UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Centre de la Imatge i la Tecnologia Multimèdia

Desarrollo de una herramienta para la visualización de protección costera.

Víctor Granado García

Director: Salvador Bolarín Molina

Grado: Multimedia

Curso: 2022-23

Universidad: Centre de la Imatge i la Tecnologia Multimèdia

Índice

Resumen

Este trabajo abarca varias técnicas de desarrollo de HDA y creación de simulaciones de fluido de manera procedural y en el que se crea una herramienta de visualización de fluidos para la protección costera.

Palabras clave

Houdini, Simulación, Flip Fluids, HDA, Procedural

Enlaces

Descarga de la herramienta:
https://drive.google.com/file/d/1eqIjfkF7AydT-xaxX_-LVxtNgUOq22B_/view?usp=sharing

Video:
https://youtu.be/1Fbrz1AkskY

Índice de tablas

Índice de figuras

Glosario

HDA: Houdini Digital Asset.

CFD: Computational Fluid Dynamics.

Bounding Box: Bounding Box es una caja 3D que envuelve una geometría o un grupo de objetos. La caja se define por sus coordenadas mínimas y máximas en X, Y y Z.

VDB: Formato de datos utilizado en Houdini y otros programas de gráficos y efectos visuales para representar volúmenes tridimensionales. VDB es una representación eficiente y compacta que permite almacenar información dentro de un campo volumétrico discreto.

Vorticity: Propiedad física que describe la rotación y la circulación de un fluido en un punto específico. Se utiliza para cuantificar la tendencia de un fluido a girar en un determinado punto del espacio.

SIGGRAPH: Organización y conferencia líder en el campo de los gráficos por computadora y las técnicas interactivas. Proporciona un entorno donde los profesionales pueden compartir conocimientos, presentar investigaciones y explorar las últimas tendencias y avances en la industria. Es un punto de encuentro importante para la comunidad de gráficos por computadora a nivel internacional.

Velocity field: Representación matemática o visual de las velocidades de un fluido en cada punto de un dominio o espacio determinado. Se utiliza comúnmente en la mecánica de fluidos y en la simulación de fluidos para describir y analizar el movimiento y la dinámica de un fluido.

Fluido compresible: Un fluido compresible es aquel que experimenta cambios significativos en su densidad debido a variaciones en la presión. Los gases son un ejemplo.

Fluido incompresible: Un fluido incompresible es aquel en el que su densidad se considera constante, independientemente de las variaciones en la presión. Los líquidos son ejemplos

SDF: Representación geométrica que se utiliza en gráficos por computadora para describir la forma y la distancia de un objeto tridimensional. En pocas palabras, un SDF asigna un valor numérico a cada punto en el espacio, indicando la distancia desde ese punto hasta la superficie del objeto.

La característica distintiva de un SDF es que la distancia se representa con un signo, lo que indica si el punto está dentro o fuera del objeto. Los valores negativos indican que el punto está dentro del objeto, mientras que los valores positivos indican que el punto está fuera. El valor cero representa la superficie del objeto en sí.

1. Introducción

1.1 Motivación

Desde mi infancia, siempre me ha fascinado el mundo del cine, especialmente la habilidad de crear escenas imposibles en la vida real y hacer que parezcan reales en la pantalla. Fue así que descubrí el vasto mundo del CGI. Al principio, mi primer acercamiento fue a través de programas como After Effects, ya que la curva de aprendizaje era sencilla. Poco a poco, fui investigando más y logré imitar algunos efectos que veía en las películas. Sin embargo, al ser un programa 2D, sentí que no era suficiente. Fue entonces cuando empecé a explorar programas 3D y cómo era posible recrear efectos como agua y fuego de manera realista.

Cuando ingresé a la universidad, contacté con exalumnos que me ayudaron a entender que lo que más me apasionaba eran los efectos visuales (VFX). Finalmente, tuve mi primer contacto con Houdini, el programa líder para crear simulaciones de todo tipo. Desde entonces, he continuado utilizando Houdini no solo para efectos visuales, sino también para cualquier tarea relacionada con la creación de gráficos en 3D.

Además, dentro del mundo de las simulaciones las que me han llamado más la atención desde un principio han sido las de fluidos. Es por esto que me gustaría adquirir y aplicar más conocimiento en este ámbito.

1.2 Formulación del problema

El objetivo de este proyecto es aportar nuevo material visual sobre las diferentes estructuras de protección costera y su posible variación como resultado del cambio climático. Además, se espera que la herramienta resultante pueda ser útil para otras personas interesadas en visualizar el comportamiento de las olas en diferentes situaciones que deseen explorar.

1.3 Objetivos generales del TFG

Las herramientas de Houdini vienen en forma de nodos, scripts o HDA y se pueden usar para una variedad de tareas, como simulaciones, creación de geometría, generación de texturas y otras cosas. En este caso, se creará un HDA porque están hechos para ser muy flexibles y adaptables, lo que permite a los usuarios cambiar diferentes parámetros para lograr diferentes efectos o comportamientos.

Al permitirles simular y visualizar el comportamiento de las olas al impactar con diferentes estructuras, la herramienta a desarrollar será de gran utilidad para los expertos en protección costera. Esto brindará a los usuarios más control y precisión sobre los resultados.

La herramienta también será útil para identificar los riesgos y desafíos provocados por el cambio climático en las regiones costeras, lo que permitirá a los expertos prever y planificar nuevos proyectos de construcción y protección. Una mejor comprensión de la dinámica de la costa y el desarrollo de defensas más efectivas podrá ser posible con la ayuda de la herramienta.

1.4 Objetivos específicos del TFG

- Lograr crear una herramienta completamente procedural.

- Estudiar las simulaciones FLIP dentro de Houdini lo suficiente como para lograr crear una herramienta lo más profesional posible.

- Conseguir un comportamiento en el agua extremadamente realista y colisiones precisas.

1.5 Alcance del proyecto

Este proyecto va dirigido a los expertos en ingeniería costera que deseen estudiar el comportamiento de los fluidos en diferentes escenarios. Especialmente en estos momentos que el cambio climático está teniendo un gran impacto en el planeta, poder investigar los problemas que puedan causar las olas al impactar en la costa permitirá tomar ciertas decisiones en base a los resultados obtenidos a través de la herramienta.

No obstante, este proyecto no se limita al grupo de usuarios descritos anteriormente, sino que también puede ser útil para artistas 3D que no tengan un gran conocimiento de Houdini pero deseen implementar fluidos en alguno de sus trabajos. Por ejemplo, haciendo uso de la herramienta recrear una playa puede resultar muy fácil y accesible para todos.

2. Estado del arte

2.1 CGI

Al permitir la creación de mundos y personajes impresionantes, las imágenes generadas por computadora (CGI) han revolucionado las industrias del cine y la televisión. Las primeras películas en 3D se hicieron en la década de 1970, pero no fue hasta The Abyss en 1989 que las imágenes generadas por computadora se usaron ampliamente en las películas. Posteriormente jugó un papel crucial en la película de 1991 Terminator 2: Judgment Day. Con el fin de ahorrar dinero, acelerar el proceso de producción y mejorar la calidad, CGI se ha vuelto cada vez más frecuente en la producción de cine y televisión. Las películas y los programas de televisión se han convertido en impresionantes experiencias de visualización como resultado de la mejora significativa de la calidad de las imágenes generadas por computadora con el tiempo.

Img 2-1: *The Abyss*, 1989.

2.1.2 VFX

VFX es el CGI utilizado en películas, programas de televisión y videojuegos para producir una ilusión o mejorar el realismo de una escena. De los efectos más simples a los más complejos, incluidas explosiones, fuego, agua, criaturas y entornos, se pueden producir utilizando efectos visuales .

Estos efectos son creados por artistas de VFX utilizando una variedad de herramientas y procesos, como modelado 3D, captura de movimiento, composición y rotoscopia. Colaboran con directores, productores y directores de fotografía para garantizar que los efectos visuales coincidan con el estilo y la visión del proyecto.

Estos se han vuelto cada vez más complejos a lo largo de los años y son necesarios para el cine contemporáneo. Se pueden usar para crear escenarios y personajes creíbles que serían desafiantes o imposibles de fotografiar en la vida real, como el mundo alienígena de Pandora en la película Avatar.

En general, VFX es esencial para la industria del entretenimiento porque permite a los cineastas realizar sus visiones imaginativas y brindar al público experiencias inmersivas.

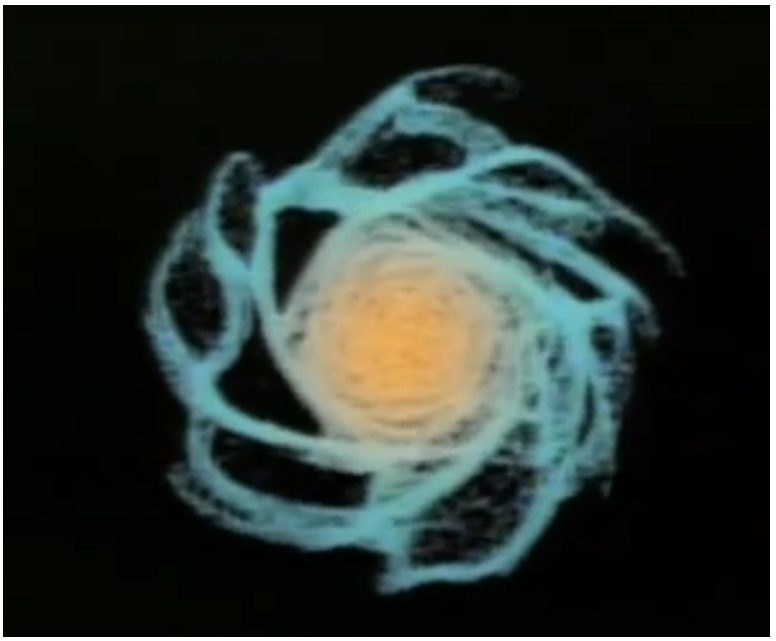
Img 2-2: *Captain Marvel*, 2019.

2.1.3 Simulaciones 3D

Las simulaciones en 3D son un proceso que permite imitar el comportamiento de objetos y fenómenos de la vida real en un entorno tridimensional. Por ejemplo, la creación de fluidos o la destrucción de objetos.

Determinar la fecha exacta del inicio de las simulaciones generadas por ordenadores resulta difícil. Sin embargo, la serie documental "Cosmos: un viaje personal" (1980) de Carl Sagan destaca por ser uno de los primeros proyectos en utilizar sistemas de partículas para representar galaxias mediante CGI.

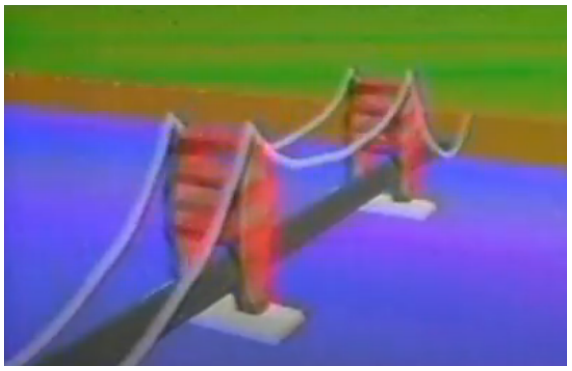
El informático James F. Blinn fue el encargado de generar estas animaciones para Sagan. Su trabajo en "Cosmos" le permitió desarrollar un sistema de visualización para la NASA.



Img 2-3: Cosmos: Un viaje personal, 1980.

Gracias al avance tecnológico, el CGI ha evolucionado considerablemente. En el año 1986, en el SIGGRAPH, Bill Viola presentó la primera simulación cloth generada por un ordenador, llamada "Flags and Waves". A lo largo de los años, estas conferencias han sido el escenario donde se han presentado los diferentes avances en la computación gráfica, incluyendo las simulaciones 3D. Entre todos los progresos que se han hecho, se pueden destacar los siguientes:

- Primeras muestras de simulaciones Soft Body avanzadas en el año 1987 de la mano de David Haumann en la Universidad de Ohio.



Img 2-4: *Dynamic simulation of flexible objects,* 1987.

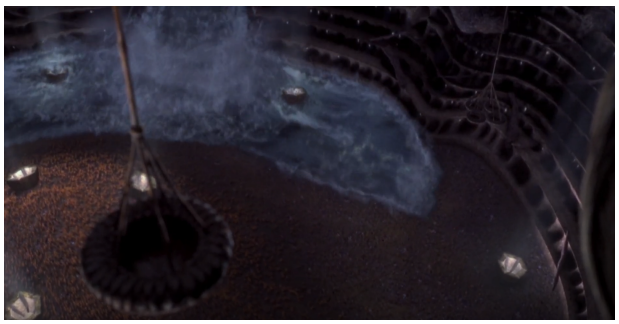
- Gran avance en los sistemas de partículas con la presentación de Particle Dreams (1988) dirigido por Karl Sims y siendo una de las primeras aproximaciones a lo que serán más adelante las simulaciones de fluidos.

Img 2-5: *Particle Dreams*, 1988.

- Simulaciones avanzadas de Humo presentadas en el SIGGRAPH de 1994 por el software Alias.

Img 2-6: *Alias Siggraph Stagereel,* 1994.

- Primera simulación de fluidos y simulación de Crowds controladas por IA integradas dentro de una película, ANTZ(1998).



Img 2-7: Antz, 1998.

Aunque se lograron avances, los procedimientos requeridos para completar las simulaciones resultaban muy costosos, especialmente en términos de tiempo. Por esta razón, en los años 90, producir simulaciones de alta calidad era un desafío. No obstante, en la próxima década el CGI experimentó un avance extraordinario, no sólo en el ámbito de las simulaciones, sino de manera general.

La trilogía de El Señor de los Anillos (2001-2003) es un ejemplo destacable de cómo se avanzó de sistemas de partículas limitados a criaturas de gran envergadura en solo unos pocos años. Sin embargo, como los propios artistas involucrados en la producción de las películas explican, no todo lo que imaginaban podía ser creado mediante simulaciones. Para la criatura Balrog, por ejemplo, habían planeado usar simulaciones de fuego para lograr efectos realistas, pero el resultado no fue satisfactorio. Por lo tanto, tuvieron que recurrir a una simulación de partículas más sencilla para completar la escena. Lo mismo ocurrió en situaciones que requerían simulaciones de fluidos, en algunos casos pudieron utilizarlas, pero en otros tuvieron que contentarse con grabaciones de agua en el set.

Img 2-8: El señor de los anillos las dos torres, 2002.

Img 2-9: El señor de los anillos las dos torres, 2002.

Fue en esa misma época en la que las películas de superhéroes se hacían un nombre dentro de la industria, por lo que la demanda de VFX aumentó considerablemente. Películas como X-Men, Spiderman-2 y Iron Man 1 demostraron la evolución del CGI y esta vez sí la completa implementación y necesidad de simulaciones 3D.

Uno de los estudios CGI más renombrados, Industrial Light and Magic (ILM), fue contratado para producir los efectos visuales de Iron Man 1 (2008). ILM se ha ganado el reconocimiento por su investigación y sus contribuciones a largo plazo en la industria CGI, además de los impresionantes resultados que ha producido. Pudieron incorporar con éxito una amplia gama de simulaciones 3D en la película, desde explosiones hasta dinámicas de destrucción, que cautivaron a los espectadores. Por tanto, se podría decir que las simulaciones 3D habían avanzado hasta el punto de que ahora eran necesarias para cualquier proyecto CGI de este calibre.

Img 2-10: Iron Man 1, 2008.

Aunque se había dado un gran paso, todavía quedaba mucho camino por recorrer hasta llegar al CGI que se conoce hoy en día.

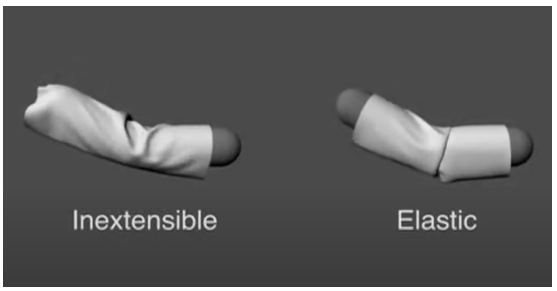
Las conferencias del SIGGRAPH seguían siendo una fuente constante de innovación año tras año. Gracias a las presentaciones realizadas durante estos eventos, se lograron grandes avances en el CGI durante la década siguiente, lo que permitió brillantes logros en esta área.

En estos 10 años (2000-2010) de evolución en el mundo del CGI, los avances que más se pueden destacar en el mundo de las simulaciones 3D son los siguientes:

- Simulaciones de fluidos avanzadas presentadas en el SIGGRAPH del año 2002.

Img 2-11: *Water Effects,* 2002.

- Simulaciones Cloth avanzadas que permiten manipular diferentes atributos de los materiales para lograr un resultado mucho más realista. Presentación expuesta en el SIGGRAPH del año 2007.



Img 2-12: *Efficient Simulation of Inextensible Cloth,* 2007.

- La película 2012 (2009) brinda al espectador entornos generados completamente por ordenador y simulaciones de destrucción extremadamente realistas y detalladas.

Img 2-13: 2012, 2009.

- Las simulaciones de gases no se quedan atrás con la posibilidad de calcularlas en la GPU, un avance presentado también en el SIGGRAPH en el año 2009.

Img 2-14: *Directable simulation of fire,* 2009.

Como conclusión, se abordará el avance de las simulaciones 3D con la trilogía de El Hobbit. Esto resulta más fácil de comparar al pertenecer al mismo universo que la trilogía de El Señor de los Anillos.

Weta Digital, fundada por el director Peter Jackson, fue la encargada de los efectos visuales en ambas películas. La empresa ha liderado la industria del cine en la creación de efectos visuales durante muchos años. Al igual que Industrial Light and Magic (ILM), Weta Digital no solo ha destacado por sus resultados sobresalientes, sino también por su contribución en el desarrollo de avances tecnológicos. La compañía ha desarrollado dos softwares propios para seguir innovando en la creación de efectos visuales.

Img 2-15: *Weta Digital*.

Uno de los cambios más importantes a la hora de grabar El Hobbit (2012) fue la implementación de paisajes generados completamente por ordenador y dejar de utilizar maquetas. Esto permitió a los artistas VFX utilizar simulaciones 3D cuando fueran necesarias.

Toda la evolución que hubo se vio reflejada a la hora de no tener demasiados problemas para poder hacer un efecto, es por ello que en toda la trilogía se pueden ver infinidad de simulaciones 3D fotorealistas. Desde largas simulaciones de fluidos hasta grandes simulaciones de fuego en combinación con destrucción.

En la actualidad, aunque las simulaciones 3D han alcanzado un nivel muy avanzado, las investigaciones en este ámbito continúan. Por lo que su aplicación no se limita únicamente al cine o la televisión, sino que pueden utilizarse en muchas otras áreas, como la visualización de datos o el tiempo real.

Img 2-16: El Hobbit: un viaje inesperado, 2012.

