if I drop an 'Export' node, Eype the node path to the object I want to receive the data and select which channels to send (in this case r?), after activating it you will see the deform rig will be modified and the leg move.

I have also put a few extra keyframes on the control rig to see the difference better.


## Rigging - Setup

If I do the same for each bone I will be able to build this sort of constraint mechanism.

At this point I must add that CHOPs are extremely efficient, after all from what I know it was built to perform in real time and therefore processing data in this context is really a good idea if you are after flexibility and performance.

Now Let's carry on.


## Rigging - Setup

Because I want to exploit the name based approach to getting and passing data I have put an expression on the Node path that picks the node name, strips the "export.". string out and substitutes it by nothing leaving me exactly the name of the bone I am about to export data too.

I do the same for the fetch so ultimately the result is that the name of those nodes will retrieve and pass the data accordingly to the appropriate bones.

The dramback is that you have to be aware that changing the name of the bone will break the system unless you go and change all these string operations so be aware.


## Rigging - Setup

I carry on adding keyframes on all the control objects to check the data is passes correctly and keep copy/ pasting nodes and renaming them to so the connection gets going.


## Rigging - Setup

Finally once everything is done and exported I tidy the scene up, color coded as the bones and move to the next step.
(H)Houdin fx File Edil Render Window Takes Help


## Rigging - Setup

For the sake of illustration the activation of these constraints is something I could control in a global way.

For this I change the parameter interface of the control rig to hold a switch that will allow me to ackivate/ deactivate these exports.


## Rigging - Setup

Here you have it.


## Rigging - Setup

And I apply a channel reference on each of the export nodes.

Like this.


## Rigging - Setup

Now I build a simple pose with translations as I have not done that yet.


## Rigging - Setup

And in this case instead of retrieving the rotations I also get the transtations, like this;
t? r?
and export the same.


## Rigging - Setup

Resulting in this.


## Rigging - Sekup

The only thing missing will be the control of eyes and fingers but at this stage you can easily set that up anyway you wank.

Also I have not built the IK side of the rig so why dont we do that now?

To setup a control rig that can switch from FK to IK we will have to do a few more things but give the control rig is actually a skeleton this will allow me to use a simple setup.

I am not after having IK/FK blending and a complex relationship, this I will leave to you.

The goal is to set the IK setup for legs and arms, still the deforming rig will not require anything extra, remember is a dumb rig.

## Rigging - Setup

I just created the controls that will manipulate the rig and put them in place so I dont bore you with the process.

Basically I got a few nulls with the sphere shape, others with cubes and line them up to the wrist reference nulls (finally makes sense right?).

After that with compensation on I did parent them to the root of the chain so they travel correctly when I transtate or rotate the character as a whole.


## Rigging - Setup

Have a closer look to the new 'L.Arm_ IKGoal' and 'L., Elbow IK Twist'.

Now the process is basically to wire these up with a solver that can handle IK solutions with Twist in account for both knee and elbows.


## Rigging - Setup

So I create a new CHOPs network to hold the solvers and drop a 'IK solver' and fill the various parameters with the corresponding nodes.

Remember we are not done yet, we need to tell the bone to ask the solver for a solution.


## Rigging - Setup

And we do that by going to the bone and in the 'Bone' tab we refer to the Kinematic solver we are interested.


## Rigging - Setup

At this point I want to offer the user an IK blending tool per limb.

For this I create on the control rig subnetwork a few parameters and set them to 1.

These parameters will be driving the blend but I want to drive the user and help him so I will also control the visibility of these objects accordingly.

Meaning, if you set it to IK I wont distract the user exposing the FK controls and viceversa.


## Rigging - Setup

And on the solver I paste a relative expression to that parameter.



## Rigging - Setup

Now let's test it by turning the IK blend off and see the leg straightens up all while the IK leg control is on a different place.

Now is time to tidy this visual effect up.


## Rigging - Setup

So on the bone itself I turn on the visibility flag and add an expression like this.

As you can see when the IK is on the object that controls the femur has disappeared.

Once again I will remind you that the arkisk will be seeing at all kimes the deforming mesh.


## Rigging - Setup

And I set it for every limb and bone.


## Rigging - Setup

And when I turn the IK blend off I wont see these controls.

Right, we have now everything ready, now would be a matter of encapsulating these to rigs under a digital asset to distribute.
of course you may want to expose these controls on the HDA and probably a few more things you may want to add like Eye controls and hand controls too.


## Rigging - Setup

Here you have the same character on a Linux box, as you can see the overlay of the "invisible" object works much better (although not perfect) but I dont have any floating curve and the animators can focus on the character rather than the controls of the character.

Notice how the character's right leg is stightly yellow, that is the effect of hovering the mouse over the "invisible" geometry, still selectable.

## Rigging - Selup

A very interesting possibility would be to have a deform rig with the multiple resolutions, or even muscles.

This could also be developed by more than one artist leading to a faster turnaround and the control of these various elements could be wrapped also on the digital asset so as you can see splitting things in this manner is actually pretty efficient.

Of course you may want to approach in another way, that is fine... for example I never use foot rolls and I wont go into why is that but you could build that mechanism very easily.

## Rigging - Sekup

of course there is a lot more to rigging, I only showed a few little things but I am sure those of you that want to go deeper will enjoy it a lot. Make sure you buy the 3DBuzz tutorials which are the best by far.

BDBUZZ HOUDINI TUTORIALS
hetp://www,3dbuzz.com/training/topic/houdini?resetFilters=True

3DBUZZ TECHNICAL RICGINC
http://www,3dbuzz.com/training/view/houdini-rigging-volume-1/skeletal-systems

3DEUZZ ADVANCED CHARACTER RIGGING
http://www, Bdbuzz.com/training/view/h9-advanced-character-rigging/character-setup-limb-control

CMIVFX CRAFT RIGGING
hetps://cmivfx.com/store/589-houdinitcraft+automobile+system+volume+1

CMIVFX AUTORIG TUTORIAL
https://cmivfx.com/store/633-houdinitriggingtanimation+fundamentals

## Animation

## Animation - Introduction

## Animation - Introduction

As you can imagine animation is like every other package a collection of basic tools, nothing funky here so my intention with this chapter is more about showing you where things are.

The key tools are;

- Animation Editor
- Dopesheet Editor

There are many other things you will want to do that are not part of the toolset in Houdini although I hope some day soon these are included but you wont see;

- Animation Layers
- Animation Mixer

Instead Houdini provides a context that will allow you to build these tools and much more therefore I will put some times in showing you how CHOPs works.

## Animation - Introduction

Let's start comparing the 'f-curve editor' or 'Channel Editor'.

In this simple example a ball is doing a bounce and as you can see a curve is made out of points with handles that can be manipulated to control the tension of the curve, if it is a continious curve, etic...

Of course the equivalent in Houdini is pretty similar although there are a few differences.


## Animation - Introduction

In Houdini the same setup is built and as you can see the 'Channel Editor' provides similar functionality.

One thing I particularly like in Softimage is that keyframes and curves mimic the object and vertex paradigm which makes manipulating them very natural.

In Houdini that is not the case and it is fair to say the interpolation is not as well done as in Softiamge.

The same for exptrapolation of curves, the setup is strange to say the least.

Nevertheless it is easy to adapt and move on as Houdini brings some useful tools too.



## Animation - Inlroduction

For example, a very useful tool is a spreadsheet like behavior to the channel editor that allows you to see the values on many channels and edit them in a quick way.

