Skolkovo Institute of Science and Technology

as a manuscript

Denis Volkhonskiy

DEEP GENERATIVE LEARNING FOR IMAGE SEQUENCES

PhD Dissertation Summary

for the purpose of obtaining academic degree
Doctor of Philosophy in Computer Science

Academic Supervisor:
Doctor of Science
Evgeny V. Burnaev

Moscow — 2022

The PhD dissertation was prepared at Skolkovo Institute of Science and Technology.
Academic Supervisor: Evgeny Burnaev, Doctor of Sciences, Full Professor, Head of the Research Center in Artificial Intelligence, Skolkovo Institute of Science and Technology.

1 Introduction

In order to obtain reliable results in many machine learning and data analysis problems including image sequence processing tasks such as video data and three-dimensional voxel data, training datasets of a significant volume are required. However, a sufficient amount of data is not always feasible to get [27; 1]. In such situations, the training datasets can be enriched with synthetic data, artificially generated by a generative model.

The generation of image sequences has a number of other applications like, for example, video content synthesis, which is important in cartoon creation [2] and video production. One can also mention 3D voxel data, which is also an image sequence. 3D voxel texture synthesis application is helpful in *digital rocks physics* [23] for new oil fields development.

In many applications, such as Porous Media generation [23], MRI data augmentation [27], there is an additional requirement for properties of generated data to be similar to those of real data. For example, in porous media modeling, it is important to have similar permeability and porosity properties of generated and real objects. For video generation, it is important to preserve pixel distribution. Even though there are several methods for image-based sequence generation, they are not capable of saving such properties or they are not able to save properties with sufficient quality [23; 28; 27]. Moreover, the existing solutions can generate image sequences of low resolution only [31; 23]. Some generative methods for video frame sequences [5; 31; 22] require up to 500 Tensor Processing Units for training, which is computationally infeasible for most users.

The present dissertation is focused on the problem of properties preserving image sequence generation. As stated above, this topic is of great value to the scientific community.

The **Goal** of this work is to develop generative models for a sequence of images capable of i) property preservation, ii) high resolution objects generation, and iii) computationally feasible. This goal leads to the following objectives:

1. to develop a method for generation of three-dimensional voxel data from two-dimensional slices with preservation of physical characteristics of objects;

2. to develop a method for generation of high-resolution three-dimensional voxel data with preservation of the physical characteristics of objects;

3. to develop a method for generation of sequences of video frames with preservation of the video pixels distribution;

4. to develop a method for removal of the background and artifacts of images, which would allow one to preserve semantically important information.

2 Key results and conclusions

The novelty of this work can be summarized as follows:

1. We proposed a new method for generation of three-dimensional voxel data from a two-dimensional slice. This method is capable of preserving two-dimensional input slices and properties such as physical characteristics of objects.
2. We proposed a new method for generation of high-resolution three-dimensional voxel data, trained only on low-resolution data. Our method outperforms existing methods in terms of quality and preserves physical characteristics of objects.
3. We proposed a new method for generation of a video sequence of images that uses much fewer computational resources than modern methods and has a similar quality. Our method can be trained on 8 Graphical Processing Units (GPUs), whereas the existing methods for video generation require up to 512 Tensor Processing Units (TPUs). Our method also saves the distribution of video pixels.
4. We proposed a new method which removes the background and artifacts from images but preserves the semantically important information of the images. Our method outperforms the existing cleaning methods and allows inpainting holes in semantically important information.

Theoretical and practical significance.

The theoretical significance of the newly developed methods is as follows:

- the new method for generation of three-dimensional voxel data from two-dimensional slices with preservation of physical characteristics of objects;
- the new method for generation of high-resolution three-dimensional voxel data with preservation of physical characteristics of objects;
- the new method for generation of sequences of video frames with preservation of video pixels distribution;
- the new method for removing the background and artifacts of images, with preservation of semantically important information.

These methods can be used by scientists and developers when creating new methods in generative modeling and solving new applied problems of image processing and analysis.

The practical significance was verified by application of our methods to porous media generation for speeding-up oil field development. Other applications, which are also demonstrated in this work, are video generation and raster-scan cleaning. The potential applications also include generation of medical, geological, or seismic data for data augmentation and analysis.

Key aspects/ideas to be defended.

1. The new method for generation of three-dimensional voxel data from a two-dimensional slice;
2. The new method for generation of high-resolution three-dimensional voxel data;
3. The new method for video generation with reduced computational requirements;
4. The new method for removing the background and artifacts from images.

Personal contribution. The author of the present dissertation obtained all the stated results. In all cases mentioned, both text and experimental results presented in the papers are the results of collaboration between authors.

Result 1 was developed in "Generative adversarial networks for reconstruction of three-dimensional porous media from two-dimensional slices", the author designed and implemented the training algorithm and conducted all the experiments.

Result 2 was developed in "User-Controllable Multi-Texture Synthesis with Generative Adversarial Networks", the author developed the method for three-dimensional data and conducted all the 3D experiments.

Result 3 was developed in "Latent Video Transformer", the author implemented the Transformer part of the algorithm and contributed to the experiments.