Img 2-17: *Real time viscoplastic deformation* (SIGGRAPH 2016)

2.1.4 Simulaciones de Fluidos

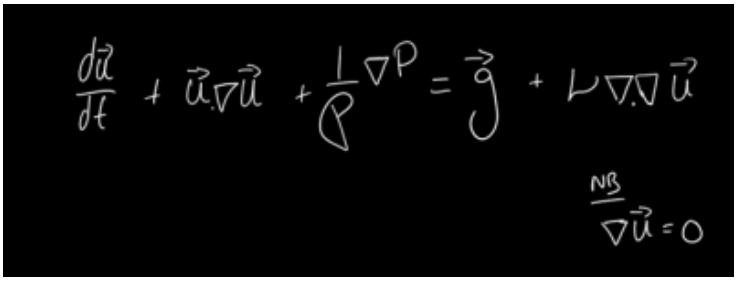
Las simulaciones de fluidos son un tipo de técnica utilizada en CGI para simular el comportamiento de los fluidos en movimiento, como el agua o gases. Son ampliamente utilizadas en la industria del entretenimiento, especialmente en la creación de efectos visuales para películas y videojuegos. Además, tienen aplicaciones en el campo de la ciencia y la ingeniería para simular el flujo de líquidos y gases en situaciones del mundo real, como el flujo sanguíneo en el cuerpo humano o la aerodinámica de un vehículo.

La mayoría de técnicas que se utilizan para este tipo de simulaciones están basadas en las ecuaciones de Navier-Stokes, un conjunto de ecuaciones que describen el comportamiento de varios fluidos incluyendo los gases, nombradas así en honor al ingeniero y físico francés Claude-Louis Navier y al físico y matemático irlandés George Gabriel Stokes.

El resultado de estas ecuaciones no es un simple número como 50, sino que es un vector que indica la velocidad, por lo que estamos hablando de velocity fields, y a su misma vez de una ecuación que describe la advección de las partículas. Es por ello que estas ecuaciones describen de qué manera se va a mover el fluido en un punto y tiempo determinado.

Estos fluidos se pueden describir con los siguientes fundamentos:

- Conservación de la masa, energía y volumen.
- La presión y la gravedad son las fuerzas principales que controlan el fluido.
- Las condiciones de los límites de la simulación, pueden ser muros u otros fluidos.
- Aceleración convectiva, es decir, la aceleración del fluido es controlada por la posición y no por la velocidad.



Img 2-18: Simplificación de la ecuación Navier-Stokes.

Simular fluidos es un desarrollo relativamente nuevo. Hasta finales de la década de 1990, los efectos como el tentáculo de agua en The Abyss se dibujaban en el ordenador frame por frame a mano.

Sin embargo, el inicio de las simulaciones de fluido se remontan a las décadas de los años 1950 y 1960 donde las simulaciones eran modeladas matemáticamente antes de que la industria del CGI apareciera. El grupo T3 del laboratorio nacional de Los Álamos dirigido por James Harlow fue el pionero de las técnicas que se emplean hoy en día para la creación de estas simulaciones.

Pero no fue hasta el año 1999 en el SIGGRAPH que Jos Stam presentó su paper científico "Stable Fluids" que cambiaría la visión de las simulaciones de fluidos, ya que propone un algoritmo estable que resuelve las ecuaciones completas de Navier-Stokes.

Basándose en un artículo de 1996 de Nick Foster y Dimitri Mataxas, que a su vez está basado en el trabajo de Harlow y Welch en CFD del año 1965, Stam mostró cómo las ecuaciones básicas de la dinámica de fluidos pueden calcularse con computadoras mucho menos potentes. Estas ecuaciones se integraron en diferentes software de animación que contribuyeron a los efectos de películas como El Señor de los Anillos y Piratas del Caribe. Pero las nuevas ecuaciones tenían dos defectos claros: no podían manejar fluidos incompresibles, y solo ayudaban a los animadores a simular, no a controlar los fluidos, por lo que supone un problema a la hora de crear efectos visuales.

En esa misma época, los años 1990, películas como Titanic estaban limitadas a crear escenas de fluidos de gran escala, como puede ser los océanos, por lo que simulaciones fotorealistas muy detalladas no eran posibles en ese momento. Es por ello que Stanley Osher y Ron Fedkiw inventaron el método Particle Level Set (PLS), para solucionar la pérdida de masa en las simulaciones.

El trabajo de Fedkiw fue altamente reconocido y llegó a ganar un premio al Logro Técnico por la Academia de Artes y Ciencias Cinematográficas.

Uno de los estudiantes de Fedkiw, Robert Bridson, trabajó junto a ILM como uno de los desarrolladores de PhysBAM, el código principal para las simulaciones físicas de la empresa. Bridson volvió a la investigación de los principales investigadores como Harlow e hizo que PIC/FLIP funcionara para flujos incompresibles mientras introducía simultáneamente su método incompresible FLIP. Además, ayudó a escribir *Squirt fluid simulator* para el estudio británico Double Negative, que se usaría en muchas películas de renombre como Harry Potter.

Img 2-19: Squirt fluid simulator, Double Negative.

Años más tarde, en 2008, estando en el Reino Unido, conoció a Marcus Nordenstam, que, con más de 20 años de experiencia en la industria, fue uno de los autores del paper presentado en el SIGGRAPH del año 2007 *Curl-Noise For Procedural Fluid Flow*, pionero del uso de FLIP para simulaciones de fuego, coautor de *Squirt fluid simulator* junto a Bridson y uno de los programadores de Zeno, software de simulación de ILM.

Nordenstam y Bridson formaron Exotic Matter en 2008, empresa creadora de uno de los programas de simulación de fluidos más importante en aquel momento, Naiad fluid sim software.

Img 2-20: Naiad Tech demo, SIGGRAPH 2009

Siendo utilizado en grandes producciones como Avatar, Harry Potter o Narnia, Naiad fue adquirida por Autodesk en el año 2012, lo que daría paso a un nuevo software de simulación de fluidos integrado en Maya, Bïfrost.

Otro software destacado de la industria es Realflow creado por la empresa española Next Limit Technologies, fundada por Víctor González e Ignacio Vargas en 1998. En 2001, se lanzó la versión inicial de RealFlow. Debido a sus características avanzadas y su interfaz fácil de usar, se convirtió rápidamente en uno de los softwares de simulación de fluidos comerciales más utilizados.

La trilogía de El señor de los anillos, 300, Pompeii, son solo algunas de las películas destacadas que han utilizado RealFlow a lo largo de los años. Con juegos conocidos como Mass Effect 3, RealFlow también se ha utilizado en la industria de los videojuegos.

En 2008, Next Limit Technologies lanzó RealFlow 4, que incluyó una serie de nuevas características como la simulación de cuerpos rígidos. En ese mismo año fueron galardonados con el Logro Técnico por la Academia de Artes y Ciencias Cinematográficas por su aportación a la industria.

Img 2-21: Realflow: Showreel by Next Limit, 2015.

Hoy en día, RealFlow sigue siendo uno de los software de simulación de fluidos más populares en la industria del entretenimiento y es utilizado por una amplia gama de estudios y artistas en todo el mundo, aunque destaca más por su uso en comerciales.

De todos estos años de investigación surgieron varias técnicas clave para las simulaciones de fluidos tal y como se conocen hoy en día.

Primero de todo se debe hacer una distinción clave para explicar los diferentes métodos, ya que se pueden clasificar de dos maneras, los eulerianos y los lagrangianos.

En un método euleriano, el fluido se describe como un conjunto de propiedades, como la velocidad, la presión y la densidad, que se definen en cada punto fijo en el espacio. Este método resulta especialmente útil para fluidos de gran escala.

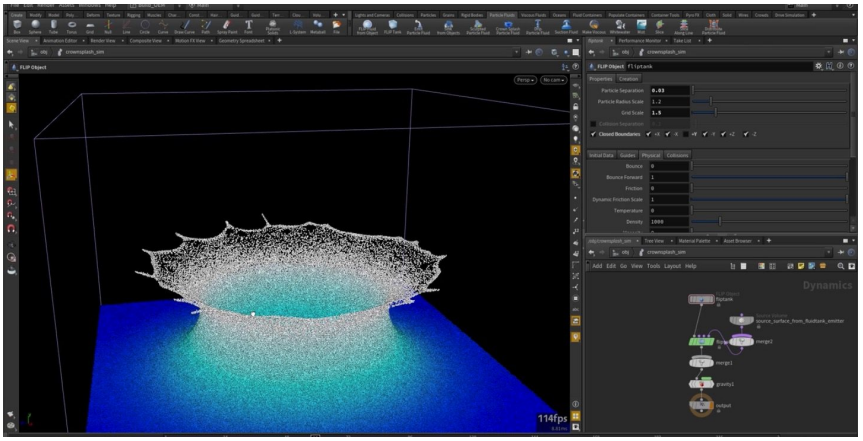
En cambio, el método lagrangiano hace un seguimiento de cada partícula individualmente a lo largo del tiempo, que puede resultar muy beneficioso para simulaciones de pequeña escala y mucho detalle.

SPH, o Smoothed Particle Hydrodynamics, es una técnica de dinámica de fluidos computacional lagrangiano que simula el comportamiento de los fluidos. La idea base detrás de la simulación SPH es representar el fluido como un conjunto de partículas, cada una de las cuales tiene características específicas como posición, velocidad, densidad y presión. Fue desarrollado por Gingold y Monaghan y Lucy en 1977, inicialmente para problemas astrofísicos.

Una de las ventajas de la simulación de fluidos SPH es que puede manejar geometrías complejas, que son difíciles de simular utilizando otros métodos. Sin embargo, las simulaciones SPH pueden ser computacionalmente costosas, especialmente a gran escala.

Otro método muy importante en la industria es FLIP (Fluid-Implicit Particle), que combina las ventajas de los métodos eulerianos y lagrangianos. Es por ello que se puede decir que es una mezcla de simulación de partículas y simulación de volúmenes.

Las partículas, usando métodos lagrangianos son realmente potentes para almacenar datos sin pérdidas pero no son tan buenas el cálculo de presiones precisas. No obstante, los volúmenes funcionan muy bien para hacer cálculos de presiones pero no a la hora de almacenar datos. De ahí que softwares como Houdini utilicen este método para el cómputo de sus simulaciones de fluidos.



Img 2-22: Flip Fluids, Houdini 16.

2.2 Houdini FX

Houdini FX es un software de animación y efectos visuales desarrollado por la compañía SideFX, presentado en el SIGGRAPH del año 1996. Es conocido por ser un software muy potente y versátil, especialmente en la creación de efectos visuales para cine y televisión.

A diferencia de otros programas de animación y efectos visuales que se basan en el modelado y animación de objetos estáticos, Houdini se enfoca en la creación de sistemas complejos y dinámicos que pueden ser controlados y ajustados de manera precisa a través de un conjunto de parámetros.

Una de las principales diferencias de Houdini con otros programas es su capacidad para crear contenido procedural. Esto significa que se puede crear una red de nodos que permite la creación de animaciones, simulaciones y efectos visuales de manera dinámica y paramétrica. Además, Houdini es capaz de simular fenómenos físicos de manera realista, lo que lo hace ideal para la creación de efectos visuales de alta calidad.

Otra característica distintiva de Houdini es su enfoque en la producción en equipo. Houdini está diseñado para facilitar la colaboración entre artistas y programadores, lo que permite la creación de proyectos complejos y la gestión de flujos de trabajo de manera más eficiente gracias a herramientas como los HDA. Además, su capacidad para manejar grandes cantidades de datos y procesarlos de manera eficiente lo hace ideal para proyectos de gran escala.

2.3 Simulaciones de fluidos para la ingeniería

Para desarrollar este trabajo también hay que entender cómo de importantes son las simulaciones de fluidos en el mundo de la ingeniería.

Las simulaciones de fluidos son una herramienta esencial para la ingeniería, permitiendo a los ingenieros comprender y optimizar el comportamiento de los fluidos en una amplia gama de sistemas y entornos.

Ser capaz de predecir con precisión cómo se comportan los fluidos en entornos y sistemas complejos es uno de los principales beneficios de las simulaciones de fluidos. Los ingenieros pueden modelar el flujo de fluidos en una variedad de sistemas, como canales, tuberías, conductos y más, mediante el uso de simulaciones de fluidos. También pueden usar estas simulaciones para mejorar el diseño de estos sistemas al identificar problemas potenciales como turbulencia de flujo, caídas de presión y otros problemas que pueden tener un impacto en el rendimiento y la eficiencia.

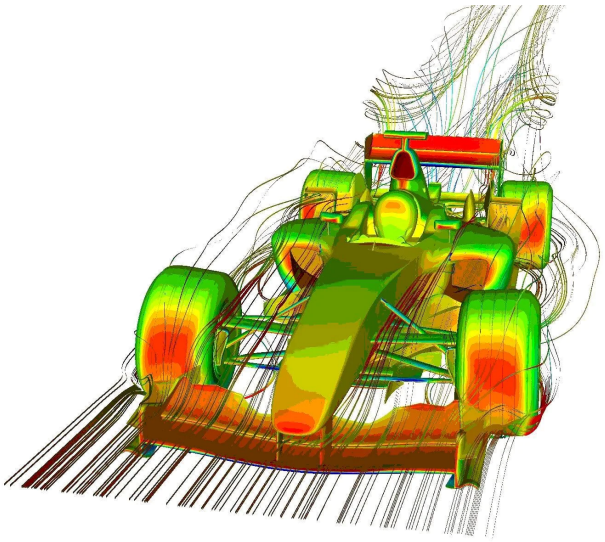
Los ingenieros pueden crear nuevos productos y tecnologías con la ayuda de simulaciones de fluidos, lo cual es otra ventaja. Por ejemplo, al simular el flujo de aire sobre la superficie del ala de un avión se pueden usar simulaciones de fluidos para optimizar el diseño de estructuras aerodinámicas. Al hacerlo, los ingenieros pueden optimizar la forma del ala, encontrar fallas de diseño y mejorar el rendimiento de la aeronave.

También se pueden usar para pronosticar cómo se comportan los fluidos en situaciones extremas, como las que ocurren en desastres naturales o accidentes industriales. Las simulaciones, por ejemplo, se pueden utilizar para modelar la dispersión química en caso de derrame o flujo de agua durante una inundación. Gracias a ello los ingenieros pueden crear planes para prevenir y

gestionar estos sucesos, reduciendo su impacto en las personas y el medio ambiente, mediante la simulación de estos escenarios.

Además son una herramienta rentable para la ingeniería, ya que pueden reducir la necesidad de prototipos y experimentos físicos. Al simular el comportamiento de los fluidos en un entorno virtual, los ingenieros pueden ahorrar tiempo y dinero, al tiempo que reducen el impacto ambiental del proceso de diseño.

En conclusión, las simulaciones de fluidos son una herramienta muy útil para los ingenieros, permitiéndoles obtener información sobre el comportamiento de los fluidos que sería difícil o imposible de obtener mediante experimentos físicos solamente. Con la ayuda de las simulaciones, los ingenieros pueden optimizar el diseño de sistemas, desarrollar nuevos productos y tecnologías y mejorar la seguridad y eficiencia en diferentes campos.

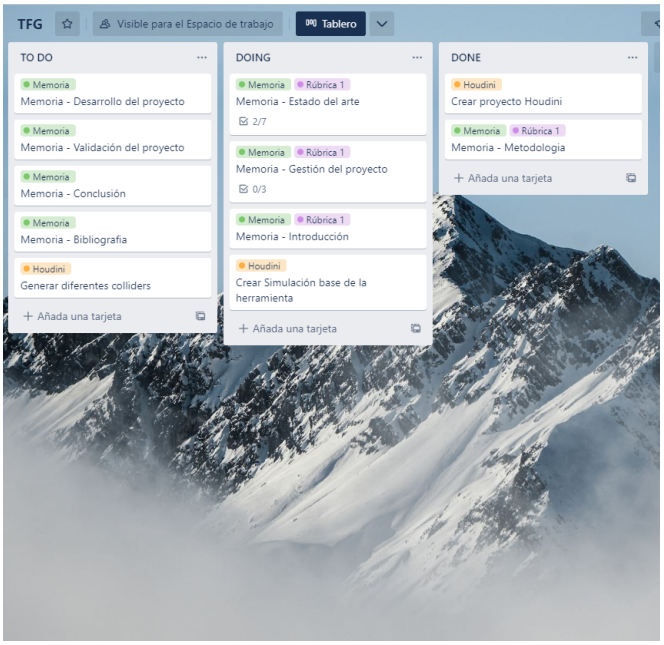


Img 2-23: Ejemplo aplicaciones simulaciones de fluidos.

3. Gestión del proyecto

Se ha implementado la metodología Kanban para la gestión del proyecto, la cual permite categorizar las tareas según su estado, ya sea completadas o en proceso de realización.

Para utilizar este método, se ha usado la plataforma Trello, ya que permite crear tableros con listas y tarjetas que representan tareas, y arrastrar y soltar las tarjetas entre las listas para indicar su estado. Por ahora están las etiquetas básicas que permiten planificar el inicio del proyecto pero más adelante se irán añadiendo más, como por ejemplo las funcionalidades de la herramienta.



Img 3-1: Metodología Kanban en Trello.

A medida que el proyecto ha ido avanzando también se ha ido actualizando el tablero, añadiendo nuevas tarjetas, ajustando fechas límites, etc..

Pese a tener una planificación previa al desarrollo del proyecto, no siempre se puede seguir con exactitud, ya que siempre pueden surgir problemas. Por ejemplo, al final de la fase 2 se tuvo que alargar el tiempo que se necesitaba para hacer la funcionalidad de las visualizaciones por la poca disponibilidad que se tenía en este proyecto por temas de la universidad.

No obstante, estos pequeños ajustes no han supuesto ningún problema a la hora de desarrollar el proyecto, ya que desde un principio se ha planificado todo para dar un margen de 3-5 días para todas las tareas.

Para organizar todo el tablero, se utilizaron 2 etiquetas principales, que distinguen las tareas relacionadas con la memoria y las de la herramienta en Houdini. Además, se crearon etiquetas para cada rúbrica, ya que de esta manera se pudieron establecer las fechas de entrega más fácilmente.

3.1 Dafo

El análisis DAFO es una herramienta de planificación estratégica utilizada para evaluar la situación actual y futura de empresas, organizaciones y personas. Las letras DAFO representan Debilidades, Amenazas, Fortalezas y Oportunidades. El análisis implica la identificación y evaluación de estos cuatro factores en relación con un objetivo específico. Las fortalezas y debilidades se refieren a aspectos internos y las oportunidades y amenazas se refieren a factores externos.