## Animation - Introduction

Or for example you can see the various curves lined up one on top of each other, something very useful when comparing mulliple channels.

Furthermore, something I love is the ability to group various curves onto group folders and save these on the left pannel (the explorer looking one) so you can access very complex selections in one click.
( Houdini Fx File Edit Render Window Takes Help O


## Animation - Introduction

You may want to configure a few things on the 'Channel Editor' so the curves are rendered in high quality, the interaction is live, etc... a bit like Softimage.


## Animation - Introduction

The 'Dopesheet' is one of the most important tools for me and specially due to the way I animate.

I like to have key poses and then follow a process reminiscent of cel animation in which I build my breakdowns, inbetweens, etc... I dont tend to offset key but poses and for that the f-curve is of not very useful but the dopesheet certainly is.

Softimage dopesheet is quite good but not great so let's have a look at what Houdini brings to the table.


## Animation - Inlroduction

At first looks a bit worst but the true power comes when you start using groups so you can use to do all your overlapping easily.

Interface wise is not great but it is not great either in any package I have found so not much here.

But one second.


## Animation - Ineroduction

You can use object scopes to look at various objects, filter these and use various templates... let's have a look at that.

First though, you can move the keys the same way as in softimage and the timeline.

## Animation - Ineroduction

Here you can see you can see the various channels by filtering them using the "Template" filter.

This is a way to block the editing of certain channels.


## Animation - Introduction

On the other hand you can filter common configurations of the channels using the Filter tool.

## Animation - Ineroduction

And as you can see changing the scope will allow me to see multiple objects at once.

Just look at the bottom left side and you will see;
lobj/sphere
and
/obj/ipr_camera/xform1


## Animacion - Ineroduction

The 'Channel Editor' hides some really interesting functionality as a grapher, something Houdini artists use a lot more than I though at the beggining.


## Animation - Introduction

Complex graphs, even noise, are displayed and are very useful to undersland that is happening with complex expressions.


## Animation - Introduction

Furthermore, the segment defined in between two regular keyframes can have its own expression, allowing you to simplify your animation by using complex functions...

## Animation - Introduction

...or even giving you the freedom to add procedural nose per segment for example.


## Animation - Introduction

As you can see it is a tool to embrace and exploit, lots of interesting things aside of normal keyframing are available and you should probably have a look at the documentation and dive into it withou delay.

HOUDINI CHANNEL EDITOR<br>hetp://www.sidefx.com/docs/houdini13.0/ref/panes/chaneditor<br>HOUDINI CHANNEL GRAPH<br>hetp://www.side $f x$.com/docs/houdini13.0/ref/panes/changraph<br>HOUDINI CHANNEL TABLE<br>hetp://www,sidefx.com/docs/houdini13.0/ref/panes/chantable<br>HOUDINI CHANNEL DOPESHEET<br>http://www,sidefx.com/docs/houdini13.0/ref/panes/dopesheet

## Animation - CHOPs

## Animation - CHOPs

Channel operations are one of the pillars of Houdini and one extremely flexible tool that can be used in a wide range of tasks, from signal processing to animation mixing, from midi operations to ik solve.

My intention is to give you a tiny introduction to CHOPs rather than writing a book about it as there is already a good book on the subject you surely will want to buy.
hEtp://www,sidefx.com/index.php? option=com content $\ddagger$ task $=$ view $\$$ id $=1268 \$$ Itemid=66

The main areas I use it for are;

- Audio processing for animation
- Animation processing
- IK processing
- Math operations

But you could use it also for

- Image processing
- Time warps
- Phonetic conversion
- Piping data in/out of Houdini to other processes
- Logic operations
- Noise generation
and more I cant even think of...
So let's start with the basics and I let you explore it further.


## Animation - Introduction

Let's start with an example to introduce you to the concept of procedural animation which CHOPs provides.

In this case I am driving the animation of an object being deformed through a path using the data from an audio track.

The idea being that things are in sync and a change in the audio track will imply a simple change.

The look is graphical but that is on purpose.

As you can see there is a chain of operations before the data is exported to the outside world. (another object in the scene)


## Animation - Ineroduction

So after loading the audio file in WAV format (in this case) I do a noise reduction by using a 'Pass Filter' chop, and yes, you may want to think CHOPs is your signal processing engine and more.

## Animation - Introduction

In this case with the audio mended a bit I proceed to convert the audio data into a sample curve.

For this I use the 'Envelop' chop that separates both audio channels into two different curves (in this case slightly different in shape).


## Animation - Introduction

And because I dont know the audio sample rate I make sure I refine the curve to something appropriate for the project, in this case sOHz .

So now I have two channels sampled at 50 Hz ready to be used.


## Animation - Introduction

The idea of this little demo project is that I want to go from something bigger than zero to the maximum so I limit the minimum to 0.099

As this geometry is deformed by a curve and the channel I am preparing is going to be fed to the scaling of the mesh along the curve I want to make sure I have a small piece of geometry to deform and the reason for the limit.


## Animation - Introduction

I may have finished with the pure manipulation of the audio but I want to make sure the movement is a bit more graceful so smoothing the curve makes a lot of sense, for this I use another 'Envelope' chop to provided that.

Using one of the many tools (in this case I changed the 'Envelope Bounds' to "Signal Power" I get what I am after, something violent but graceful.


## Animation - Ineroduction

And of course to make my life easier later on I create a multiplier that servers the purpose of a global dimmer/booster so I can get the right feeling, in the first part of the track I want a big surprise effect, on the second just a reminder so I also animate it.


## Animation - Introduction

And finally export the data to an object chop node and indicate which channels I want to export and to which parameter they should be going to.


## Animation - Introduction

In this case the data is sent to this 'pathdeform' tool I got from Orbolt and in particular to the 'To' parameter that controls where is the deformation arriving in relationship with a curve.

If you want the tool just visit this link;
hetp://www,orbolt.com/asset/aswaab::soptools pathdeform
Which is similar to the Softimage "Deform by curve" operator (although I had to add a bit of functionality myself to match it)... I just cant live without that tool.


## Animation - CHOPs

CHOPs was developed for high performance operations so the more you use it the better and faster your scene will be.

Not only that, it will be more modular and therefore you will be able to reuse it and since you can embed a CHOP subnetwork in your asset it is obvious it is the perfect tool to manipulate motion capture, audio and also pipe that data to whatever object you are interested.

So let's look at CHOPs from another angle, let's modify the geometry of a polygon mesh using this wonder tool.

Let's start with a simple grid, configured to 'Mesh', then let's add a null that will serve to our bridge to the 'chop network' we have embedded.

NOTE: This example is inspired in Jeff Wagner tutorial as I found it a true revelation. Link in the final page of the guide.

## Animation - CHOPs

Sometimes is hard to understand the relationships of many many nodes put together in an apparent random order so why dont we force Houdini to "graph" for us the influence on Subnets by pressing ' $d$ ' in the network editor and on the 'Dependency' tap we turn on that?

Now you have a nice line connecting these nodes.

The thinking is that my grid may change later on for something else so having a null as the bridge that passes the data is a very very convenient approach which will allow you to work much faster.


## Animation - CHOPs

Here I have built a simple network that gets some geometry from the outside wortd, in this case from the null we just saw and applies a nose to the $Y$ component of every point.

For this I use the 'Geometry' chop that will bring the $P$ attribute (point position) into chops for all 3 axis, $t x$, ty and tz.

As you can see the graph in the 'motion view' window is displaying rows and columns that are related to the point order of the grid.

Genius.


