Super-resolution Fluid flows using Machine Learning

Fabian Greavu, Johan Andrey Bosso

22/09/2019

Outline

Intro

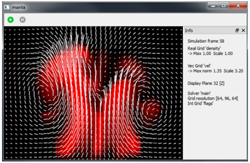
- **Fluid Flow** is a part of fluid mechanics and deals with fluid dynamics. Fluids such as gases and liquids in motion are called as fluid flow. The motion of a fluid subjected to unbalanced forces.
- In computer graphics, generating 2d and 3d fluids is an interesting task. Generating super-resolution fluid flows is a difficult task
- One solution is to use Machine Learning to train models that can generate those flows and use CNNs to generate super-resolution frames based on the ones generated before.

Software used

- **Mantaflow** is an open-source extensible framework targeted at fluid simulation research in Computer Graphics.
  It written in C++ but it can be used by python with given interfaces and automatically parallelizes all operations.
- **TempoGAN** is an open-source library (available at Github) that uses Mantaflow to syntethize four-dimensional physics fields with neural networks and addresses the super-resolution problem for fluid flows..
- All those software use **Tensorfow** to train data, **CUDA** with cnn for parallel computation (if not given then **OPENMP** on CPU).

Mantaflow

- Mantaflow can be cloned from bitbucket repo and all installation instructions can be found on main website.
- Once installation is completed it allows to use some demos in the example folder to view results using default pre-trained models on a simpole GUI



Mantaflow Overview

A first obstacle when you work with mantaflow on python is the lack of documentation: being a C++ library interfaced with python all documentation is in C++, so a brief overview is needed.

- **Solver**:Mantaflow environment revolves around **the solver**, wich is the base object for any others structure in mantaflow. Teh solver defintion requires a gridSize, which is a vector with the size in each axis, and the dimension of desired space so
  gs = vec3(32, 32, 32) dim = 3

  s = Solver(name='main', gridSize=gs, dim=dim)

  define a 3D environment with size 32 on each axis.

Mantaflow overview

- **Geometry**:To simulate a real scene it needs to place some object in our Solver like obstacle or a system of particles. Mantaflow provides some base geometry like **Box** or **Sphere**, which require a size definition and coordinates in space, better if they are given in term of solver size-ratio to simplify changes. After the object initialization, we anchor it to the grid and set as obstacle

```
obstacle1 = Box(parent=s, p0=gs * vec3(0, 0, 0), p1=gs * vec3(0.4, 0.6, 1.0))
obstacle1.applyToGrid(grid=flags, value=FlagObstacle)
```

Mantaflow overview

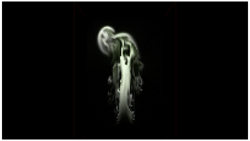
- **Fluid** :Being a fluid simulation environment the most important object in mantaflow is fluid, which is defined as a set of particles that fill a space. First of all we define a particle size. Unlike what was said for obstacles particles size must be independent from solver size, this allow you to simulate higher resolution in fluid if you increase the solver size.

Mantaflow overview

- **Grid System**: The simulation core are grids, which apply some effects on unfixed object, so if you want to introduce pressure in your world you have to define a presure as a **RealGrid** a velocity as a **MACGird** and a phi as a **LevelSetGrid**.
  Mantaflow combines behaviors given by grids to render the scene frame by frame.

tempoGAN

- tempoGAN can be cloned from github repo. Readme file shows how to proceed with installation based on mantaflow.
- It is a mantaflow repo with additional scripts (python) for training and running. The main script is **tempoGAN.py** that starts the training. A training example python script allows us to start a quick training with pre-setted parameters.



Paper

Goal: represent a first approach to synthesize four-dimensional physics fields with neural networks. There already are some basic approaches to the problem but not in a deep way. The main contributions are:

- A novel temporal discriminator, to generate consistent and highly detailed results over time
- Artistic control of the outputs, in the form of additional loss terms and an intentional entangling of the physical quantities used as inputs
- A physics aware data augmentation method
- A thorough evaluation of adversarial training processes for physics functions

GAN

To better understand the mechanisms behind all the work, some definitions have to be done.

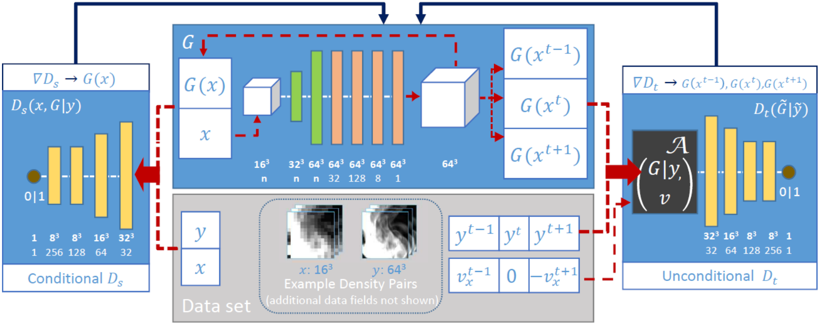
- GAN: Generative Adversarial Network.It is a type of construct in neural network technology that offers a lot of potential in the world of artificial intelligence. A generative adversarial network is composed of two neural networks: a generative network and a discriminative network. These work together to provide high-level simulation of conceptual tasks.

tempoGAN algorythm

Training is splitted into three main steps:

- 1: training a spatial Discriminator D_s with normal GAN loss for Discriminators
- 2: training a novel temporal Discriminator D_t with normal Discriminator loss but with sequence data input
- 3: training the Generator with the adversarial loss from both D_s and D_t, a novel layer loss term, and a regularization l_1 loss

Algorythm components



Alg. 1 tempoGAN training algorithm

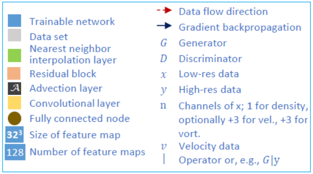
while training **do**

Train D_s with $\mathcal{L}_{D_s} = -\mathbb{E}_m[\log D_s(x, y)] - \mathbb{E}_n[\log(1 - D_s(x, G(x)))]$

Train D_t with $\mathcal{L}_{D_t} = -\mathbb{E}_m[\log D_t(\widetilde{Y}_{\mathcal{A}})] - \mathbb{E}_n[\log(1 - D_t(\widetilde{G}_{\mathcal{A}}(\widetilde{X})))]$

Train G with $\mathcal{L}_G = -\mathbb{E}_n[\log D_s(x, G(x))] - \mathbb{E}_n[\log D_t(\widetilde{G}_{\mathcal{A}}(\widetilde{X}))]$

$$+\mathbb{E}_{n,j}\lambda_f^j \left\|F^j(G(x)) - F^j(y)\right\|_2^2 + \lambda_{L_1}\mathbb{E}_n \left\|G(x) - y\right\|_1$$



The code

- As said before the repository is open-source and can be cloned and ready to go.
- Some little twicks are needed to make CUDA work along with Tensorflow and mantaflow in the background. At last it is ready to be runned.
- Having small harware computation eqipped, it isstill possible to download Models for 2d and 3d pre-trained and make them run on the hardware equipped