	Positivos	**Negativos**
Origen Interno	**Fortalezas** - Haber hecho trabajos personales FLIP con Houdini. - Gran motivación para estudiar simulaciones de fluidos. - Tener una base sólida del programa.	**Debilidades** - Limitación del hardware. - Falta de tiempo para el cómputo de las simulaciones.
Origen Externo	**Oportunidades** - Aprendizaje de simulaciones FLIP en Houdini. - Conseguir un proyecto con una alta calidad. - Acceso a diferentes fuentes de información.	**Amenazas** - No conseguir el objetivo o los resultados deseados. - Fuentes de información confiables.

A medida que ha ido avanzando el proyecto se ha ido modificando la tabla, ya que se han encontrado nuevas situaciones que pueden verse involucradas en el proyecto.

3.2 Riesgos y plan de contingencias

A medida que se avanza en el proyecto, es posible que surjan diversos riesgos que se deben considerar. Por esta razón, es esencial crear un plan de contingencia que incluya soluciones posibles para cada problema que pueda surgir, con el fin de poder reconducir el proyecto y evitar que se desvíe del objetivo inicial.

Riesgos	Soluciones
Fallo de hardware.	Utilizar otro ordenador hasta que se solucione.
No lograr desarrollar todas las utilidades de la herramienta.	Explicar las complicaciones que han habido y cómo se han intentado solucionar.
Pérdida de los archivos del proyecto.	Crear copias de seguridad en dos discos distintos.
Pérdida de progreso por crasheo del programa.	Generar archivos de respaldo cada 10 minutos y en momentos específicos. Por ejemplo, antes de empezar a simular.
Falta de tiempo para generar todas las simulaciones necesarias para explicar la herramienta.	Generar todas las simulaciones de todos modos pero con menor resolución.

3.3 Análisis inicial de costes

Si se tratara este proyecto como uno real, se debería calcular un presupuesto considerando el costo de los materiales, software y horas de trabajo. Para el ordenador se ha establecido que tendrá una vida útil de unos 6 años, aunque se estima que el periodo de amortización sea de unos 3 años, por lo que su coste mensual será de 93,75€.

Se ha utilizado como referencia para fijar el precio de cada fase el trabajo que corresponde a un artista VFX freelance en España que, aunque puede variar mucho, la media es de unos 30.000 € anuales.

Para determinar el precio de la luz en cada fase, se ha estimado que en la fase 2 y 3 será cuándo el ordenador necesitará más potencia por el cálculo de las simulaciones y cuando más tiempo se estará usando. Para ello se ha fijado una potencia de 5kW, lo que resulta en 76€ mensuales, 2,5€ por día, y si se estima trabajar unas 4h diarias serían 0.625/h.

	Precio producto	Precio por hora	Horas de trabajo	Precio total
Material				
Ordenador	3000€	0,8€	160h	125€
Disco seguridad	150€			150€
Total material				275€
Software				
Houdini INDIE	269€			269€
Total software				269€
Fase 1				
Simulación base	15€/h		30h	450€
Luz		0,625€/h		18,75€
Fase 2				
Desarrollo de la herramienta	15€/h		70h	1050€
Luz		0,625€/h		42,75€
Fase 3				
Finalización de la herramienta	15€/h		60h	900€
Luz		0,625€/h		37,5€
Total Fases			160h	2.499€
TOTAL				3.043€

4. Metodología

El proyecto se ha planificado en tres fases. La primera fase consiste en la definición de todas las funcionalidades y la creación de una simulación base que servirá como punto de partida para el desarrollo de la herramienta. En esta etapa, se dedicará tiempo a la investigación y análisis de los requisitos necesarios para la creación de la simulación.

La segunda fase se centrará en añadir todas las funcionalidades necesarias a la herramienta. Es en este momento en que se dará vida a la herramienta en sí misma, y se integrarán todas las características que se han planificado previamente. Se deberá realizar una planificación detallada de todas las funcionalidades a incluir, y en qué orden se desarrollarán para lograr el mejor resultado posible.

La última fase del proyecto será la etapa en la que se llevará a cabo la finalización de todas las utilidades de la herramienta y se realizará un análisis para detectar posibles errores. Es importante asegurarse de que todas las funcionalidades están bien integradas y que la herramienta en su conjunto funciona de manera eficiente y coherente. Esta fase también implica la realización de pruebas rigurosas para garantizar que la herramienta está funcionando de acuerdo a lo previsto, y la corrección de cualquier problema encontrado antes de la entrega final.

Las pruebas abarcarán la evaluación en usuarios con diferentes perfiles y conocimientos, tanto del programa como del mundo de los efectos visuales en general. De esta forma, se pueden identificar posibles problemas relacionados tanto con el rendimiento de la herramienta como con su comprensión. Además, al completar la herramienta se realizará un proceso de optimización en el que se intentará buscar el mejor proceso para el cálculo de cada funcionalidad, de esta manera los usuarios podrán llegar a ahorrarse mucho tiempo y la visualización en tiempo real no se verá perjudicada.

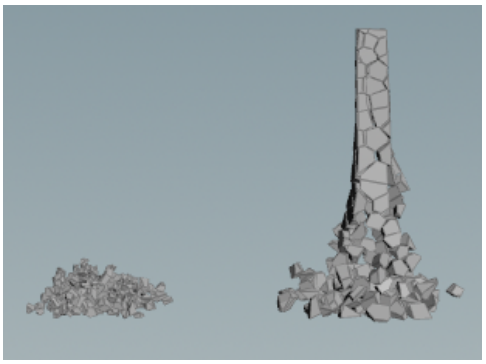
5. Desarrollo del proyecto

Como se ha comentado en el apartado anterior, este proyecto está dividido en tres fases, y la primera fase corresponde a la definición de las funcionalidades de la herramienta y realizar la simulación base. A pesar de eso, se ha visto que para desarrollar el proyecto de una manera más ordenada, en la primera fase se van a realizar todas las funcionalidades de la carpeta Setup y todo lo demás se ha movido a la segunda fase, aunque la tercera fase sigue siendo la misma.

Todo este cambio se ha ido documentando en los apartados que se han visto afectados a lo largo del desarrollo del proyecto, en especial los que tienen que ver con la gestión del proyecto. Por ejemplo, el análisis inicial de costes se ha modificado porque la estimación de horas de cada fase no fue la correcta, ya que la segunda se ha alargado.

Antes de empezar, se van a explicar ciertas características de Houdini para poder entender mejor cómo se va desarrollando la herramienta.

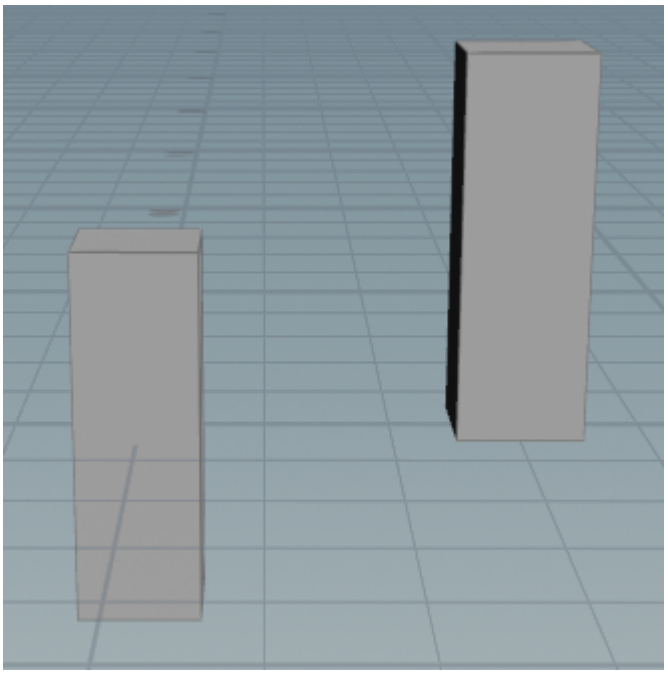
A diferencia de otros programas, Houdini trabaja en metros, por lo que resulta muy práctico a la hora de utilizar valores de objetos del mundo real. Lo cual está relacionado con las simulaciones, porque la escala de una simulación afecta directamente al comportamiento de los objetos. Por ejemplo, no cae de la misma manera un edificio de 5 metros que uno de 50.



Img 5-1: Ejemplo simulación RBD objeto 5m y 20m.

Como se puede observar en la imagen, el objeto de 20m cae mucho más lento pese a tener los mismos parámetros, es por eso que se deben respetar las unidades siempre que se vaya a simular.

Otro aspecto a tener en cuenta es cómo se crean las geometrías en Houdini. Siempre que se genera una nueva geometría lo hace con el centro en el eje de coordenadas (0,0,0). Esto puede ser útil en muchas ocasiones pero una desventaja en otras, por ejemplo, si se quiere escalar una Box en +Y y se hace sin realizar ninguna modificación, el objeto se escalará tanto en +Y como en -Y, por la posición del centro. Para cambiar esto, se puede copiar el parámetro que controla la escala Y y pegarlo en el centro en Y multiplicado por 0.5. De esta manera siempre que se escale el centro se posicionará y el objeto se escalará en +Y. Esto se aplica a todos los ejes.



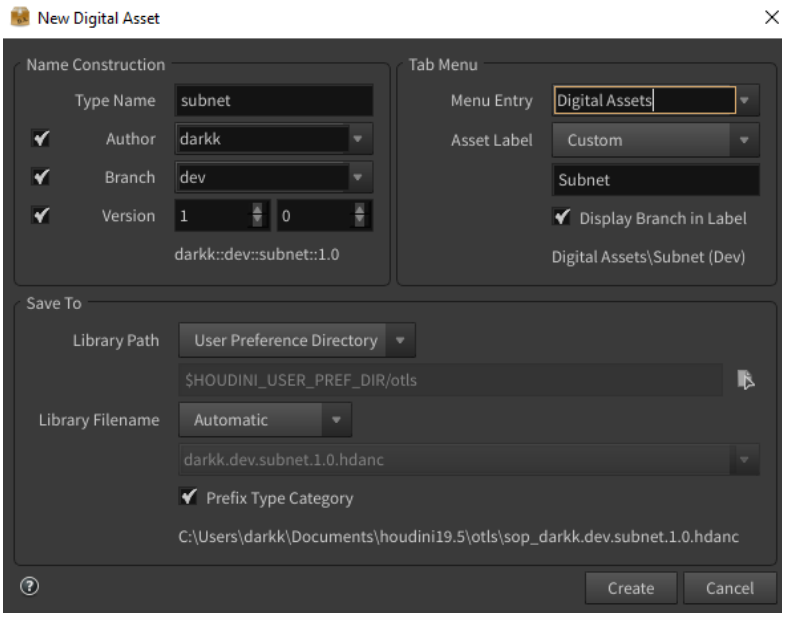
Img 5-2: Ejemplo creación geometrías Houdini.

En la imagen, el objeto de la izquierda tiene el centro en el eje de coordenadas (0,0,0), por lo que siempre que se escala en Y lo hace en los dos sentidos. En cambio, el centro de la geometría de la derecha se ha modificado para que sea siempre la mitad de la escala en Y, por lo que se escalará solo en +Y.

Por último, se va a explicar cómo se crea un HDA y cómo se configura y modifican los parámetros.

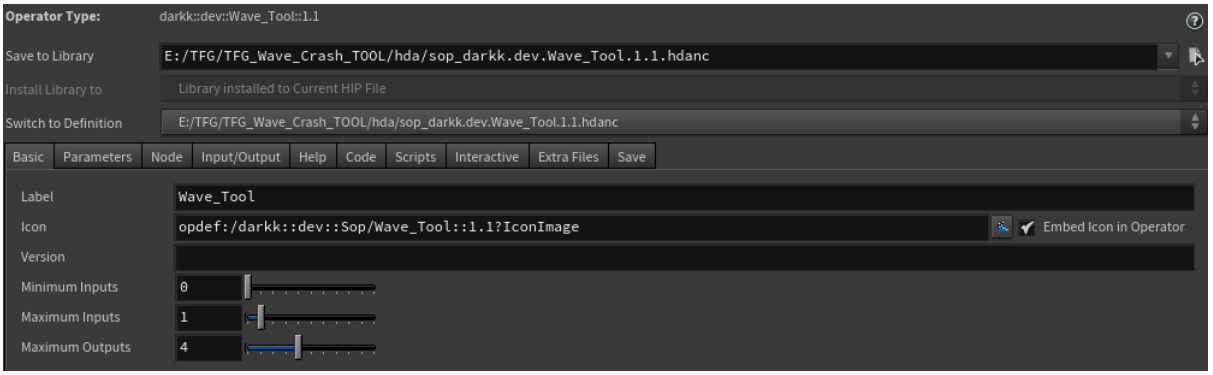
En Houdini, las escenas se construyen utilizando redes de nodos. Por ejemplo, los nodos de objeto representan personajes, accesorios, simulaciones o transformaciones en la escena, mientras que los nodos de geometría crean o modifican la geometría. Estas redes de nodos se pueden convertir en nodos personalizados reutilizables con sus propias interfaces, un digital asset.

Para crear un HDA primero de todo hay que tener unos nodos iniciales que se agrupan en un solo nodo creando una subnet. Después se crea el HDA sobre la subnet y se pueden ajustar diferentes parámetros antes de crearlo como se puede ver en la imagen de abajo.



Img 5-3: Interfaz creación HDA.

Una vez creado, se abre otra interfaz en la que se controla todo el HDA.



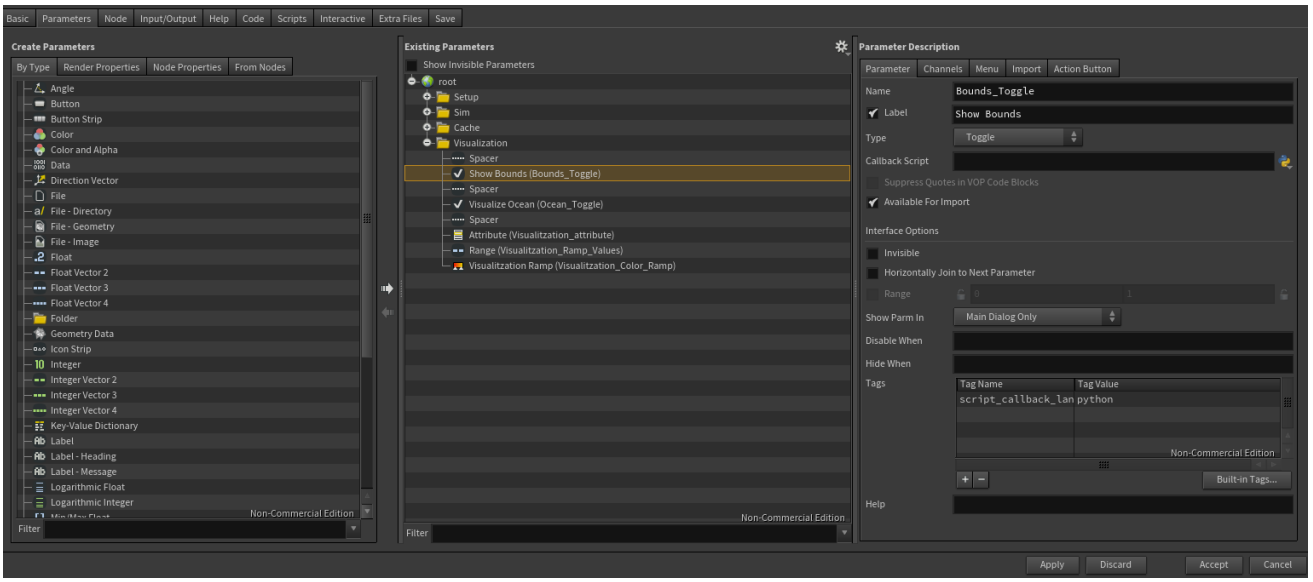
Img 5-4: Interfaz HDA.

De todas las pestañas, las que realmente se van a utilizar son Basic, Parameters e Input/Output, ya que las otras son para funcionalidades de HDA más complejas.

La carpeta Basic abarca todos los parámetros que tienen que ver con el HDA en general; el nombre que tendrá el nodo, el icono, la versión y el número de inputs máximos y mínimos así como outputs máximos para que funcione.

En cambio, la carpeta Parameters es la que se encarga de gestionar todos los parámetros que se incluyen en en HDA, por lo que es la más importante de todas. Se divide en 3 zonas diferentes, la del lateral izquierdo que se utiliza para acceder a cualquier tipo de parámetro que existe dentro de Houdini, que se arrastran a la zona central, que representa en forma de árbol los parámetros que se están utilizando. Finalmente, en la zona lateral derecha se pueden ver los diferentes parámetros que componen el parámetro seleccionado.

En esta imagen se toma como ejemplo el parámetro Bounds Toggle, que es un toggle para cierta funcionalidad. Al tratarse de un parámetro toggle, se puede acceder a ciertos parámetros de la zona de la derecha, como puede ser el nombre. Además, la pestaña de Channels permite visualizar a qué otros parámetros está vinculado y su valor inicial.



Img 5-5: Ejemplo parámetros HDA.

5.1 Fase 1

Una vez reajustado el planteamiento de la metodología del proyecto, primero de todo se ha creado una lista para poder ordenar las funcionalidades.

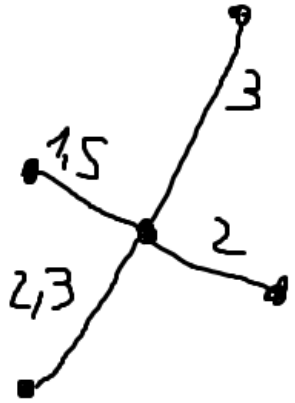
- Diferentes colliders para la simulación.
- Guardar en disco las simulaciones y colliders.
- Visualizador de varios atributos.
- Visualizar el océano que dirige la simulación.
- Control total de la dirección y velocidad de las olas.
- Añadir colliders secundarios.
- Escalar y mover el dominio de la simulación.
- Control total en la inicialización de las partículas.
- Varios parámetros para controlar los colliders.

Todas estas funcionalidades se han ordenado en 4 carpetas: Setup, Sim, Cache y Visualization.

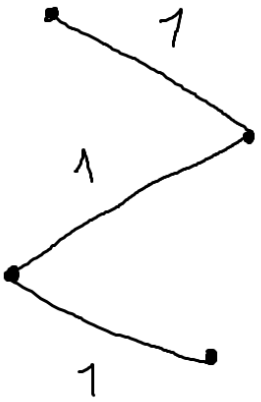
En la carpeta Setup se encuentra todo lo relacionado con la inicialización de las partículas, las transformaciones del dominio de la simulación y los diferentes colliders. Por lo que lo primero que se ha hecho para la creación del HDA es el contenedor de las partículas.

En Houdini hay diferentes maneras para generar puntos dentro de un contenedor, por lo que se ha tenido en cuenta el que mayor rendimiento tiene tanto para la simulación como para el HDA en general.