## Animation - CHOPs

Then on the noise node I configue the type of noise, the amplitude, offset and period to match my taste... something I will go back later to make sure I am happy.

Conveniently I have renamed the noise channel 'ty' so the addition I am about to perform between to channels "by name" works.


## Animation - CHOPs

And on the 'Math' chop I add the ty component together as I dont want to affect the other components, meaning I just want a vertical displacement of points.

I still cant see the outcome of the whole operation so I will now go back to my SOP context and carry on.


## Animation - CHOPs

And here I toggle on the dependency graph as we did before to see where the data is going to after the CHOP network.

And as you can see the channell SOP node points to what has been processed inside the network called 'chopnet1'.

Now the deformation is visible on the geometry and I can start tweaking the operations until I get the look I am after.


## Animation - CHOPs

Of course you can use a huge range of tools like waveform generators, layering of data by means of addition, etc... you can filter the data in many ways, apply audio filters and as you can imagine building an audio equalizer is pretty simple in Houdini.

JEFF WAGNER CHOP TUTORIAL THIS FINAL SECTION WAS INSPIRED hktp://www, sidefx.com/index.php? option=com conkent\&kask=view $\ddagger$ id $=573 \neq$ Itemid $=216$

LIST OF CHOP NODES
htep://www.sidefx.com/docs/houdini13.0/Modes/chop/

CESTURE CHOP
help://www, sidefx.com/docs/houdini13.0/nodes/chop/gesture
SHIFT CHOP
hetp://uww, sidefx.com/docs/houdini13.0/Modes/chop/shift

PHONEME CHOP
hetp://www,sidefx.com/docs/houdini13.0/nodes/chop/phoneme
LOOKUP CHOP
hetp://www.sidefx.com/docs/houdini13.0/nodes/chop/lookup

## Animalion - Devices

## Animation - Devices

In XSI you have the possibility to use your mouse, keyboard or a PC1600 Midi controller.

I have use them countless times in production by capturing a live stream of data that is later manipulated, from the mouse I piped the data to the forces of a physics simulation, from the keyboard I used it to trigger actions and 'edit' the motion capture live, from the PC1600 I use it to map facial muscles to the sliders and keyframe it from there.

So if you enjoy technology as I do I will do a very quick overview of how to use external devices to get data into Houdini as you will find it is another dimension altogether.


## Animation - Devices

In XSI if you go to the animation module you will find the 'Devices' menu which allows you to record data from a number of limited input devices.


## Animalion - Devices

By selecting any of the standard 3 devices you can record daka which can later be used for a number of things.

If you want more devices you will have to write your own driver (not hard but annoying) and the way the data is used is mostly a "fcurve generation" mentality.


## Animation - Devices

In Houdini this goes beyond it because it is just another channel input and as in XSI you will find the keyboard and mouse...

Here 'mouse', 'keyboard' and 'audio in' can be used to create channels, trigger events, count, etc...

The 'record' chop will store the data which can later be processed further in a number of ways.

It is interesting to note that using the keyboard various modes you can do complex flow control on your network which will be really liberating experience.


## Animation - Devices

For example with the audio in you can see mee talking and generating a channel on the fly.


## Animation - Devices

And with the mouse you will see my $x$ and $y$ coordinates are mapped to two new channels called Ex and Ey.

Houdinifx File Edit Render Window Takes Help


## Animation - Devices

I could also use MIDI IN/OUT to pass data to various channels and these could drive (Let's say) the facial muscles of a character.

## Animation - Devices

And if we consider another computer a device I why not use the 'network', 'pipe in' and 'pipe out' chop nodes to send and receive signals?

The idea of having a computer listening to another either via a network port or a pipe (read about linux to know more) is fascinating as you could actually build easily clusters of computers working together towards one solution.

In this case my computer "flat" is receiving data through the port $\$ 000$ which I would use to drive (for example) the eyes of the character in real time for a show.


## Animation - Devices

And here the 'pipe in' node is reading from a pipe file located in t tmp, again I could use two machines running houdini passing data from one to the other in this fashion.


## Animalion - Devices

Furthermore you can use it with other tools like the 'phoneme' chop to be able to do quick lipsync and although it is not strictly a device is the combination of text and live input that I find fascinating.

Here is the curve generated by the world "WELCOME".


## Animation - Devices

The best recomendation I can give to anyone getting into procedural animatin and particularly CHOPs is to buy "Simultaneous Music, Animation and Sound Techniques with Houdini" and also have a Look at Derivative's Touch Designer 088 which is the real time engine for events based on Houdini.

## HOUDINI CHOPS BOOK

http://www,sidefx.com/index.php?option $=$ com content $\$$ task=view $\$$ id $=1268 \$$ Itemid $=66$

DERIVATIVE<br>hetp://www,derivative.ca<br>INPUT SOURCES REFERENCE<br>hetp://www.billbuxton.com/InputSources.heml

## Texkuring

## Texturing - Introduction

## Texturing - Introduction

Houdini, like any other 3D application use the same principles for UV creation so it is not surprising they are similar-ish, only naming and "friendliness" will be different.

Anyhow the same situation happens with Houdini than with XSI, Maya or anything else pretty much, creating UVs is so good and efficient using UVLayout that I would recommend to still use that route.
but...

Of course the point of Houdini is that it is procedural and if you want to take advantage of it, that is a very good reason to invest a tiny bit more time and actually do your UV creation in Houdini.

Furthermoe, UV creation can be used for many other things so having the data generation in Houdini makes a lot of sense.

So let's start with the very basics.

## Texturing - Projections

## Texturing - Projections

Let's review the types of UV projections available in XSI (not discussing the unfold one)


## Texturing - Projections



## Texturing - Projeckions



## Texturing - Projections

## And their equivalent in Houdini

Orthographic YZ projection


Orthographic XY projection


Orthographic XZ projection


## Texturing - Projections

Polar projection


Spherical projection


Rows \& Columns projection


## Texturing - Projections



Face projection


## Texturing - Projeckions

These are the equivalences between XSI and Houdini so you can better identify the tools.

| XSI | Houdini | Houdini Node |
| :---: | :---: | :---: |
| Planar ( $X Y, Y Z, X Z$ ) | Orthographic ( $X Y, Y Z, X Z$ ) | UV Texture \& UV Project |
| Cylindrical | Cylindrial | UV Texture \& UV Project |
| Spherical | Polar | UV Texture \& UV Project |
| Spatial | $n / a$ | n/a |
| n/a | Plastic Wrap | UV Project |
| $n / a$ | Rows \& Columns | UV Texture |
| Cubic | UV unwrap | UV Unwrap |
| Camera | Camera | UV Project |
| Unique | Face | UV Texture |

## Texturing - Projections

So as you can see is pretty much the same, only thing you will miss are the very useful Cubic projection $\$$ spatial projection.

How to achieve these? will go into that later on the next guide.

But let's dive into some of the differences on the setup to achieve these and how can we get projection handles.

## Texturing - Projeckions

The important thing to remember is that you have two nodes to keep in mind;

- UV Project SOP
- UV Texture SOP

Let's start using the UV Texture one.
Houdini does not preview automatically the UV space like XSI does so apart from adding the projection node you have to add also a visualization one..


## Texturing - Projeckions

So by adding a "UV Quick Shade SOP" node we can how visualize what is happening with our projection and thus have a much closer experience to that from XSI.

The point is to edit the projection node but look at the visualization node... of course you can have multiple branches going on in parallel to compare things so exploring options is a breeze.


## Texturing - Projections

All the options are in the 'Texture Type' drop down so I will suggest you play with it.


