MASTER THESIS

Creation of a Procedural Crack Generation Tool

Vigneshwar Viswanathan

MSc Computer Animation and Visual Effects

NCCA

Topics

Contents

Chapter 1 Introduction about the thesis

Chapter 2 outlines the background research and the works that have been done previously in the field of computer graphics and visual effects.

Chapter 3 explains the technical background and the concepts involved in this dissertation.

Chapter 4 describes how the digital asset is implemented and also the methods that have been followed in Houdini to achieve the same and also the options that are included in the graphical user interface of this HDA.

Chapter 5 shows the results of this otl by using one of the simulations in the production pipeline. A background plate is used and the simulation is integrated with that using a compositing software.

Chapter 1

Introduction

In the computer graphics literature, cracking and fracturing the objects has always been an interesting topic to be dealt with. These days many films delineate these effects using various computer graphics techniques and take the film to the level above so that the audience can feel the impact the films create. Before the arrival of these techniques**,** most films dealt with physical effects that ended up costing a lot of money and also one was not able to art direct and control the physical effects as they want the visuals to look like.This was one of the major drawback that lead to creation of the effects in the CG.For example, one cannot expect the crack to be produced in ground or a building to be destroyed apart as they want to be when using physical effects. So with the use of computer graphics, the user can get the variety of controls like the length and speed of the crack, the area of impact or how the building can be blown apart into pieces, which part to be shattered first, how many chunks to be shattered and all other minute details of the shot.In recent days,most of the dynamics involves the traditional keyframe animation so that the user can have great amount of control over the effect they want to create.

The main purpose of this project is to create a procedural tool that generates cracks according to the various inputs given by the user in the form of a Houdini digital asset.This OTL has been made mostly out in surface and particle context respectively and also vertex expression language commonly called as VEX programming has also been used.The final effect created can be used as a post production effect on the live action movies.

This tool (Houdini Digital Asset) lets the user to produce realistic cracks.This project also attempts to cover the different types of cracks formed mainly in surfaces like ground and glass.A shader has been written in the latter case to produce the glass crack. The user can either model or run the simulation using this digital asset. This digital asset also provides various techniques to produce cracks on the surface. The user could plug in a single curve or multiple curves using which the cracks can be modelled or animated in the 3d object. Using this HDA, the user can even break the geometry into shards to create a shattering effect in the dynamic context.

Chapter 2

Causes of the formation of cracks

One of the primary reasons why cracks are generated is because of the tensions applied to an object.Different types of materials produce different types of cracks.For example the cracks in rocks occur due to shear stress under the surface.Reasons for crack vary from metallic surfaces to non metallic surfaces.Crack also occur due to the shrinkage of the objects surface area.The cracks in the mud is due to the drying of the mud faster on the surface and thereby accumulating stress which later on subsequently develops as cracks to relieve the stress that was previously developed.In the same manner,the glass crack are also developed due to evolving of stress.

History

Research and Previous Work

The simulation of cracks and fractures are one of the subjects that is used much frequently in feature films recently.The initial research on this topic was done way back in 1988 by Terzopoulos and Fleischer and by Norton in 1991.They presented a general technique for modelling the viscoelastic and plastic deformations which used metric tensors to define energy functions that calculated the deformation over curves. Norton animated the 3D solid object which when subjected to large strain, resulted in breakage. However there were some drawbacks using these methods which included artefacts that occurred during this process. The other works done in this subject are modelling static crack patterns and fractures induced by explosions. The static crack pattern created by dry mud used a Spring Mass system attached to an immobile substrate was proposed by Hirota and his colleagues.Mazarak et al. used a voxel based approach [1]. Recursive pattern generator has been proposed by Neff and Fiume that divided a planar region into polygonal shards that flew apart when it is subject to a spherical wave. [1].Previously finite elements have been used to produce various type of fractures. The fracture mechanics has also been used in engineering apart from the computer graphics.

Hayley N. Iben and James F. O'Brien proposed a method of generating crack surface patterns.They followed a physical based approach. According to their method, the algorithm generated cracks from a stress field by using heuristics over a triangulated mesh. This method is an iterative process where cracks can be created by evolving the stress field over time.[2]

Fig 1.01(Hayley N.Iben and James F.O`Brien)

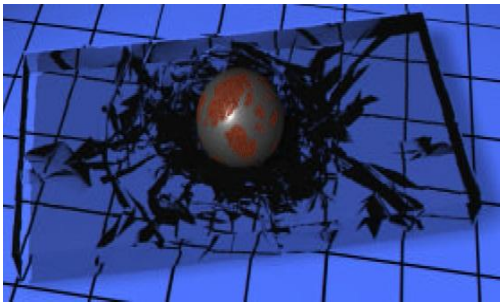


Fig 1.02 (James F.O`Brien and Jessica K.Hodgins,1999)

Most users also refer to the paper Graphical modelling and animation of Brittle Fracture which determines the stress tensors computed over a finite element model. Here the author causes the deformation by using a differential technique to define the set of differential equations that describe the aggregate behaviour of the material.

Rigid body simulations, demolitions, destructions, cracks and fractures are some of the stuffs that one could get from rigid body dynamics. These days most of the production companies have their own simulation pipelines and their own proprietary tools that are built with the use of the custom built physics engines that enable the technical directors and the effects animators to have great control over the simulation. One of the commonly used physics example is Bullet graphics engine. Using this many companies incorporated this graphics engine into their production pipeline. One example is that Sony Pictures Imageworks used their physics engine for the core work they did at the 2012 feature film which demanded a faster and a workable pipeline.

Production Examples

Some of the examples of the simulation of the cracks are done in feature films most recently.

Fig 2.01,2.02,2.03

The above displayed are some of the pictures taken from the film 2012.These shots have been difficult and there is a science of creating this says Volker Engel, one of the Co-Producer and the Visual effects supervisor of the feature film 2012.The whole idea was an inspiration from this film directed by Roland Emmerich.This was one of the films that displayed the maximum level of realism that a film could achieve by the use of computer graphics techniques.

Chapter 3

Technical Background

This tool has been created using SideFX Houdini 11.This 3d software allows the user to have much control over the simulation and also for the reason as one could easily make it as a tool in the form of a Houdini digital asset. The biggest advantage of using Houdini unlike any other 3d application is that, it allows the user to have a procedural workflow. One of the good example of its use can be seen in the feature film Invictus where a crowd system was needed to fill the football stadium. ”Houdini was the hub of the Crowd system while Houdini’s Mantra served as our primary renderer for the crowd/stadium work at CIS Vancouver” says Peter Bowmar,Head of 3d and Technology at CIS Vancouver. The other main example of Houdini is the digital asset workflow. Asylum used Houdini for the feature film Terminator Salvation that allowed several different artist to work on the same rig simultaneously and finally making it easier to bring the parts to the final output. Jeff Willette, Technical Director of Asylum says “Houdini digital assets let you experiment the different techniques and flush them through the entire pipeline easily and know you can undo or modify them later”.

In this tool Houdini’s surface operations or otherwise called as SOPS have been used. Most of the operations and core functionality of the digital asset is made in the Surface contexts of Houdini. The same tool can be implemented in Maya using Mel or C++ in the form of a plugin.Hscript (Houdini general purpose scripting) is used to create this digital asset procedurally.