La primera forma en la que se ha pensado ha sido utilizando el nodo "Particle Fluid Tank", que permite generar puntos dentro de un contenedor. El problema de este nodo es que puede resultar algo lento, ya que muchos parámetros no son necesarios para el HDA que se va a crear. La alternativa que se ha pensado ha sido utilizar otro nodo muy similar pero que simplemente genera puntos a partir de una geometría dada, "Points from Volume". Sin embargo, tiene el mismo problema que "Particle Fluid Tank", muchos de los parámetros no son realmente necesarios y puede ralentizar la herramienta. A pesar de esto, si se ajusta el parámetro Jitter Scale a un número mayor que 1, hace que el nodo sea lo más eficiente posible, ya que no necesita hacer cálculos extra para ordenar los puntos. Por lo que esta es la manera en la que se inicializa el contenedor de las partículas en el HDA.



Img 5-6: Jitter Scale > 0.



Img 5-7: Jitter Scale = 0 .

Como se puede observar en las imágenes de arriba, siempre que el Jitter Scale sea = 0 la distancia entre los puntos será igual entre todos. En cambio, cuando es mayor a 0 la distancia va a ser distinta dependiendo del valor que se le asigne al parámetro.

Por ahora, se tienen los puntos que se han generado utilizando el nodo "Points From Volume". Sin embargo, no se puede modificar ninguno de los parámetros ni mover el contenedor. Es por ello que se necesita definir qué parámetros se podrán controlar a través del HDA.

Dentro de la carpeta Setup, se ha creado una subcarpeta llamada Container, en la que se añadirán todos los parámetros que controlan el estado inicial de las partículas. Dentro de esta subcarpeta se han añadido los siguientes parámetros, de los cuales es importante tener en cuenta qué tipo son y qué función cumplen:

- **Particle separation** – Float – Controla la cantidad de partículas que tendrá la simulación.

- **Jitter Scale** – Float – Controla la cantidad de randomización que se aplica a la posición de los puntos.

- **Jitter Seed** – Float– Controla la seed de randomización.

- **Top Padding** – Float – Controla la altura que tendrá el dominio de la simulación.

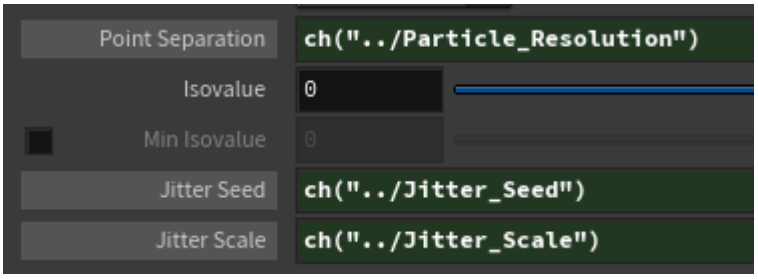
- **Translate** – Vector3 – Controla el vector de transformación del contenedor en los 3 ejes.

- **Size** – Vector2 – Controla el vector de escala del contenedor en los ejes XZ.

- **Water Level** – Float – Controla el nivel del agua.

Una vez definidos, se han añadido los parámetros a la interfaz del HDA y se han inicializados sus valores en la ventana de Channels, como ya se ha comentado anteriormente. No obstante, esto simplemente los ha añadido a la interfaz y no se han vinculado con los nodos que controlan cada funcionalidad.

Particle Separation, Jitter Scale y Jitter Seed son parámetros que se encuentran en el mismo nodo de "Points from Volume", por lo que simplemente hay que conectar los dos Channels. Para ello se puede copiar la referencia del parámetro del HDA y pegarla directamente en el que corresponde.



Img 5-8: Ejemplo ruta parámetros HDA.

Como se puede observar en la imagen, estos 3 parámetros tienen una ruta de referencia a los valores que se controlan a través del HDA.

El parámetro Top Padding es un poco más complejo de implementar, ya que al tratarse de una herramienta procedural, muchas funcionalidades tienen un efecto directo sobre otras. Por lo que primero se han implementado las relacionadas con las transformaciones del contenedor de las partículas.

Tanto el parámetro Translate y Water Level se controlan a través de dos nodos "Transform", en uno se ha referenciado el vector de translación en el parámetro Translate y en el otro el vector de escala en el ScaleY. El parámetro Size se controla directamente desde la Box en el que se generan los puntos y, como para el HDA se necesita que las escalas se produzcan sólo en el eje -Z, y no en Z y -Z, se ha implementado lo ya explicado, copiar la escala en el centro y multiplicarla por 0.5.

El último parámetro que queda por configurar es el Top Padding, por lo que primero se debe crear la geometría que representa las dimensiones del contenedor de la simulación y por último añadir el parámetro que va permitir manipular la altura. El nodo "Bounds" es perfecto ya que realiza justo lo que se necesita, generar la Bounding Box del objeto para después añadir el nodo "Ends" que la convierte en Wireframe.

Como esta geometría ha sido creada a partir del nodo que controla las transformaciones del contenedor de las partículas se escala y traslada sin tener que volver a copiar los parámetros Translate y Size. Sin embargo, la altura de la Bounding Box todavía no se puede manejar a través del parámetro Top Padding, por ello se ha referenciado en el parámetro dentro del nodo "Bounds" que permite ajustar la altura de la Bounding Box, Upper Padding en el eje Y.

Img 5-9: Canal que controla el parámetro Top Padding del HDA.

Una vez acabada la carpeta Container, se deben explicar los parámetros que conformarán la carpeta Wall, que se encargan de controlar el muro que sujeta los Colliders:

- **Enable** – Toggle – Controla si el Wall está activo o no.

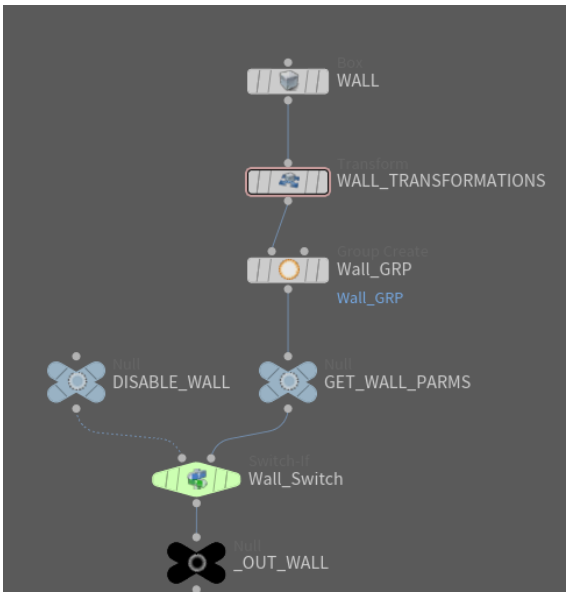
- **Height** – Float – Controla la altura del Wall

- **Width** – Float – Controla el ancho del Wall.

Para esta funcionalidad del HDA se debe tener en cuenta que no solo depende de los parámetros de la carpeta Wall, sino que se debe mover y escalar junto al contenedor de las partículas, ya que de esta manera se mantiene en todo momento un trabajo procedural.

Para generar la geometría se ha utilizado una Box a la que se le ha cambiado el centro en los ejes Y y Z para que cuando se escalen lo hagan en +Y y -Z, tal y como se ha visto ya con el contenedor de las partículas. Haciendo uso del nodo "Transform", el parámetro que debe mover esta geometría es el mismo que mueve el contenedor de las partículas, por lo que se copia esa referencia y se pega en el parámetro Transform del nodo "Transform" creado para la Box. En este mismo nodo se copian también las rutas que controlan la escala, la escala en X se controla por el parámetro que también controla la escala en X del contenedor de las partículas, la escala en Y por el parámetro Height del Wall y la escala en Z por Width.

Finalmente, para poder activar y desactivar el Wall haciendo uso del parámetro Enable, se utiliza el nodo "Switch-If", que permite cambiar entre dos inputs dada una expresión. En este caso, la expresión es la ruta del parámetro toggle Enable, que devuelve valor 1 si está activa y 0 si no lo está. Es por ello que en el primer input, que es cuando está desactivado el toggle, se conecta un nodo "Null", que su función es no hacer nada, por lo que no se generará ninguna geometría. En cambio, para el segundo input, se conecta todo lo explicado en el párrafo anterior, ya que en este caso el toggle está activo.



Img 5-10: Estructura de nodos para la creación del Wall.

Siguiendo con la carpeta Setup, se ha creado otra subcarpeta llamada Collider, en la que se accede a los parámetros que controlan los diferentes colliders de la simulación:

- **Enable** – Toggle – Controla si el Collider está activo o no.
- **Collider** – Ordered Menu – Controla qué Collider está activo.
- **Height** – Float – Controla la altura del Collider.
- **Width** – Float – Controla el ancho del Collider.
- **Arc Radius** – Float – Controla el radio del Collider Arc.
- **Arc Width** – Float – Controla el ancho del radio del Collider Arc.
- **Spike Direction** – Float – Controla la dirección del Collider Spike.

Para la creación de las geometrías de esta funcionalidad se ha empezado creando una Box que sirve de referencia para realizar todas las transformaciones necesarias, ya que de esta manera se hace tan sólo una vez y no una por cada Collider. Una vez más, la escala se trabaja tan solo con el eje +Y y -Z y el parámetro Width del Collider ya se puede copiar en la escala Z. Como el Collider también debe moverse junto al contenedor de las partículas se utiliza de nuevo un nodo "Transform" en el que se copia la ruta del parámetro Translate del contenedor y se pega en el parámetro Translate del nuevo nodo. Además, el Collider también debe mantener la escala en X del contenedor de las partículas, por lo que este valor es controlado por el parámetro Size X del contenedor.

Una vez creada la geometría de referencia, se han generado los diferentes colliders de la siguiente manera:

Para el Collider Box se ha creado una Box sin modificaciones, ya que todo se moverá y escalar a través de la geometría de referencia.

Para el Collider Arc se ha utilizado el Collider Box y se ha creado un Tube que se sustrae de la Box utilizando una booleana. Como hay colliders que tienen parámetros del HDA que únicamente funcionan en ellos mismos, se deben generar en el collider y no en la geometría de referencia. De esta manera, el parámetro Arc Width se controla con la escala X del tubo y el parámetro Arc Radius con la escala en Y y en Z del tubo.

El siguiente es el Collider Spike, que la gran parte del trabajo es de modelado. Se ha empezado con una Box a la cual se le han juntado dos edges de la parte superior para conseguir la forma en punta que se desea. Seguidamente se han seleccionado los dos puntos de la parte superior y se les ha conectado un nodo "Transform" para poder controlar el parámetro Spike Direction, cuya función es mover en el eje Z los dos puntos seleccionados anteriormente. Cabe destacar que este parámetro sólo puede tener valores entre -0.5 y 0.5, que ha sido configurado a través de la interfaz de parámetros del HDA.

Img 5-11: Valores del parámetro Spike Direction limitados.

Por último, el Collider Custom será generado o importado por el propio usuario en Houdini y se conectará al primer input del HDA. De esta manera se abre la posibilidad de que el Collider se modele dentro de Houdini o en otro programa para después implementarlo en el HDA.

Una vez todos los colliders han sido creados, hay que hacer que todos se ajusten a las medidas que proporciona la geometría de referencia, que, como se ha explicado anteriormente, es la que realmente controla los parámetros a nivel general de los colliders. Por ende, se ha utilizado el nodo "Match Size", que permite modificar las transformaciones del objeto conectado al primer input utilizando como referencia el objeto del segundo input.

Además, el parámetro toggle que activa o desactiva el Collider todavía no ha sido vinculado, aunque funciona de la misma manera que el Enable del Wall. Por lo tanto, se ha utilizado la misma técnica creando un "Switch-If", pero se ha creado otro nodo igual con la diferencia de que es controlado por el parámetro toggle del Wall, ya que si el Wall no está activo tampoco lo estará el Collider.

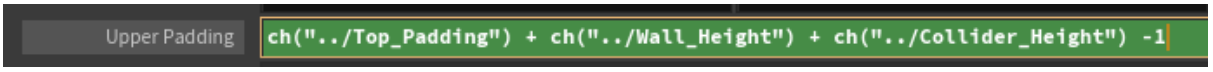
No obstante, todavía no se puede seleccionar cada Collider, ya que el ordered menu no está vinculado a nada. Para ello, es importante saber que un ordered menu es un array, por lo que se utilizará una técnica muy similar a la de activar o no el Collider. En este caso se ha utilizado un nodo "Switch" que permite tener tantos inputs como se desee, pero se deben ordenar de la misma manera que están ordenados en el parámetro del ordened menu, ya que cada elemento del array debe coincidir con el del switch. Una vez se han ordenado correctamente, se ha copiado el array del parámetro del menu y se ha referenciado en el array del switch.

Asimismo, hay que reposicionar el Collider para que se sitúe siempre por encima del Wall, para eso se ha usado una expresión muy simple en un nodo "Transform".

Img 5-12: Expresión BBOX para obtener el valor del YMAX de un nodo específico.

Esta expresión devuelve el valor de las dimensiones máximas en un eje de un nodo específico, en este caso la Y máxima del Wall.

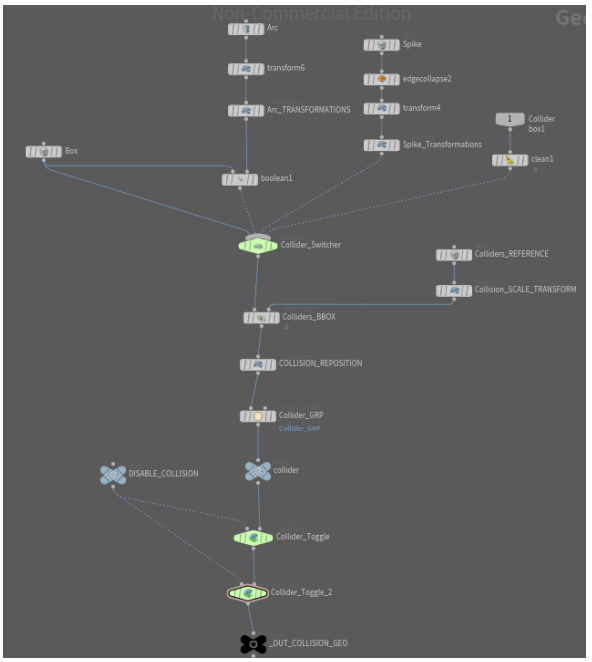
Finalmente, como ya se ha mencionado anteriormente, muchos parámetros del HDA dependen de varios valores para que la herramienta funcione correctamente y de manera procedural. Es por eso, que la expresión que controla el parámetro Top Padding de la subcarpeta Container debe ser modificada para que se ajuste a la altura del Wall y del Collider.



Img 5-13: Expresión que controla el Top Padding.

Esta nueva expresión tiene en cuenta todo lo necesario para que el número que aparece en el parámetro Top Padding sea realmente la distancia que hay entre el punto más alto de los colliders y la altura máxima del contenedor.

No obstante, hay que crear otro nodo "Bounds" con otra expresión, ya que cuando el Collider no está activo no se debe tener en cuenta su altura. Por lo que al tratarse de una condición, se vuelve a utilizar el nodo "Switch-If", que depende del parámetro toggle Enable del Collider.



Img 5-14: Estructura de nodos para la creación de los colliders.

La siguiente funcionalidad a crear es la Ramp, que se situará siempre delante del Wall y será otra colisión a tener en cuenta para la simulación:

- **Enable** – Toggle – Controla si la Ramp está activa o no.

- **Height** – Float – Controla la altura de la Ramp

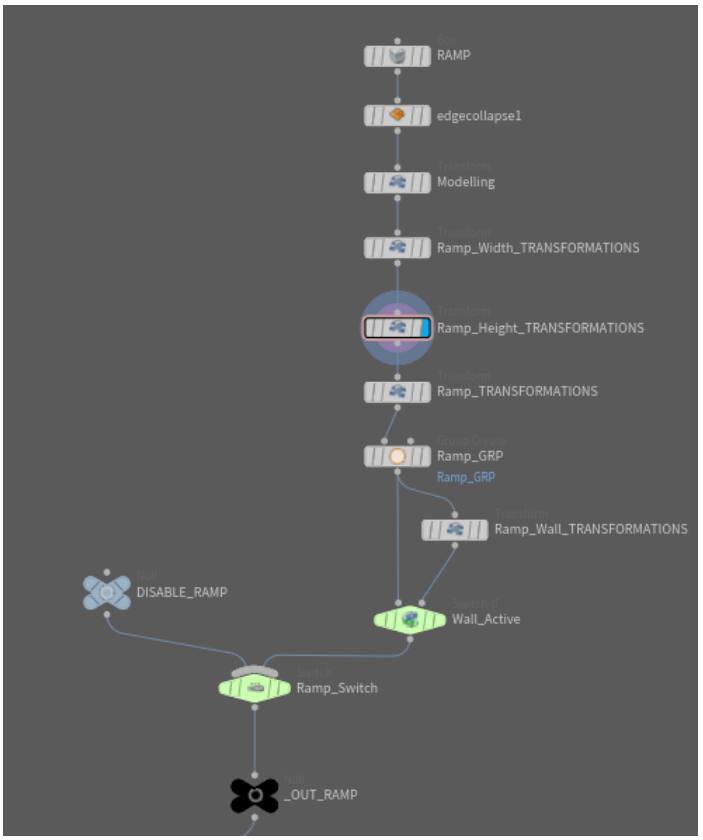
- **Width** – Float – Controla la longitud de la Ramp.

Una vez más, se ha comenzado con el modelado de la geometría a partir de una Box a la que se le han juntado dos edges de la parte superior. En esta misma Box ya se puede copiar la referencia de la escala en X del contenedor de las partículas, ya que la Ramp debe ocupar siempre la amplitud máxima. Siguiendo la misma técnica que en las otras geometrías, se ha usado el nodo "Transform" para aplicar los parámetros del HDA. Se han seleccionado los dos puntos frontales en el eje Z y se trasladan según el parámetro Width. De la misma manera se han seleccionado los dos puntos superiores de la geometría y se trasladan en el eje Y con el parámetro Height.

Para las transformaciones de la Ramp en referencia al contenedor, se ha utilizado otro nodo "Transform" al que se le han copiado en X e Y las transformaciones del contenedor. En cambio, para el eje Z, como la rampa debe situarse siempre en frente del Wall, se ha aplicado de nuevo la expresión BBOX para obtener la distancia que debe ser desplazada la Ramp, en este caso la amplitud del Wall.

Por último, para el parámetro enable se ha utilizado el mismo método que en las otras funcionalidades, utilizando un "Switch-If". No obstante, la Ramp sí que puede estar activa aunque el Wall no lo esté, por lo que la distancia que se desplaza en Z debe ser corregida dependiendo del estado. Con ese objetivo se ha creado otro "Switch-If" que varía entre los inputs dependiendo del toggle del

Wall. En el primer input se ha conectado la Ramp sin el nodo "Transform" que lo desplaza en Z y en el segundo se ha conectado con ese nodo.



Img 5-15: Estructura de nodos para la creación de la Ramp.

