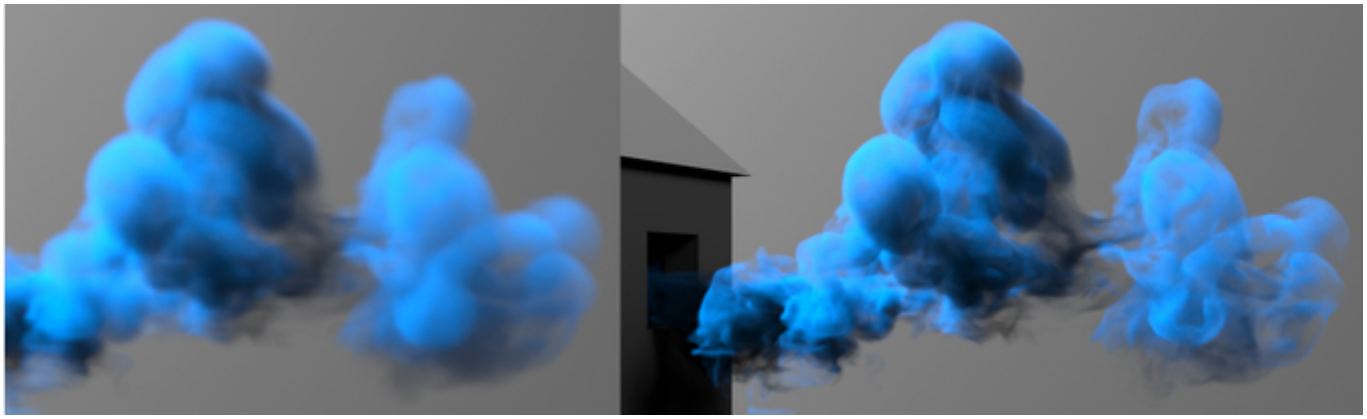
Volumetric Super-Resolution Tool



What is this?

Well, glad you ask, it's the bleeding edge of technology!

This is a tool using neural networks to up-scale low-res simulations to allow for a novel workflow for Pyro/Smoke FX tasks. The idea is that after receiving approval for a low-res sim (which usually happens quite quickly) instead of having to redo the sim in a higher render-able resolution (changing the behaviour in almost all cases), the artists can now sometimes take this shortcut of converting the low-res sim into a higher res one, potentially cutting a pyro FX task in half. While the paper's suggested speeds aren't matched (3x slower due to their superior hardware) it still outperforms running the higher res simulation in most cases. And this leaves out the main advantage: You can simply up-res a frame for look-dev the end of the sim, whereas otherwise it would have to be ran from the start and waited on for hours.