Particles and dynamics

The particle systems has been first created in the year 1983 by William T.Reeves for the feature film StarTrek II: The Wrath of Khan which showed the Genesis effect

Particle and dynamic operations which are otherwise called as POPs and DOPs in Houdini have also been used. These days particles and dynamic operations have been used widely to produce stunning visual effects in the computer graphics industries. These tools have been created with the use of C++ with enormous amount of maths and physics involved in it. Using these the user can create variety of effects like the simulation of dust,debris,smoke,snow,fire,a flock of bird or fishes,etc.

In this thesis, particles play a major role. Particles have been used to produce dust when the crack generation happens and also the use of dynamics helps one to create random chunks when crack is getting generated. Basically this digital asset is made in the view of creating a crack generation in ground and this dissertation is not concentrated in making the cracks work for every type of object object. Particles are produced inside the POP(Particle operator) network in Houdini which is inside

the SOP network. A simple dynamics network have also been set so that the user could get the chunks blow when the crack propagates. The main reason why particle context is used widely rather than the dynamics context is speed. One could get a efficient solution and also the cost of calculations is minimal in particles.

Vex

Vector expression in Houdini are shortly called as VEX.VEX allows the user to modify the data.VEX was initially a shading language whose paradigm has been extended to work on other types of data than the pixels to be shaded. It can work on geometric point attributes, particle attributes, frame pixels, etc.It works on per point basis depending upon the context one works on. It defines relations and equations. The given below is one such example where the velocity increases linearly with time

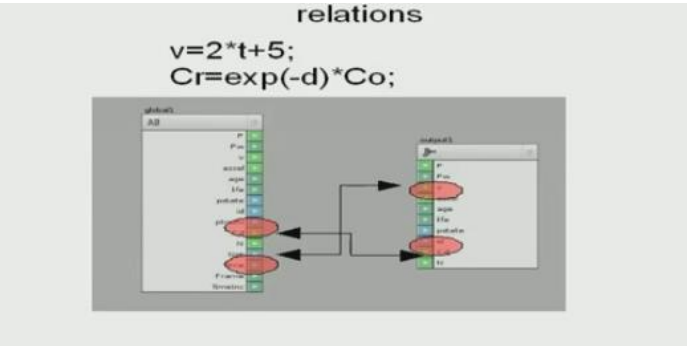


Fig 3.00

The below are some of the principles of vex

-VEX does not create data,it only modifies data

-Executed in a parallel manner

VEX also comes in visual form called as Vector Operators (VOPS) that can be used to create custom operators swiftly in Houdini.

In this thesis, the glass crack shader has been written in Houdini VEX.The user has given various parameters to get the type of glass he can get procedurally.

Development Environment

SideFX Houdini 11 for Windows was used for this project. The nodes were named using the function so that one could easily get to know what is happening inside the network if he follows it. Also a help file is embedded with the digital asset that provides a reference note of what each parameter does in

the digital asset. The nodes were also coloured to enhance the visibility and also sticky notes have been used inside the network to make it easy for the third person who investigates the network to understand the setup

Chapter 4

Implementation

The primary aim of this master project is to create one or more small tools that allows the user to generate the cracks so that it could be animated as per the animator's requirement and to finally wrap this setup as a digital asset so that it could be easily used by the effects animator to produce a simulation. The other aim of this master project is to integrate it with backplate.This tool provides the user to draw the curves over a geometry they want to be cracked. Accordingly the geometry was cracked and animated. Also the user could plug in multiple curves using a merge node and when plugged into the second input of this digital asset and accordingly cracks were generated. By using this tool the geometries were cut according to the curves the user draws to the object and it was cracked. Also the geometry can also be shattered when it is subjected to any dynamic forces.

Basic Idea

Even before getting ready to making a digital asset the first idea is to make the curves as an input which will enable the user to draw the curves and crack the geometry accordingly. By doing this the user gets a control of how long the crack grows and also which part of the geometry the crack was being generated.

Two forms of digital asset has been made in this dissertation and also a glass crack shader is created .The former allows to get ground cracks and shattering stuff to be done and the latter allows to produce glass crack patterns.

TOOL I – CrackGround OTL

The user draws a curve to produce the cracking effect.Particle context has been extensively used to create this effect.

This is really a straight forward tool that gives the end user to produce realistic cracks very easily. The GUI is straight forward enabling the user to achieve the effects efficiently

The given below figure represents the UI of this tool and soon will be followed by the description of what each parameter does in this OTL

Fig 4.0

Curve type-The user can either draw a single curve or a multiple curve.The user can also plug in a L-system.

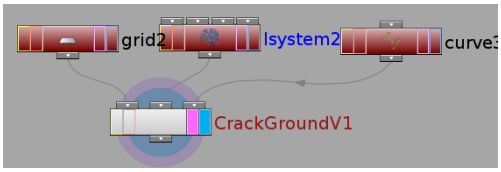


Fig4.1

NOTE: Since the thesis covers only the cracks formed on ground surface,only 2D surfaces like ground and glass are focussed.

Animate Speed-The user can animate the speed the crack moves. Value 0 indicates null value which means that the animation has not begun and value 1 indicates the completion of the animation.

Amount of Chunk-The user gets control of the number of chunks .This is done with the help of Scatter SOP which scatters points in the geometry.

Curve Thickness-The user can control the breadth of the crack.

Offset Border-Allows the user to adjust and refine the border of the ground after the noise applied.

Improve Triangulation-Allows the user to get chunks of very smaller size.

Invert Color-The tool mainly works by transferring the color to the geometry.Inverting the color does the opposite of it.

Offset-Offsets the noise parameter which is inside the VOPSOP.

Frequency-The user can adjust the frequency.

Sharpness-The user can control how rough and sharp each shard has to be.

Gravity-Controls the speed of the chunks falling.

Smoke Particles-Turns ON/OFF the smoke particles which are then sent to dops to be made as smoke.

Debris size-The user can control the size of the debris.

Debris Amount-The user can give birth the amount he wants to see.

Debris Variance-This allows to vary the debris fall which would result in a realistic effect.

Show Debris- Turns ON/OFF the debris.

TOOL II – Shatter and Crack OTL

The user can model the cracks and also do shattering by just drawing random curves in the top view of the object

The given below describes the GUI of the shatter and crack OTL and following the image is the description of what each parameter does in the OTL.

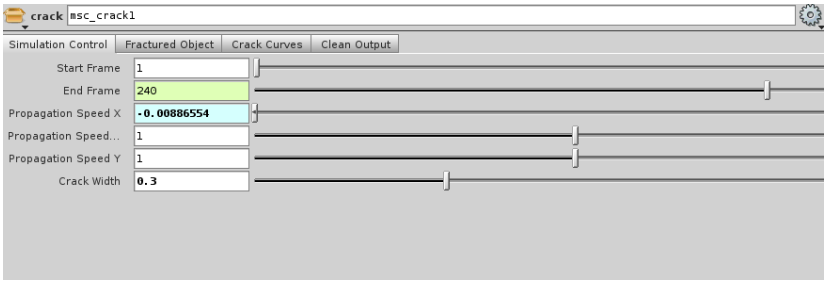


Fig 4.2

In the **Simulation Control** tab the user could control the start frame and end frame.Propogation speed x,y and z define the speed of the crack appearing in the surface.Crack width can be used to control the width of the curves.

