Weaving The Druun's Webbing

Jacob Rice
Walt Disney Animation Studios
Burbank, USA

Figure 1: The Druun with exposed webbing.

ABSTRACT

The Druun in Walt Disney Animation Studio's "Raya and the Last Dragon" is a unique character, both in design and in implementation. Over the course of this film, unique solutions were designed to overcome the technical challenges of having a creature made of both a fluid and a web like structure. In this talk we will present the techniques used to bring one of the Druun's amazing features to life: the webbing.

CCS CONCEPTS

• **Computing methodologies** → **Physical simulation**.

KEYWORDS

Optimal Transport, Relative Neighborhood Graph, Voronoi Diagrams, Fluid Simulation, Webbing

ACM Reference Format:
Jacob Rice. 2021. Weaving The Druun's Webbing. In *SIGGRAPH '21: Aug 01–05, 2021.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3450623.3464647

1 INTRODUCTION

In the film, the Druun is a constant threat to humanity that is ever changing in form. From creative development, it was evident that we needed to create a character that is menacing while keeping its form abstract and fluid. To represent this visually, the Druun is composed of two elements: a viscous fluid dough, and an internal webbing structure that travels within the core of the dough. To achieve this look, the webbing is created by a series of simulations driven by the dough's fluid simulation.

Since the character appears in a large portion of the film and the dough fluid simulation is complex, it was paramount that the webbing simulation could be completely automated to allow artists to focus on the Druun's acting performance. This talk presents an inside look into how the dough simulation is turned from a fluid into a coherent web-like pattern, spanning the interior of the simulation, without any user input.

2 RELATED WORKS

2.1 Tracking Fluid Motion

Fluid surface tracking has long been used to add detail to fluid surface, including ripples, white water and foam [Ihmsen et al. 2012]. These techniques advect new particles using the velocity field from the original simulation. However due to the chaotic nature of the Druun's dough fluid simulation, and its tendency to launch particles away from the bulk of the fluid, these techniques were ill suited for our task. Since the webbing effect spans the interior of the dough simulation, the primary goal is to capture the low frequency motion of the input simulation while maintaining an even distribution of points within the fluid itself. This can be re-framed into a point set mapping problem [Solomon 2018], where the aim is to find the best possible mapping from the tracking points to the fluid's particles, at every frame.

2.2 Generation of Web-Like Structures

There have been many approaches to generating web-like structures from point clouds using Voronoi diagrams [Worley 1996]. However, in practice, the web structures resulting from the Voronoi diagram are difficult to art direct in motion: the cells of the Voronoi diagram tend to rotate around each other rather than stretching in predictable ways. In contrast, related approaches with similar visual characteristics, like Relative Neighborhood graphs [Hoff 2016],

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
SIGGRAPH, Aug 01–05, 2021, Virtual
© 2021 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-XXXX-X/18/06.
https://doi.org/10.1145/1122445.1122456

are constructed using the input points as the vertices of the graph, making them much more art-directable when in motion.

3 TRACKING FLUID SIMULATIONS

We capture low frequency motions in fluid simulations by transporting a set of points towards a target set of points in an iterative process. By using the optimal mapping between the two point sets, the process minimizes the distance needed to travel from any single point to any target point per frame.

Algorithm 1: Fluid Tracking Operation. Optimal mapping is computed from the previous timestep's position to scattered particle position of current timestep.

input : Druun Dough Simulation Particles X_t, Tracking Particles P_t, where t is time.

$X'_t \leftarrow$ filter(X_t);
$field \leftarrow$ generateDensityField(X'_t);
$Y \leftarrow$ scatterParticles($field$);
$P_t \leftarrow$ optimalTransport(P_{t-1}, Y);

The fluid tracking operation, as described in Algorithm 1, consists of three main operations: (1) filter the incoming point set to remove regions of low density (e.g. fly off particles), (2) convert the points to a density field and scatter random points uniformly within that density field, and (3) find a coherent mapping from one frame to another between the random point sets by optimal transport. The first two steps of the process are straightforward to achieve in Houdini through the use of the VDB From Particles SP and the Scatter SOP. The Optimal transport step is implemented using Numpy and a Python SOP in Houdini. We follow the same basic steps outlined in Peyré [2011].