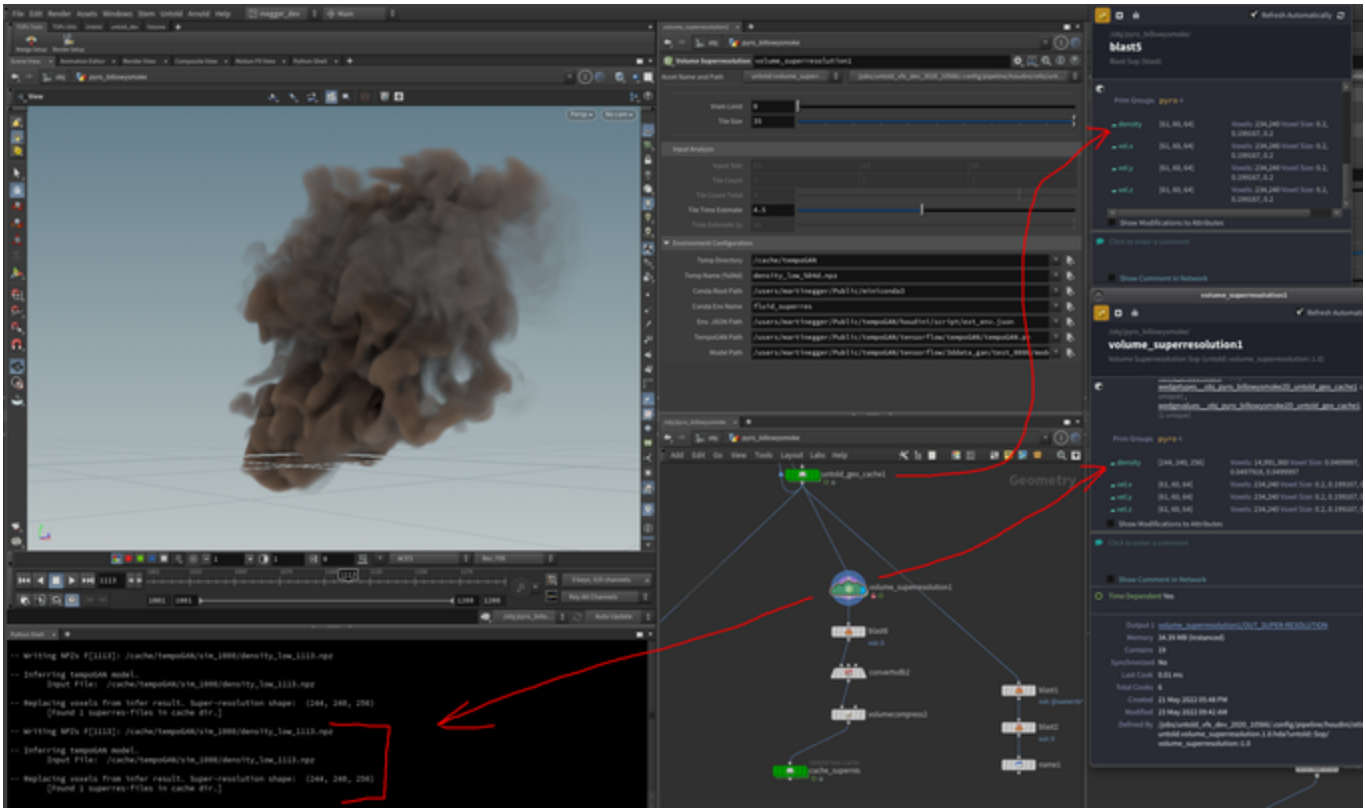
This tool is a Houdini implementation of the paper "TempoGAN", released on Siggraph 2019 Link to Paper / Link to GitHub. The original paper worked with an open-source (but unfortunately very niche) format for volume simulations (MantaGen) and the core had to be rewritten a lot to work with Numpy (Python array library) formats.
tempoGAN_prev_v5_three_fixed.mp4