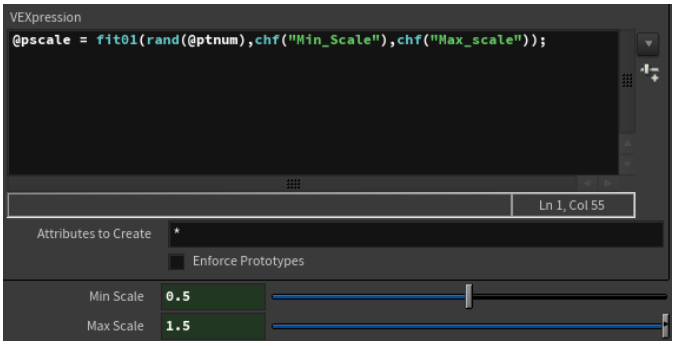
Como última funcionalidad dentro de la carpeta Setup, se encuentra la generación de Rocks dentro del contenedor de partículas, lo que va a permitir más variedad e interés cuando se vaya a simular:

- **Enable** – Toggle – Controla si las Rocks están activas o no.
- **Amount** – Int – Controla la cantidad de Rocks generadas.
- **Seed** – Float – Controla la seed de Randomización de la posición de las Rocks.
- **Relax iterations** – Int – Controla en qué medida se separan las Rocks entre ellas.
- **Min_Scale** – Float – Controla la escala mínima de las Rocks.
- **Max_Scale** – Float – Controla la escala máxima de las Rocks.

Para poder generar las Rocks se han utilizado diferentes técnicas de scatter y randomización para lograr un aspecto orgánico. Es por eso que se ha empezado con una grid en la que se han añadido ya los parámetros de escala y traslación del contenedor de partículas en un nodo "Transform", ya que las Rocks deberán moverse con el contenedor también.

No obstante, siempre que haya un collider no se deben generar Rocks, por lo que se han sustraído de la grid toda geometría que está dentro de los colliders, esto incluye la Ramp, el Wall y el Collider. Para ello se ha utilizado el nodo "Group", que permite agrupar puntos del primer input y realizar las operaciones necesarias con la geometría del segundo input, en este caso agrupar las que interseccionan con los colliders. Tras agrupar los puntos se ha utilizado el nodo "Blast" para eliminar la geometría del grupo creado anteriormente.

Una vez creada la grid con todos los parámetros bien configurados, se han generado los puntos en los que se van a instanciar las Rocks usando el nodo "Scatter", en el que se pueden pegar ya la ruta de los parámetros Amount, Seed y Relax iterations. En cambio, los dos parámetros que controlan la escala se han generado a partir de un código utilizando el atributo @pscale.



Img 5-16: Código para controlar la escala de las Rocks.

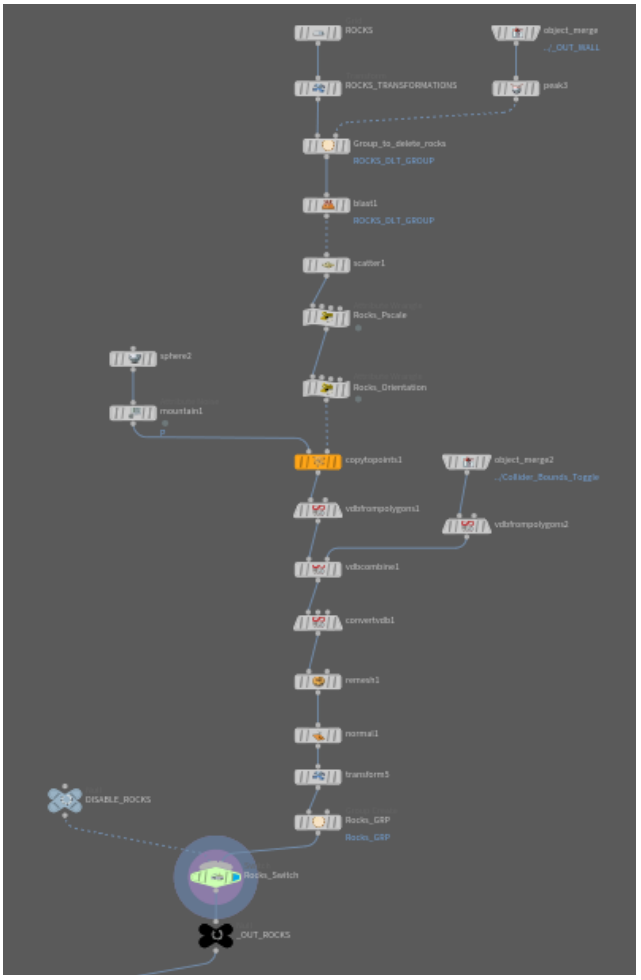
Este código se ha escrito dentro del nodo "Attribute Wrangle" y utiliza el lenguaje de Houdini llamado Vex. Como el atributo que controla la escala es @pscale, se han generado números random para cada punto utilizando la función rand(). Sin embargo, esta función devuelve valores entre 0 y 1, y lo que se busca es que sean valores entre Min_Scale y Max_scale, los parámetros del HDA. Por eso mismo se ha usado la función fit01(), a la que se le han creado dos Channels y añadido a la interfaz para poder vincularlos con los parámetros Min_Scale y Max_scale.

Por otra parte, todavía no se han generado las Rocks, para ello se ha utilizado el nodo "Copy to Points", ya que permite copiar la geometría del primer input a los puntos del segundo input. En el primer input se ha creado una esfera y con un nodo "Mountain" se le ha añadido ruido para reproducir el aspecto de una roca. Como ya se han configurado los parámetros de escala en los puntos, el nodo "Copy to Points" lo aplica automáticamente a las esferas que se han copiado.

Como último paso, se han eliminado las partes de las Rocks que sobresalen de los límites del contenedor. La primera técnica en la que se ha pensado ha sido en la de agrupar las partes que sobresalen y después eliminarlas utilizando un "blast". No obstante, en este caso, esta técnica resulta algo lenta y poco eficiente por la cantidad de cálculos que se deben hacer para eliminar todas las geometrías necesarias. La manera en la que se ha pensado para evitar estos problemas, ha sido utilizando VDB, ya que permite hacer cálculos muy precisos sin prácticamente afectar al rendimiento del HDA. Además, con esta técnica, las Rocks que se generen una al lado de la otra se juntarán como una única, dándole un aspecto más orgánico.

Primero de todo se han convertido en SDF tanto las Rocks como los Bounds del contenedor de las partículas haciendo uso del nodo "VDB from Polygons". Seguidamente, se ha utilizado el nodo "VDB Combine" en modo SDF intersection para mantener las partes de las Rocks que están dentro del contenedor. Por último se ha vuelto a convertir los VDB en geometría con el nodo "Convert VDB".

Finalmente, el parámetro toggle Enable se ha implementado de la misma manera que todos los otros, ya que es el que controla si las Rocks están activas o no.



Img 5-17: Estructura de nodos para la creación de las Rocks.

Para finalizar, el contenedor de las partículas ha sido modificado, ya que donde haya colliders no deben haber partículas. En este caso, se ha utilizado la misma técnica que se ha usado anteriormente para eliminar partes no deseadas de una geometría. Se han juntado en un grupo todos los puntos del contenedor de las partículas que están dentro de los colliders y, seguidamente, se han eliminado con el nodo "Blast".

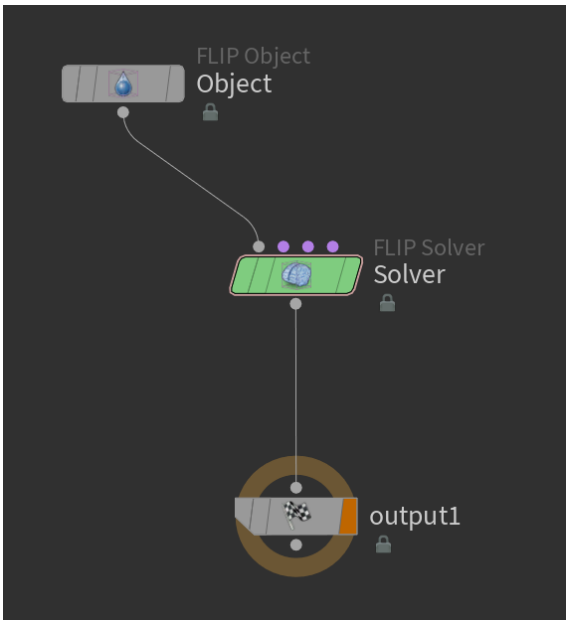
Con este último apartado concluye la primera fase del desarrollo del proyecto.

5.2 Fase 2

Como ya se ha explicado, la planificación de la segunda fase ha sido modificada, por lo que en esta se van a acabar de hacer todas las funcionalidades del HDA. Esto incluye las carpetas Sim, Cache y Visualization y la simulación.

Para una mejor organización, se ha empezado con la simulación, porque de esta manera más adelante resulta más sencillo crear y vincular los parámetros del HDA.

Cualquier tipo de simulación dentro de Houdini debe de estar dentro de una DOP Network, es decir, ya no se trabaja a nivel de objeto sino que se ha pasado a otro contexto que funciona de manera distinta. Toda simulación necesita como mínimo 2 nodos para que funcione: el Solver y el Object, uno se encarga de realizar todos los cálculos y el otro de almacenar datos, respectivamente. Como para la herramienta se está trabajando con FLIP, se utilizaran los nodos que corresponden a este tipo de simulación, FLIP Solver y FLIP Object.



Img 5-18: Ejemplo simulación básica FLIP.

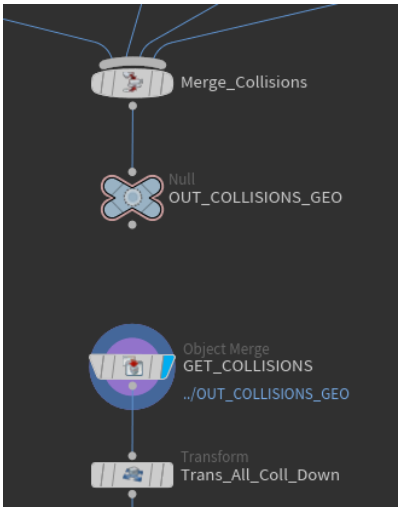
Para especificar qué objeto fuera de la DOP corresponde a las partículas que irán a la simulación se debe copiar la ruta en el parámetro SOP Path del FLIP object. Además se ha puesto el input como Particle Field, ya que, como se ha visto en la Fase 1, se ha creado un contenedor de partículas para la simulación.

Otro factor importante cuando se trabaja con FLIP, es el dominio de la simulación, puesto que la simulación de volúmenes que ocurre internamente en FLIP debe tener un dominio. Este dominio debe ser igual a los parámetros que ya se han asignado de los Bounds del contenedor, por lo que simplemente se han referenciado.

No obstante, si se prueba la simulación en este punto, se puede ver cómo las partículas se quedan quietas, esto se debe a que no hay ninguna fuerza aplicada para que se puedan mover. Es por ello que la primera fuerza que se ha añadido es una de las más importantes, la gravedad. En este punto, si se visualiza la simulación se puede observar como las partículas caen, pero con un problema, van desapareciendo. La causa de esto es que el dominio de la simulación no está cerrado en ninguno de los ejes y las partículas lo atraviesan, ya que el parámetro que se encarga de esto, que es Closed Boundaries, no ha sido modificado. Para prever este problema, se ha cerrado el dominio en todos los ejes excepto en +Y, porque si hay partículas que salen disparadas hacia arriba no deben colisionar.

Por ahora se tiene una simulación básica de FLIP en la que tan solo hay fuerza de la gravedad. Por consiguiente, todavía hay que añadir los colliders y la fuerza que moverá las partículas. Como lo único que hay por el momento son los colliders se han añadido a la simulación de la siguiente manera.

Primero de todo, para mantener el HDA bien organizado y entendible, se han juntado todos los colliders en un solo null utilizando el nodo "Merge". Con el nodo "Object Merge" se ha llamado al null en otra zona de la Network para poder gestionar las colisiones de la simulación y se han movido levemente hacia abajo para prever problemas en la simulación.

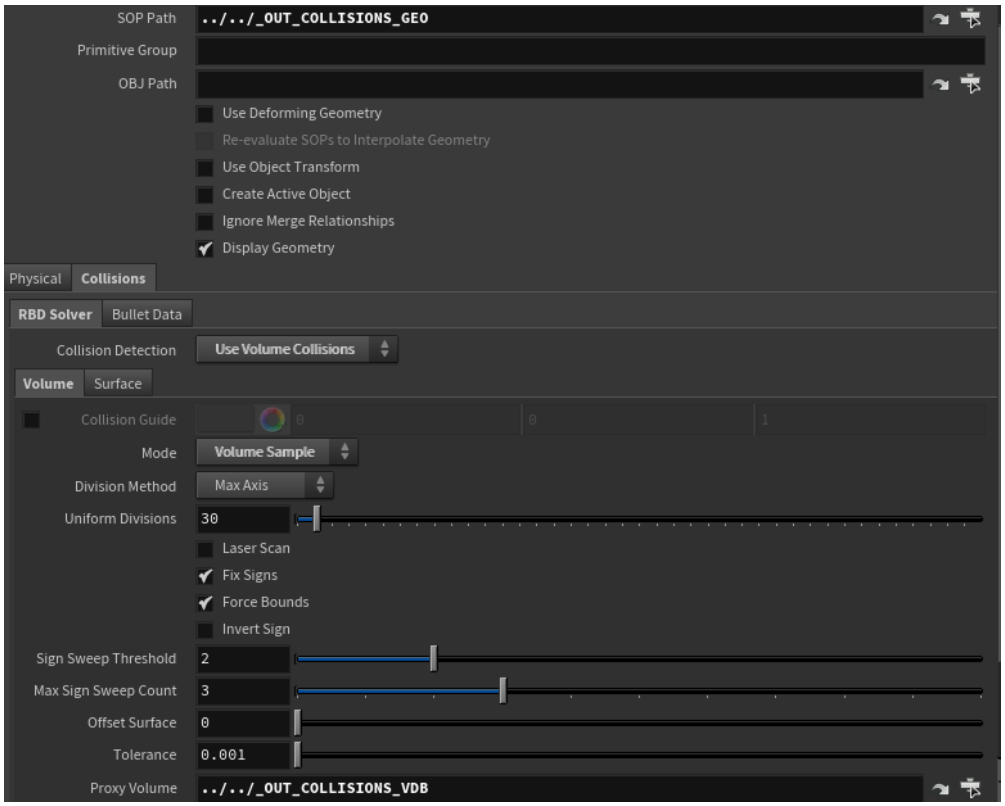


Img 2-19: Organización de colliders para la simulación.

Un aspecto importante a la hora de preparar colisiones para simulaciones es que las colisiones se pueden calcular de varios modos, como por ejemplo, con Heightfield, Ray Intersect o incluso con VDB, que es el que se ha usado. Para ello se ha extraído la geometría de los colliders por una parte y por la otra el VDB con el nodo "Collision Source".

Para poder introducir todo esto dentro de la simulación se ha utilizado el nodo "Static Object" dentro de la DOP y se ha conectado a la izquierda de un "Merge" junto al Solver. Este punto es muy importante, ya que dentro del contexto DOP los merge tienen en cuenta el orden en el que se conectan los nodos, porque una de las características de este nodo es que los inputs que se conectan a la izquierda afectan a los de la derecha, y es eso precisamente lo que se quiere, que los colliders afecten al comportamiento de las partículas.

Una vez configurado los nodos, se ha pegado la ruta de la geometría y del VDB en el "Static Object", que tiene como modo de cálculo Volume Sample.



Img 5-20: Configuración del Static Object usando cálculos de VDB.

El siguiente elemento que se va a añadir a la simulación es la fuerza que mueve a las partículas. La primera manera en la que se ha pensado ha sido actualizando la velocidad en Y de las partículas que se sitúan más alejadas de los colliders para generar de esta manera olas. Sin embargo, esto puede resultar algo poco natural y con un movimiento muy forzado.

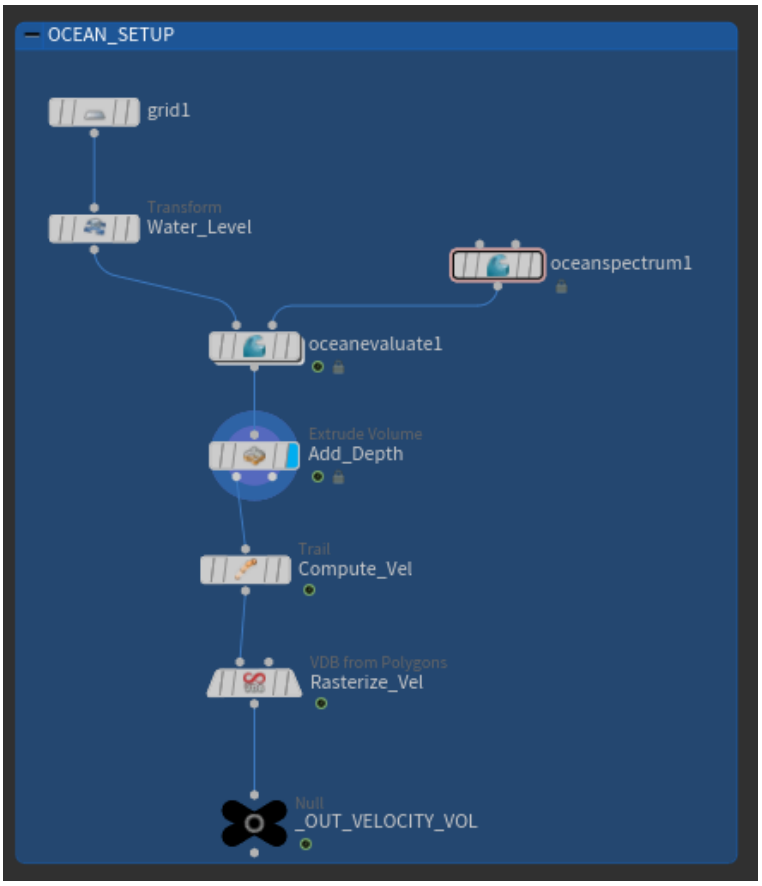
Es por eso que se ha decidido utilizar una técnica que usa las propias herramientas de océanos que ofrece Houdini, ya que de esta manera sí que se consigue un resultado orgánico. Por esa razón, primero de todo se va a crear un océano al cual se le va a extraer la información de la velocidad y se transferirá a las partículas.

Para generar un océano en Houdini se necesitan tan solo 2 nodos: "Ocean Spectrum" y "Ocean Evaluate". El nodo "Ocean Spectrum" se encarga de manejar todos los parámetros que conforman el océano, en cambio el "Ocean Evaluate" deforma la geometría del primer input en base a los parámetros del "Ocean Spectrum".

Como primer paso se ha creado una grid con las dimensiones del contenedor de las partículas, ya que el espectro del océano debe abarcar el mismo tamaño. Además, como la altura de las partículas puede variar según el parámetro Water Level, se ha utilizado un nodo "Transform" para desplazarlo en Y ese valor.