We scatter new points as opposed to using our incoming points as tracking points because it gives a more regular distribution, which helps to prevent the webbing from bundling up in later simulation steps. However, the random distribution of generated tracking points at each frame causes random motion of P. To overcome this, we smooth out the particle motion through Laplacian style smoothing on the individual particle's motion paths.

4 WEAVING THE WEBBING

At the start of the webbing generation, described in Algorithm 2, a set of 20 or so webbing particles is generated around each fluid-tracking particles from the previous stage. These newly generated webbing particles are used to create the web; the fluid-tracking points are used as targets for the web points to follow.

We use optimal transport to track the motion of the fluid-tracking particles with the webbing particles. The webbing particles are mapped to the fluid-tracked particles using explicit constraints. Since the mapping is not bijective, our method allows extra degrees of freedom for the webbing particles as the webbing particles can map to different fluid-tracked particles over frames.

The Relative Neighborhood graph operation is performed at every simulation step to introduce new edges into the graph at a constant rate; however, this is an additive operation, meaning we only remove edges of the graph if specific conditions are met.

Algorithm 2: Generate Webbing Operation. All non-constraint operations are computed in a point-wise manner. Constraint operations are handle on a per constraint basis in parallel. Optimal transport mapping is also solved globally.

input : Webbing particles P_i, webbing connections C_i, Tracked Particles X_i

if $length(C_i) \geq restLength(C_i)$ **then**
  remove(C_i);
end
$C \leftarrow relativeNeighborhood(P_i, P)$;
updateRestLength(C);
$P_i = lerp(P_i, OptimalTransport(P_i, X), .5)$;
weightedSmoothing(P);

To prevent flickering, we modify the technique described by Hoff [2016]: we remove edges of the graph if the constraint has either been alive longer than a pre-specified number of frames (typically a random value fit between a minimum and maximum age per edge), or has stretched beyond a pre-specified length. We use a weighted Laplacian smoothing on the graph based on the age in frames of any given webbing connection. This age weighting parameter is exposed to the artist as a slider-editable per-simulation constant. This creates the illusion that the webbing connections are smoothly forming and smoothly breaking, since this age value will ultimately drive the thickness of our output mesh. Lastly, once the entire simulation is solved, we convert the edge graph into a renderable mesh through the use of the VDB toolset within Houdini.

5 CONCLUSION

The Druun's webbing is a complex multi-stage simulation that starts from a fluid simulation and becomes a complex network of webs. The Druun is featured in well over 100 shots in "Raya and the Last Dragon", and the webbing itself was solved in a completely automated way, requiring user intervention in less than 5 shots. The webbing's individual components are as useful as the whole system; we believe that new advances in Optimal Transport theory will play a key role in designing the future of effects animation, due to its wide range of utility. We hope to continue to explore the utility of relative neighborhood graphs as well, due to their ease of construction and their elegant appearance.

REFERENCES

Anders Hoff. 2016. On Generative Algorithms: Differential Lattice · inconvergent. https://inconvergent.net/generative/differential-lattice/

Markus Ihmsen, Nadir Akinci, Gizem Akinci, and Matthias Teschner. 2012. Unified Spray, Foam and Air Bubbles for Particle-Based Fluids. *Vis. Comput.* 28, 6–8 (June 2012), 669–677. https://doi.org/10.1007/s00371-012-0697-9

Gabriel Peyré. 2011. The Numerical Tours of Signal Processing - Advanced Computational Signal and Image Processing. *IEEE Computing in Science and Engineering* 13, 4 (2011), 94–97. https://hal.archives-ouvertes.fr/hal-00519521

Justin Solomon. 2018. Optimal Transport on Discrete Domains. arXiv:1801.07745 [math.OC]

Steven Worley. 1996. A Cellular Texture Basis Function. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96)*. Association for Computing Machinery, New York, NY, USA, 291–294. https://doi.org/10.1145/237170.237267