FlumeNet: A neural network model for generating videos of flume experiments

Júlio Hoffimann Mendes[1]* (**Category:** Physical Sciences)

Abstract
In geomorphology research, flume experiments are used to study patterns of landscape evolution, and to understand the physical processes by which these patterns are created on the surface of the Earth. Understanding these processes is important for assessing risk of environmental disasters (e.g. floods in urban areas) and for modeling natural resources such as oil & gas and groundwater. Although various numerical models were proposed in the literature for approximating flow and sediment transport captured in flume experiment videos, these models often show limited resemblance to the records and/or are quite expensive to calibrate and run. In this work, a neural network model is proposed for generating new videos of the flume, which are laborious to obtain otherwise, but that are important for geomodeling and uncertainty quantification studies (e.g. statistical hypothesis testing). The network is trained on a sequence of frames recorded in a flume tank across various experiments designed with different boundary conditions. The videos generated by the network are assessed qualitatively on the basis of visual inspection and quantitatively with return level plots from extreme value theory and autocorrelation statistics from variogram analysis.

Keywords
video generation — surface processes — landscape evolution

[1] *Department of Energy Resources Engineering, Stanford University*
***Corresponding author**: juliohm@stanford.edu
Project code: `https://github.com/juliohm/FlumeNet`

Contents

Introduction

Surface processes (e.g. water flow in rivers and deltas) are constantly reworking the landscape of our planet with perhaps the most diverse patterns of sediment displacement known to humanity. Capturing this diversity is important for advancing our knowledge of systems [Murray et al., 2009], and for sustainable exploitation of natural resources by future generations, as resources are stored under ground by surface processes. From a modeler's perspective, great diversity comes with great uncertainty. Although quantifying uncertainty about physical processes of the past is understandably hard, modeling this uncertainty explicitly is crucial for assessing the risk of floods in urban areas, and for estimating reserves of natural resources, among other reasons.

In order to model uncertainty, statistical methods require hundreds or thousands of observations, yet it is only recently that the geomorphology community started to collect high-resolution image data from a few (< 10) flume experiments designed with complex boundary conditions [Bufe et al., 2016]. Such advances in data acquisition via sophisticated experiment apparatus is what motivates this project and report.

We aim to develop a statistical learning method with neural networks that is capable of reproducing the spatial patterns of flow captured in flume videos, and that could potentially be used to augment the data for subsequent Monte Carlo studies and hypothesis testing. Differently than traditional basin filing modeling [Paola, 2000], our goal is to let the data speak for itself. The present work is unique as there is no previous attempt in the literature to apply modern machine learning methods to extract insights from this type of data.

In section 1, we quickly describe the dataset used in this project, and introduce the proposed neural networks for video generation. In section 2, we compare side-by-side the original videos recorded in the tank with the videos synthesized by the neural networks. In the same section, we derive useful statistics for assessing the quality of flow pattern reproduction. Finally, in section 3, we summarize the findings and discuss possible future work.

1. Data and Methods

In a recent investigation published in nature geoscience, [Bufe et al., 2016] generated a dataset comprising seven high-resolution videos of flume experiments under different boundary conditions (e.g. uplift rate, sediment discharge) in an attempt to understand the forces that control relief creation and landscape flattening in sedimentary basins. Through an ongoing research collaboration, we've been investigating the extent to which flow patterns recorded in the tank can be reproduced with advanced statistical methods.

1.1 Data Preprocessing and Augmentation

Frames extracted from the videos at a rate of 0.5 frames per minute (fpm) were organized on separate folders representing distinct flow regimes (see Table 1). Each frame (or image) in the dataset comes in high-resolution with 3939×5931 pixels (see Figure 1).