Acto seguido se ha conectado la geometría al primer input del nodo "Ocean Evaluate" y como segundo input el "Ocean Spectrum". Para lograr que el grid tenga profundidad, el nodo "Extrude Volume" ha resultado realmente útil porque cumple con lo que se necesita.

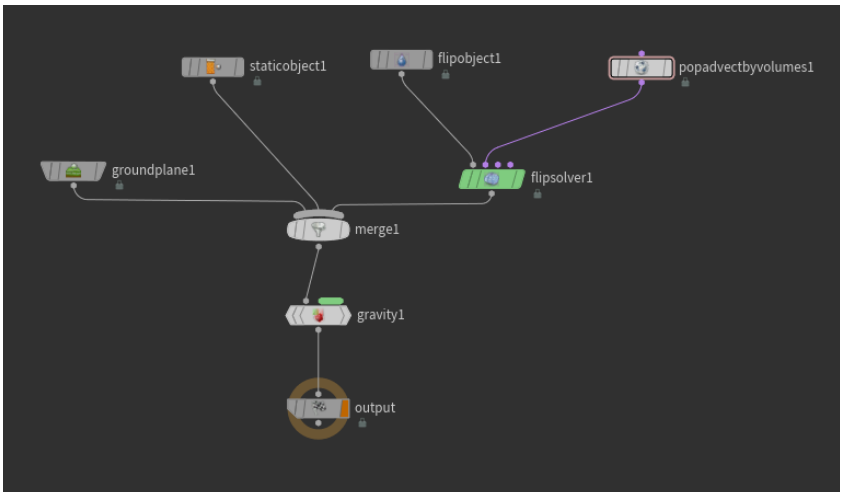
Finalmente, para extraer la velocidad de la geometría se ha hecho con el nodo "Trail", ya que con la opción Compute Velocity se calcula el atributo @v de los puntos. No obstante, para poder añadir esta velocidad a las partículas se debe hacer utilizando VDB, por lo que con el nodo "VDB from Polygons" se convierte el atributo @v en vel, que es como se llama el field de velocidad cuando se trata de volúmenes.



Img 5-21: Estructura de nodos para la creación del océano.

Una vez generado el volumen que moverá a las partículas en la simulación, se puede volver a la DOP Network para aplicar la velocidad generada.

El nodo que se encarga de hacer todo esto es "POP Advect by Volumes", que toma como ruta el volumen vel que se ha creado anteriormente y esto actualiza la velocidad de las partículas, ya que se ha conectado al segundo input del Solver. El tipo de advección que se ha utilizado ha sido Update Force, porque en vez de cambiar la velocidad ajusta la aceleración de las partículas, lo que resulta en un movimiento más natural. En cambio, para el método de advección, se ha usado Trace, puesto que permite cálculos más precisos aunque eso conlleva más tiempo de cómputo.



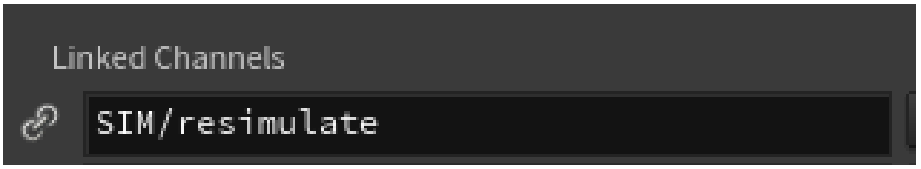
Img 5-22: Estructura de nodos de la simulación.

Como se puede observar en la imagen, se ha añadido el nodo "Ground Plane" para evitar que algunas partículas se cuelen por debajo de los colliders. Este nodo genera un plano infinito en XZ a nivel del suelo, es decir, Y=0.

Todo esto debe ser configurado para que se pueda usar a través de la interfaz del HDA, por lo que primero de todo ha creado la carpeta Sim que controla todo lo relacionado con la simulación:

- **Reset Simulation** – Button – Controla el botón que resetea la simulación.
- **Start Frame** – Int– Controla el frame de inicio de la simulación.
- **Global Substeps** – Int – Controla la cantidad de substeps de la DOP.
- **Min Substeps** – Int – Controla la cantidad mínima de substeps del Solver.
- **Max Substeps** – Int – Controla la cantidad máxima de substeps del Solver.
- **Cache Memory** – Int – Controla la cantidad máxima de Caché de la simulación que se almacena en la Ram.

Primero de todo, para poder configurar el parámetro Reset Simulation, hay que tener en cuenta que no se puede copiar y pegar la ruta directamente sobre el botón que corresponde, por lo que se ha escrito manualmente el parámetro del botón, que en este caso es resimulate, en los Channels vinculados del parámetro Reset Simulation, tal y como se muestra en la imagen de abajo.



Img 5-23: Ejemplo vinculación de parámetros manualmente.

Todos los parámetros restantes se han vinculado como hasta ahora, copiando la referencia. Reset Simulation, Start Frame, Global Substeps y Cache Memory se encuentran en la DOP Network de la simulación, ya que son parámetros que controlan la simulación en general. En cambio, Min Substeps y Max Substeps se encuentran en el Solver de la simulación.

Para los parámetros que controlan aspectos más específicos de la simulación se han creado dos subcarpetas: Solver y Waves. Los parámetros que se encuentran dentro de la subcarpeta Solver son los siguientes:

- **Grid Scale** – Float – Controla la resolución de la simulación de volúmenes.

- **Velocity Transfer** – Ordered Menu – Controla qué método se utiliza para transferir la velocidad a las partículas.

- **Add Vorticity Attribute** – Toggle – Controla si el atributo Vorticity está activo o no.

- **Add Rest Attribute** – Toggle – Controla si el atributo Rest está activo o no.

- **Reseed Particles** – Toggle – Controla si se utilizará la técnica de Reseed para evitar pérdida de volumen en zonas donde hay pocas partículas.

Y los de la subcarpeta Waves:

- **Direction** – Float – Controla la dirección del viento.

- **Speed** – Float - Controla la velocidad del viento.

- **Directional bias** – Float – Controla en qué medida las olas siguen la dirección del viento.

- **Advection Scale** – Float – Controla la fuerza de la advección de las partículas.

Los parámetros de la carpeta Solver se encuentran en el nodo "Flip Solver" de la simulación y los de la carpeta "Waves" en el espectro del océano.

Recapitulando, la herramienta permite al usuario controlar todo lo relacionado con el contenedor de las partículas, el Collider, el Wall, la Ramp y las Rocks. Además, ya puede visualizar la simulación y ajustar los parámetros que controlan el movimiento de las olas así como otros adicionales del Solver.

Sin embargo, no puede guardar la información de la simulación en disco para poder acceder a ella directamente sin tener que simular de nuevo, por lo que lo siguiente que se va a hacer es configurar todo lo necesario para que el usuario pueda guardar la simulación y el Collider en disco.

Todo esto se ha configurado para la carpeta Cache, en la que se han creado dos subcarpetas en modo tab, una para el caché de las colisiones y otra para el de la simulación:

- **Voxel Size** – Float – Controla la resolución del VDB que se usará para calcular las colisiones.

- **Save to Disk** – Button – Controla el botón que permite guardar en disco.

- **Base Name** – String – Controla el nombre de base que tendrán los archivos que se guarden en disco.

- **Version** – Integer – Controla el número que se añade a la string del Base Name para crear varias versiones.

- **Load from Disk** – Toggle – Controla si se carga la colisión del disco o no.

Guardar las colisiones antes de entrar en la simulación puede ser realmente útil, ya que, al cargar la geometría desde el disco, el ordenador no debe calcular el VDB generado para las colisiones mientras simula. Es por ello que una vez generado el VDB con el nodo "Collision Source", se han conectado dos "ROP Geometry Output", uno para la geometría y otro para el VDB, que es el nodo que se encarga de guardar cualquier tipo de dato dentro de Houdini en disco. Para generar la ruta en la que se guardarán, se ha utilizado una técnica que consiste en concatenar diferentes strings y rutas. Se ha tomado como ejemplo la ruta de la geometría:

$HIP/cache/Collisions/`chs("../Collision_Cache_Name")`_v0`chs("../Collision_Cache_Version")`/`chs("../Collision_Cache_Name")`_GEO_v0`chs("../Collision_Cache_Version")`.bgeo.sc

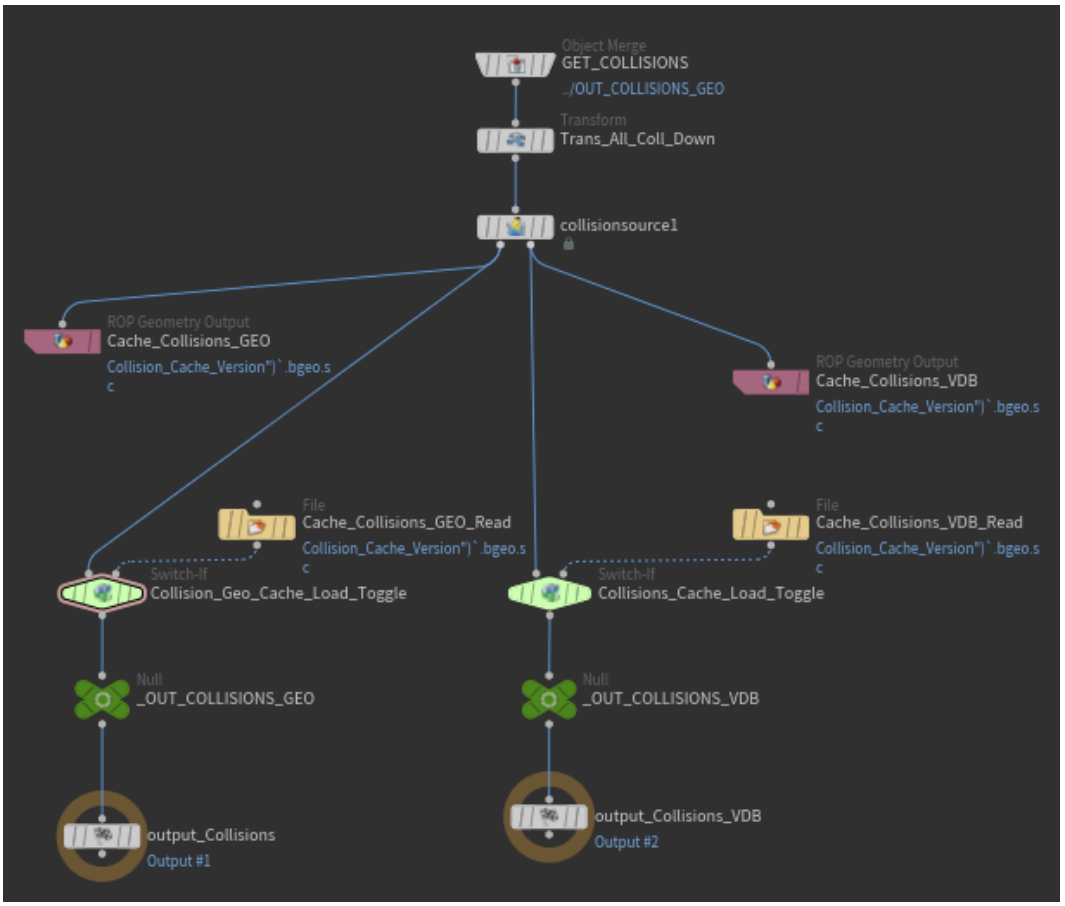
Primero de todo, **$HIP** es una variable global dentro de Houdini que permite acceder a la ruta en la que se ha guardado el proyecto, **/cache** se utiliza para acceder a la carpeta "cache" de la ruta **$HIP,** y lo mismo con **/Collisions.** Cabe destacar que si una carpeta no existe Houdini la creará automáticamente, por lo que la carpeta "Collisions" se generará una vez se guarde geometría en disco. En cambio, la carpeta "cache" no, porque es una de las que vienen por defecto cuando se crea un nuevo proyecto de Houdini.

Para poder generar otra carpeta que guarda diferentes colisiones con diferentes nombres y versiones, se han utilizado los parámetros string que se han añadido al HDA previamente. El primero de todos es Base Name por lo que se pegará la referencia directamente después de añadir "/", ya que se debe cerrar la carpeta anterior que es "Collisions". Fuera de ese string, se ha escrito **_v0** para después añadir la string que contiene el valor de la versión y finalmente añadir de nuevo "/".

Sin embargo, por ahora solo se han creado las carpetas que mantienen todo organizado, pero no se guardaría ninguna geometría. Es por ello que se han vuelto a copiar las dos referencias de las strings pero con dos modificaciones, la primera es que, antes de añadir **_v0** se ha puesto qué es lo que se está guardando el disco, en este caso la geometría. La segunda es que ya no se ha puesto "/" al final de la string, porque ya no se crean carpetas sino que se ha añadido **.bgeo.sc**, el formato nativo de Houdini.

Finalmente, para poder cambiar entre cargar geometría del disco y visualizar la que se genera en tiempo real, se ha utilizado la misma técnica que se ha ido usando hasta ahora cuando se trata de un toggle, los "Switch-If". En el primer input se conecta la geometría que no está guardada en disco, es decir, la que sale directamente del nodo "Collision Source", y en el segundo input la que se carga desde el disco con el nodo "File", que tiene la misma ruta que "ROP Geometry Output" para poder acceder a los archivos.

Este proceso se ha hecho tanto para la geometría como para el VDB y el parámetro Save to Disk del HDA controla ambos botones para guardarlos en el disco, ya que siempre se deben guardar los dos a la vez.



Img 5-24: Estructura de nodos para el caché de las colisiones.

Para el caché de las simulaciones se ha utilizado la misma técnica pero con variaciones, ya que hay que añadir otros procesos:

- **Start/End/Inc** – Vector3 – Controla el Frame de inicio, el final y el incremento.

- **Save to Disk** – Button – Controla el botón que permite guardar en disco.

- **Base Name** – String – Controla el nombre de base que tendrán los archivos que se guarden en disco.

- **Version** – Integer – Controla el número que se añade a la string del Base Name para crear varias versiones.

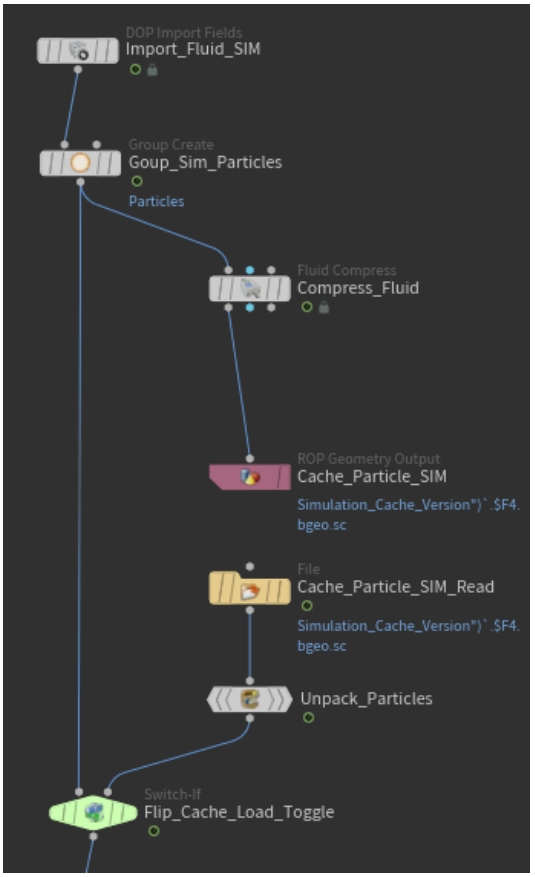
- **Load from Disk** – Toggle – Controla si se carga una simulación desde el disco o se visualizará una en tiempo real.

En primer lugar, como ya se ha explicado anteriormente, las simulaciones se llevan a cabo en otro contexto, DOP, por lo que se han extraído los datos de las partículas con el nodo "DOP Import Fields", ya que tan solo es necesario obtener los puntos. Seguidamente, se han agrupado y se han comprimido con el nodo "Fluid Compress", puesto que este tipo de simulaciones ocupan mucho espacio en memoria. Lo único que hay que tener en cuenta para utilizar este nodo es el parámetro Particle Separation, así que se ha referenciado el que se utiliza en el HDA.

Una vez las partículas están listas para ser guardadas en disco, se ha utilizado de nuevo el nodo "ROP Geometry Output", en el que se ha utilizado la misma técnica de concatenación de strings y rutas haciendo uso de los parámetros Base Name y Version.

No obstante, si se cargan los datos de la simulación con el nodo "File" se ven los puntos empaquetados, es por eso que se ha usado el nodo "Unpack", con el que obtienen de nuevo todos los puntos y atributos.

Por último, de nuevo, para cambiar entre cargar la simulación guardada en disco y la que se puede ver en tiempo real se ha utilizado el nodo "Switch-If" controlado por el parámetro toggle Load From Disk.



Img 5-25: Estructura de nodos para el caché de la simulación.

Como último paso para terminar la herramienta se ha creado la carpeta Visualization, en la que el usuario puede manejar diferentes parámetros y atributos para tener mayor control sobre el HDA:

- **Show Bounds** – Toggle – Controla si se muestran los Bounds del contenedor o no.

- **Visualize Ocean** – Toggle – Controla si se muestra el océano que controla las olas en la simulación.

- **Attribute** – Ordered Menu – Controla qué atributo se utiliza para darle color a las partículas.

- **Range** – Vector2 – Controla el rango de valores que tiene la rampa de color.

- **Visualization Ramp** – Ramp RGB – Controla los colores que se muestran en las partículas que depende de los parámetros Attribute y Range.

Tanto Show Bounds como Visualize Ocean se han configurado como todos los toggle de la herramienta, haciendo uso de "Switch-If" y "Null". Por una parte, para configurar Show Bounds se han necesitado crear dos diferentes, ya que cuando se guarda en disco una simulación también se deben guardar los Bounds porque sino se mostrarían siempre los que hay en tiempo real.

Por otra parte, para configurar Visualize Ocean se ha cogido el espectro del océano que se utiliza para mover las partículas dentro de la simulación y se visualiza a través de una grid. Lo que permite al usuario poder observar en tiempo real los parámetros que controlan el espectro.

Para crear la rampa que permite visualizar los atributos con colores, primero se han configurado los parámetros Attribute y Range y para ello hay que extraer la información de la velocidad de las partículas.

