# V-Flow: Deep Unsupervised Volumetric Next Frame Prediction

Alon Shtern
Department of Computer Science
Technion, Israel Institute of Technology
Email: ashtern@campus.technion.ac.il

Ron Kimmel
Department of Computer Science
Technion, Israel Institute of Technology
Email: ron@cs.technion.ac.il

***Abstract*—Predicting the temporal dynamics of three-dimensional images is an important means for analyzing volumetric data. We propose an unsupervised learning of a multi-scale optical flow based approach for predicting the next frame of a sequence of volumetric images. The fully differentiable model consists of specific crafted modules that are trained on small patches. To test the proposed approach, we ran unsupervised experiments on synthetic incompressible two-fluid Navier-Stokes simulation and real magnetic resonance imaging (MRI) of the cardiac cycle. Comparison of a spatial version of our architecture to recent methods in predicting the next frame of movie sequences shows significant quantitative and visual improvements.**

## I. Introduction

Motion lies at the core of dynamical systems. One way to understand the motion of forms and structures in images is through *optical flow* [1], [2], which is an approximation of the motion of objects in an image, and its computation was traditionally based on local spatial and temporal derivatives in a given sequence of images. That is, in two dimensions it tries to specify how much the semantic content of each image pixel moves between adjacent images, while in three dimensions it specifies how much the content of each volume element (voxel) moves between adjacent volumes. While several solutions to deep optical flow and video prediction are well established [3], [4], [5], [6], [7], [8], [9], [10], [11], volumetric temporal evolution learning remained unexplored. Part of the difficulty in devising a robust and efficient 3D optical flow is due to the large number of possibilities by which each voxel can move. Another difficulty is that in many situations, such as three dimensional dynamical systems that involve fluid or gas, the *brightness constancy* [1] assumption is invalid. Moreover, unlike the two dimensional case, where there exist some benchmarks with ground-truth optical flow, volumetric datasets lack such supervised information.

The approach introduced here, is motivated by several papers that predict the next frame of movie sequences, knowing the past and the present frames, in an unsupervised manner. The end-to-end differentiable architecture is based on multi-scale optical flow prediction. Each pyramid level consists of a deep generative network, which is designed as a series of convolution layers with element-wise multiplication modules, followed by rectified linear units. The generative network recursively refines the future optical flow estimation and simultaneously adjusts the last frame that the warping module operates on. We refer to the proposed architecture as “V-Flow.”

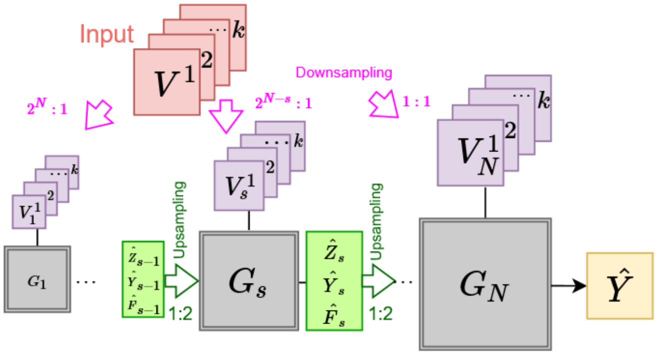


Fig. 1: Multi-scale architecture.

The key contributions of V-Flow include the following features.

- We introduce a *latent frame* which is a variation of the last frame. Each pixel of the latent frame is moved in keeping with the optical flow to produce the next frame prediction. Optimizing for both the latent image as well as the optical flow is important for two reasons. First, the use of the latent frame is beneficial for situations where the brightness constancy assumption is invalid. Second, if quantitative performance (mean square error) is the important measure, then it is best to somewhat blur the images before moving them.
- The model is trained without any supervision effort, by minimizing the reconstruction error between the predicted volumetric frame and the ground truth next frame. The optimizer minimizes the absolute error criteria aggregated with a *volumetric gradient difference loss* function.
- Given previous volumetric frames, the system predicts 3D future optical flow. Multiple scales are combined linearly in a *Laplacian pyramid* like fashion.
- The neural network is specifically designed for the task of optical flow prediction and consists of multiple *convolution and multiplication* layers. A novel three dimensional *warping module* is introduced, comprised of 3D grid generator and a *linear tri-sampler*.

## II. Related Efforts

As learning of three dimensional volume prediction is yet unexplored, we draw inspiration from papers related to video prediction. Prediction of both two and three dimensional video dynamics is a challenging problem due to its complexity and the inherent intensity ambiguity in image sequences. Srivastava *et al.* [4] introduced Long Short Term Memory (LSTM) networks [12] to learn representations of video sequences in an unsupervised manner. Lotter *et al.* [5] designed a neural network architecture, inspired by the concept of *predictive coding* to continually predict the appearance of future video frames, using a deep, recurrent convolution network with both bottom-up and top-down connections. Patraucean *et al.* [6] used a convolution LSTM network that integrates changes over time and an optical flow [1], [2] prediction module that extends the Spatial Transformer [13] by using a per-pixel transformation for each position instead of a single transformation for the entire image. This approach is somewhat related to supervised deep optical flow models like DeepFlow [9], FlowNet [10], and SPyNet [11].