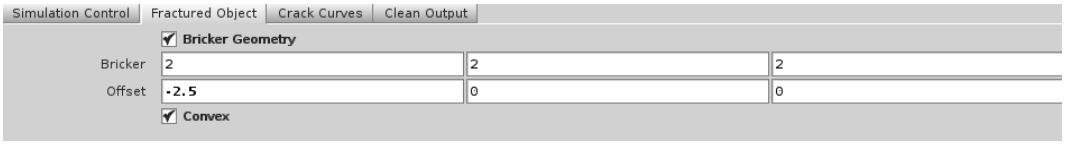


Fig 4.3

Bricker geometry parameter is used to add the extra polygons for the geometry to control cracking.This bricker node works as a polysplit sop too dividing the geometry.Using offset we can specify where a orgin to be locacted and the convex toggle is to convex the polygon

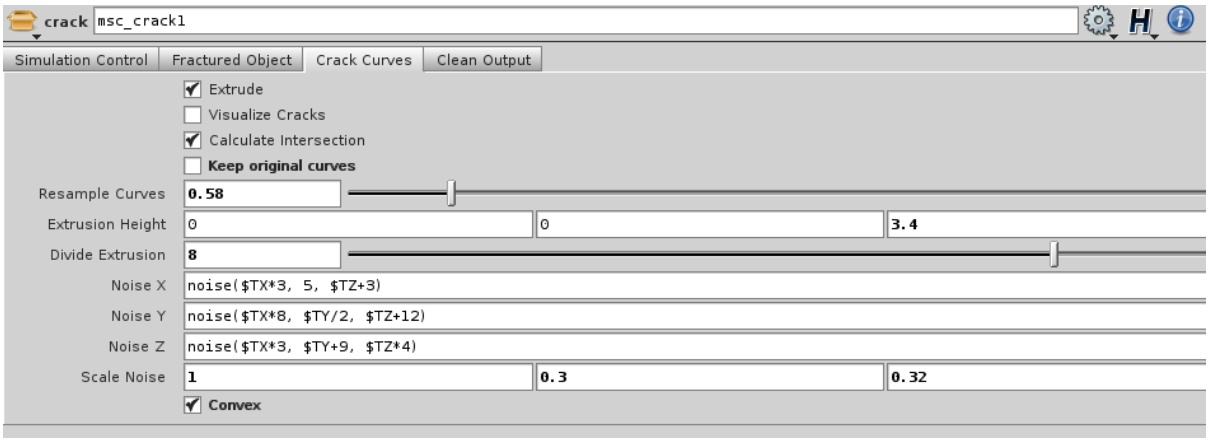


Fig 4.4

Extrude is used to extrude the geometry.

Visualize cracks lets the user to see the cracks as they are generated by the hda.

Calculate intersection is used to see if curves are inside geometry.We can speed up things by using this option.

Resample curves lets the user to resample the curves as the name suggests.

Extrusion Height allows the user to adjust the height of the extrusion.This parameter is copied to a polyextrude node inside the setup.

Divide Extrusion divides the extrusion.

Scale noise scales the noise

Convex adds triangulated mesh and to convex the cracks if needed.

Noise X **noise($TX * 3,5,$TZ+3)**

Where **noise** is a custom noise function existing in Houdini.

Noise Y **noise($TX*8,$TY/2,$TZ+12)**

Noise Z **noise($TX*3,$TY+9,$TZ*4)**

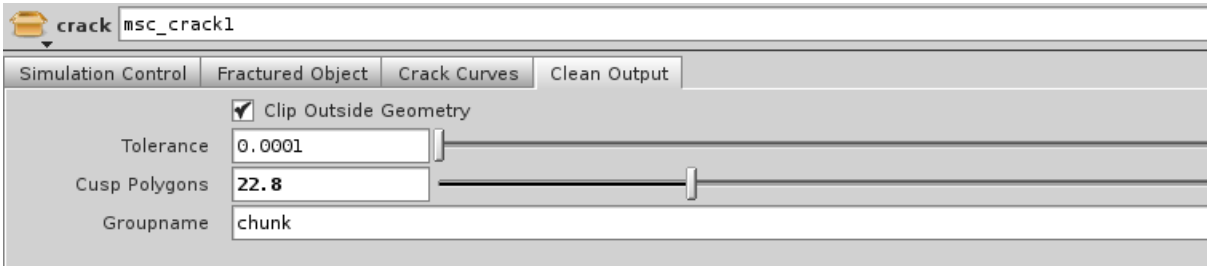


Fig 4.5

Clip Outside Geometry clips outside geometry for removing artifacts.

Cusp polygons is to increase the divisions in the polygons.Its done by copying this to a edgecusp node.

Basic Pseudo Code

- Take the input(curve) from the user.
- Extrude the curve so that the curves cuts through the geometry.
- Adding noise to the curves and to the extruded curves.
    - A foreach iteration
    - Grouping the cut pieces using connectivity sop
        - Foreach
            - Boolean operations using a cookie sop
        - End of foreach
    - End of Foreach

Starting to set up the network

Here is the sequence of images that guides the user about what is happening in the setup of this network.

Step 1: User draws the curve.

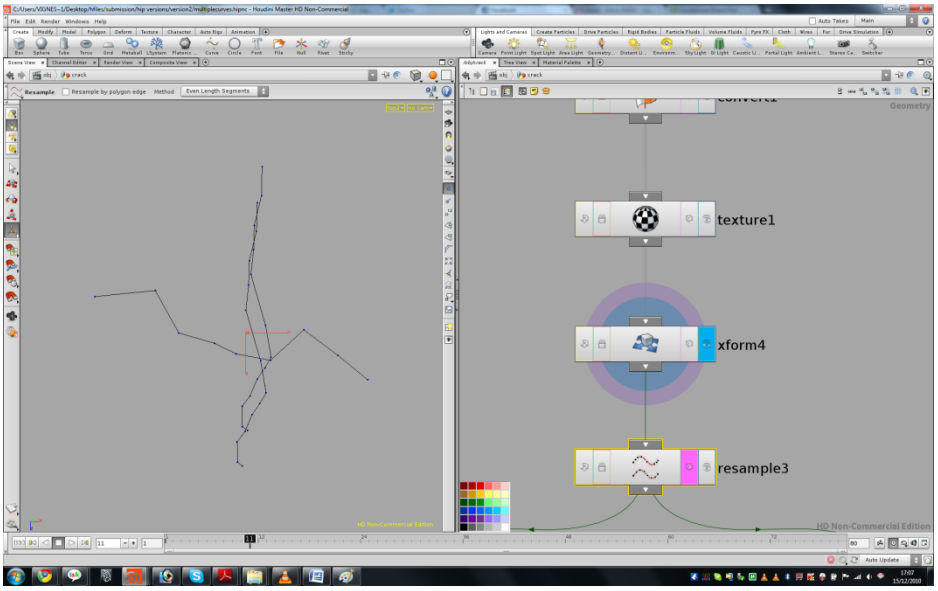


Fig4.6 getting the user input for generating cracks

In the above figure the user gets the curve and adds a **uvtexture** node.By adding this node,it allows the user to identify the ends of the curve and also this uvtexture attribute plays a very important role in animating the width of the curve which will be discussed later.

Step 2: increasing the resolution of the cracks

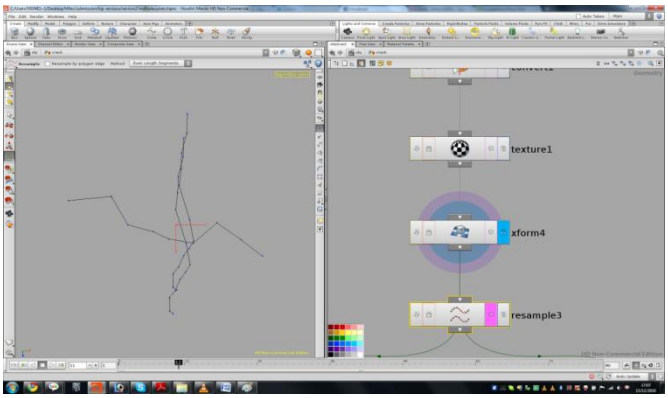


Fig 4.7 curves are resampled

After getting the input from the user, the curves are resampled so that one could get a finer curves that could be used to generate the crack in the geometry we provide.