## Texturing - Projeckions

If we now look at the 'UV project' node you will see the functionality is similar but be careful, it is not.
'UV Project' is the equivalent of Texture projections in XSI and has manipulators, can be animated, linked, etc.. and I tend to favour it.
'UV Texture' is the equivalent of the cluster creation in XSI, meaning the UV attribute is defined on the spot (the same as freezing you texture projection in XSI).

Please bear in mind I have not fully explored these and I may be missing something.

## Texturing - Projections

Now if you enter into editing mode for the 'UV project' tool you will be able to see and manipulate the actual texture projection.

So this is very similar to XSI as you can see.


## Texturing - Projections

For example, the rotation of the icon will produce these UV setup.


## Texturing - Projeckions

There are a few things you want to make sure you keep in mind when using UV texture projection in Houdini.
Like any other 3D software the approach is having points to define the geometry and unique vertices per face to hold the texture, and you can choose how to apply your projection, either the geometric points or the vertices.

The issue is easy to see, as points are shared when they wrap the texture indexes are mapped badly always on that joint so try to use always vertex mode and the issue will be solved.

Point projection


Vertex projection


## Texturing - Projeckions

If you use the fix boundaries with polygon meshes you will see they do an excellent job but it is not enough...

Fix boundary seams OFF


Fix boundary seams ON


## Texturing - Projeckions

Even if you subdivide it to death you will see the 'fix boundary seam' does not completely solve this well known problem every single 3D application have.

Fix boundary seams ON with few subdivisions


Fix boundary seams ON with lots of subdivisions


## Texturing - Projeckions

Another interesting approach with some objects is to generate a nurbs or primitive version of these, apply the UVs using the Nurbs tools and then convert them to polygons which will result in really good UV/s for free pretty much.

In this case the Texture type is using the 'Uniform Spline' to get the job done.


## Texturing - Projections

So far these are the very basics, I will move into UV manipulation, Unwrapping and pelting on the next guide so we can see the potential of these tools.

## HOUDINI UV TEXTURING TUTORIALS

hetp://www.sidefx.com/index.php?option $=$ com_content\&task=view\$id $=996 \neq$ Itemid $=265$

> PT QUINT UVs $1+2$ UV PROJECT hetps://vimeo.com/12547162 + https://vimeo.com/12547587 http://www.sidefx.com/docs/houdini13.0/shelf/uvproject UV TEXTURE hetp://www.sidefx.com/docs/houdini13.0/hodes/sop/texture UV UNWRAP http://www.sidefx.com/docs/houdini13.0/shelf/uvunwrap UV QUICKSHADE

## Texturing - UV Manipulakion

## Texturing - UV Manipulation

Now that we have the basics it is time to look into manipulating these UVS in a semi-traditional way.

It is a worry of mine always how truly procedural my work is as these operations require hand input, at some point like I did you accept the fact that it just cant be $100 \%$ procedural unless there is a major effort so I only invest the time on the situations where I am building a tool rather than a model.

Also you will sooner or later want to color code or tag somehow those nodes that are direct user manipulations and also those that require user input parameters so you can idenfity where in the big long network those moments are.

## Texturing - UV Manipulation

UV EDIT
This node allows you granular (per vertex or per poink) maniputation of the UVs.

As usual these are really cumbersome operations and I try not to do them but of course, it will eventually happen.

In this model (from a scan) there is a set of UVs already prepared so if you put down a 'UVEdit' node you will be able to manipulate them.


## Texturing - UV Manipulation

Just go to the viewport and press 'Space+s' to swap view to UV space.


## Texturing - UV Manipulation

Here I selected a bunch of vertices and moved them.


## Texturing - UV Manipulation

## UV BRUSH

This tool allows for more artistic friendly manipulation of UVs and although is not as nice as UVLayout it does a pretty good job.


## Texturing - UV Manipulation

If you enter the tool you will be ready to paint and if you right click the pop-up menu will appear with the options for the tool.

You can select between drag, smooth, size of the brush, etc...


## Texturing - UV Manipulation

## UV PELTT

Houdini has had pelting on their arsenal months after it was introduced at Siggraph and over the years this has been the case with many other technologies that find a very early adopter, Alembic, VDB, etc...

So UV Pelt that exactly what is says in the tin, pelts away the mesh.


## Texturing - UV Manipulation

My preferred way to use it is to preselect the edgest to cut in a group that I can later switch quickly.

In this case as I have a model that already has UVS I am going to also show how to delete them.

So I inspect with the middle the node information and you can see amongst quite a bit of information that it has Point attributes ( $P$ ) and vertex altributes (uv).

Meaning you have points and a UV set.



## Texturing - UV Manipulation

And the UVS dont look nice to me SO...



## Texturing - UV Manipulation

... I drop an 'Attribute' node that can be used to create but also delete them and on the 'Vertex' tab I select on the drop down the "uv" attribute.

Now we have blasted them.


## Texturing - UV Manipulation

I select the edges, create a group from the viewport typing "Tab>Group" and the group is created and connected with the pattern of edges I selected.

At this point I add the 'UV Pelt' node and the only thing left is specify the group to use.


## Texturing - UV Manipulation

Then I add a 'UV Quickshade' to see the effect and tweak the parameters on the 'UV Pelt' to give me something on the lines of what you can see on the right picture.

The "Hint Primitive" is a way to select which "fragment" of the geometry will be pelted (represented by a blue square on the geometry).

Please make sure you spend a bit of time with some simple geometry to get to understand a few things.


## Texturing - UV Manipulation

And here you can see the quick result I produced.
of course this is just an illustration of the tool so remember you may want to spend more time and get a much more uniform result than mine.


## Texturing - UV Manipulation

A good thing to remember with UV generation is that actually it is very useful to save the data to disk and make sure you dont overwrite it as other tools may play a part (like Mari) as want these to get final.

This is a good example of "Data freeze" I was talking on the "Data Management" guide.

Also you can rely on houdini attribute transfer to mimic XSI GATOR tools so transferring the UV from one mesh to another is not very difficult thing to do.

The question will remain... do I use the UV toolset in Houdini?
of course but for many things I prefer UVLayout so it is always of the piece at work that will dictate which process is better.

By the way, there is a very nice UV generation tool in Orbolt that you should check out as this is a good example of what you can do with Houdini in a procedural manner.

TdG UVgen
https://www.orbolt.com/asset/Twan_de_Graaf::TdG_UVgen

## Texturing - Projeckions

And once again, a few interseting links to visit.
UV EDIT
hetp://www.sidefx.com/docs/houdini13.0/Modes/sop/uvedit
UV BRUSH
http://www.sidefx.com/docs/houdini13.0/nodes/sop_state/uvbrush
UV FUSE
hEtp://www.sidefx.com/docs/houdini13.0/Modes/sop/uvfuse
UV PELT
hetp://www.sidefx.com/docs/houdini13.0/Modes/sop/uvpelt
LAYER SOP
hetp://www,sidefx.com/docs/houdini13.0/hodes/sop/layer

## Rendering

## Rendering - Introduction

## Rendering - Introduction

Houdini's rendering with Mantra deserves a book in itself so I wont try to do much other than point to some good setting and concepts to keep in mind.

Also since Mantra shares a lot of concepts form Renderman my suggestion is to read a bit about Renderman and you will get a better insight on Mantra.

So let's start point to the key elements you need to know in order to produce good looking, artifact free and efficient rendering first, then point to some advanced features you will explore for yourself later on.