Ranzato *et al.* [3] defined a recurrent network architecture inspired from language modeling, predicting the frames in a discrete space of patch clusters. Brabandere *et al.* introduced the Dynamic Filter Network, where filters are generated dynamically conditioned on an input and demonstrated the effectiveness of the dynamic filter network on the task of video prediction. The Video Pixel Networks proposed by Kalchbrenner *et al.* [7] use a generative video model, that reflects the factorization of the joint distribution of the pixel values in a video. Oh *et al.* [14] proposed an action conditional auto-encoder model, and predicted next sequences of Atari games from a single screen image. To deal with the inherently blurry predictions obtained from the standard Mean Squared Error (MSE) loss function, Mathieu *et al.* [8] proposed a multi-scale architecture, and improved the quality of predicted images by using a Laplacian pyramid [15] and an image gradient difference loss function.

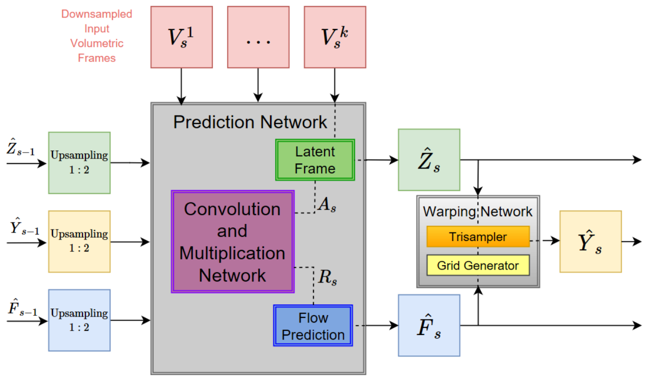


Fig. 2: Generative network $Gs$.

## III. Architecture

### A. Optical flow process

Suppose we are given an input sequence of $k$ volumetric frames (patches) $V^1, V^2, \dots, V^k \in \mathbb{R}^{c \times t \times h \times w}$ with $c$ color channels and grid size of $t \times h \times w$. We would like to predict the next frame $Y \equiv V^{k+1} \in \mathbb{R}^{c \times t \times h \times w}$. We assume the existence of an underlying optical flow process that deforms a latent frame $Z \in \mathbb{R}^{c \times t \times h \times w}$ into $Y$. The latent frame $Z$ is guided by an optical flow vector field $F \in \mathbb{R}^{d \times t \times h \times w}$ $(d = 3)$, such that each pixel of $Y$ is found by moving the corresponding pixel of $Z$ in line with the displacements coded in $F$. This can be expressed by

$$Y(i, x, y, z) = Z(i, \tilde{x}, \tilde{y},\ \tilde{z})$$

where $\tilde{x} = x - F(1, x, y, z)$, $\tilde{y} = y - F(2, x, y, z)$, $\tilde{z} = z - F(3, x, y, z)$, and $i \in \{1 \dots c\}$, $x \in \{1 \dots t\}$, $y \in \{1 \dots h\}$, $z \in \{1 \dots w\}$. We denote the operator that takes $Z$ and $F$ as inputs and calculates $Y$ as the *FlowWarp* operator.

Our goal is to estimate the latent frame $Z$ and the optical flow $F$. Let $\hat{Z} \in \mathbb{R}^{c \times t \times h \times w}$, $\hat{F} \in \mathbb{R}^{d \times t \times h \times w}$ be the estimations of $Z$, $F$, respectively, and let $\hat{Y} \in \mathbb{R}^{c \times t \times h \times w}$ be the next frame prediction of $Y$ that is calculated by $\hat{Y} = FlowWarp(\hat{Z}, \hat{F})$. We wish $\hat{Y}$ to be as close as possible to the target $Y$. This can be formalized by introducing a loss function between the predicted next frame and the ground truth next frame. Let $Loss(\hat{Y}, Y)$ denote this loss function. Therefore, $\hat{Z}$, $\hat{F}$ are found by optimizing

$$\hat{Z}, \hat{F} = \underset{Z,F}{\operatorname{argmin}}\, Loss(FlowWarp(Z, F), Y).$$

A basic approach would set $\hat{Z}$ to be exactly equal to $V^k$. Here, we allow some leeway. We introduce the auxiliary variable $A \in \mathbb{R}^{c \times t \times h \times w}$. $A$ is linearly combined with $V^k$ to get the latent frame $\hat{Z}$.