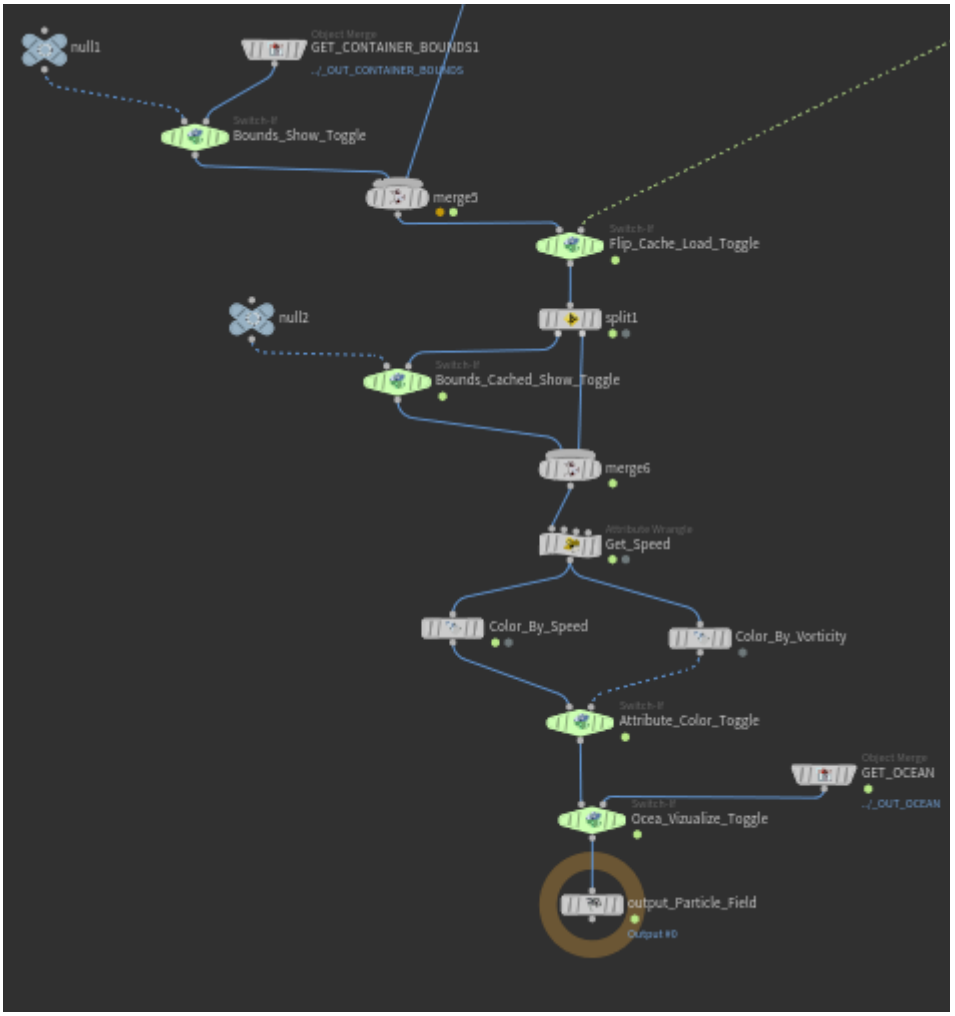
Como ya se ha explicado anteriormente, velocity es un vector, pero para poder visualizar la velocidad se necesita la magnitud del vector, es decir, un float. Para eso se ha utilizado un código en Vex:

```
1 f@speed = length(v@v);
```

Img 5-26: Ejemplo extraer magnitud de un vector en Vex.

Con este código se le asigna a un nuevo atributo llamado @speed la magnitud del vector @v utilizando la función length(). Acto seguido se han conectado dos nodos "Color", uno que es controlado por el atributo @speed y otro por el atributo @vorticity. En ambos nodos se ha referenciado el parámetro Range, que permite manipular el rango de velocidad de las partículas para aplicar el color. También se ha referenciado la rampa de color en ambos.

Finalmente, el toggle que controla qué atributo se utiliza para aplicar el color, se ha configurado de nuevo con un "Switch-If", y toma como primer input el nodo "Color" con el atributo @speed y el segundo input el nodo "Color" con el atributo @vorticity.



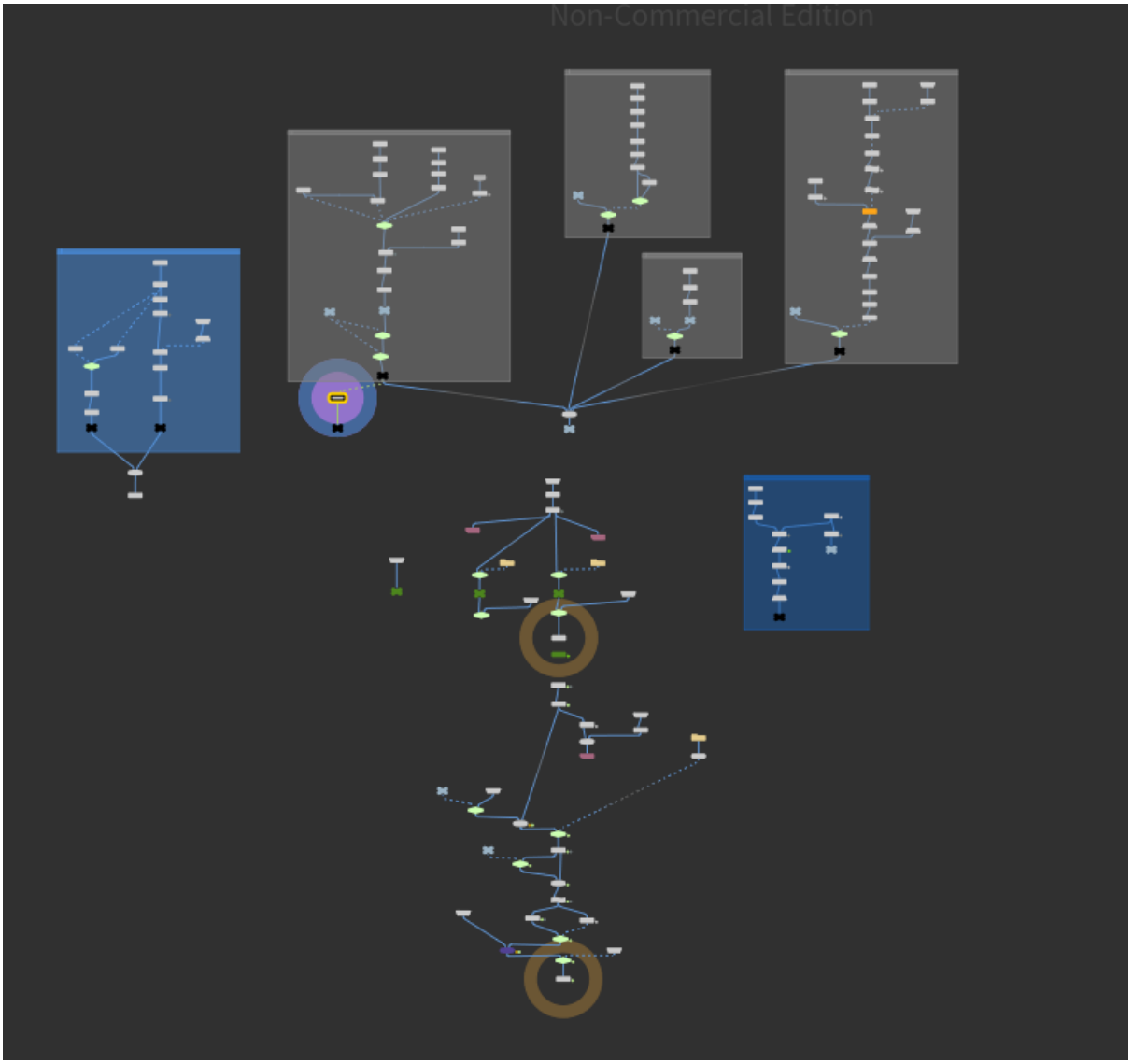
Img 5-27: Estructura de nodos para los parámetros de visualización.

Para finalizar esta Fase se ha mejorado la interfaz del HDA, ya que aunque alguna de las funcionalidades no estén activas todavía se pueden manipular sus parámetros. Se ha tomado como ejemplo los parámetros que controla el Collider Arc:

Img 5-28: Ejemplo deshabilitar parámetros.

Como se puede ver en la imagen, se han puesto una serie de condicionales para que el parámetro Arc Width se deshabilite. El primero de todo es que siempre que el Collider Type, que es el ordered menu donde se seleccionan los colliders, no sea Arc este parámetro se va a deshabilitar. También se debe deshabilitar cuando el toggle del Collider o el toggle del Wall estén desactivados.

Esto se ha hecho con todos los parámetros que dependen de un toggle que los activa.



Img 5-29: Estructura de nodos HDA.

Esta es la red de nodos que se ha creado por ahora pero está sujeta a cambios, ya que en la Fase 3 se buscará mejorar el rendimiento de la herramienta y posibles errores.

5.3 Fase 3

Como ya se ha explicado en los apartados anteriores, esta fase se encarga de encontrar posibles errores de funcionamiento y probar la herramienta con usuarios. Es por ello que este apartado está directamente vinculado con los siguientes apartados, ya que las pruebas que se realizarán con los usuarios servirán también para direccionar la validación del proyecto y ayudarán a sacar las conclusiones del proyecto.

Se ha decidido que antes de intentar buscar posibles errores, se va a realizar una primera prueba en usuarios y, si es necesario, después de tomar las decisiones en base a los resultados obtenidos realizar una segunda prueba.

Para obtener los datos necesarios de los usuarios antes y después de la prueba se ha creado un formulario para poder guardar la información de qué perfil de usuario es y qué feedback va a dar, ver en anexo.

Al momento de probar la herramienta se le pidió a los usuarios que compartieran su pantalla, de esta manera si surge algún problema o hay algo que falla pero los usuarios no se dan cuenta, todo queda registrado para poder realizar un análisis posteriormente. Por ejemplo, uno de los usuarios a los que se le realizó la prueba usaba una versión distinta, por lo que el HDA no funcionaba correctamente por un fallo en las librerías de uno de los nodos utilizados. Es por ello que a la hora de extraer las partículas de la DOP network ya no se usa el nodo "Dop Import Fields" sino que se hace desde el propio nodo de DOP. De esta manera se puede utilizar en cualquier versión del programa.

La primera pregunta del formulario es si los usuarios piensan si es necesario un manual para poder utilizar la herramienta, a la que el 80% respondió que sí. Por eso mismo, se ha decidido hacer un PDF en el que se explica para qué sirve la herramienta, cómo se instala el HDA y qué cosas se pueden llegar a hacer. Este manual se podrá descargar junto al HDA, ver en anexo.

La siguiente pregunta es acerca de la navegación e interfaz, ya que es una parte fundamental para que la herramienta funcione de una manera coherente. Todos los usuarios respondieron que la experiencia en cuanto a esto fue satisfactoria, ya que al haber sido organizada con diferentes carpetas y apartados es muy fácil identificar las funcionalidades. Es por ello que la estructura del HDA se va a mantener como está.

En cuanto a qué parámetros se podrían añadir o modificar, prácticamente no se presentaron propuestas pero sí que varias personas coincidieron en que los colliders podrían tener más parámetros para conseguir diferentes formas con más facilidad. Sin embargo, desde un principio se pensó en que los 3 colliders que vienen por defecto fueran lo más simple posible y que los usuarios más experimentados o que simplemente quieran utilizar geometrías más complejas lo hicieran a través de la opción de collider custom. Por esto mismo, ninguno de los parámetros ya existentes se van a modificar ni se van a añadir nuevos.

Otra de las preguntas es qué funcionalidad añadirían a la herramienta y en esta surgieron dos ideas bastante interesantes. La primera es la posibilidad de renderizar la escena, es decir, crear una malla a partir de las partículas, añadir materiales y poder renderizar. No obstante, por lo que abarca el proyecto esto se sale de los límites que se pusieron a la hora de qué funcionalidades se crearían, ya que, no tan solo es el tiempo que se necesita para implementar todo esto sino que también no es algo necesario para el objetivo principal que tiene la herramienta.

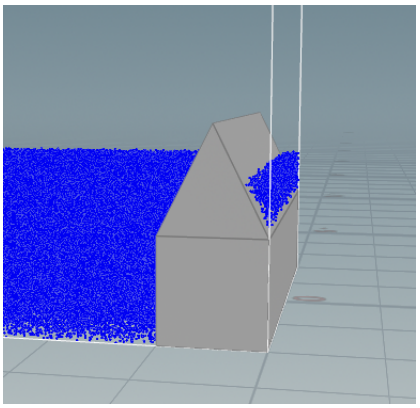
En cuanto a la otra propuesta, se trata de poder incluir más objetos en medio del contenedor de las partículas, esto sí que se ha decidido incluir porque no es algo que requiera mucho tiempo para implementarlo y además puede ser realmente útil el hecho de poder añadir geometría extra. De esta manera el usuario puede experimentar diferentes comportamientos del agua con más objetos.

Para poder añadir esta opción primero se ha creado un nuevo input para que los usuarios puedan conectar la geometría que quieran. Además, como el usuario es libre de poner cualquier geometría y posicionarla como quiera no se ha creado ningún parámetro para manipularla, ya que no es necesario.

Como desde un principio se ha creado todo pensando en posibles modificaciones lo único que se ha tenido que hacer es conectar esta geometría que viene dada por el segundo input del HDA al nodo "merge" que junta todas las colisiones de la simulación. Después en el nodo de caché se ha tenido que cambiar a que se puedan guardar más frames, ya que la geometría puede estar animada. Es por eso que se ha incluido $F4 a la string que guarda la ruta del caché.

A raíz de esta implementación se ha visto que hay un proceso que se puede optimizar, ya que el nodo "object merge" que coge todas las colisiones para extraer los puntos del contenedor que están en el interior de la geometría lo hace una por una, es por eso que se ha cambiado para que lo haga todo una vez.

Respecto a la pregunta de si que si hay alguna funcionalidad que se debería quitar todos los usuarios han concluido que cada una de las funcionalidades tienen una tarea realmente importante dentro del HDA. No obstante, dos usuarios encontraron dos errores que sí pueden entorpecer el uso de la herramienta. El primero de todos es que cuando se utiliza el collider Spike y el nivel del agua está por encima del Wall se generan partículas por detrás del collider lo cual supone un problema porque no deberían de estar ahí.



Img 5-30: Problema con el collider Spike.

Como se puede observar en la imagen las partículas no son eliminadas correctamente, esto se debe a que el objeto que se ha usado para eliminarlas es el propio collider de Spike y obviamente no las elimina. Para solucionar esto se ha creado un cubo y se ha situado de tal manera que la cara delantera quede siempre en el valor del parámetro Spike Direction, por lo que cuando ese valor se modifica la caja se posiciona de manera correcta y los puntos se eliminan como deben. También se le han asignado todos los valores necesarios de escala y transformaciones generales del HDA. Además, esta geometría solo se debe activar cuando se use ese collider, por lo que de nuevo hay que hacer uso de nodos "Switch-If" para desactivar la geometría cuando sea necesario.

El otro problema con el que se han encontrado los usuarios ha sido la falta de optimización en las rocas, ya que la generación de estas es muy costosa y ralentiza la herramienta significativamente. Primero de todo, se ha decidido eliminar el "remesh", que mejoraba los polígonos de la geometría. Pese a bajar la resolución esto no supone un problema porque el motivo por el que se pusieron las rocas fue para poder generar más interés en el movimiento del fluido. También se ha reducido el número de polígonos en la esfera inicial, ya que de esta manera es menos costoso para el ordenador calcular los ruidos y el "Copy to points". Por último, en el nodo "VDB combine", que se encarga de eliminar la geometría de las rocas que se encuentra fuera del contenedor, se ha

cambiado el parámetro Resample a B match A, porque usando este método se logra hacer que el volumen se calcule más rápido.

Con la última pregunta de la encuesta, todos los usuarios estuvieron de acuerdo en añadir una descripción para cada parámetro. Para ello, en la interfaz de los ajustes del HDA cada parámetro tiene un apartado llamado Help en el que se puede escribir información. De tal manera que una vez el usuario pone el cursor en el nombre del parámetro se abre una ventana pequeña en la que se puede leer esa información.

Finalmente, para poder identificar errores y posibles mejoras que los usuarios no han detectado se ha hecho un análisis profundo de todas las funcionalidades de la herramienta, tanto de manera individual como conjuntamente.

Lo primero que se ha encontrado es que cuando se quieren guardar en caché las colisiones de la simulación que no están animadas hay un error porque la ruta se cambió a que siempre se guardaran múltiples frames. Pese a no ser la mejor solución se ha decidido que cuando no haya geometría animada se guardarán todos los frames necesarios igualmente, ya que no se ha encontrado ninguna otra solución.

Otra cosa que se ha decidido mejorar ha sido los Inputs/Outputs del HDA. Los dos inputs que hay se han configurado para que no sea obligatorio tener una geometría conectada siempre. No obstante, pese a tener esa opción puesta sigue saliendo un error cuando no hay ninguna geometría conectada, pero esto se debe a un error del propio programa. La solución se ha encontrado en los foros y es simplemente conectar nodos "Null" en los nodos input dentro del HDA.

Después, para los Outputs finalmente se ha decidido poner 3, el primero para las partículas y las geometrías, el segundo para únicamente las geometrías y el tercero para los VDB creados para las colisiones de la simulación. Además se

ha creado también un grupo para el objeto de colisiones extra, de esta manera los usuarios pueden acceder individualmente a cada una de las funcionalidades.

En último lugar, cuando se aplican transformaciones en el eje Y al contenedor, el container de la simulación no se ajusta correctamente ya que no se tiene en cuenta ese parámetro. Por eso mismo, se ha añadido la ruta del parámetro translate Y del HDA y se ha configurado correctamente para que finalmente toda la herramienta funcione bien.

Con estos ajustes se da por finalizada la tercera fase y también el desarrollo del proyecto.

6. Validación del proyecto

Como método de validación del proyecto, se ha utilizado la fase 3, la cual ha desempeñado un papel crucial en la obtención de feedback por parte de los usuarios y en la realización de pruebas finales exhaustivas en la herramienta. Gracias a esta etapa, podemos considerar que el proyecto se encuentra en su fase final, habiendo alcanzado los objetivos propuestos.

Es satisfactorio destacar que la herramienta en su totalidad funciona correctamente y que se han logrado implementar todas las funcionalidades planificadas. Si bien siempre hay espacio para mejoras y refinamientos, el hecho de que la herramienta cumpla con su propósito principal es un logro significativo. A través de un riguroso proceso de desarrollo y pruebas, hemos asegurado que cada componente y funcionalidad operen de manera fluida y sin problemas, brindando una experiencia satisfactoria a los usuarios.

Además, se han creado una serie de videos en los que se pueden apreciar las diversas aplicaciones del HDA. Estos videos permiten visualizar cómo la herramienta puede ser utilizada en diferentes contextos y escenarios, resaltando su versatilidad y capacidad para generar efectos visuales realistas relacionados con el agua. Estos ejemplos concretos ayudan a ilustrar el potencial y las posibilidades que la herramienta ofrece a los profesionales y artistas del 3D.

En resumen, el proceso de validación en la fase 3 ha sido fundamental para garantizar el cumplimiento de los objetivos del proyecto. La herramienta se encuentra en pleno funcionamiento, habiendo superado las pruebas y recibido el feedback de los usuarios. Los videos demostrativos complementan la validación, mostrando las aplicaciones y el potencial del HDA.