Table 1. Number of frames for each flow regime.

run1	**run2.1**	**run2.2**	**run3.1**	**run3.2**	**run4**
1638	171	1339	113	1256	1256
run5	**run6.1**	**run6.2**	**run7.1**	**run7.2**	**TOTAL**
2404	141	1033	59	2106	11516

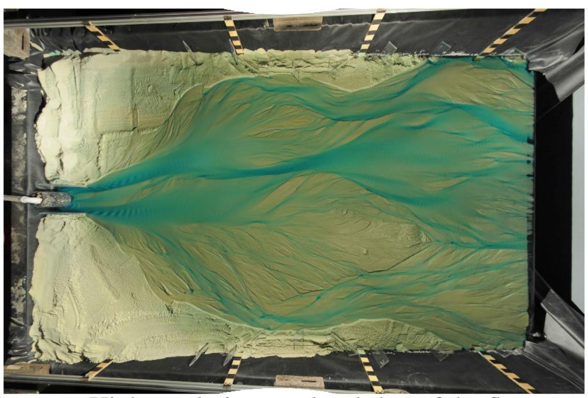

Figure 1. High-resolution overhead shot of the flume tank.

Cropping & upscaling We crop the pixels in the images that are outside of the sandbox and upscale the resulting images to a more manageable resolution with 150×100 pixels for training the neural network models (see Figure 2).

Thresholding To further reduce the dimension of the problem, we convert the RGB images into binary images where the white color represents areas of active (or intense) water flow. This is done by first converting the images from RGB to HSV color space and by picking the appropriate hue value for the blueish color with a given tolerance range (see Figure 3).

Augmentation We augment the dataset of 11516 images by flipping them horizontally at random. Because the water flow has a preferential direction from the top to the bottom of the tank, we do not flip the images vertically. We consider random crops of similar size as a future augmentation technique.

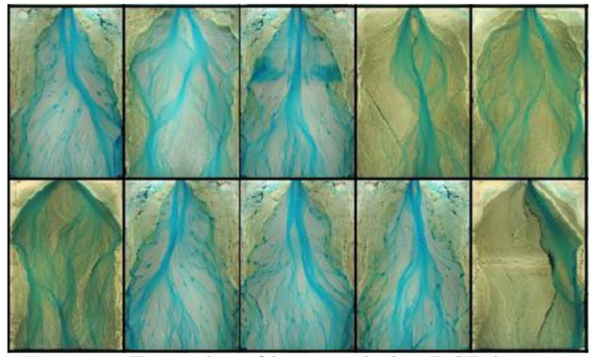

Figure 2. Examples of low-resolution RGB images.

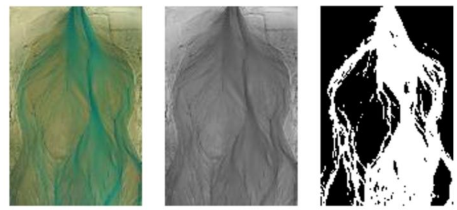

Figure 3. Thresholding RGB images into water flow patterns.

Normalization All images are normalized to contain pixel values in $[0,1]$ with a straightforward division by the maximum possible integer value of 255.

Optical flow Besides training the neural network models with frames directly, we also attempt to train the networks with optical flow images generated for each dataset. See Figure 4. Predicted optical flow images can be used to warp initial frames and synthesize new videos.

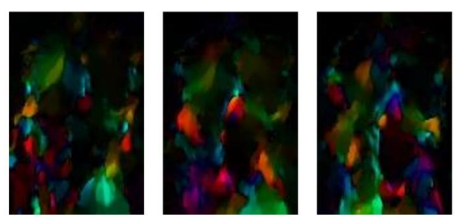

Figure 4. Example optical flow images computed from RGB datasets. Colors represent direction of flow.

Train-Dev-Test split All the runs in Table 1 are included in the training set except run3.1. The images in run3.1 and run3.2 come from the same experiment, and are expected to have similar distribution. The hundred frames in run3.1 are saved for the dev set. In the future, at the time of deployment or publication, a new experiment should be designed just for testing the trained network models.

1.2 Proposed Neural Network Models

Regardless of the model, the problem setup is the same. A window of past frames $\boldsymbol{x}_p = (I_{t-p+1}, \ldots, I_{t-1}, I_t)$ is defined by looking p steps into the past, and the goal is to predict a window of frames $\boldsymbol{y}_f = (I_{t+1}, I_{t+2}, \ldots, I_{t+f})$ with f steps into the future. For this project, we concentrate on the case

$p \geq 3, f = 1$ of single step prediction, and unroll the neural network iteratively in order to produce new videos of arbitrary length based on a small collection of initial p frames.

The training set $\mathscr{D} = \left\{\boldsymbol{x}_p^{(i)}, \boldsymbol{y}_f^{(i)}\right\}_{i=1,2,\dots,m}$ is therefore a collection of m slices of the videos. These slices are fed into the following neural network models.

TorricelliNet Inspired by one of Torricelli's equations of motion from classical mechanics

$$x_{t+\Delta t} = x_t + v\Delta t + a\frac{\Delta t^2}{2}$$

in which a particle with velocity v and constant acceleration a moves from locations x_t to $x_{t+\Delta t}$ in the time interval Δt, we design a convolutional neural network with three modules:

1. velocity module

$$\left.\begin{aligned} \boldsymbol{z}_p^{(i)} &= Conv(\boldsymbol{x}_p^{(i)}, kern = 5, pad = 2) \\ \boldsymbol{a}_p^{(i)} &= ReLU(BatchNorm(\boldsymbol{z}_p^{(i)})) \end{aligned}\right\} \text{L times}$$

$$\boldsymbol{v}(\boldsymbol{x}_p^{(i)}) = Conv(\boldsymbol{a}_p^{(i)})$$

2. acceleration module

$$\left.\begin{aligned} \boldsymbol{z}_p^{(i)} &= Conv(\boldsymbol{x}_p^{(i)}, kern = 5, pad = 2) \\ \boldsymbol{\alpha}_p^{(i)} &= ReLU(BatchNorm(\boldsymbol{z}_p^{(i)})) \end{aligned}\right\} \text{L times}$$

$$\boldsymbol{a}(\boldsymbol{x}_p^{(i)}) = Conv(\boldsymbol{\alpha}_p^{(i)})$$

3. prediction module

$$\begin{aligned} \boldsymbol{z}_p^{(i)} &= Conv(\boldsymbol{x}_p^{(i)} + \boldsymbol{v}(\boldsymbol{x}_p^{(i)}) + \frac{\boldsymbol{a}(\boldsymbol{x}_p^{(i)})}{2}, kern = 1) \\ \boldsymbol{y}_f^{(i)} &= Sigmoid(BatchNorm(\boldsymbol{z}_p^{(i)})) \end{aligned}$$

The prediction module combines the first and second derivatives with the past frames to produce a new frame. The kernel size of 1 is used to shrink the volume (or depth) from any given number of hidden channels to the number of output channels.

SliceNet In an attempt to capture very small changes in flow patterns, a recurrent neural network model is proposed. First, 50 evenly spaced rows of the frames are forward in time using different GRU units. Then, another set of 50 GRU units is used to fill in the gaps between the predicted rows as illustrated in Figure 5. The hidden states of the GRU units are fed into a dense layer with sigmoid activation to produce frames with valid pixel values in $[0, 1]$.

The proposed neural network models are trained using all three different color spaces RGB, GRAY, and BW, and optical flow images. Each configuration requires a different loss function $\mathscr{L}_o$. For binary images, we use the binary cross-entropy loss whereas for continuous images, we use both L_1

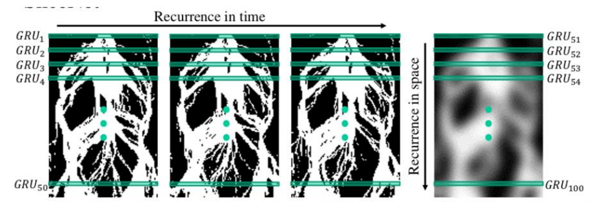

Figure 5. SliceNet architecture. GRU units (1-50) are used to predict evenly spaced rows in the future frame, and another set of GRU units (51-100) is used to fill in the gaps.

and L_2 loss. For the SliceNet, an additional total variation term is added to the loss to enforce continuity between the slices:

$$\mathscr{L}(\boldsymbol{y}, \bar{\boldsymbol{y}}) = \mathscr{L}_o(\boldsymbol{y}, \bar{\boldsymbol{y}}) + \sum_i \left| \left\| \boldsymbol{y}_{i,:} - \boldsymbol{y}_{i+1,:} \right\|_1 - \left\| \bar{\boldsymbol{y}}_{i,:} - \bar{\boldsymbol{y}}_{i+1,:} \right\|_1 \right|$$

The optimization is performed with the Adam optimizer and with various learning rates. The best obtained results are reported in the next section.

2. Results and Discussion

We assess the videos synthesized by the neural networks with visual inspection, and with relevant statistics computed on the validation set.