Theoretically, there are many solutions for $Z$, $F$ that are equivalent. But in practice, because we are setting $\hat{Z}$ as a linear combination of $V^k$ and $A$, we observe that the network converges to a specific solution, such that $FlowWarp(V^k, \hat{F})$ is a good estimation of $Y$. Essentially, we see that the auxiliary variable $A$ can restore the brightness constancy assumption for some situations where it is invalid. It also absorbs most of the blurring effect inherent in video prediction. Hence, in the evaluation phase, it is possible to generate sharper predictions by omitting $A$ altogether. We denote this option as V-Flow-Sharp. Deriving the latent frame $Z$ and the optical flow $F$ is challenging due to the fact that the movement of each pixel between adjacent frames has many degrees of freedom. Therefore, to estimate $\hat{Z}$ and $\hat{F}$ we adopt a multi-scale architecture.

### B. Multi-scale network

**Overview.** The structure of the multi-scale network is based on pyramid decomposition [16], [15], [8], [11] and is made of a series of generative networks that follow one another as illustrated in Figure 1. The generative networks use down-sampled versions of the input sequence to make their predictions, ranging from the lowest to the highest resolutions. The predictions are recursively passed on as starting points for consequent pyramid levels, and are refined in such a way that the output of the last generative network reaches the desired original resolution.

More specifically, let $s \in \{1 \dots N\}$ be the index of the pyramid level. Let $d_s(\cdot)$ be the downsampling function that decimates a frame by a factor of $2^{N-s}$, and let $V_s^1, V_s^2, \dots, V_s^k \in \mathbb{R}^{c \times t_s \times h_s \times w_s}$ be a down-sampled version of the input volumetric frames in increasing resolution, such that $V_s^j = d_s(V^j)$, $\forall j \in \{1 \dots k\}$. The network consists of a sequence of $N$ generative networks denoted by $G_s$. Let $\hat{F}_s \in \mathbb{R}^{d \times t_s \times h_s \times w_s}$ be the optical flow estimation, and $\hat{Z}_s \in \mathbb{R}^{c \times t_s \times h_s \times w_s}$ be the latent frame of the $s$ level of the pyramid. The generative network $G_s$ receives the down-sampled input volumetric frames, and the up-sampled products of the preceding network $G_{s-1}$, computes $\hat{F}_s$ and $\hat{Z}_s$ by a multi-layered neural network, and outputs the next frame prediction $\hat{Y}_s \in \mathbb{R}^{c \times t_s \times h_s \times w_s}$ by warping the frame $\hat{Z}_s$ according to the optical flow $\hat{F}_s$. Then, $\hat{F}_s$, $\hat{Z}_s$ and $\hat{Y}_s$ are upsampled by a factor of two and passed to the succeeding generative network $G_{s+1}$. This is done recursively from the lowest resolution to the finest one.

**Generative network.** The generative network is shown in Figure 2. On the left of the Figure is the prediction module and on the right is the warping module. Two sources feed the generative network. The first is the given input sequence of volumetric frames which are down-sampled to the current resolution. The second information source is the set of up-sampled products of the preceding generative network.

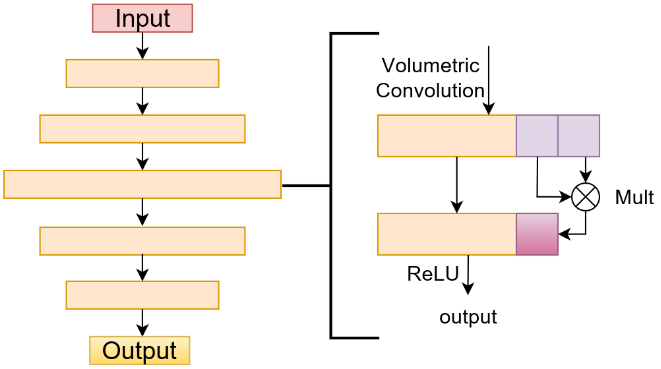


Fig. 3: Convolution and multiplication neural network.

**Prediction module.** The prediction module calculates the optical flow $\hat{F}_s$ and the latent frame $\hat{Z}_s$. It includes a neural network that is specifically designed for the task of optical flow prediction.

**Convolution and multiplication neural network.** The multi-layered neural network outputs the flow residual $R_s \in \mathbb{R}^{d \times t_s \times h_s \times w_s}$ and the auxiliary variable $A_s \in \mathbb{R}^{c \times t_s \times h_s \times w_s}$, that are used to calculate $\hat{F}_s$ and $\hat{Z}_s$. It is based on a classic structure of volumetric convolution layers followed by rectified linear units (ReLU). To better capture the temporal correlations between adjacent frames, we insert an element-wise multiplication block, as shown in Figure 3. The configuration we used in our experiments are given in Table I. In all layers, the size of the multiplication block is $1/4$ of the overall number of feature maps.