Usage

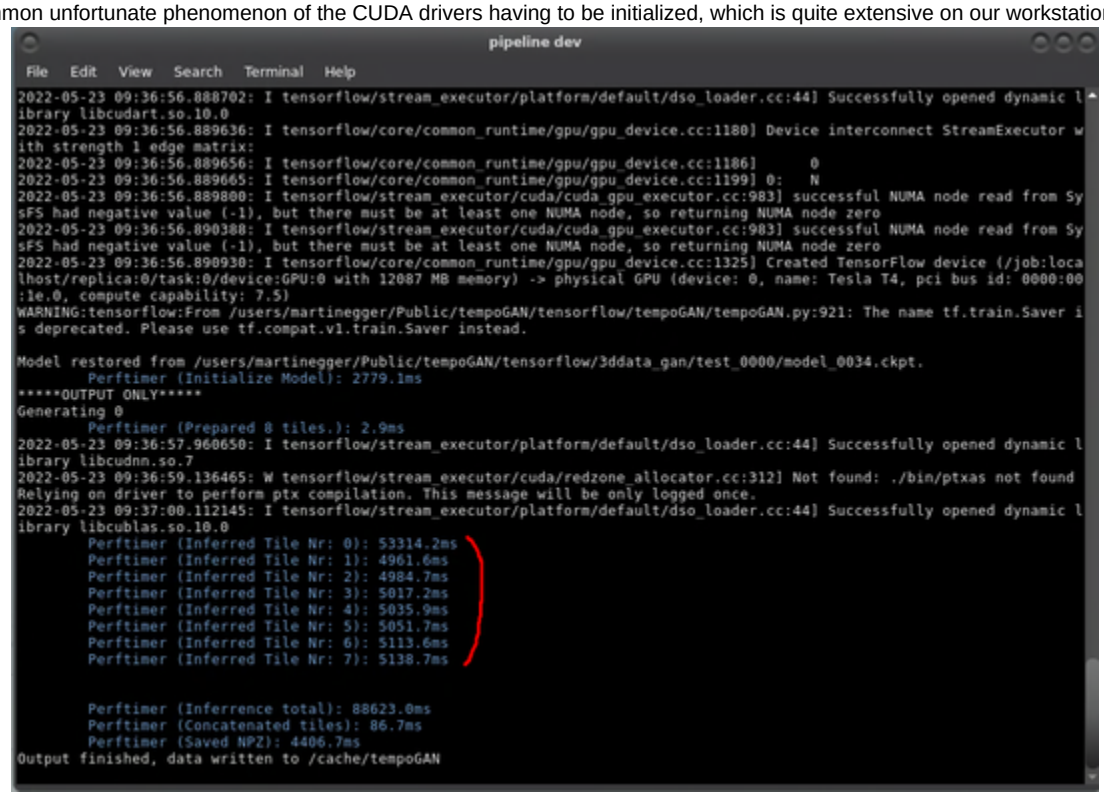
The implementation is simply a node you drop down before caching with a normal geo-cache that turns your low res into high res by a fixed factor of 4! The video above is of three test simulations, with the current implementation taking about 80 seconds for a frame to upscale 64^3 to 256^3. While the local evaluation is dependent on the GPU, it can fortunately also be easily sent to the farm, not even requiring a GPU (though it will be a negligible 20% slower with our beefy farm CPUs).

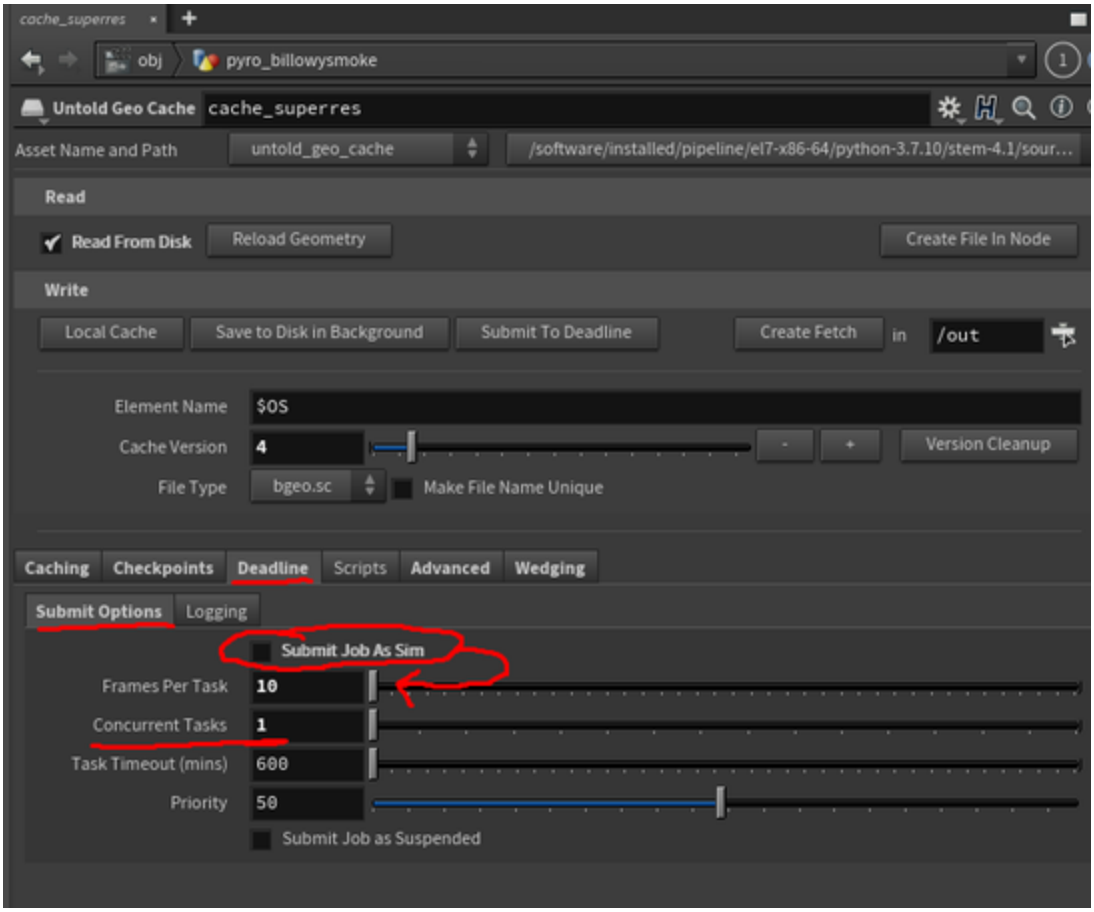
Required input: In order for the tool to work, the input has to be a density and three velocity volumes (not VDBs) and they need to have the exact same dimensions, so it will not work for velocities at half resolution.