7. Conclusiones

Cuando elegí este trabajo, lo hice porque siempre me ha apasionado el mundo del 3D y, en particular, la creación de efectos visuales, especialmente aquellos relacionados con el agua. Este proyecto ha representado un desafío completamente nuevo para mí, ya que nunca antes había creado un HDA tan complejo y con tantos parámetros. Además, la fase de investigación previa me permitió adentrarme aún más en la forma en que se crean estos efectos y comprender su historia.

Durante el proceso de desarrollo, enfrenté varios desafíos que me ayudaron a aprender y aplicar nuevos conocimientos en simulación de fluidos y modelado 3D. La complejidad del HDA me llevó a explorar diferentes enfoques y técnicas para lograr el efecto deseado, lo cual resultó en un aprendizaje significativo y un crecimiento en mis habilidades técnicas.

Aunque estoy satisfecho con el resultado alcanzado, soy consciente de que siempre hay espacio para la mejora. Por ejemplo, el rendimiento del HDA podría optimizarse aún más para lograr una mayor eficiencia y permitir su uso en escenas más complejas. Además, existen oportunidades para explorar nuevas características y expandir las funcionalidades de la herramienta en futuros trabajos o versiones.

En resumen, el desarrollo de este proyecto me ha permitido sumergirme en el apasionante mundo de los efectos visuales 3D, especialmente en la creación de efectos de agua. A través de una exhaustiva investigación y el enfrentamiento de desafíos técnicos, logré alcanzar los objetivos planteados, aunque reconozco que siempre hay margen para mejorar. Este proyecto ha sido una experiencia enriquecedora que ha fortalecido mis habilidades y conocimientos en el campo de los efectos visuales, y estoy emocionado por seguir explorando nuevas oportunidades en este fascinante campo.

7.1 Líneas de futuro

Si se deseara continuar este proyecto en el futuro, se considerarian varias acciones clave para su mejora y expansión. En primer lugar, se buscaría optimizar el rendimiento del HDA con el objetivo de lograr una visualización fluida y en tiempo real de los fluidos en escenarios de protección costera cada vez más complejos. Esto requeriría investigar y aplicar técnicas de optimización que aprovechen al máximo la capacidad de procesamiento del hardware disponible.

Además, se trabajaría en perfeccionar las simulaciones de fluidos en el HDA, centrándose en mejorar la precisión y realismo de los efectos generados. También se explorarían técnicas específicas para simular fenómenos relacionados con la protección costera, como la erosión y sedimentación, para lograr resultados más fieles a la realidad.

Se dedicaría atención a la incorporación de nuevas funcionalidades y características al HDA, con el fin de brindar una experiencia aún más completa y versátil. Entre las posibilidades se encuentra la integración de opciones de exportación que faciliten la utilización de los resultados en otros software y flujos de trabajo, como Unreal Engine. Asimismo, se consideraría la adición de parámetros necesarios para permitir la renderización de la escena, ya que varios usuarios lo echan en falta.

Estas mejoras y propuestas permitirían avanzar en el desarrollo del proyecto, potenciando su capacidad de visualización y simulación de fluidos para la protección costera. Al buscar la optimización del rendimiento, la mejora de la precisión y la adición de funcionalidades adicionales, se espera que el HDA sea aún más útil y versátil para los usuarios, satisfaciendo sus necesidades y brindándoles herramientas más completas para sus proyectos.

8. Bibliografía

- SideFX. Houdini Documentation [En línea]. [Consulta: 23 Noviembre 2022] Disponible en: https://www.sidefx.com/docs/

- Matt Estella. CgWiki [En línea]. [Consulta: 23 Noviembre, 2022] Disponible en: https://www.tokeru.com/cgwiki/Main_Page

- Computer animation history [En línea]. [Consulta: 29 Noviembre, 2022] Disponible en: https://computeranimationhistory-cgi.jimdofree.com/

- J.J.Monaghan. Smoothed Particle Hydrodynamics [En línea]. 1992. [Consulta: 3 Diciembre , 2022] Disponible en: https://articles.adsabs.harvard.edu/cgi-bin/nph-iarticle_query?bibcode=1992ARA%26A..30..543M&db_key=AST&page_ind=0&plate_select=NO&data_type=GIF&type=SCREEN_GIF&classic=YES

- Jousef Murad. Smoothed Particle Hydrodynamics - Advantages & Limitations [En línea]. 2020. [Consulta: 3 Diciembre, 2022] Disponible en: https://www.youtube.com/watch?v=0KBTqN08Pd4

- CgChannel. Naiad. [En línea]. 2012. [Consulta: 8 Diciembre, 2022] Disponible en:https://www.cgchannel.com/tag/naiad/

- Douglas Enright, Stephen Marschner, Ronald Fedkiw. Animation and Rendering of Complex Water Surfaces [En línea]. 2002. [Consulta: 16 Diciembre, 2022] Disponible en: http://physbam.stanford.edu/~fedkiw/papers/stanford2002-03.pdf

- Veena Parthan. The Era Of Fluid Simulations In Hollywood [En línea]. 2022. [Consulta: 15 Enero, 2023] Disponible en: https://semiengineering.com/the-era-of-fluid-simulations-in-hollywood/

- Mike Seymour. Bifröst: the return of the Naiad team with a bridge to ICE [En línea]. 2013. [Consulta: 15 Enero, 2023] Disponible en: https://www.fxguide.com/fxfeatured/bifrost-the-return-of-the-naiad-team-with-a-bridge-to-ice/

- CineFix - IGN Movies and TV. Top 10 VFX Innovations in the 21st Century! [En línea]. 2015. [Consulta: 21 Enero, 2023] Disponible en: https://www.youtube.com/watch?v=G0cPeEAhzWU

- Stuart Fox. And the Oscar Goes To: Fluid Simulation Algorithms! [En línea]. 2008. [Consulta: 1 Febrero, 2023] Disponible en: https://www.popsci.com/entertainment-gaming/article/2008-02/and-oscar-goes-fluid-simulation-algorithms/

- James VanWagnen. What Is Computational Fluid Dynamics (CFD) - And How Does It Really Work? [En línea]. 2022. [Consulta: 1 Febrero, 2023] Disponible en: https://www.fidelisfea.com/post/what-is-computational-fluid-dynamics-cfd-and-how-does-it-really-work#:~:text=The%20first%20commercial%20CFD%20software,for%20a%20variety%20of%20industries

- Jos Stam. Stable Fluids [En línea]. 1999. [Consulta: 14 Febrero, 2023] Disponible en: https://dl.acm.org/doi/pdf/10.1145/311535.311548

- Robert Bridson, Matthias Muller-Fischer. Fluid Simulation [En línea]. 2007. [Consulta: 2 Marzo, 2023] Disponible en: https://www.cs.ubc.ca/~rbridson/fluidsimulation/fluids_notes.pdf

- Mike Seymour. The science of fluid sims [En línea]. 2011. [Consulta: 6 Marzo, 2023] Disponible en: https://www.fxguide.com/fxfeatured/the-science-of-fluid-sims/?highlight=fluid%20history

- Mike Seymour. Science of fluid sims: Part 2 - Realflow [En línea] 2012. [Consulta: 6 Marzo, 2023] Disponible en: https://www.fxguide.com/fxfeatured/science-of-fluid-sims-pt-2-realflow/?highlight=fluid%20history

- Mike Symour. Wipe out: 'Poseidon' Fluid Simulations [En línea] 2006. [Consulta: 10 Marzo, 2023] Disponible en: https://www.fxguide.com/fxfeatured/wipe_out_poseidon_fluid_simulations/?highlight=fluid%20history

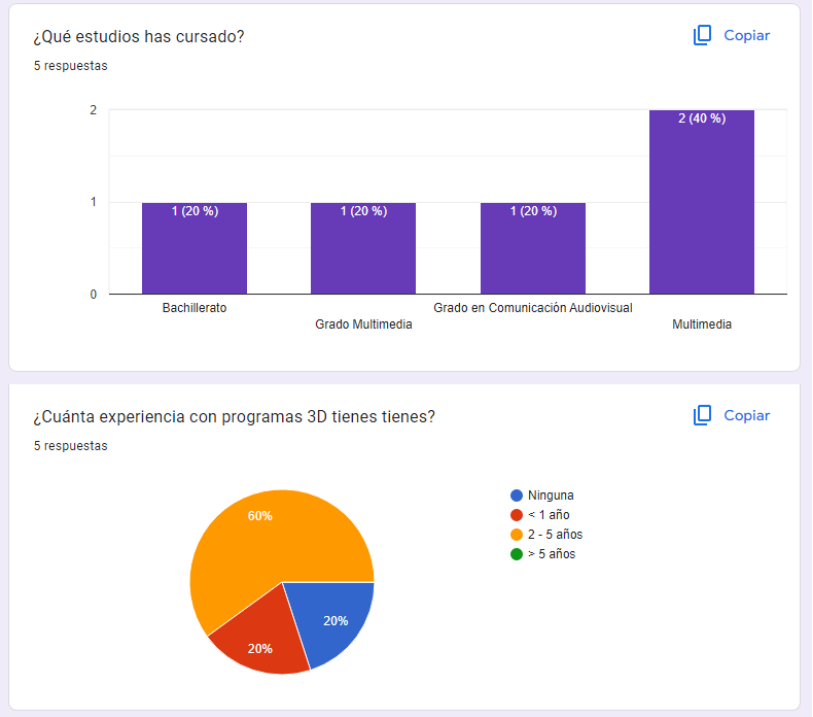
- Mike Seymour. The Tech Behind the Tools of Avatar Part 2: Naiad [En línea]. 2010. [Consulta: 10 Marzo, 2023] Disponible en: https://www.fxguide.com/fxfeatured/the_tech_behind_the_tools_of_avatar_part_2_naiad/?highlight=fluid%20simulation

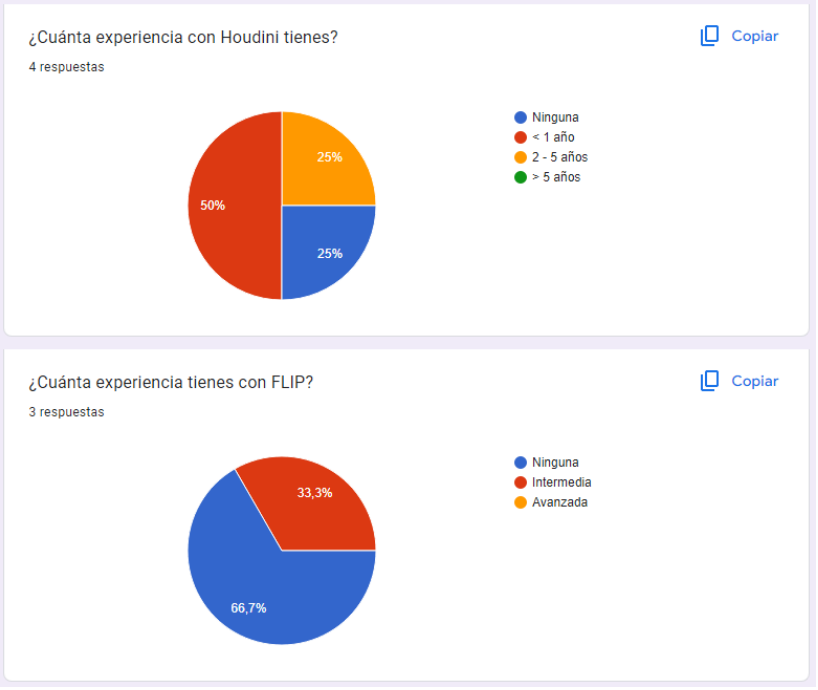
- Mike Seymour. The Simulation of Game of Thrones [En línea]. 2019. [Consulta: 16 Marzo, 2023] Disponible en: https://www.fxguide.com/fxfeatured/the-simulation-of-game-of-thrones/?highlight=fluid%20simulation

- Mike Seymour. Battleship: tactical water and fluid sims (updated) [En línea]. 2012. [Consulta: 16 Marzo, 2023] Disponible en: https://www.fxguide.com/fxfeatured/battleship-tactical-water-and-fluid-sims/?highlight=fluid%20simulation

9. Anexos

9.1 Formulario

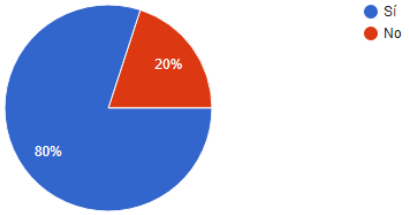




Segunda Parte

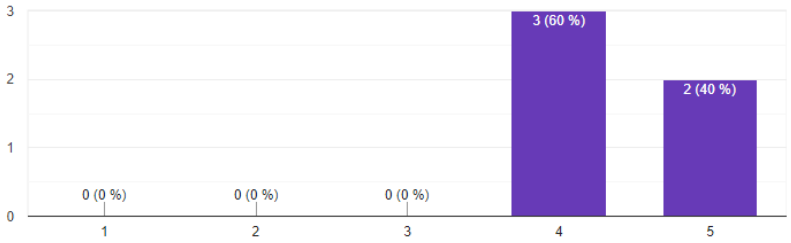
¿Crees que es necesario un manual para usar la herramienta? Copiar

5 respuestas



¿Qué te ha parecido la navegación e interfaz de la herramienta? Copiar

5 respuestas



¿Qué parámetros añadirías o modificarías?

5 respuestas

- Poder modificar los coliders con más parámetros
- Ninguno.
- No.
- Mas variedad en los parámetros de los colliders
- Nada

¿Qué funcionalidades añadirías a la herramienta?

5 respuestas

- Poder renderizar el proyecto
- Probar a añadir más variedad de objetos 3D para experimentar las diferentes reacciones en el agua.
- Poner vegetación y vida animal.
- Poder renderizar la escena, ya que para temas de visualización puede ir muy bien
- Poder poner más elementos dentro del agua

¿Hay alguna funcionalidad que quitarías?

5 respuestas

No

No, creo que en el caso de esta propuesta es interesante tener una gran variedad de funcionalidades ya que se puede utilizar de muchas maneras diferentes y cada persona la puede llegar a necesitar para ciertas cosas en concreto.

No.

¿Has encontrado algún problema de funcionamiento? Si es que sí ¿Cuál?

5 respuestas

No

Sí, al cambiar el objeto 3D a spike y subir el nivel del agua, hay algunas partículas que se quedan aisladas detrás del objeto 3D, cuando realmente no deberían haber partículas en esa zona en concreto.

No.

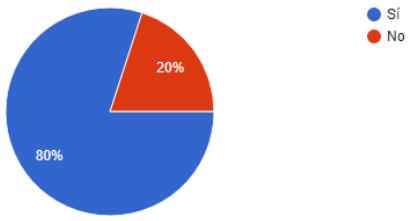
Las rocas van un poco lentas por lo que sería importante intentar optimizarlas un poco

No, ninguno

¿Crees necesaria una pequeña descripción para cada parámetro?

Copiar

5 respuestas



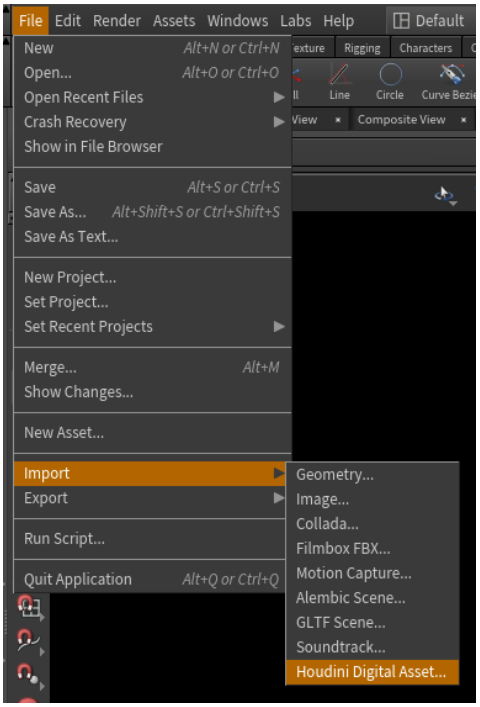
9.2 Manual herramienta

Manual de uso herramienta

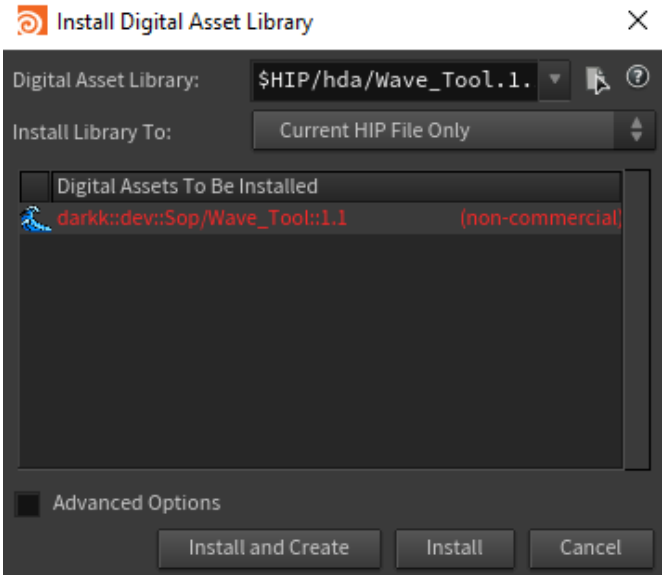
Si es nuevo en el programa se recomienda ver un video introductorio acerca de la navegación e interfaz de Houdini. Los enlaces de abajo son vídeos introductorios al programa.

https://www.youtube.com/watch?v=2CSmc720b_4
https://www.youtube.com/watch?v=l6ifK4kcBWQ

Una vez descargado el archivo del HDA se va a instalar dentro del programa, para ello se debe ir a File → Import → Houdini Digital Asset.



Se abrirá una ventana en la que se debe seleccionar el archivo previamente descargado y pulsar el botón Install and Create.



Una vez creado el nodo se recomienda no tocar el parámetro Particle Separation hasta que todo el Setup haya sido configurado y se vaya a simular. Además se recomienda leer la información de cada parámetro antes de manipularlo si se desconoce cuál es su función.

El HDA se divide en 4 carpetas: Setup, Sim, Cache y Visualization.
En la primera se encuentra todo lo relacionado con la creación de los colliders y la configuración del contenedor de las partículas.

En la carpeta Sim se recomienda tocar solo los parámetros de la subcarpeta Waves, ya que los otros son para gente con más experiencia en el programa.

Para poder guardar la información de la simulación y los colliders se debe usar la carpeta cache, ya que permite guardar y cargar al mismo tiempo archivos desde el disco. Además, se debe crear un proyecto de Houdini para que se pueda cachear todo.

Por último, en la carpeta visualization se recomienda usar el Visualize Ocean cuando se manipulan los parámetros de la subcarpeta wave.