## Rendering - Introduction

Remember your ROP context (Render OPerators) is more than a renderer, you have nodes that do not render but affect how it is rendered (like batch or shell).

Therefore I will suggest you explore the Mantra render node in combination with other nodes to make sure you get a good picture of the power of ROPs.

Once again, remember the network does NOT describe hierarchies but data flow and as such you can split and join building some truly automated processes that will allow you to go home early.

Yes, go home early.
(Because the computer is that one that should be working 24/7 rather than you)

## Rendering - Overview

## Rendering - Overview

Let's explore the various settings in the Mantra node first.
You will see it is divided into various sections

- Main, to configure the command and camera along with the IFD file generated (the baked scene desciption) and a few other execution options and importantly the "show in Viemport Menu" flag that will allow you to either show this ROP on the viewport drop-down or not.

18, Mantra mantral : H. (1) (3)
Asset Name and Path ifd /Library/Frameworks/Houdini.frameworkNersions/13.0.256/Resources/houdini/otls/OPlibScripted.ot


```
    Valid Frame Range Render Any Frame
        Start/End/Inc
    Render With Take (Current) \hat{ }
Main Objects Properties Scripts 
        Camera /obj/caml
        Command mantra
        Disk Fil
        [Block Until Render Complete
        I Create Intermediate Directories
        Initialize Simulation OPs
        \ Show In Viewport Menu
Valid Frame Range Render Any Fra
Render Any Frame f
Render With Take (Current) \(\hat{\rightharpoonup}\)
\begin{tabular}{|l|l|l|l|}
\hline Main & Objects & Properties & Scripts \\
\hline
\end{tabular}
Camera / \(\mathrm{lobj} / \mathrm{caml}\)
ommand mantra
\[
\begin{aligned}
& \text { Block Until Render Complete } \\
& \text { Create Intermediate Directories } \\
& \text { Initialize Simulation OPs } \\
& \text { I Show In Viewport Menu }
\end{aligned}
\]
```


## Rendering - Overview

- Objects, to select which objects and lights will appear on the render in various forms.

Be aware it is common to explicitly refer to objects by their full path it is IMO a very bad practice.

The same way you will be better abstracting your filesystem references with environment variables you should abstract your scene references.

The way to do that in Houdini is by using Bundles which are equivalent to XSI groups.

This means adding one more object on your 20 ROPs is a single operation of adding an object to a bundle instead of changing 20 ROPs..

## Rendering - Overview

- Properties, to configure the specifics of the engine, how to sample the scene, arbitrary outputs (framebuffers) and a big number of parameters you will get familiar sooner than you think.

More on this Later as it is truly big.


## Rendering - Overview

- Scripts, which will allow you to run pre and post render scripts, pre and post frame scripts.

Using this you can create folders if they dont exist, log the render in your own way, copy the frames, etc...

Now that we have seen a quick overview let's dive into some important aspects to keep in mind when using Mantra as your renderer and specially the Physically Base Render engine.

```
M, Mantra mantral
    **)
Asset Name and Path ifd /Lbrary/FFrameworks/Houdini:frameworkNersions/13.0.256/Resources/houdin//otis/OPlibScripted.ot
Render Render Control.-
    Valid Frame Range Render Any Frame 合
        Start/End/lnc
    Render with Take (Current) &
MMain Objects Properties Scripts
\ Pre-Render Script \square & & Hscript &
\checkmark Pre-Frame Script \square\ & & Hscript %
\checkmark Post-Frame Script 
\checkmark Post-Render Sc.. \square\ Hscript &
```


## Rendering - Overview

Diving into the Properties you will see a number of subgroups, these are all related to the Mantra engine so be aware if you use Renderman, 3DLight, Arnold, etc... this will be totally different.

So in the Output tab you will see the picture defaults to "ip" which is the interactive viewer, meaning the picture is not saved on disk. If you want to save it on disk you will want to put your file path here.

The pixel filter defaults to "Gaussian 22" which is great for draft mode in a production but may want to use "Sinc" filtering instead which sharpens a bit the image although it is a bit slower.

Quantization refers to how Mantra full fledge computations are stored into a smaller bit depth saving disk space. Nowadays with EXR as the standard it is common to use 16 bit float but if you want to produce sbit for example this will be your setting.

## Rendering - Overview

When in the 'Output Options' you will see I have added two environment variables to the ROP, this is stored in your image so it is handy if you are using a renderfarm which machine dealt with it and which artist is the author of that render.


## Rendering - Overview

In the 'Render' tab you will select the PBR engine which is the one I normally use... very similar to Arnold.

The important thing to remember is that you have some memory management tools in here and configuration flags in case you want to decouple maps generation (may be a two stage render approach?)


## Rendering - Overview

In the sampling is the meat of the process and gets very interesting.

The critical relationship with XSI is your;

- Pixel samples which determine the number of pixel samples used to determine the pixel color. Big numbers up to $16 \times 16$ are possible but beware this will have a big impact on your render times.
- Min/Max Ray samples, when using 'Ray Variance Aliasing' you will determine the min/max of ray samples and again the impact on your rendertimes will be profound.
- Noise Level, when reducing the noise level you will force the sampling to be increased so be aware this will again have a huge impact on your render times.

Make sure you switch off the 'Sample lock' which will store the first frame sampled frame and reuse it producing a paper like effect that it is only useful for quick previews.

## Rendering - Overview

In the Shading tab you will see also various critical elements to be configured that will have a major impact on your render times and image look so beware this should be something to define early and stick to it (if possible)

The Reflect and Refract Limit are obvious and easily overriden at shader level so you could actually optimize the scene heavily but using this approach but be aware without that kind of control on a per shader basis your render times can skyrocket (with glass like materials specially)

The Diffuse Limit is your second light bounce and IMHO critical for realistic rendering but be aware more light will be introduced in your scene and therefore the image will change.

Color space will drive sampling optimisations taking advantage of how is this image going to be viewed later so try to leave it in Gamma 2.2.


## Rendering - Overview

Raytracing Bias is your way to tell the render the size of the world so if you use very small scenes and you encounter shading artifacts like "broken shadows" you may want to put the number down even further. Equally on really big scenes you may find yourself going on the opposite direction.

Color Limit is critical when using PER as it is used to clamp values and eliminate super bright spikes that will force you to up the shading even more.


## Rendering - Overview

In the Statistics you will be able to have some meaninful output for analysis.

Alfred style progress is always useful so make sure you turn it on.

Furthermore you can have a python script being called after every tile which gives you a lot of control although of course this will slow down a bit your renderer.


## Rendering - Overview

So what are the good values you use Jordi? I tend to set my mantra ROPs for DRAFT mode or PRODUCTION mode and have two sets of values that are my starting point. My Production mode is heavily based on a post by Jason Iversen so credit to him for sharing his knowledge.

| PARAMETER | DRA ${ }^{\text {PT M M }}$ | PRODUCTION MODE |
| :---: | :---: | :---: |
| Pixel Samples | $4 \times 4$ | $7 \times 7-10 \times 10$ |
| Min/Max Ray Samples | 1/9 | 1/9-2/9 |
| Noise Level | 0.1 | 0.08 |
| Refleck/Refrack/Diffuse Limils | $4 / 4 / 1$ | $4 / 4 / 1$ |
| Color Space Camma | 2.2 | 2.2 |
| Color Limil | 4 | 4 |
| Sample Lock | ON | OFF |

## Rendering - Overview

So after this quick overview I will simply point you again to the documentation you should have a look.