TABLE I: Network configuration

| | Volumetric next frame prediction | |
|---|---|---|
| | # feature maps | kernel size |
| G1 | 64,128,64 | 5,3,3,5 |
| G2 | 64,128,256,128,64 | 5,3,3,3,3,3,5 |
| G3 | 64,128,256,128,64 | 7,5,5,5,5,5,7 |
| | Video next frame prediction | |
| | # feature maps | kernel size |
| G1 | 128,256,128 | 3,3,3,3 |
| G2 | 128,256,128 | 5,3,3,5 |
| G3 | 128,256,512,256,128 | 5,3,3,3,3,3,5 |
| G4 | 128,256,512,256,128 | 7,5,5,5,5,7 |

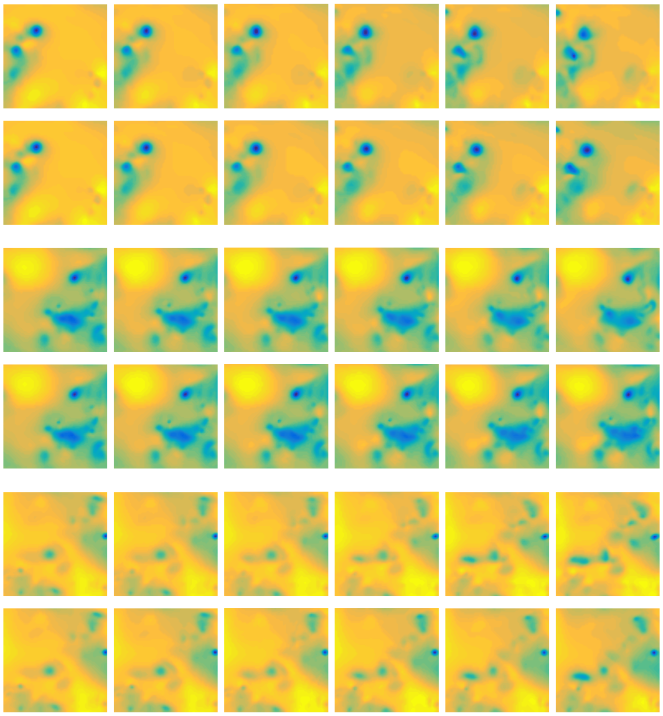


Fig. 4: Turbulence prediction. In each of the 3 examples, the first row are the 6 ground truth pressure slices, and the second row is their respective predictions.

**Flow prediction.** Let $u(\cdot)$ be a function that increases the resolution of the flow prediction of the preceding pyramid level by a factor of two. The optical flow estimation $\hat{F}_s$ is obtained by applying a pyramid style refinement

$$\hat{F}_s = \beta u(\hat{F}_{s-1}) + R_s.$$

The weight $\beta$ regulates the propagation of the optical flow. We typically initialize $\beta = 0.7$.

TABLE II: Mean Square Error (MSE) of pressure predictions (in micro-units).

| Method | 1st Frame | 2nd Frame | 3rd Frame | 4th Frame | 5th Frame | 6th Frame | 7th Frame | 8th Frame |
|---|---|---|---|---|---|---|---|---|
| V-Flow (test) | **3.7** | **21.0** | **67.7** | **160.6** | **342.3** | **639.7** | **1031.4** | **1509.1** |
| V-Flow-Sharp (test) | 23.3 | 109.2 | 280.4 | 500.9 | 780.6 | 1095.5 | 1434.1 | 1781.9 |
| Last (test) | 68.3 | 243.7 | 477.4 | 734.3 | 995.3 | 1251.8 | 1500.1 | 1739.5 |
| V-Flow (train) | **3.7** | **21.1** | **67.0** | **158.8** | **335.9** | **619.6** | **983.1** | **1416.4** |
| V-Flow-Sharp (train) | 23.0 | 107.3 | 276.8 | 494.2 | 771.3 | 1082.1 | 1415.6 | 1757.0 |
| Last (train) | 74.1 | 259.3 | 499.9 | 758.9 | 1018.8 | 1270.4 | 1510.1 | 1737.4 |

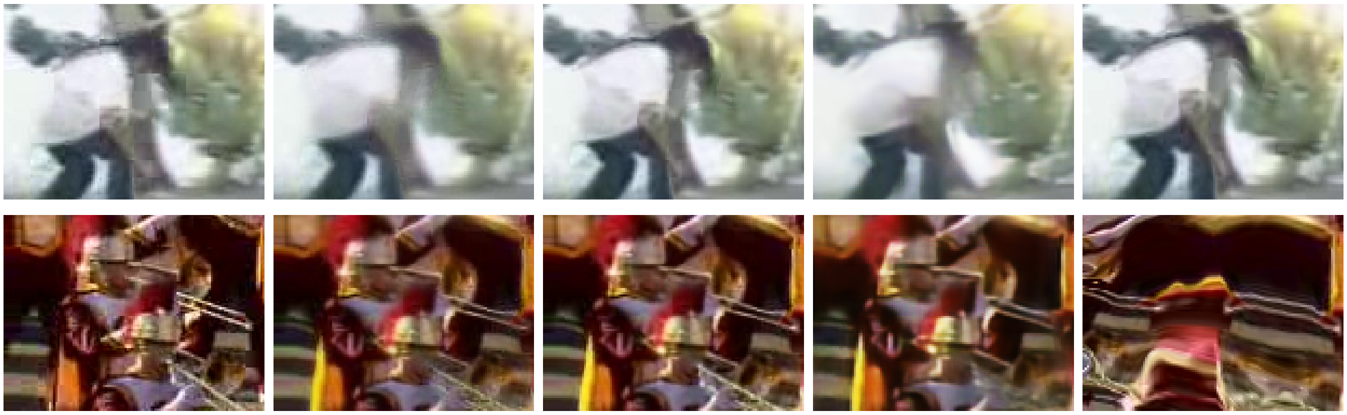


Fig. 5: Comparison of different methods to predict the next frame from UCF101. Left to right: ground truth, V-Flow, V-Flow-Sharp, adversarial, optical flow.

