Stylizing Volumes with Neural Networks

Mike Navarro
Walt Disney Animation Studios
Burbank, USA

Jacob Rice
Walt Disney Animation Studios
Burbank, USA

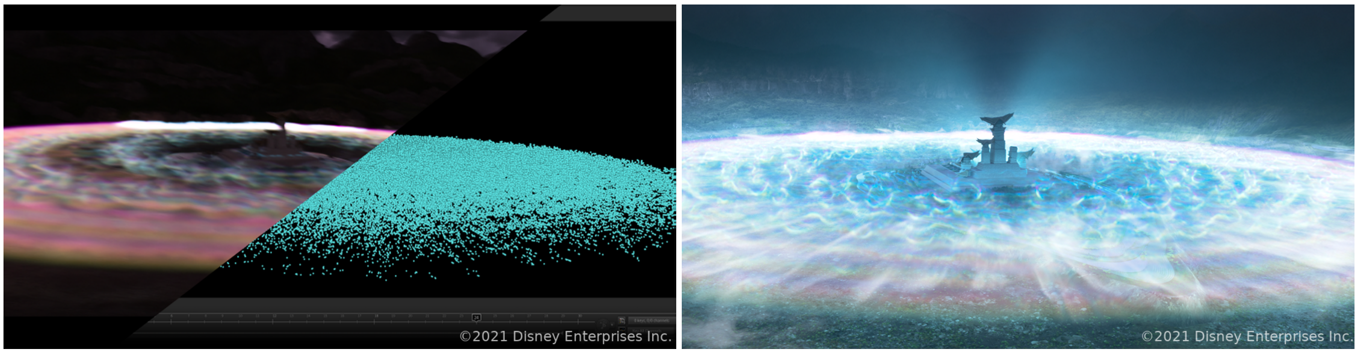

Figure 1: Example images: Source Volume and Final Render after Neural Style Transfer.

ABSTRACT

In "Raya and The Last Dragon", a blast of energy rings across the desiccated lands of the ancient world of Kumandra. This climactic story point represents a powerful force of magic and transformation, and as such, is art directed to be composed of stylized wave patterns and harmonic textures, as if created by sound vibrations. To achieve this, we artistically stylize a simulated volume using Neural Style Transfer. In this talk, we describe the integration of deep learning-based tools into our effects pipeline to accomplish this.

CCS CONCEPTS

• **Computing methodologies** → **Physical simulation**; **Neural Style Transfer**; **Feature Optimization**.

KEYWORDS

neural network, style transfer, style optimization, deep learning, image net, machine learning, lagrangian, volumes, fluids

ACM Reference Format:
Mike Navarro and Jacob Rice. 2021. Stylizing Volumes with Neural Networks. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Talks (SIGGRAPH '21 Talks), August 09-13, 2021.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3450623.3464652

1 INTRODUCTION

The ability to simulate fluid-dynamics has advanced to produce stunning realism in recent years. However, the process of creating stylized shapes and patterns within simulated volumes remains tedious, and is limited in scope and flexibility. We instead incorporate a method for effects artists to choose a 2D style image from which stylistic structures and shapes are automatically transferred to the volume's density field.

In collaboration with Disney Research Studios (DRS), we integrate Lagrangian Neural Style Transfer (LNST) [Kim et al. 2020] into production (Figure 1). Inspired by Neural Style Transfer (NST) methods for images [Gatys et al. 2015], LNST employs a pre-trained Convolution Neural Network (CNN) to transfer styles from a given image to a density field.

In the following sections, we present how we implemented the approach of [Kim et al. 2020] on "Raya and The Last Dragon", and how we further customize our pipeline to meet the needs of this ambitious effect.

2 APPROACH

2.1 General LNST Worflow

2.1.1 Volume Deconstruction: Houdini reads the density and velocity fields from the simulated volume to advect particles that inherit position and density attributes. To ensure sufficient point density, and to faithfully represent the original density field, "kernel size" and "density threshold" parameters are introduced along with vex code to populate additional points into sparse regions of the point cloud. To ensure temporal consistency, per-point "Id" attribute values are added and verified to be unique.

2.1.2 Point Cloud Conversion and Export: The point cloud with 3 per-point attributes is converted to a bundle of 3 numpy arrays and exported.

2.1.3 Inputs to the Stylizer Script: The stylizer has 2 main inputs; The point cloud file and the parameters file. Since the Id values are unique and temporally consistent, velocity can be computed dynamically. The parameters file includes a dictionary of hyperparameters such as: iterations, transmittance, learning rate, kernel size, and the style target image path. Additional file paths and custom parameters are also included for pipeline specific tasks.

SIGGRAPH '21 Talks, August 09-13, 2021, Virtual Event, USA

ACM ISBN 978-1-4503-8373-8/21/08.
https://doi.org/10.1145/3450623.3464652

2.1.4 Stylizer Script Execution: Style transfer runs as a standalone process, in Python 3.6 and Tensorflow 1.15, with additional python modules for particle to grid conversions. Internally, the points are rendered or "splatted" to a flattened image that is passed through the pre-trained neural network to be stored as a hierarchical representation of features. Similarly, the target style image is passed through the network in the same way, so that structural features can be compared and statistically matched. Once features are correlated and optimized for maximum similarity, the source density data is reconstructed with the most structurally similar shapes from the 2D image. The result is exported as numpy arrays.