Step 3:

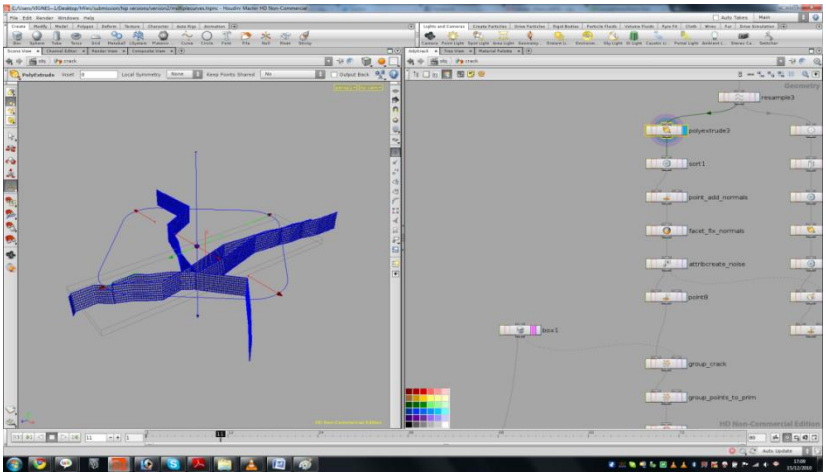


Fig 4.8 Here we extrude the curves till the curves intersect the geometry.

Step 4:

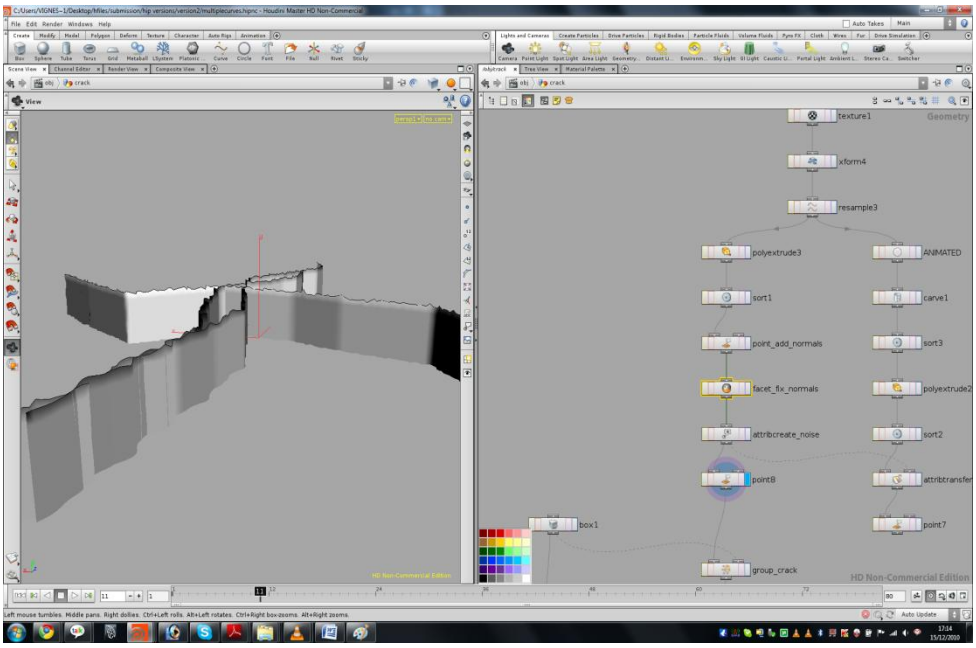


Fig 4.9 adding noise to the top of the curve

In the previous image a random noise pattern on the crack is generated using a **createattribute sop**

$NX * noise($TX*3, 5, $TZ+3) * ch("noisescalex")

$NY * noise($TX*8, $TY/2, $TZ+12) * ch("noisescaley")

$NZ * noise($TX*3, $TY+9, $TZ*4) * ch("noisescalez")

In the above expressions, the custom noise based function is multiplied by the normal information of the curves and then it is multiplied by the value the user provides so that the user could give in a random value.The noise gets scaled by the ch("../../scalenoise3") parameter and then moved in the direction of the normal in the Z direction denoted by $

These noise can also be added to the point information so that we get a surface of more sharp edges .

Step 5:

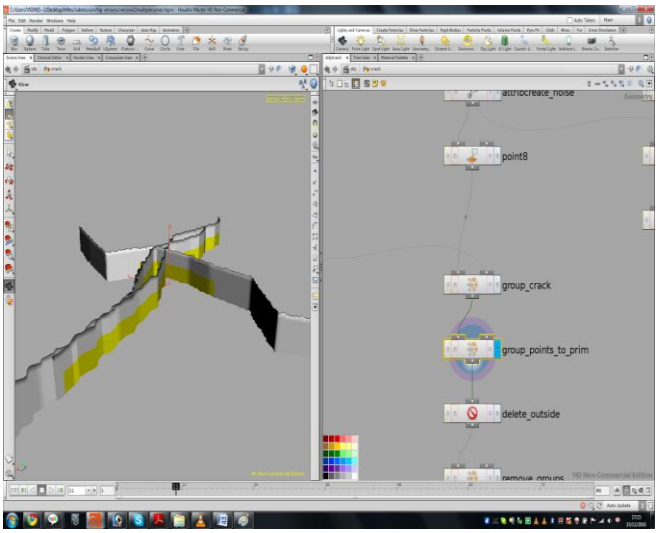


Fig 4.10selecting the intersecting portion of the curve

In this step the user groups the entire surface in points that intersect the geometry firstly and then converts these point group nodes to the primitive groups so that the primitives that are not intersecting the geometry can be removed.

Step 6:

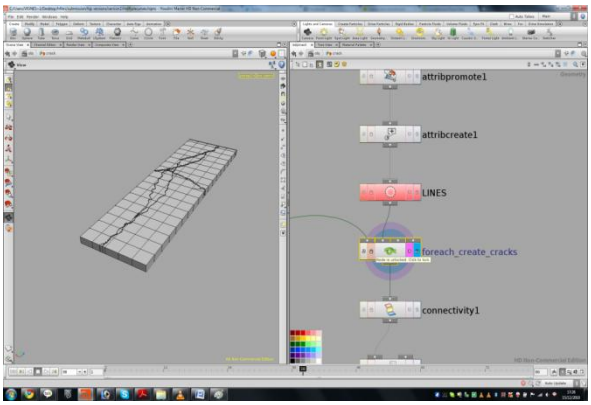


Fig 4.11 FOREACH loop

A kind of iteration has to be followed for the number of curves when the user plugs in, so that for each curve the cracking operation can be performed. Houdini offers a node called **foreach sop** which is similar to the looping statements that exist in the programming languages. This node iterates the process till a certain condition is satisfied. This is where the each box is cut into different pieces.In the outer loop we check how many curves are there present. These are the main cracks and we crack the input geometry in a subsequent order.

The delete1 node deletes every curve except the current curve.

The following expression does the above mentioned

($LINE - 1) == stamp("..", "totcuts", 0)

$LINE is the attribute that was created outside the loop and the value of this is set to $LINE+1 since some expressions the value of Houdini starts counting from 0.