> Before the tool is included in our Untold-TD repository, it sits at the following location:
>
> `/jobs/untold_vfx_dev_2020_10566/.config/pipeline/houdini/otls/untold.volume_superresolution.1.0.hda`
>
> To use it for your project, simply copy it over to your project's OTL folder (by replacing the job title in the path).



In order to track progress, while it is evaluating, take a look at the console from which Houdini was launched. There, all the progress will be written to (a lot of tensorflow test, but also some metrics on evaluation, like how long a single tile takes to up-res). The first tile inference time spike is a common unfortunate phenomenon of the CUDA drivers having to be initialized, which is quite extensive on our workstation's T4 GPUs.





The tool has defaults set to access (the culprit) @ Martin Egger 's Public user folder (`/users/martinegger/Public/`, where both the paper and the Mini-Conda environment necessary for the evaluation sit. Read and execution permissions are set for everyone. To make adjustments simply copy the tempoGAN folder someplace else and modify the path that the HDA points to accordingly.

Look

Look wise, the outcome is greatly influenced by the provided ML-model which was trained by the authors of the paper on very smooth training data, leading to the up-res being just as smooth.

Retraining

The solution to the above problem is training further models with more production-oriented data. Those could match simulation types like the preview's smoke simulations with a lot of disturbance. (Others potentially being fire/explosions). As of time of this writing, re-training has some technical hurdles which need a bit more attention, but that is to be done soon.

So far, an automated output (in the required numpy format) has happened here: `/jobs/untold_vfx_dev_2020_10566/build/fx/MLdata/h_MLdata/hip/generic/ML_upres_data_prep`

Here, 10 sims were generated, to be used for training: `/jobs/untold_vfx_dev_2020_10566/build/fx/MLdata/h_MLdata/geo/generic/ML_upres_data_prep/npz3d`

Here is the demonstrating video of the original paper:

Outlook

The main ToDo here remains training new models and creating a drop-down parameter on the asset making selecting various models intuitive.

However, there was a second iteration of the paper released in the following year “Multi-Pass GAN”, improving the process significantly by reworking the approach and the model architecture, leading to 2-3x speed gains. Link to Paper / Link to GitHub. Unfortunately, the new implementation lacks much of the old paper’s flexibility to handle arbitrary input (with automatic slicing and non-uniform axis lengths, and not being restricted by powers of 2). Unfortunately, this means that an even more invasive re-write of the source code would have to be done than to make it work with the Numpy format, which might take a good amount of time and effort.

Given that the tool works flawlessly on our CPU farm, reducing cost for evaluation, implementing this second iteration of the paper became less relevant.

Here is also a backup upload of the HDA: