Everybody's an Effect: Scalable Volumetric Crowds on Elemental

Sasha Ouellet
sasha@pixar.com
Pixar Animation Studios

Aksel Taylan
akselt@pixar.com
Pixar Animation Studios

Arnold Moon
amoon@pixar.com
Pixar Animation Studios

Paul Kanyuk
pkanyuk@pixar.com
Pixar Animation Studios

William Reeves
bill@pixar.com
Pixar Animation Studios

Figure 1: Volumetric crowds from *Elemental* do a "wave" at an air ball game. ©Pixar.

ABSTRACT

Crowd animation and rendering is challenging enough with hard surface models, but the world of *Elemental* takes this to a new level by immersing the viewer in a teeming metropolis populated by sentient air, fire, and water, in the form of volumetric characters. By building a new Houdini-Engine character pipeline based on blended simulation caches and extending our proprietary crowd pipeline to approximate non-skeletal deformation with blendshapes, we were able to choreograph, deform, shade, and light an absurd number of voxels. The complex underlying physical simulation and shading process called *hexport* [Coleman et al. 2020] we used to create the hero look of our main characters took roughly 400 cpu hours per shot, and afforded us the ability to only have about 2.5 characters on screen per shot on average. In the end, each shot on *Elemental* had an average of 162 additional volumetric crowd characters. Thus our challenge was to create those 162 characters with visual fidelity as close as possible to the 2.5 hero characters, despite forgoing hexport. By building a solution as a Houdini Engine [SideFX [n. d.]] procedural, with UsdSkel [Universal Scene Description [n. d.]] deformed meshes as input, we deferred the expensive computations until render time. However, given some shots could have as many as 30,000 volumetric characters, our solution had to execute on the order of several seconds a character to even be feasible, if painful, at scale. Furthermore, IO and storage limits meant the results could not be cached on disk and needed to remain in memory at render time, thus constraining our memory footprint. Accordingly, our pipeline factored as much complexity as possible into pre-process stages, and leaned heavily on level of detail, both for inputs to the render time procedural, and in minimizing the resulting voxels.

KEYWORDS

keywords.

ACM Reference Format:
Sasha Ouellet, Aksel Taylan, Arnold Moon, Paul Kanyuk, and William Reeves. 2023. Everybody's an Effect: Scalable Volumetric Crowds on Elemental. In *Proceedings of SIGGRAPH Talks.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

SIGGRAPH Talks, 2023
2023. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Submission ID: ID. 2023-02-13 21:42. Page 1 of 1–2.

1 BACKGROUND

Our approach to this challenge built on the solid foundations created for the film *Soul*, which also featured large numbers of volumetric characters. On that project, the *hexport* [Coleman et al. 2020] process was developed as a formalized system for voxelizing and processing volumetric characters. The original formulation of hexport, however, depended on each frame's processing being completely independent, and maximally parallelizable. That was leveraged to create an even further optimized volume rasterization system that could occur as a render time procedural [Ouellet et al. 2020]. By contrast, on *Elemental*, physical simulations were required as a first stage of hexport, and thus required long serial computations that dramatically increased turnaround time and complexity. Furthermore, the number of volumetric characters and shots with these characters meant the lattice based "reves" rasterization used in [Ouellet et al. 2020] was too slow to be used in practice, and thus a new system was needed.

2 HOUDINI ENGINE

Rather than revisit this challenge with another handwritten render time procedural, our team decided to adopt Houdini Engine as a framework for developing a scalable character representation. Houdini Engine allowed the logic of the render time procedural to be expressed as a node graph and leverage existing geometry and volume processing techniques rather than relying entirely on C++. The obvious benefit was ease of development and maintenance, but the downside was incurring overhead costs for creating so many Houdini Engine instances at render time. As an early test, we tried creating a simple Houdini Engine network to apply volume deformation to a single 200 frame simulation of the main character's fire simulation for 5000 characters. All 5000 characters deformed in under 2 hours, which indicated the Houdini Engine overhead was minimal enough to build a volume deformation pipeline with the framework.

3 CROWDS AND RIG APPROXIMATION

While Houdini Engine provided a promising framework for render time volume deformation, this still left the challenge of choreographing and deforming the underlying geometry of the characters at the required quality. For skeletal deformation, we use a non-negative least squared optimizer to find the best fit linear blend skinning weights to approximate our hero rigs [Kanyuk et al. 2018], which was "good enough" for most crowds work. Our air, fire, and particularly water characters, however, had very non-skeletal motions that were poorly learned by this system. Our rigging artists were using the recently developed *Profile Mover* [De Goes et al. 2022] technology to give animators the ability to keyframe points across the character surfaces to smoothly deform their surfaces in all sorts of water/fiery/gaseous ways. Attempts had been made to learn to derive corrective shapes whose weights could be driven by a neural network [Radzihovsky et al. [n. d.]], but it turns out a far simpler heuristic proved almost as accurate, and was far easier to deploy. Even though our character rigs are largely built without blendshapes and have highly non-linear controls, a simple linear approximation where a blendshape is generated for the minimum and maximum of each animation control was surprisingly effective. Further improvements were made by analyzing the set of all available crowd animation and instead generating blendshapes for the minimum and maximum controls used in actual practice. We also added the ability to have in between blendshapes, but the improvements were minor. While not perfect, this amazingly straightforward heuristic provided fast and expressive deformations that were invertible back to animation controls for touch ups. By encoding these crowd rigs with UsdSkel joint animation and blendshapes, the mesh deformation scalable challenges were addressed.