The $LINE attribute gets a value that is checked against the iteration of the value in the foreach loop. If both values match, then all the curves except this get deleted.

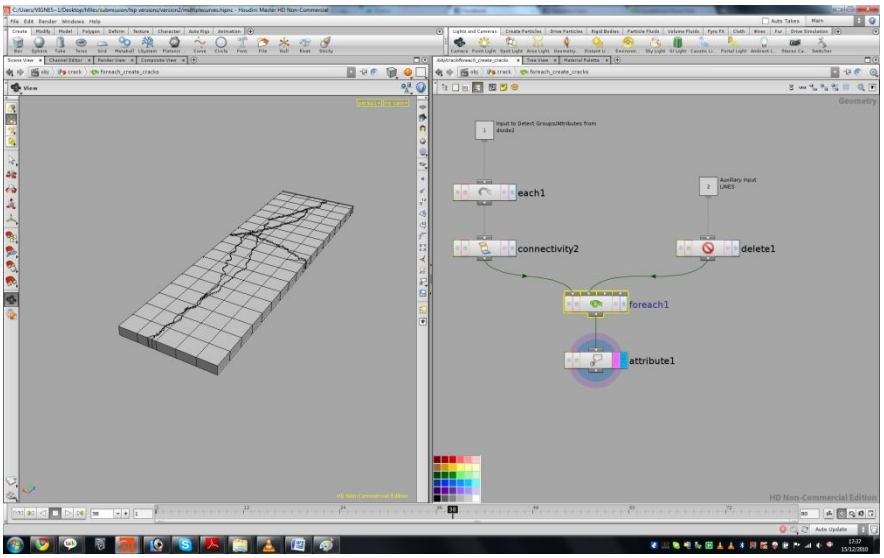


Fig 4.12 Inside the foreach loop

Here is where the Boolean operations take place inside our network.

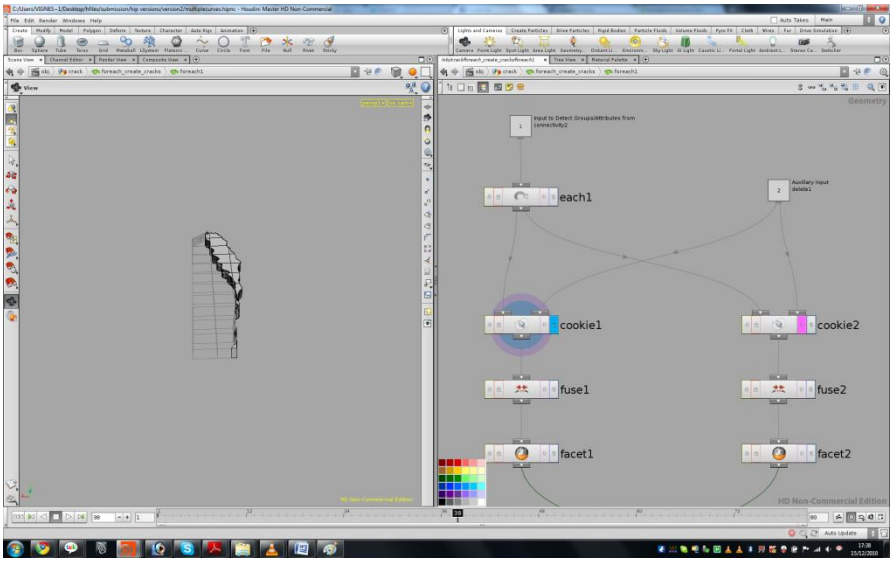


Fig 4.13 Boolean operation using a cookie sop

Here in this foreach loop we crack the input geometry. Then we fuse on both the pieces of the geometry and apply facet sop to get the correct normals.

This is the core setup that does takes place inside our tool. Boolean operations are one of the key things to be noted down since it's a way where the geometry is created using the intersection of two or more objects. Till the arrival of Houdini 11, this is the only method that existed wherein the intersection of two or more volumes had been calculated. After the launch of Houdini 11 in the market, the geometry intersections can be done using a far more efficient and suitable approach with the use of isooffset sop and also with the new sop in Houdini 11 by name volumemerge." Using this method we get less artefacts" says Jeff Wagner, the senior mathematician and developer of Houdini in the old school Houdini blog in the SideFX website.

After the primitives are cut, some sort of animation was needed to show that cracks grows. So the animation can be shown in the output. The original group of the lines/crack curves are needed so we promote this primitive attribute to a point attribute using an attributecreate.grouping is done if the value of the attribute line was greater than 1.

$TX – 0.5 * ($F/$NFRAMES) * point("../"+opinput(".",0),$PT,"uv",0) * $NX * (1 - $CONSTRAINED)

Where **$TX** is the point position

0.5 is just a scaling factor. It could be a any number or a user parameter that could be controlled in the digital asset.

$F/$NFRAMES is where the animation happens

$F- is the current frame

$NFRAMES- is the total number of frames set in the timeline.

point("../"+opinput(".",0),$PT,"uv",0) is an expression that could also be written as **$MAPU** which gets the length percentage on the curve. This expression gets the u value of the curve.

$NX-normal pointing in X axis.

(1-$CONTSRAINED) – here we normalize the normals to avoids the scaling in the noise.

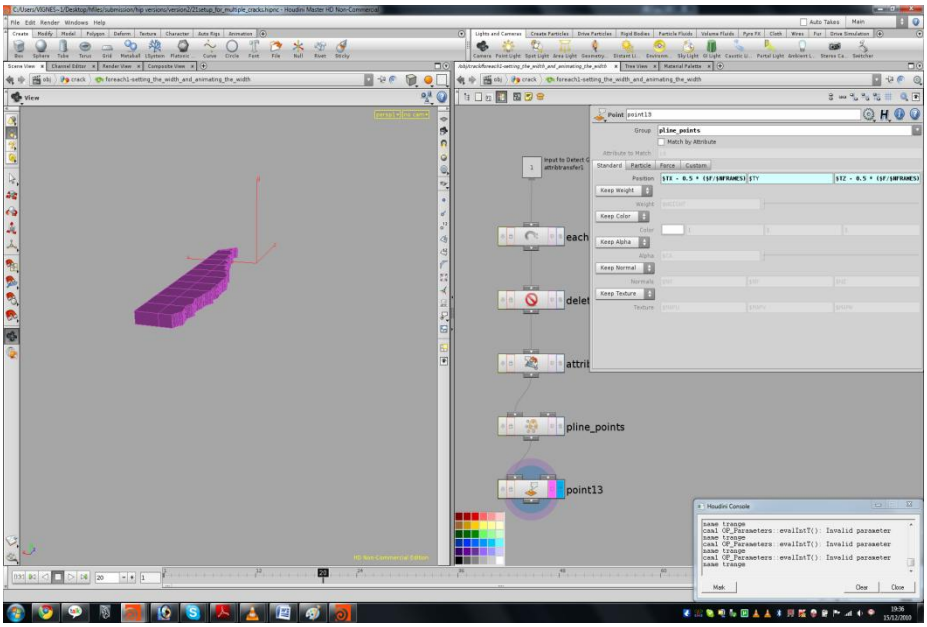


Fig 4.14 Animation of cracks

Enhancing the setup

The above discussed was the basic setup we had. There were some problems and artefacts encountered in creating multiple cracks for the previous network setup created. For instance if the user draws a weird curve or many number of curves intersecting each other, the setup failed for some reason as it created lots of artefacts and the crack animation also failed because of this. So a new method has to be made from the previous setup so that the network accepts multiple curve input and

generates multiple cracks and also that avoids problems and artefacts during geometry intersection.The setup of the network is almost the same except the new foreach and some other nodes makes to tool error free and reduce artifacts.

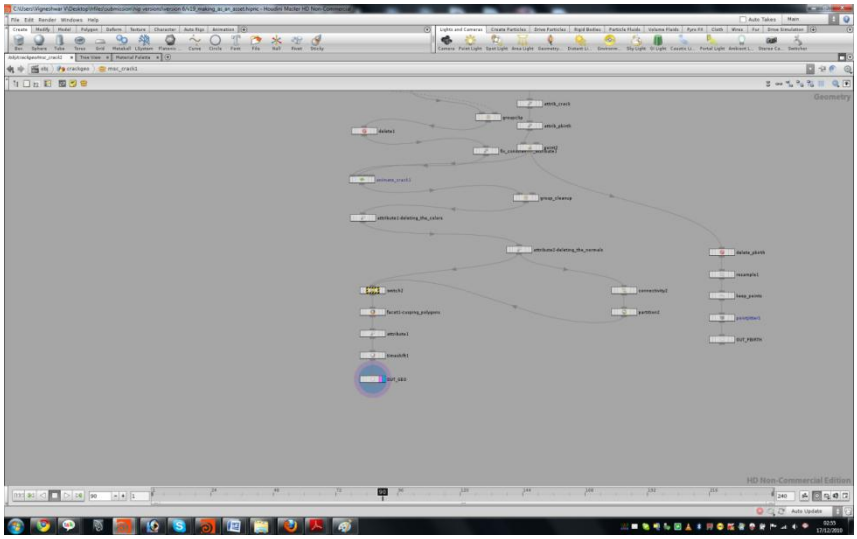


Fig 4.15 the main network(contd)

This is the main network that was setup in the process of making the digital asset.Most of the problems and the artefacts have been fixed.This is the final version of the tool and this is made in order to fix all the problems that persisted before and this setup was aimed to get a error free tool that the user could use for his purpose.This setup is similar to the previous setup except of the new addition of the foreach loop at the beginning.

Inside ReadyCurves FOREACH

The network is setup in such a way so that the operation loops for all the curves.Firstly in here attributes are added on the first and the last point and then the curve is extruded.After the extrusion of the curve,we sort the points.Then the attribute is created for making custom noise from the user.The following are the expressions to make this

eval(chsraw("../../pointnoise1"))* ch("../../scalenoise1")*$NX

eval(chsraw("../../pointnoise2"))* ch("../../scalenoise2") * $NY

eval(chsraw("../../pointnoise3"))* ch("../../scalenoise3") * $NZ

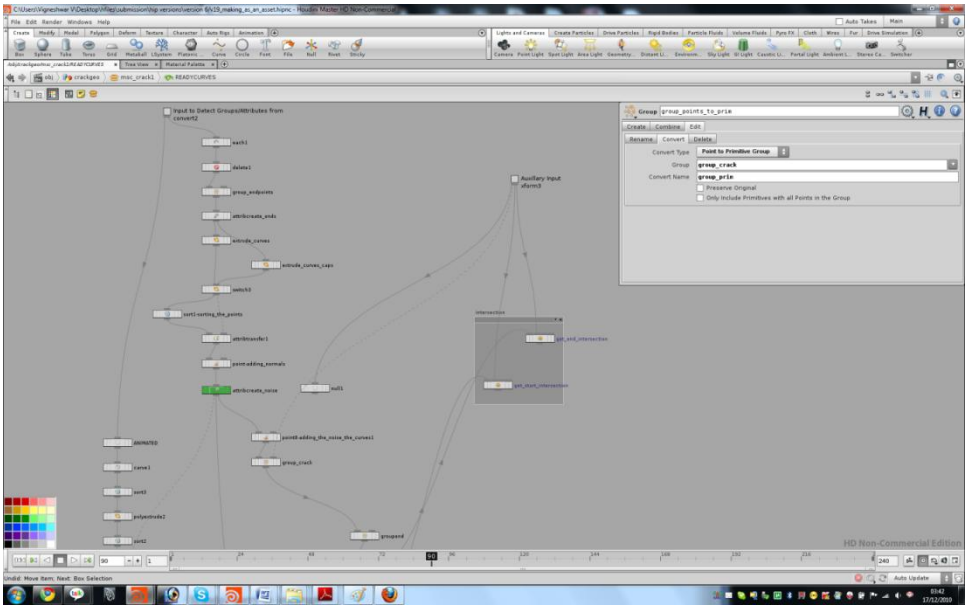


Fig 4.16 setup inside the Readycurve FOREACH

eval(chsraw("../../pointnoise3"))

Since the Houdini digital asset does not get to know about the point information in $PT attribute,we have to make use of the above expression.So that the strings can be executed and a value can be got.

ch("../../scalenoise3") * $NZ

Here we scale the noise by multiplying the scale parameter given by the user with the normals.

Then there are subsequent group nodes in the network that checks if the start point and the end point were inside the geometry.Then we extend to the surface of the geometry with the use of ray sop.We can see this operations inside the nodes get_end_intersection and get_start_intersection,

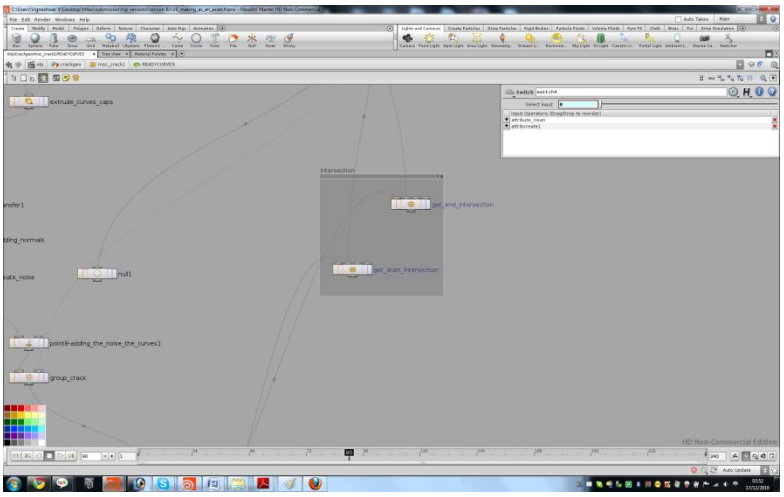


Fig 4.17 setup inside the Readycurve FOREACH(contd.)

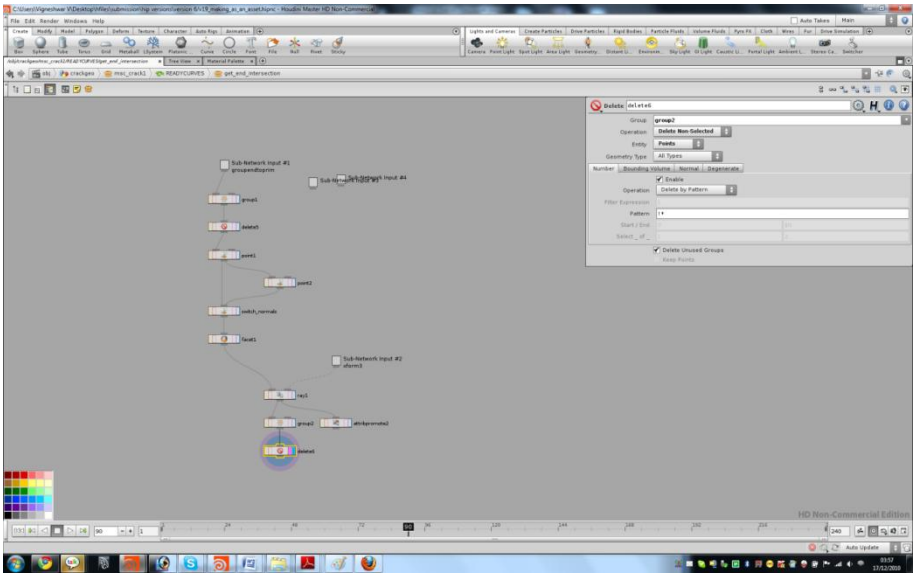


Fig 4.18 get_end_intersection node

Here an attribute is created that measures the distance from the points to the surface.In the xform_end and xform_start extension happens.