TABLE III: Mean Square Error (MSE) of MRI predictions (in milli-units).

| Method | 1st Frame | 2nd Frame | 3rd Frame |
|---|---|---|---|
| V-Flow (test) | **3.2** | **8.0** | **13.0** |
| V-Flow-Sharp (test) | 3.6 | 9.1 | 14.4 |
| Last (test) | 5.1 | 12.5 | 18.8 |
| V-Flow (train) | **2.9** | **7.3** | **11.9** |
| V-Flow-Sharp (train) | 3.3 | 8.4 | 13.4 |
| Last (train) | 4.7 | 11.5 | 17.4 |

**Latent frame.** The latent frame $\hat{Z}_s$ is determined by linearly combining the last input volumetric frame $V_s^k$ and the auxiliary variable $A_s$ as follows

$$\hat{Z}_s = \alpha V_s^k + A_s.$$

$\alpha$ is a weight that controls the power of the last frame in $\hat{Z}$. We initialize $\alpha = 1.0$.

**Warping module.** The refined optical flow $\hat{F}_s$ drives a 3D optical flow warping module. This module is implemented similarly to the method of Patraucean *et al.* [6] for video prediction. A three dimensional grid generator represents the optical flow as a dense transformation map that maps $\hat{Z}_s$ to $\hat{Y}_s$. A novel trisampler module follows. It uses the map to interpolate $\hat{Z}_s$ linearly, and effectively moves voxels of the estimated latent frame $\hat{Z}_s$ to obtain $\hat{Y}_s$. Hence, we have

$$\begin{aligned}\hat{Y}_s &= FlowWarp(\hat{Z}_s, \hat{F}_s) = \\ &= \text{Trisampler}(\hat{Z}_s, \text{Grid Generator}(\hat{F}_s)).\end{aligned}$$

### C. Training

The model is trained by minimizing the reconstruction error between the predicted next frame and the ground truth next frame. One way is to minimize the $L_p$ distance

$$L_p(\hat{Y}, Y) = \left\| \hat{Y}_s - Y_s \right\|_p^p.$$

In our multi-scale architecture we use $p = 1$.

Another option is to penalize the differences between the prediction gradients and the image gradients. We adapt the gradient difference loss (GDL) of [8] to our volumetric setting. Thus, we define the volumetric gradient loss (VGDL)

$$\begin{aligned}&L_{\text{VGDL}}(\hat{Y}, Y) = \\ &\sum_{x,y,z} \Big( ||Y_{x,y,z} - Y_{x-1,y,z}| - |\hat{Y}_{x,y,z} - \hat{Y}_{x-1,y,z}||+ \\ &\qquad ||Y_{x,y,z} - Y_{x,y-1,z}| - |\hat{Y}_{x,y,z} - \hat{Y}_{x,y-1,z}||+ \\ &\qquad ||Y_{x,y,z} - Y_{x,y,z-1}| - |\hat{Y}_{x,y,z} - \hat{Y}_{x,y,z-1}|| \Big).\end{aligned}$$

The total loss combines the $L_1$ and the $L_{\text{VGDL}}$ loss functions with different weights, and is expressed by

$$\text{Loss}(\hat{Y}_s, Y_s) = \lambda_{L_1} \sum_{s=1}^{S} L_1(\hat{Y}_s, Y_s) + \lambda_{\text{VGDL}} \sum_{s=1}^{S} L_{\text{VGDL}}(\hat{Y}_s, Y_s).$$