## RENDERING BASICS

hetp://www,sidefx.com/docs/houdini13.0/render/render
UNDERSTANDING MANTRA
http://www.sidefx.com/docs/houdini13.0/render/understanding
RENDERING AS PART OF A WORKFLOW
hetp://www.sidefx.com/docs/houdini13.0/render/batch
PYTHON IN MANTRA RENDERS
hetp://www,sidefx.com/docs/houdini13.0/render/python
SOHO
hetp://www.sidefx.com/docs/houdini13.0/render/soho
LIGHTING AND RENDERING WEBINAR
http://www.sidefx.com/index.php?option=com_content\$task=view\$id=2022\$Itemid=354

## Rendering - Sampling

## Rendering - Sampling

When talking about sampling we should be careful as there is the tendency to confuse the terms;

The process in which the render engine filters the pixels to procede natural looking and artifact free images is based on the idea of using a 'pixel filter algorithm' that combines 'sub-pixel samples' to eliminate jaggies, moire and other artifacts.

The final result is then "quantized" so we convert the purest data coming from the render engine onto images we can display (8bit, 10bit, etc...)

During this process there are two types of elements we want to filter, the actual object edges and the textures, for this reason I have prepared a little scene to analyse the various elements we will find.

## Rendering - Sampling

Here is the interface for the 'Mantra ROP' output tab. Notice the naming.

The pixel filter is the element you are looking for to set, my suggestion is to stick to gaussian $2 \times 2$ or gaussian $3 \times 3$ in case of heavy motion blur.

Also look at Catmull-Rom $2 \times 2$ which is a very good filter too.


## Rendering - Sampling

This is the scene and we will be analysing the next areas.


## Rendering - Sampling



## Rendering - Sampling



Texture looks ok although a bit noisy.

Causs $3 \times 3$


Texture gets very mushy in the far distance.

Calmul-Rom $3 \times 3$


Lots of artifacts on the far distance.

Barlett $2 \times 2$


Not many artifacts on the distance.

Hanning $2 \times 2$


Not many artifacts on the distance.

Sinc $3 \times 3$


Very sharp mids but far distance has lots of artifacts.

## Rendering - Sampling

Causs $2 \times 2$


Texture looks ok, not too blurry, not too sharp.

Convenient across most situations, also with motion blur.

Causs $3 \times 3$


Texture looks a bit too blurry, we loose fine detail.

Convenient for big motion blur shots.

Catmul-Rom $3 \times 3$


Texture looks very sharp.

Convenient for most situations without motion blur.

Barlett $2 \times 2$


Texture looks ok, a tiny bit blurry.

Convenient for high detail, texture heavy shots.

Hanning $2 \times 2$


Texture looks sharp.

Convenient for high detail, texture heavy shots.
$\sin c 3 \times 3$


Texture looks really sharp and nice.

Convenient across most situations without motion blur.

## Rendering - Sampling

Gauss $2 \times 2$


Shadows and textures on the edges look good.

Gauss $3 \times 3$


Textures are too blurry, shadows are nice.

Catmul-Rom $3 \times 3$


Textures are very sharp, shadow a tad noisy which means we will have to increase the amount of subpixel samples.

Barlett $2 \times 2$


Texture are not too bad, shadows not too bad either.
x


Textures are not too bad, shadow is better.

## Rendering - Sampling

Again, a few links to carry on the training.

MANTRA ROP<br>help://www.sidefx.com/docs/houdini13.0/nodes/oul/ifd<br>RENDERING QUALITY<br>hkep://www,sidefx.com/docs/houdini13.0/render/qualily<br>IN DEPTH FILTERING IN RENDERMAN<br>hklp://www,renderman.org/RMR/sL/PRMan Filkering/Filkering In PRMan.himl<br>RENDERMAN IMAGING PIPELINE<br>hetp://penguin.ewu.edu/RenderMan/RMS $2.0 /$ section 4 .heml<br>(see figure 4.2)

## Rendering - Mokion Blur

## Rendering - Motion Blur

True Motion blur is something XSI artists using mental ray have been afraid to use due to the massive increase on render times but I will ask you to reset your brain in this area.

Houdini motion blur is really really fast adding very little to the rendering time and if use in combination with the parameters exposed in the mantra renders to optimize sampling you may find in can even go faster in some cases.

I will finish this guide with a little set of settings you will want to keep always at hand.

## Rendering - Motion Blur

I have prepared two different scenes to play with and analyse some of the idiosyncrasies of Houdini's Mantra motion blur approach.

First of all regarding the motion blur positioning which is not centered by default.

This sphere is moving fast from frame 1 to frame 10 and without motion blur you could not tell there is any motion of course, so let's activate it going to the Mantrassampling tab.


## Rendering - Motion Blur

So if we turn it on you will notice the sphere has "moved" forward, this is because Houdini does by default has a camera shutter time of 0.5 which is equivalent to 180 degrees and by default there is no configuration for centering around the key frame.

Coming from animation this is fundamental for me, the pose I built is the most important moment in the animation as it has been the result of an intellectual process, this is no random position and as such I want to have the shutter time offset so they match in the centre.


## Rendering - Motion Blur

So what I do is pick the camera and find the 'shutter Time' parameter which directs the amount of motion blur.

Then offset by the same amount the 'Shutter Offset' on the 'Mantra ROP' and voila, the sphere is in the same place now.

## Rendering - Motion Blur

It is quite useful to actually build an expression so if I pass the scene to someone else they dont need to worry about touching the ROP, just the camera.

Also keep in mind that hovering over these parameters will give you a good help sometimes.

Also it is important to remember the 'Motion Factor' parameter will decrease the amount of pixel samples done to speed up the render.

It is really useful on very fast moving scenes to decrease the pixel samples as you wont see the textures well.

In this example you could compare the texture sample with Motion Factor 0 or 1 and wont notice a thing but the render time decreasing noticeably.


## Rendering - Motion Blur

If we now analyse rotation mokion blur with this nice propeller courtesy of "crabcad" website.
hetps://grabcad.com/library/3-bladeprop

And animate it over the course of 260 frame from 0 to $-1,000$ degrees (really fast rotation) we can see some of the effects of these controls.


## Rendering - Motion Blur

So Let's go to the 'Mantra ROP'


## Rendering - Motion Blur

Now let's activate the motion blur option and see how the position has changed?


## Rendering - Motion Blur

Let's offset the camera shutter speed to center the blades.

This is the equation -(ch".././obj/cam1/shutter")*0.6


## Rendering - Motion Blur

Now let's increase the
'Transformation time samples' so we have more time slices to define the curved motion.

As you can see it is 3 is not enough.


## Rendering - Motion Blur

And even more... 4 seems good but still we have straight lines so time to push it higher in our production 'Mantra ROP'.



## Rendering - Motion Blur

With 6 it is now working as I need.

Also keep in mind you wank to toggle off ALWAYS the 'Sample Lock' flag as you want to recompule the image samples, otherwise will look like paper noise over the render.


## Rendering - Motion Blur

But of course you dont want to render a scene with such 'Xform Time samples' if only one object requires.

What you want is fine control to be able to say "this object requires $X$ samples"

For this you add an attribute to the object by editing editing the Parameter interface, look for the Rendering tab and then find the $\times$ form paramter in the samplig tab.

Add it to your object as you see in the picture.


## Rendering - Motion Blur

And I tweak the parameter to 6 which will override the parameter on the 'Mantra ROP'.

The scene has been setup to 2 yet the propeller is transformed 6 times to have smooth motion blur.


## Rendering - Motion Blur

So I guess I have arrived to the end of the rendering chapter, here is again a list of things worth looking at.