2.1 Visual Inspection

Based on p initial frames, we synthesize new videos of the flume experiment by unrolling the network. These videos are compared side-by-side with the original records as illustrated in Figure 6 and Figure 7.

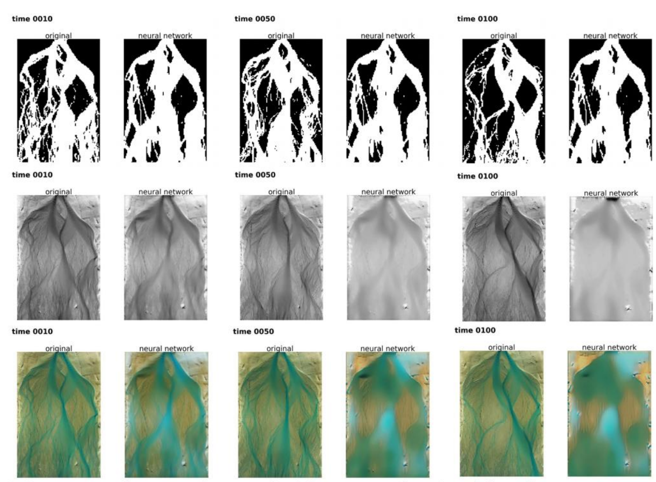

Figure 6. Screenshots of videos synthesized by the TorricelliNet for BW, GRAY, and RGB datasets.

Full videos are available at `https://vimeo.com/album/5055962`. For the BW dataset, the neural network quickly looses its ability to mimic the flow dynamics and just copies the same frames forward in time. For the GRAY and RGB datasets, the frames quickly become blurry.

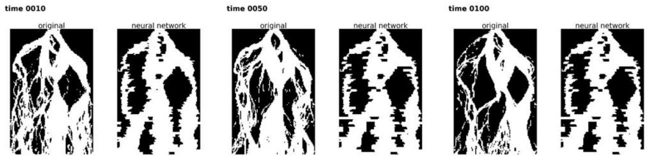

Figure 7. Screenshots of videos synthesized by the SliceNet for BW dataset.

2.2 Return Levels and Variograms

For assessing the performance of the network quantitatively, we introduce two statistics based on the difference process. The difference process $d_t = \|I_{t+1} - I_t\|_1$ is the time series of L_1 norms between consecutive frames. Its normalized version $d_t^* = d_t/d_1$ is computed for the original and synthesized videos.

Return levels We would like the original and synthesized videos to have the same return levels [Beirlant et al., 2005]. This means that on average big changes in flow patterns should take the same amount of time to happen.

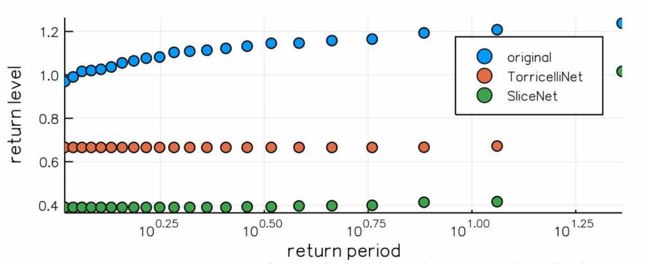

Figure 8. Return levels for original and synthesized videos.

In Figure 8, the proposed neural networks models underestimate return levels of the true (or original) phenomena. This is in agreement with the previous visual inspection, in which we noticed that very similar frames were copied forward in time after a short time horizon.

Variograms We would like to reproduce the autocorrelation of the process [Matheron, 1971]. In other words, given any time interval separating two frames in the synthesized video, we would like the correlation coefficient to be equal to that of the original video.

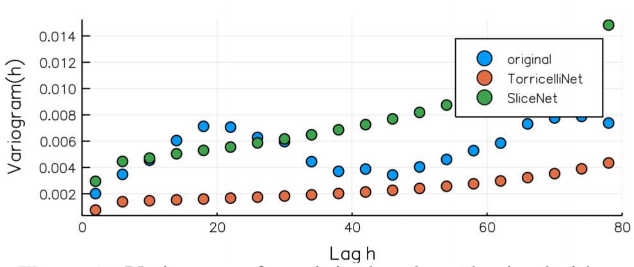

Figure 9. Variograms for original and synthesized videos.