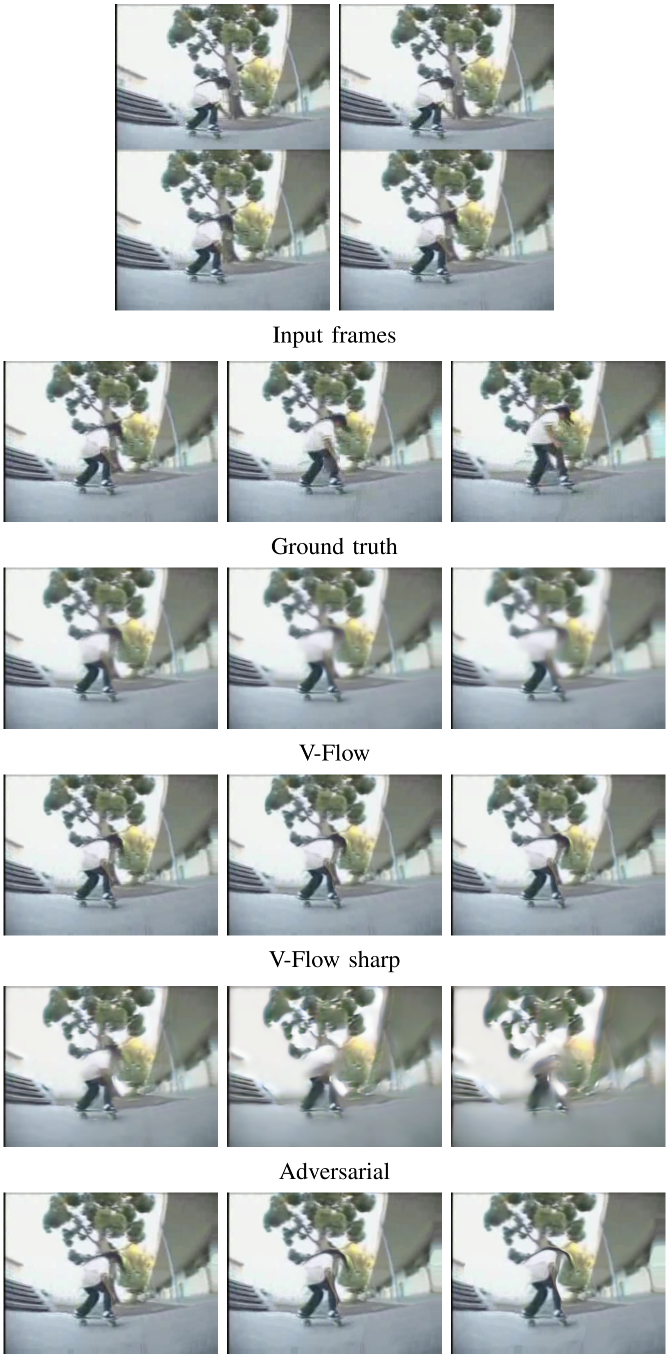
In our experiments the parameters $\lambda_{L_1}$ and $\lambda_{\text{VGDL}}$ are set to 1.0 and 0.5, respectively.

## IV. Experiments

To assess qualitatively and quantitatively the behavior of the proposed architecture and its components, we ran unsupervised

TABLE IV: PSNR, SSIM and sharpness results on UCF101 dataset

| | 1st Frame | | | 2nd Frame | | | 3rd Frame | | |
|---|---|---|---|---|---|---|---|---|---|
| Method | PSNR | SSIM | Sharpness | PSNR | SSIM | Sharpness | PSNR | SSIM | Sharpness |
| V-Flow | **32.10** | **0.93** | **25.65** | **29.06** | **0.91** | **25.10** | **26.47** | **0.87** | **23.95** |
| V-Flow-Sharp | 31.65 | **0.93** | 25.50 | 28.58 | 0.90 | 24.91 | 25.97 | 0.86 | 23.74 |
| Mathieu *et al.* [8] | 31.47 | 0.91 | 25.38 | 27.45 | 0.87 | 24.69 | 24.56 | 0.82 | 23.64 |
| Last | 28.50 | 0.89 | 24.58 | 26.21 | 0.87 | 24.15 | 24.56 | 0.84 | 23.26 |
| EpicFlow *et al.* [17] | 31.97 | **0.93** | 25.57 | 28.46 | 0.90 | 24.80 | 26.16 | **0.87** | 23.76 |