## RENDERING BASICS

hetp://www,sidefx.com/docs/houdini13.0/render/render
UNDERSTANDING MANTRA
hEtp://www.sidefx.com/docs/houdini13.0/render/understanding

MANTRA ROP
hetp://www, sidefx.com/docs/houdini13.0/nodes/ouk/ifd

RENDERINE QUALITY
hlep://www.sidefx.com/docs/houdini13.0/render/quality
RENDERING AS PART OF A WORKFLOW
hklp://www,sidefx.com/docs/houdini13.0/render/batch

PYTHON IN MANTRA RENDERS
htep://www.sidefx.com/docs/houdini13.0/render/python

## Compositing

## Compositing - Introduction

## Compositing - Introduction

Like softimage provides a compositing tool that not long ago was very up to day but the rapid evolution of Nuke has defined its use more for small precomps and image processing tasks inside Houdini than full on compositing work.

The same than softimage the lack of a tracker and few other things has made this really good compositing tool a bit of a little brother but please pay attention to it because will help you enormously in your 3D work if you know how to exploit it.

Let's go through the basics and then find out how to use it for other things.

I will be using an image from STOCKVAULLT.COM free catalog so I should also thank them for the material.

## Compositing - Basics

## Compositing - Basics

The first thing you will encounter is that the 'CoP' (compositing operators) context is literally a brother of every other context.

Once you get on it each composite is seen as a chain of operations too so fits perfectly with Houdini approach and phylosophy.

In this case I have 3 ROP objects containing each a compositing operation, a bit like opening 3 nuke scripts in one single scene.

Furtheremore, they can of course talk to each other and contain other context networks inside so it can be extremely effective if used seriously.


## Composiking - Basics

Inside the 'comp1' COP network you can see I have put down a file node to read in this still.

The options are obvious and some nice details like having the name of the node depend on the name of the file are available.


## Composiking - Basics

Using the information icon you can extract information of what is inside the file, the image planes, color information, resolution, frame range, etc...


## Compositing - Basics

Also very handy is the fact that the histogram is very accessible without having to drop nodes so you can analyse what is going on with the image.


## Compositing - Basics

And a nice timeline in which you can get a quick understanding of the various clips in your comp on how they are synchronised.

## Compositing - Basics

The process is very simple, in this case a 'fonk' node is responsible of adding some text.

As you can imagine you can use expressions and call python functions from these fields as we have seen across this guides.


## Compositing - Basics

And by adding a 'composite Cop' you can layer on on kop of each other.

Similar to your Nuke merge node.
Then I used the handle to reposition the text so it is more readable.


## Compositing - Basics

Finally to write out you put a 'ROP file output' and using the various options for EXR I proceed to render the comp.

We have done our first comp so let's see a bit the toolset around so you can explore it later.


## Compositing - Basics

If I drop a 'color correct COP' you will see the normal values are presented but you can ask to see the curves for the operation from which you can actually affect the effect.

This is a more visual way of dealing with the manipulation of course.

## Compositing - Basics

The same with the levels node that exposes quite a few controls.

Pretty much the visual handles will be your tool of choice so explore it.


## Compositing - Basics

And all the operations are easily accessible the usual way, pressing tab.

All your new tools will also be installed here as we have seen before and of course you have the possibility of writting your own effects in VEX which is a real powerful too.

If you look at side FX orbolt website you will see quite a few new operators other people are building and you can download them to have an even more complete set of tools.


## Composiking - Basics

So here are a few things to look at if you are interested in the compositing area... and you should.

COMPOSITING BASICS<br>hetp://www, sidefx.com/docs/houdini13.0/composite/<br>IMAGE VIEWER INTERFACE<br>hktp://www.sidefx.com/docs/houdini13.0/ref/views/imgview \#ref.imgview compare<br>COMMON EFFECTS<br>hkp://www.sidefx.com/docs/houdini13.0/composite/common effects<br>CUSTOM COPS WITH VOPS<br>help://www.sidefx.com/docs/houdini13.0/composile/comp_vops<br>LUTS<br>help://www.sidefx.com/docs/houdini13.0/composike/luks<br>DEEP RASTER EFFECTS - DEEP COMPOSITING<br>help://wwwsidefx.com/docs/houdini13.0/composite/deepraster

## Compositing - Image Processing I

## Compositing - Image Processing I

By far the most interesting aspect of having a compositor inside your 3D package is the ability to pass data from the 3D scene to the compositing engine and vice-versa.

For example, applying effects on the footage before it is used in the viewport is very useful but it is also to process material that may or may not be used again in the scene.

Furthermore there are datasets that are much better manipulated in the form of a bitmap, like Digital Elevation Maps (DEM) and thus having the possibility of building geometry on the fly without writting code is priceless.

So let me start using the compositor to process image sequences in the scene.

## Compositing - Image Processing I

Here I have a sequence I recorded in a pool this morning and wanted to use to visualise how to manipulate the data the same way you would do in XSI.


## Compositing - Image Processing I

We are used to turn on the 'Rotoscope Options', load a sequence and then be able to manipulate the in-out points, offset, scale and use a set of basic color transformations on the clip.

Normally we use a lot the in-out points and offset so let's focus on how ko do that in houdini as if it was XSI.


## Compositing - Image Processing I

We start by creating a camera, something Houdini does not do for you automatically.

The resolution settings are something really weird so you will find yourself changing them always but I would like something contextual, meaning whatever footage I read I want that resolution on my camera parameters so let's do just that.


## Compositing - Image Processing I

On our "COPs context" I create a new composition and call it 'ingest' so it is obvious its duty.


## Compositing - Image Processing I

Then put a 'file' node to load a sequence of images...


## Compositing - Image Processing I

And once loaded I look at the resolution which happens to be greyed out, meaning I cant change it but I can reference it so...


## Compositing - Image Processing I

I simply right click and copy the parameter to use it later.


## Compositing - Image Processing I

And 'Copy reference parameter' in the camera resolution so it builds an expression pointing to my compositing file node.


## Compositing - Image Processing I

Resulting in these two expressions.
Now the resolution of the 'file' node in my compositing is read by my camera so if I change the file my camera will adjust.


## Compositing - Image Processing I

To read the 'file' node path you can directly point to the file field and copy the parameter to later copy the reference parameter in the camera background field.

This will allow you to see the rotoscope but this time is coming from a a compositing context.

Of course you can use the standard read like we saw on the camera guide in the first chapters.


## Compositing - Image Processing I

I could easily add the XSI
functionality for in-out points by using a trim node and adjusting it as you wish.

In this case I am using absolute frame numbers but you have other options at your disposal and this graphical representation of both clips is very handy.

On the top is the full sequence, on the bottom my trimmed version with some graphical manipulators to adjust it easily.
(Not bad for a technical too! ;-)


## Compositing - Image Processing I

To achieve the offset behaviour you may want to use the 'shift' node that again will allow you to move frames left and right.

You will want to configure your file in to hold the first and last frame instead of turning to black frames or worst, giving you an error.

Again easy to manipulate too with the handles.


## Compositing - Image Processing I

Also you can use the output of the whole network after many operations in your camera window...

In this case I am emulating the full XSI functionality... (kind of)


## Compositing - Image Processing I

And this is how you read that node from your camera.


## Compositing - Image Processing I

But as houdini tries to op cache the textures out you will want to make sure it is not a static one by adding changing this;
op/img/ingest/OUT_ROTOSCOPE
to
op/img/ingesk/OUT_ROTOSCOPE[\$F]
and voila!!! it works.


