Bournemouth University

Large Scale Simulations with USD in Houdini Solaris

Christopher Hosken

Personal Inquiry
Valery Adzhiev
Bournemouth University
5 May 2025

Introduction

FX simulations are pivotal in modern-day animations and live-action movies, and the type of shots that films require are getting more and more complex. This has led to studios developing unique pipeline systems. However, the ever-increasing popularity of USD is starting to take over and as an aspiring Generalist TD, I felt that it is important to understand USD and how it behaves, specifically in the FX domain. In this inquiry I explored how to create large-scale simulations and combine them in Houdini Solaris with USD. In order to do this I used a dragon animation from Creature Station, orchestrated a shot, and then did FX on it.

For ease of completion, I broke this inquiry into multiple stages.

1. Orchestrating the FX shot. This consisted of taking assets and animations and composing them into a USD stage. Doing this allowed me to gain a basic understanding of how USD works in Houdini Solaris. It also reflected the layout and animation part of the pipeline, which is only briefly touched on in this inquiry.

2. Rigid Bodies and Vellum. I destroyed a part of the shot using Rigid body (and/or vellum) simulations and then brought them into the scene's Solaris stage.

3. Fluids. I created a fluid simulation with whitewater and brought that into Solaris.

4. Pyro Fire and Smoke. I simulated fire and smoke, converted it into VDBs, and put it in Solaris.

5. Final compilation. I combined all these elements (with lighting) into one USD file that was rendered and composited.

Each stage involved a unique round of research, both visually (as the final goal is a visual artefact) and technically (the methods required for creating the artefact). The final artefact of this inquiry was a rendered fx shot from a compiled USD scene containing all the simulations.

Before beginning the project, some of the challenges that I foresaw were around simulation times (especially for fluids) and cache sizes. I was also aware of the amount of iteration that will need to be done for each FX element. To combat this I set up personal deadlines for each element as well as sacrificed simulation quality for quicker (and smaller) simulation caches. This dramatically helped with my project and I was able to complete it on time.

Annotated Bibliography

This annotated bibliography includes a few of the resources that helped me develop an efficient FX pipeline for Houdini Solaris and USD.

1. **SideFX, 2025. *Solaris and Karma* [online]. sidefx.com. Available from: https://www.sidefx.com/docs/houdini/solaris/index.html [Accessed 12 Mar 2025].**

    Houdini Solaris and Karma are built around USD, and understanding how they work will allow for an optimal USD pipeline. The SideFX documentation helped explain a USD pipeline inside of Solaris, including layer breaking, primvars, stages, variants, and all the necessary LOP (Lighting Operator) nodes.

2. **CG Forge, 2019. *Learning Solaris - Part I - CG Forge* [online]. YouTube. Available from: https://www.youtube.com/watch?v=_X9sl5d_ObE&list=PL2SMrYpOIl0Pj13o_OoV6kCnNV35VBA6e [Accessed 5 May 2025].**

    However, the SideFX documentation did not fully explain how to use the LOP nodes, which is where other resources, such as CG Forge's *Learning Solaris* series came in use. I was able to see a practical example of how USD pipelines were structured and what specific nodes were used for. I was able to understand practically how layer breaks, sub layering, references, and exporting works.

Numerous techniques exist for getting RBDs (Rigid Bodies) into Solaris from the SOP context. I compared 3 different techniques to find which was the easiest to set up, fastest to update, and smallest in file cache size.

3. **SideFX, 2025. *Solaris and Karma* [online]. sidefx.com. Available from: https://www.sidefx.com/docs/houdini/solaris/index.html [Accessed 12 Mar 2025].**

    The first technique consisted of reading out the simulated geometry from SOPs into Solaris, either directly using a SOP Import node or through a

USD render ROP node. This was the least effective method as it took the most amount of time to read out, and also resulted in massive file caches.

Although this technique didn't work for RBD's, it was perfect for exporting out vellum simulations. I came to the conclusion that SOP Imports work best for general SOP -> USD conversions. However when it comes to large RBD caches a SOP Import is not the way to go.

4. **Tim van Helsdingen, 2024. *How to use the RBD Procedural in Houdini 20.5* [online]. YouTube. Available from: https://www.youtube.com/watch?v=8Ivsd_zb1Q0 [Accessed 17 Mar 2025].**

The second technique was to use a Houdini RBD Procedural. This proved promising at the start but I found it to be more complex to use, and slower than other methods. I hope to potentially explore the Houdini RBD Procedural properly sometime in the future.

5. **Mats Nyman, 2024. *Houdini & Solaris Tutorial: Dealing With RBDs in USD* [online]. YouTube. Available from: https://www.youtube.com/watch?v=E_jbT3JthSs [Accessed 17 Mar 2025].**

The third technique used a LOP "Transform by Points" node. I found this to be the easiest to set up, fastest to run, and it didn't require any pre-caching to work. The Transform by Points matches a collection of points to stage geometry based on a shared name. This was optimal as it only had to load the geometry once, and then update the position every frame. It could also store velocity and other point-based attributes to help with rendering. I decided to go with this technique for the shot.

6. **SideFX, 2025b. *Houdini Procedural: Ocean* [online]. Sidefx.com. Available from: https://www.sidefx.com/docs/houdini/solaris/houdini_ocean_procedural.html [Accessed 12 Mar 2025].**

Merging fluid simulations with Oceans proved quite challenging. Originally, I was surfacing the fluid and then tyring to displace it in the SOP context, bringing it in with a SOP Import Node. However, this would eat up *huge* amounts of storage and was incredibly slow. However, Houdini provides a tool called the "Houdini Ocean Procedural". This tool reads in an ocean spectrum and a base mesh (alongside a mask), and applies a realistic ocean surface, all at render time. This dramatically reduced how many mesh points needed to be cached as well as providing a much more realistic ocean than what would have been made from scratch. For the whitewater, I simulated millions of particles and saved it as a USD file. I could then import these particles and they would play back at an optimal speed. The particles could then be converted into volumes if needed. The one downside to writing out particles is that configuring their size (pscale) was extremely difficult. I would have to make chances and then re-export, which could take up to an hour.

7. **AJUKking, 2024.** ***Reddit - The heart of the internet*** **[online]. Reddit.com. Available from: https://www.reddit.com/r/Houdini/comments/197phew/loading_animated_vdbs_in_lops [Accessed 11 Apr 2025].**

    Although USD has its own way of describing volumes, VDBs are generally considered to be the standard for storing volume data. Luckily, USD supports VDBs. Generally, creating a VDB is quite simple; just export it from your software of choice. Importing it is where the trouble starts. How should you do it? Do you use an asset reference? Do you use a geometry cache? Do you use a volume with a path to your VDB? All of these decisions affect how your VDB is loaded and saved into your USD file. I found that the easiest way to do it was to use a volume node with a file path to the VDB. This is how the volume node was intended to work. One problem with VDB's is that they can't be "empty" (as in, no density). *"This is a feature of OpenVDB, it does this by default to support on-demand loading of grid voxels. It's pretty useful for processing of big volumes."* (Blender 2020). This means that some VDB simulations result in a "frame hold" on the last visible density, which is not ideal. For example, my fireball would float in the air when it should have been invisible. To solve this, a LOP prune node is used to animate the visibility of the volume.

Conclusion

Overall, I found this project to be satisfying and I feel that I was able to fulfil all the criteria I hoped to meet at the start of this inquiry. I have been able to grow as both an FX Artists and Pipeline TD, and I hope to continue building upon this knowledge and applying to my future projects and beyond! Given more time, I would have loved to explore more in-depth how USD procedurals work and how to "purely" optimize cache files. However, for what I have done with this inquiry I am very satisfied with it.

Being able to iterate on each step of the pipeline was extremely useful, and by the time I had got to the VDB / Pyro iteration, I felt I had a solid grasp on Houdini's LOP nodes and how to use them.

I think the second iteration was my most successful as it involved the most exploration and experimentation to get working. The other iterations seemed like they had (mostly) sound solutions. Going back, I wished I looked at the Houdini RBD Procedural and how it works as that may be a better option in some FX shots. There were some issues with the Houdini Ocean Procedural that I had to fix in compositing, and getting the fluid sim to blend into the rest of the ocean was a tough challenge.

Although this wasn't part of the inquiry, I am glad that I chose to continue working on the project to do lighting & rendering, being able to see the whole production pipeline let me make FX changes in advance to help with rendering.

This assignment has given me the time and space to explore a part of the CG/VFX industry that excites me. I look forward to using what I've learnt in future assignments, and I hope that I (one day) will be able to pass on this information to other aspiring FX and Pipeline artists.

References

AJUKking, 2024. *Reddit - The heart of the internet* [online]. Reddit.com. Available from: https://www.reddit.com/r/Houdini/comments/197phew/loading_animated_vdbs_in_lops [Accessed 11 Apr 2025].

Blender, 2020. *2.9 OpenVDB holds on to an open file after loading VDB frame.* [online]. Blender Projects. Available from: https://projects.blender.org/blender/blender/issues/75901 [Accessed 6 May 2025].

CG Forge, 2019. *Learning Solaris - Part I - CG Forge* [online]. YouTube. Available from: https://www.youtube.com/watch?v=_X9sl5d_ObE&list=PL2SMrYpOIl0Pj13o_OoV6kCnNV35VBA6e [Accessed 5 May 2025].

CG Spectrum, 2021. *Destroy a Pirate Ship featuring KitBash3D Assets* [online]. YouTube. Available from: https://www.youtube.com/watch?v=p9NEr-O74pA [Accessed 17 Mar 2025].

Creature Station, 2024. *SIDEQUESTS* [online]. Creature Station. Available from: https://creaturestation.com/sidequests [Accessed 12 Mar 2025].

Mats Nyman, 2024. *Houdini & Solaris Tutorial: Dealing With RBDs in USD* [online]. YouTube. Available from: https://www.youtube.com/watch?v=E_jbT3JthSs [Accessed 17 Mar 2025].

SideFX, 2025a. *Solaris and Karma* [online]. sidefx.com. Available from: https://www.sidefx.com/docs/houdini/solaris/index.html [Accessed 12 Mar 2025].

SideFX, 2025b. *Houdini Procedural: Ocean* [online]. Sidefx.com. Available from: https://www.sidefx.com/docs/houdini/solaris/houdini_ocean_procedural.html [Accessed 12 Mar 2025].

Tim van Helsdingen, 2024. *How to use the RBD Procedural in Houdini 20.5* [online]. YouTube. Available from: https://www.youtube.com/watch?v=8Ivsd_zb1Q0 [Accessed 17 Mar 2025].

Voxyde VFX, 2023. *Dragon Fire Breath - Houdini VFX Tutorial* [online]. YouTube. Available from: https://www.youtube.com/watch?v=Ogzm2gnkMTU [Accessed 17 Mar 2025].