Input frames

Ground truth

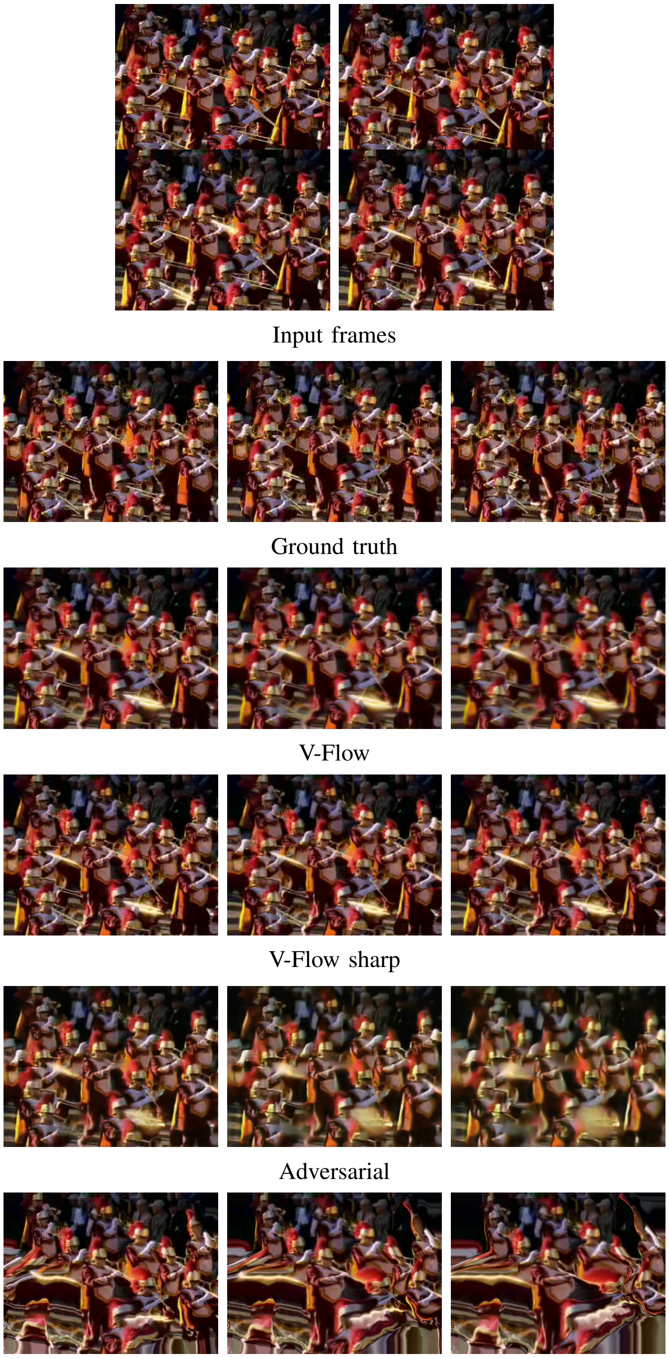
V-Flow

V-Flow sharp

Adversarial

Optical flow

Fig. 6: Comparison of different methods on UCF101.



Input frames

Ground truth

V-Flow

V-Flow sharp

Adversarial

Optical flow

Fig. 7: Comparison of different methods on UCF101.

experiments on synthetic as well as real datasets. In order to predict more than one frame, we apply the model recursively by using the newly generated frame as an input. In all our experiments we used four input frames. Our implementation is based on Torch library [18] and was trained on a TITAN-X nvidia GPU with 12GB memory. The optimization was done using *adagrad* [19]. The spatial and volumetric configuration have 15,808,667 and 16,648,465 trainable parameters, respectively.

### A. Two-fluid Navier-Stokes simulation

Fluid dynamics has a wide range of applications, including calculating forces and moments on aircraft, determining the flow rate of water through pipelines, and predicting weather conditions and ocean currents. The motion of viscous fluid substances is governed by the Navier-Stokes equations, which are a non-linear set of differential equations that describes the flow of a fluid whose stress depends linearly on flow velocity gradients and pressure.

We test our architecture on the homogeneous buoyancy driven turbulence database [20], [21] that simulates the turbulence of incompressible two fluids of different molar masses by solving the Navier-Stokes equations. We note that the numerical solution of the Navier-Stokes equations for turbulent flow is extremely difficult and the computational time in some situations becomes infeasible for calculation.

The database simulation grid is of size $1024^3$ at 1015 time frames. The input to our model is the pressure volume derived from the solution of the Navier-Stokes equations. The pressure does not fully define the temporal evolution of the buoyancy driven turbulence. Nevertheless, we try to predict the future pressure volumes. We randomly extract sequences of 20 time frames and of size $64^3$ from the simulation grid. The frames were normalized so that their values are mapped to the interval $[-1, 1]$. We used 144 sequences for training and 49 for testing.

Figure 4 shows examples of the predicted pressure at six consecutive future time frames using the proposed volumetric optical flow architecture. Table II shows the mean square error for predicting eight future frames. We compare V-Flow to the baseline of the last frame, and to V-Flow-Sharp on both the train and test sets. We see that the model generalizes well. We see that in this case the latent variable is very important, since it can predict the mixing of the two fluids. Hence there is a gap between the performance of V-Flow and of V-Flow-Sharp.

### B. Volumetric MRI

Magnetic Resonance Imaging (MRI) is considered the gold standard test to accurately assess the heart's squeezing ability. Analyzing the heart's motion is important for estimating the amount of blood ejected from the left ventricle with each heartbeat. The 2015 Data Science Bowl (DSB) dataset [22] consists of cardiac MRI images in DICOM format across the cardiac cycle, with a minimum of 8 slices at each time frame. We have extracted the region of interest [23] of size $128 \times 128$ from each slice. We trained the multi-scale architecture on $8 \times 16 \times 16$ patches on 455 training sequences of volumetric data to predict the next volume in the sequence. We tested the model on 174 test sequences. Table III shows the performance of the proposed approach for the testing and training sequences. Again, we see that the algorithm generalizes well. Remark: careful inspection of the results shows that there is no significant flow across slices. This can be explained by the fact that the slice thickness is between 4 to 10 times larger than the pixel spacing within slice.

### C. Video prediction

To quantitatively compare our approach to existing methods, we had to downscale our approach to two dimensions. For training, we used a subset of the Sports1m dataset [24]. All frames were down-sampled to a $240 \times 320$ pixel resolution and normalized. We train our network by randomly selecting temporal sequences of patches of size $64 \times 64$ pixels.