## Compositing - Image Processing I

Even with that nasty bug you can see the potential of mixing contexts, just be aware of it and make sure you read the documentation.

COMPOSITING PREFERENCES<br>hetp://www,side fx.com/docs/houdini13.0/ref/windows/mainprefs<br>COMPOSITING BASICS<br>http://www,sidefx.com/docs/houdini13.0/composite/<br>SOME EXAMPLES<br>hetp://www,sidefx.com/docs/houdini13.0/examples/cop2<br>COPVOPS<br>hetp://www.sidefx.com/docs/houdini13.0/composite/comp -vops<br>PYTHON COPS<br>hetp://www,sidefx.com/docs/houdini13.0/hom/cookbook/multi_stamp/

## Composiking - Image Processing II

## Compositing - Image Processing II

In this second part I will show you how to integrate your COPs networks onto your rendering pipeline with a little example.

Normally I do quick slap comps to check all frames are ok along with resolution, colorspace etc... Also serves the purpose of publishing it to a "dailies" style mechanism to see where are we and although pushes some extra workload on the render farm it is worth it.

But unlike using XSI or Maya or any other tool I rather not write software for this simple task. Houdini allows you to do just that and much more sophisticated processes without a line of code so let's have a little look at this.

## Compositing - Image Processing II

In this case I will do just that, a Little slap comp processing right after my renders.

I will start where I left it so bear with me the image ingest process is also there.

I have added a couple of lights, a camera and also I have added in my ROP context a mantra node rendering to my Amp folder (for the sake of illustration but you should render to your production folder)


## Compositing - Image Processing II

And it renders looking something as exciting as this ;-)


## Compositing - Image Processing II

Then after configuring it all I add a new comp network in my COP context.


## Compositing - Image Processing II

Inside the network I simply read the background from my other COP network (the ingest one).

Also read the sequence of images rendered using a file read that references the name of the mantra node so changing the name or location of the render wont break the comp recipe.

Scale it to the correct resolution so I protect myself from low-res renders and comp both things.

Then a null provides a referencing hook.


## Compositing - Image Processing II

A hook that is used in my compositing ROP node.

So now after I render the image sequence that compositing network is also rendered and a hierarchical system is now in place.


## Compositing - Image Processing II

If I render it locally you will see the progress bar and after a while the image will be saved to disk.


## Compositing - Image Processing II

Looking something like this.


## Compositing - Image Processing II

The interesting element in using Houdini's compositor as an image processor is that all the information needed for the compositing is already in the scene, from camera resolution, matrix, file names, etc... plus the procedural nature of the rendering context means that you can build very sophisticated processes.

Another good use would be to process textures before rendering (like screens on $t v$ 's) which are driven by scene parameters and later rendered as textures on those tv's.

If on top of this you imagine embedding a compositing network (or many) inside a tool and triggering the compositing from the interface of the tool you can easily put in the hands of your artists some truly serious capabilities without code.

And I insist all this "without code" approach of mine is due to the experiences in the past in which simple things like this lead to hundreds of lines of code that someone has to maintain, debug, deploy and ultimately have a cost nobody seems to account for.

My only suggestion is to make sure you sketch a few guidelines on your studio on etiquette.

## Compositing - Image Processing II

And finally a few more pieces of the compositing puzzle... enjoy

LINEAR WORKFLOW<br>help://www, sidefx.com/docs/houdini13.0/render/linear<br>COPS GENERATORS<br>help://www.sidefx.com/docs/houdini13.0/Modes/cop2/-generators<br>CLOBAL EXPRESSION VARIABLES<br>hep://www sidefx.com/docs/houdini13.0/expressions/.globals<br>DEEP CAMERA MAPS<br>http://www.sidefx.com/docs/houdini13.0/render/dem<br>DEEP RASTERS<br>hklp://www.sidefx.com/docs/houdini13.0/render/deeprasters

VEX \& VOPS

## VEX $\ddagger$ VOPs - Ineroduction

## Compositing - Introduction

VEX stands for Vector EXpressions and it is an omnipresent part of Houdini, something to be learnt as the true power of the software is easily apparent when you unleash VEX.

Vex is a general purpose high-performance language (close to $C /\left(C_{++}\right)$aimed at writing custom nodes and shaders.

VOPs are the graphical representation of those VEX language and it could be seen as the beloved sofimage ICE environment... but on steroids.

Ultimately the interesting thing is that like VEX, VOPs are omni-present and easier to use so the comparison with ICE is pretty obvious but dont be fooled, it is not a direct transtation.

For example, VOPs play in various context thus writting a new compositing node in VEX is trivial, the same in VOPs (the graphical way) and tends to be called VOPCOPs.

For the motion module, the mighty CHOPs it tends to be called VOPCHOPS and so on and so forth for the rest of the contexts.

Particularly interesting is the use of VOPSOPs (this is ICE) and also VOPSHOPs (your own graphical shader builder).

These new tools built with VEX or VOPs can be later encapsulated onto OTLs to be deployed in your studio making your pipeline extremely powerful.

## Compositing - Introduction

The various contexts in which you write VEX programs or vop "programs" are;

- Shading context allows you to write custome programs for every stage of the rendering pipeline using the same language.
- Displacement
- Fog
-- Light
-- Shadows
- Surfaces
- Modelling context (SOPs) allows you to manipulate points, color, veolcities, etc...
- Rendering context (ROPS) allows you to perform
- Compositing context (COPS) allows you to write complex compositing operations easily.
- Particles context (POPs now DOPs) allows you to manipulate particles are your will allowing you to build and extend the already huge library of operations.
- Channels (CHOPs) allows you to manipulate arbitrary channel data in any way you want.
- Image 3D (3D textures) can be generated in the standalone program 'i3dgen' and Later used in Houdini.

Furthermore VEX is used in other areas of Houdini like Fur to extend the already sophisticated toolset as needed.

VEX \& VOPs - Basic concepls

## VEX $\&$ VOPs - Surface operations

## thanks

## Thanks

Looking back at the result of what in the beginning was a guide to help the transition is is pretty incredible how it morphed on its own onto a huge set of guides... I could now have planned it but clearly the motivation has remained and with the help of countless people and friends seems I have arrived to the end.

Of course I owe all of you a massive thank you for your constant feedback, corrections and insights, but I want to take advantage of this moment and put some very special names down in recognition to the very long journey that has taken me to where I am today.

First of all my family and specially my mother and brother who are no longer with me but I remember every day for their kind spirit and support.

Second to those who live with me day in and day out and have to deal with all the not so good things I throw around... it is not easy, so thank you for the energy and patience.

Professionally I must confess the are key people in my life that have marked my trajectory and those deserve an special mention.

Jose Maria Aragonés, a true visionary, Pat Joseph and Robin Shenfield who have been a beacon of light in this industry and together had the guts to give me the biggest chances of my life, stephen venning who hired me at The Mill on what has been quite an amazing ride, Hitesh Patel, my mentor for most of my career while at The Mill, a true mega artist and the best vfx supervisor you can get.

## Thanks

During the 15 years I have been using Softimage and then XSI I made lots of friend but their support and tools literally made my work possible in its form.

Now the nice folks at Side Effects deserve a huge amount of respect for the incredible tools they produce, their honest approach and support (the best bar none) which makes the tough challenges ahead of me much easier to deal with.

Alongside me there has always been amazing artists that have made me look better than I am, mostly because I like working with people better than me and at the risk of missing those people I rather send a group hug instead of trying to name them all... you know who you are for sure.

## the end?