2.1.5 Volume Reconstruction: To reconstruct the volume, we repeat the previous steps in reverse. Stylized points are loaded and restored to world space and their densities are transferred back to Houdini Volumes using a cubic splatting kernel. By utilizing the "kernel size" and "density threshold" values from both, initial advection as well as the LNST process, we restore the stylized volume to its original quality and resolution. These point-to-volume (and volume-to-point) conversion methods that we apply in Houdini give us the ability to accurately preserve volume detail at a production level. It's even a good practice to pass a simulated volume through this entire conversion process without LNST and compare the reconstructed volume to the original. This serves as a verification of the volume's integrity, as well as provides feedback for data optimization.

2.2 Additional Optimizations

2.2.1 Point Cloud Normalization: We normalize the point cloud from world space to camera space to minimize the domain size. The domain size calculation is simplified by minimizing world space translations and provides measurements to accurately adjust the voxelsize, as well as preserve volume resolution. This normalization helps control the culling of points not visible to the camera, further optimizing the dataset. Another benefit to normalization is the removal of any 3d perspective distortion introduced by the virtual camera. Since LNST is based on a 2D image style transfer method, the minimization of perspective delivers the best results, especially for cameras with short focal lengths. This distortion is easily reintroduced after stylization.

2.2.2 Domain and Grid Calculations: Calculating the domain and grid resolution parameters in Houdini, gives the artists valuable control during tuning and optimization. We use Houdini's native tools to construct the domain from primarily two inputs: the advected point cloud's bounding box and the voxelsize of the original simulated volume. The result from just these inputs gives domain and grid dimensions that are accurate and reliable. And, since we've modified the point cloud with transformations and normalization, the modified inputs reliably give us values that accurately define only the necessary computation space for the style transfer point cloud. Furthermore, if we wish to iterate on a smaller region within the domain (to artistically tune the optimizer's style settings) this method is also accurate and reliable. We use this custom domain calculation method to compute style transfer at higher resolutions with an existing point cloud by decreasing the voxelsize, which, in turn results in a finer grid resolution. We take advantage of this method for tuning the stylized results over many iterations. Also, since the stylization process repeatedly performs point-to-grid conversions, it is occasionally necessary to pad the domain to allow grid densities to expand beyond tight bounding boxes. Having this domain calculation scheme within Houdini, the original LNST code from DRS remains unmodified. This delivers the most flexibility to effects artists, allowing tuning to remain in Houdini, as opposed to external LNST scripts.

2.3 Large-Scale Datasets

2.3.1 Clustered Partitions and Distributed Tasks: Extremely large data sets require partitioning points into clusters for distributed processing in parallel. Point clusters of similar size also improve overall compute efficiency. Additional Houdini and python tools were developed to manage the complexities introduced by partitioned point clusters.

2.3.2 Compress Id Numeric Range: Large gaps in the Id attribute's numeric range results in the need for placeholder array elements to be created during style transfer. Even though these placeholders are given position coordinates outside the domain (and subsequently ignored by the optimization process), the memory, computing and storage demands increase and overall efficiency decreases. The solution for this is to process the points through a particle solver in Houdini, solely for the purpose of establishing unique Id values with as few gaps as possible.

CONCLUSIONS AND FUTURE WORK

The energy blast effect on "Raya and The Last Dragon" challenged us to embrace a new approach for art-directing volumetric effects. By allowing effects artists to start with a target style image from which visual features are transferred automatically to a density field, final results are rich in textures and patterns that would have previously taken countless hours to achieve. And while existing techniques have been through many cycles of development, LNST's potential is barely realised, with many possible paths for future development.

Render Queue Utilization: We developed pipeline and queue tools to manage LNST tasks specifically for the blast effect. The biggest challenges were in managing tasks that loaded multiple frames of data, often with overlapping frame ranges. This workflow was necessary for LNST, but is unlike most pipeline tasks at Disney Animation, so there remains considerable work to extend render queue processing to the full range of LNST options.

Future Work: The visualization of feature maps and other hidden layers within the neural network would provide more artist feedback from inside the network. Other statistics for the fine-tuning of style parameters would strengthen artistic control of the style transfer. Also, more sophisticated cluster and partition management tools for the queue and pipeline would allow greater load balancing and increased efficiency.

REFERENCES

Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. 2015. A Neural Algorithm of Artistic Style. arXiv:1508.06576 [cs.CV]

Byungsoo Kim, Vinicius C. Azevedo, Markus Gross, and Barbara Solenthaler. 2020. Lagrangian Neural Style Transfer for Fluids. *ACM Trans. Graph.* 39, 4, Article 52 (July 2020), 10 pages. https://doi.org/10.1145/3386569.3392473