We evaluated the quality of our video predictions on 738 test videos from the UCF101 dataset [25]. We compute the Peak Signal to Noise Ratio (PSNR), the Structural Similarity Index Measure (SSIM) [26] and sharpness of the images as in [8]. As some of the images in the UCF101 dataset do not involve any motion, we use the approach presented in [8] and compute the different quality measures only in the regions where the optical flow is higher than a fixed threshold. In some of the sequences, the last frame predicts the next frame almost perfectly. These sequences are discarded. We evaluate our architecture with and without modifying the last frame. We compare to the baseline last image and to the adversarial learning method of Mathieu *et al.* [8], [27]. We also include the optical flow method that extrapolates the pixels of the next frame by using the optical flow from the last two frames [17], [28]. The results are given in Table IV. We see that V-Flow presents a significant improvement of all measures. Figures 5, 6 and 7 show examples of the next frame prediction on test sequences from the UCF101 dataset. Although V-Flow present the best quantitative results, its output is blurred. By applying the optical flow on the last frame, we get much sharper images which visually appear realistic. The results of the adversarial net are blurry and there are some artifacts. The optical flow images are usually sharp and visually appealing, but in Figure 5 the skate-boarder's head is squeezed and the trumpeters are filtered out

## V. Conclusions

A volumetric optical flow based next frame prediction method has been presented. We defined an underlying optical flow process that is flexible enough to model a range of problems. We used neural networks at each level of the volumetric pyramid to estimate the nature of the optical flow process, and trained the networks in an unsupervised manner.

As an implication of the ideas proposed in this paper, consider the way dynamical systems are currently being analyzed and simulated. Classical theoreticians try to model systems by postulating compact physical equations (usually differential ones) that represent the observed dynamics. Then, experimental researchers often make measurements to validate the theoretical model and assumptions. If the results do not match the model, scientists try to come up with a different model and repeat their evaluations. Our architecture goes the other way around. From the measurements themselves the network captures the governing "equations". Within this scope, validation is based on the networks prediction capabilities, and the networks architecture, which is somewhat flexible, replaces the exact rigid classical model. We think that this is a promising alternative approach that scientists can exploit in order to analyze dynamical systems.

## References

[1] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[2] B. D. Lucas, T. Kanade *et al.*, "An iterative image registration technique with an application to stereo vision," 1981.

[3] M. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra, "Video (language) modeling: a baseline for generative models of natural videos," *arXiv preprint arXiv:1412.6604*, 2014.

[4] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised learning of video representations using lstms." in *ICML*, 2015, pp. 843–852.

[5] W. Lotter, G. Kreiman, and D. Cox, "Deep predictive coding networks for video prediction and unsupervised learning," *arXiv preprint arXiv:1605.08104*, 2016.

[6] V. Patraucean, A. Handa, and R. Cipolla, "Spatio-temporal video autoencoder with differentiable memory," *arXiv preprint arXiv:1511.06309*, 2015.

[7] N. Kalchbrenner, A. v. d. Oord, K. Simonyan, I. Danihelka, O. Vinyals, A. Graves, and K. Kavukcuoglu, "Video pixel networks," *arXiv preprint arXiv:1610.00527*, 2016.

[8] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," *arXiv preprint arXiv:1511.05440*, 2015.

[9] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, "Deepflow: Large displacement optical flow with deep matching," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1385–1392.

[10] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," *arXiv preprint arXiv:1504.06852*, 2015.

[11] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," *arXiv preprint arXiv:1611.00850*, 2016.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[13] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 2017–2025.

[14] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh, "Action-conditional video prediction using deep networks in atari games," in *Advances in Neural Information Processing Systems*, 2015, pp. 2863–2871.

[15] E. L. Denton, S. Chintala, R. Fergus *et al.*, "Deep generative image models using a laplacian pyramid of adversarial networks," in *Advances in neural information processing systems*, 2015, pp. 1486–1494.

[16] P. Burt and E. Adelson, "The laplacian pyramid as a compact image code," *IEEE Transactions on communications*, vol. 31, no. 4, pp. 532–540, 1983.

[17] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Epicflow: Edge-preserving interpolation of correspondences for optical flow," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1164–1172.

[18] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, no. EPFL-CONF-192376, 2011.

[19] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[20] J. Pulido, "Homogeneous buoyancy driven turbulence data set," 2015.

[21] Y. Li, E. Perlman, M. Wan, Y. Yang, C. Meneveau, R. Burns, S. Chen, A. Szalay, and G. Eyink, "A public turbulence database cluster and applications to study lagrangian evolution of velocity increments in turbulence," *Journal of Turbulence*, no. 9, p. N31, 2008.

[22] Kaggle and B. A. Hamilton, "Second annual data science bowl," *Kaggle*, 2015.

[23] Kaggle, A. Newton, and B. A. Hamilton, "Left ventricle segmentation tutorial," 2015.

[24] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[25] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.

[26] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[27] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[28] P. Dollár, "Piotr's Computer Vision Matlab Toolbox (PMT)," https://github.com/pdollar/toolbox.