All the above functions and network setup are same as explained in the basic idea column earlier.

Shattering

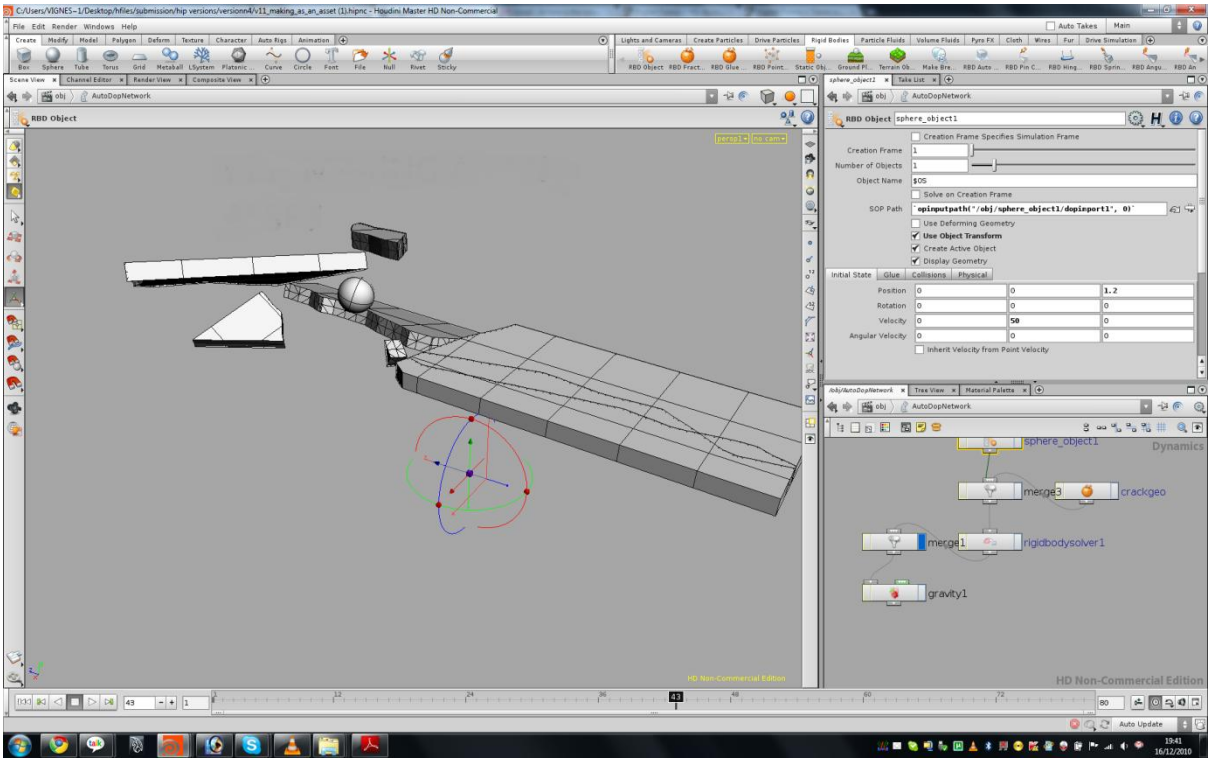


Fig 4.19 Shattering

Using this setup the shattering can also be done. The user can plug in any number of curves and based on that shards can be cut and by clicking RBD fractured object fractures can be made and when it is subjected to any force we see the shattering.

TOOL III – Glass crack shader

A shader has been written in vex context of Houdini.This shader enables the user to produce surface crack.The crack formed in the tool represents the cracking that appears in the glass.

The following is the GUI of the shader and following the image is the explanation of what one each parameter does

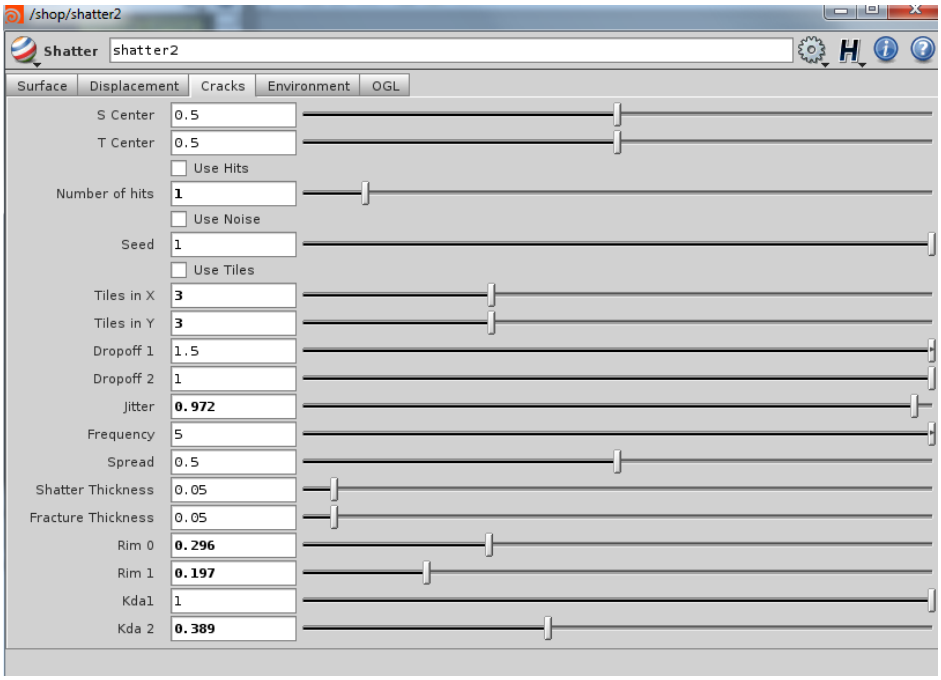


Fig 4.20 GUI of glass crack shader

S Center and T Center offsets the centre position of the crack.

Use Hit-Turn ON/OFF.

Number of hits- produces hits on the surface.

Use Noise- use noise is to offset the perpendicular cut of crack.Without this option, the crack will be straight.

Seed- controls the randomization.

Tiles in X and Y-draws the tiles row wise and column wise which acts as a centre point for the cracks

Dropoff 1 and 2- controls the crack from boundary edge and is creating long fractures at the outer edge of crack respectively.

Jitter-This parameter increases the amount of minute cracks.Atleast a minimum of 0.250 is advised for this parameter.

Frequency-This parameter controls how frequent the cracks occur.Large cracks are produced when the value is low.

Spread- Increases the space between cracks.

Shatter Thickness-Increases the thickness of the edges in crack.

Fracture Thickness- Increases the thickness of the fracture.