In Figure 9, the ciclicity of the empirical variogram computed on the normalized difference process is not reproduced by the proposed neural network models. The correlation lengths in the videos synthesized by the neural networks are much larger than those present in the true phenomena.

Among the two proposed neural network models, the SliceNet produces frames with greater variability. Learning curves in Figure 10 indicate an efficient optimization. Similar learning curves were seen while training the TorricelliNet, but are omitted here to save space in the report.

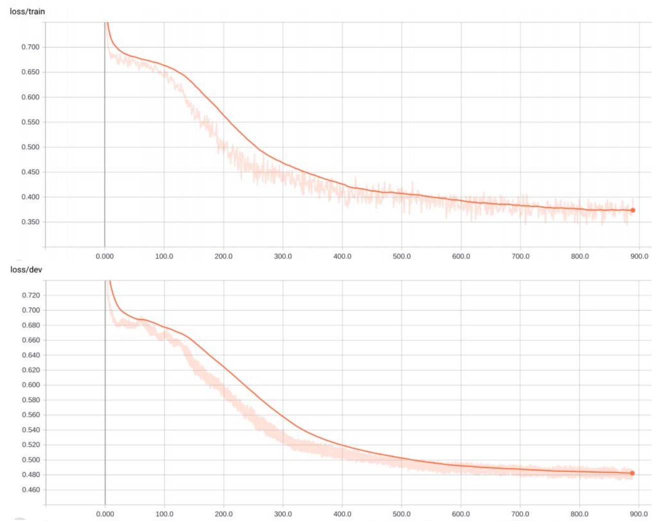

Figure 10. Learning curves for SliceNet on BW dataset.

3. Conclusions and Future Work

The neural network models proposed in this work are far too simple to accommodate the complexity of flow and sediment transport recorded in flume experiments. Despite the various attempts to train the networks with different color spaces, architectures, and loss functions, all fail to reproduce statistics of interest such as return levels and autocorrelation.

The SliceNet architecture together with the total variation loss is promising. Additional work is needed to eliminate artifacts in between neighboring GRU units.

Training the networks on optical flow images instead of raw frames did not improve the results considerably. Future work should include a more careful investigation of video synthesis by means of warping frames with optical flow.

Acknowledgments

I would like to thank teaching assistants for their valuable feedback, maintainers of the PyTorch project for this great tool, and AWS for providing computing resources.

References

J. Beirlant, G. Dierckx, and A. Guillou. Estimation of the extreme-value index and generalized quantile plots. *Bernoulli*, 11(6):949–970, Dec 2005. ISSN 1350-7265. doi: 10.3150/bj/1137421635. URL http://dx.doi.org/10.3150/bj/1137421635.

Aaron Bufe, Chris Paola, and Douglas W. Burbank. Fluvial bevelling of topography controlled by lateral channel mobility and uplift rate. *Nature Geoscience*, 9(9):706–710, Aug 2016. ISSN 1752-0908. doi: 10.1038/ngeo2773. URL `http://dx.doi.org/10.1038/ngeo2773`.

G. Matheron. *The Theory of Regionalized Variables and Its Applications*. Centre de Morphologie Mathématique Fontainebleau: Les cahiers du Centre de Morphologie Mathématique de Fontainebleau. École national supérieure des mines, 1971. URL `https://books.google.com/books?id=TGhGAAAAYAAJ`.

A. Brad Murray, Eli Lazarus, Andrew Ashton, Andreas Baas, Giovanni Coco, Tom Coulthard, Mark Fonstad, Peter Haff, Dylan McNamara, Chris Paola, and et al. Geomorphology, complexity, and the emerging science of the earth's surface. *Geomorphology*, 103(3):496–505, Feb 2009. ISSN 0169-555X. doi: 10.1016/j.geomorph.2008.08.013. URL `http://dx.doi.org/10.1016/j.geomorph.2008.08.013`.

Chris Paola. Quantitative models of sedimentary basin filling. *Sedimentology*, 47:121–178, Feb 2000. ISSN 0037-0746. doi: 10.1046/j.1365-3091.2000.00006.x. URL `http://dx.doi.org/10.1046/j.1365-3091.2000.00006.x`.