4 FIRE AND AIR DEFORMATION

The main input to the Houdini Engine procedural (the "HDA") is the rest pose volumetric data that is calculated offline after each build of the character. The offline process is a full hexport of an "a-pose" shot of each character variant, followed by extra post processing to convert volumes from Field3D to VDB, and create a MIP-pyramid of voxel resolutions up to 64x coarser than the original. The chosen MIP level is determined by the LOD of the crowd character, which is fixed per shot. Additionally, there is a global voxel size scale that is calculated via the screen space size of the character, which is used as a final downsample on the output deformed volumes [Cook et al. 2007]. The scalar is continuous and can change over time. Since there might not necessarily be a direct correspondence between rest pose shot volume data on a particular frame, and an arbitrary frame in a shot, the HDA is responsible for remapping the shot frame to an existing frame in the rest pose data as well as blending between the tail and head of the frame range to produce believably loop-able rest pose volumes. The core deformation algorithm is a re-rasterization from points after the points have been bound and warped against the input geometric surfaces. There are controls for each input volume to determine the number of point samples per voxel and target bind surface(s).

5 WATER DEFORMATION

Water characters differ from fire and air in that they are a mixture of surfaces and volumes. Point deformation is used for the surface data: the body, the hair simulation, and the bubbles, while the aforementioned volume deformation algorithm is used for one of two volumetric components, the hair foam. The other component, the interior volume, is generated at render-time in the Houdini Engine procedural rather than deformed from the rest pose, as there were no significant look differences from this cheaper approach. Several optimizations are made at lower LODs, such as applying procedural noise - modulated based on distance from a given point to the hair seam - instead of deforming the cached rest-pose simulation. Other optimizations include lowering the resolution of the interior volume, and culling fine features such as bubbles and hair foam.

6 STATS AND CONCLUSIONS

The performance stats bear out the success of this approach. Hero characters running *hexport* took between 5 and 8 minutes a frame to compute their voxels. By contrast our HoudiniEngine procedurals, the times ranged from 4 to 50 seconds depending on the element type and LOD. This allowed us to render our average of 162 crowd characters a shot and helped bring Element City to life.

REFERENCES

Patrick Coleman, Laura Murphy, Markus Kranzler, and Max Gilbert. 2020. Making Souls: Methods and a Pipeline for Volumetric Characters. In *ACM SIGGRAPH 2020 Talks* (Virtual Event, USA) *(SIGGRAPH '20)*. Association for Computing Machinery, New York, NY, USA, Article 28, 2 pages. https://doi.org/10.1145/3388767.3407361

Robert L. Cook, John Halstead, Maxwell Planck, and David Ryu. 2007. Stochastic Simplification of Aggregate Detail. *ACM Trans. Graph.* 26, 3 (jul 2007), 79âĂŞes. https://doi.org/10.1145/1276377.1276476

Fernando De Goes, William Sheffler, and Kurt Fleischer. 2022. Character Articulation through Profile Curves. *ACM Trans. Graph.* 41, 4, Article 139 (jul 2022), 14 pages. https://doi.org/10.1145/3528223.3530060

Paul Kanyuk, Patrick Coleman, and Jonah Laird. 2018. Mobilizing Mocap, Motion Blending, and Mayhem: Rig Interoperability for Crowd Simulation on Incredibles 2. In *ACM SIGGRAPH 2018 Talks* (Vancouver, British Columbia, Canada) *(SIGGRAPH '18)*. Association for Computing Machinery, New York, NY, USA, Article 51, 2 pages. https://doi.org/10.1145/3214745.3214803

Sasha Ouellet, Daniel Garcia, Stephen Gustafson, Matt Kuruc, Michael Lorenzen, George Nguyen, and Grace Gilbert. 2020. Rasterizing Volumes and Surfaces for Crowds on Soul. In *ACM SIGGRAPH 2020 Talks* (Virtual Event, USA) *(SIGGRAPH '20)*. Association for Computing Machinery, New York, NY, USA, Article 30, 1 pages. https://doi.org/10.1145/3388767.3407374

Sarah Radzihovsky, Fernando De Goes, and Mark Meyer. [n. d.].

SideFX. [n. d.].

title = Project Website: https://openusd.org Universal Scene Description. [n. d.].