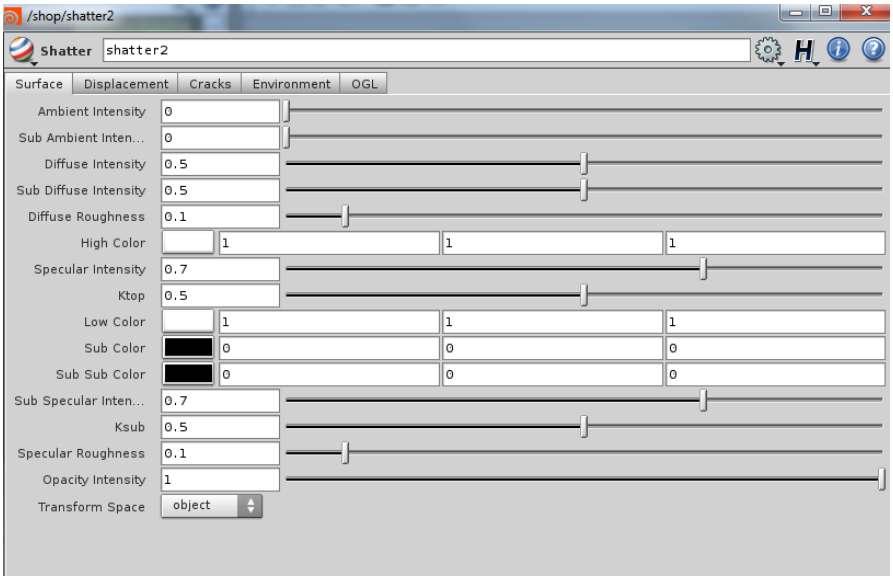


Fig 4.21 Fig 4.20 GUI of glass crack shader

Surface tab

The surface tab controls the lighting calculations.

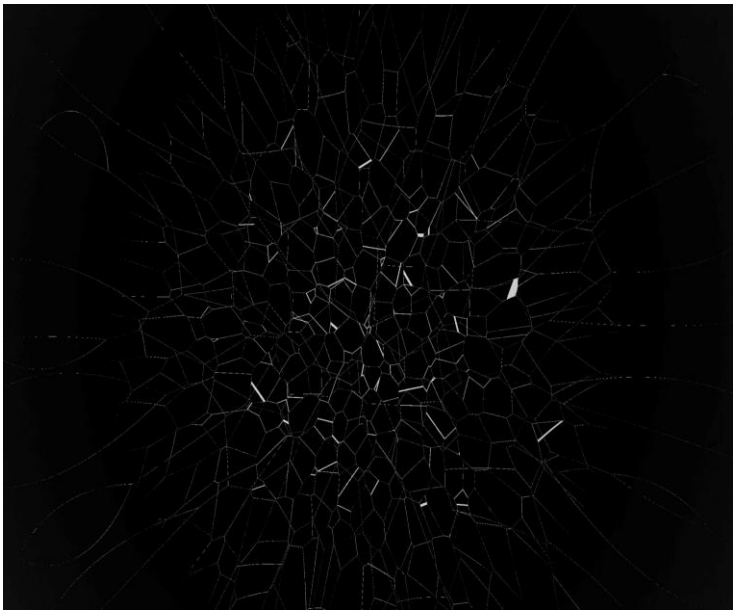
The other tabs follow the usual controls which every shader in Houdini possesses.

Results and Conclusion

Outputs of groundcrack tool

Tool I-groundcrack otl

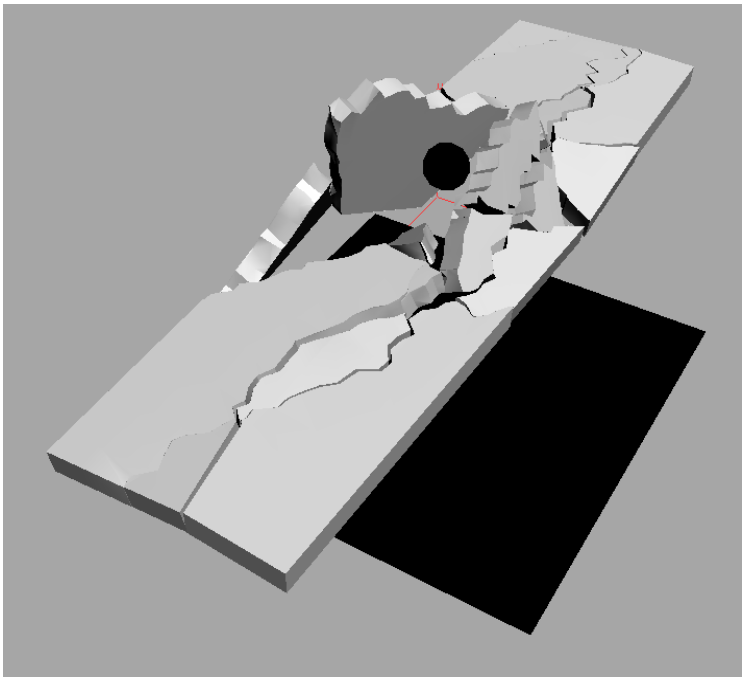
Outputs of glass crack shader tool



Tool-II glass crack shader

TOOL –III Crack and Shatter generator tool

Tool III-crack and shatter generator



Tool III-crack and shatter generator

This crack generator tool can be used in a production pipeline very easily as it is not that difficult to use since the digital asset is fairly straight forward.Also the user can shatter the object according to the wish by drawing curves with respect to the object from the top view to visualize where the users wants to create the fracture.

Future Works

A base setup has been made by following the paper.Due to time constraints the setup was not completed.The glass crack shader is procedural and it can be developed to produce subsurface scattering effects and also displacements to look further more realistic.

There were several methods to achieve the effect and implement as a tool but one could get more control according to the curve they draw,thats the main reason why this tool was created.As a beginner i tried to learn several things,few things have been tried to make it work still more efficient.Lot of new things in the SOP's context have been learnt by doing this dissertation.There are some offsets and i have worked till my best to give the error free setup.

References:

1. James F.O`Brien and Jessica K.Hodgins.(1999).Graphical Modeling and Animation of Brittle Fracure,SIGGRAPH'99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques.URL: http://portal.acm.org/citation.cfm?id=311550 [Accessed 17 December 2010]
2. Hayley N.Iben and James F.O`Brien(2006).Generating surface crack patterns.Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer Animation. URL:http://graphics.cs.berkeley.edu/papers/Iben-GSC-2006-09/ [Accessed 17 December 2010]
3. 2010.Houdini Documentation .URL: http://www.sidefx.com/

Video Tutorials

4. Getting started with digital assets,video tutorial. URL: www.digitaltutors.com [Accessed 8 December 2010]
5. H9 Bridge Asset,video tutorial.URL: www.3dbuzz.com [Accessed 4 December 2010]
6. H9 Elevator Asset,video tutorial. URL: www.3dbuzz.com [Accessed 7 December 2010]
7. Fxphd online Houdini class:HOU202-Intermediate Techniques for Houdini TDs. URL: www.fxphd.com [Accessed 17 December 2010]
8. Fxphd online Houdini class:HOU201-Intermediate Techniques for Houdini TDs. URL: www.fxphd.com [Accessed 17 December 2010]
9. Side Effects Tutorials,video tutorial.URL:www.sidefx.com URL:www.sidefx.com [Accessed 27 October 2010]
10. http://sfdm.ca.scad.edu/faculty/mkesson/vsfx419/wip/winter09/tyler_rodenburg/subsurface_effects_02.html [Last accessed on Mar25,2011]

Forums

11. Odforce Forum, URL: http://forums.odforce.net [Accessed 27 October 2010]