Result 4 was developed in "Deep Vectorization of Technical Drawings", the author developed the method for cleaning raster-scan images and contributed to the cleaning experiments.

3 Publications and approbation of research

First-tier publications

1. *Denis Volkhonskiy, Oleg Sudakov, Ekaterina Muravleva, Denis Orlov, Boris Belozerov, Evgeny Burnaev, Dmitry Koroteev* Generative adversarial networks for reconstruction of three-dimensional porous media from two-dimensional slices. Physical Review E, Q2 Journal. Indexed by SCOPUS, Web of Science.

2. *Egiazarian Vage*, Oleg Voynov*, Alexey Artemov, Denis Volkhonskiy, Aleksandr Safin, Maria Taktasheva, Denis Zorin, Evgeny Burnaev* Deep Vectorization of Technical Drawings. ECCV 2020, CORE A. Indexed by SCOPUS.

Second-tier publications

1. *Rakhimov Ruslan*, Denis Volkhonskiy*, Alexey Artemov, Denis Zorin, Evgeny Burnaev* Latent Video Transformer. VISAPP 2021, CORE B. Indexed by SCOPUS.

∗ — Equal Contribution

2. *Aibek Alanov*, Max Kochurov*, Denis Volkhonskiy, Daniil Yashkov, Evgeny Burnaev, Dmitry Vetrov* User-Controllable Multi-Texture Synthesis with Generative Adversarial Networks. VISAPP 2020, CORE B. Indexed by SCOPUS.

Reports at conferences and seminars

1. "Latent Video Transformer" talk at the *VISAPP* conference, Online, 2021;
2. "Steganographic generative adversarial networks" talk at the *ICMV* conference, Amsterdam, 2019;
3. "Reconstruction of 3D Porous Media from 2D Slices" talk at the *Multiscale methods and high performance scientific computing* conference, Moscow, 2018;
4. "Inductive Venn-Abers Predictive Distribution" talk at the *COPA 2018* conference, Maastricht, 2018;
5. "Inductive Conformal Martingales for Change-Point Detection" talk at the *COPA 2017* conference, Stockholm, 2017.
6. Poster presentation, Skoltech & MIT Conference: Shaping the Future: Big Data, Biomedicine and Frontier Technologies, Russia, 2017

Patents

1. *Denis Volkhonskiy, Oleg Sudakov, Dmitry Koroteev, Ekaterina Muravleva, Evgeny Burnaev, Leyla Ismailova, Denis Orlov* System for recovery of rock sample three-dimensional structure. RU2718409C1

The author has also contributed to the following publications

1. *Denis Volkhonskiy, Ivan Nazarov, and Evgeny Burnaev* Steganographic generative adversarial networks. ICMV 2019, CORE C. Indexed by SCOPUS.

2. *Denis Orlov, Mohammad Ebadi, Ekaterina Muravleva, Denis Volkhonskiy, Andrei Erofeev, Evgeny Savenkov, Vladislav Balashov, Boris Belozerov, Vladislav Krutko, Ivan Yakimchuk, Nikolay Evseev, Dmitry Koroteev* Different methods of permeability calculation in digital twins of tight sandstones. Journal of Natural Gas Science and Engineering, Q1. Indexed by SCOPUS.

3. *Denis Volkhonskiy, Evgeny Burnaev, Ilia Nouretdinov, Alexander Gammerman, and Vladimir Vovk* Inductive conformal martingales for change-point detection. In Conformal and Probabilistic Prediction and Applications. PMLR 2017. Indexed by SCOPUS.

4. *Ilia Nouretdinov, Denis Volkhonskiy, Pitt Lim, Paolo Toccaceli, and Alexander Gammerman* Inductive venn-abers predictive distribution. PMLR 2018. Indexed by SCOPUS.

$*$ — Equal Contribution

4 Contents

The dissertation topic is disclosed in the following chapters, each chapter summarizes the corresponding paper.

In 4.1, we proposed a new method for generation of three-dimensional voxel data from two-dimensional slices, with preservation of physical properties. In 4.2, we proposed a new method for the generation of three-dimensional voxel data of high resolution with preservation of physical properties. In 4.3, we proposed a new method for video generation, which utilizes much less computational resources than modern methods while having a similar quality. In 4.4, we proposed a new method for image cleaning, with preservation of semantically important information.

4.1 Generative adversarial networks for reconstruction of three-dimensional porous media from two-dimensional slices

Many branches of earth sciences require the solution of the rock study problem on the micro-level. However, this requires a significant number of representative samples which are not always feasible. Obtaining new rock samples is complicated: developing a rock deposit with a subsequent digitization process is required. This makes relevant the problem of generation of samples with similar properties. Synthetic samples could be used together with real samples for digital rock study.

One of the most promising techniques for data generation is based on generative adversarial networks (GANs) [13]. GANs learn complex probability distributions directly from samples. The first paper on the use of GANs in the context of 3D porous media generation [23] considered the task of synthetic images generation. But no additional information (such as 2D slices) was used in the generation step.

In the present work, we propose a new GANs-based deep neural network architecture capable of efficiently generating 3D structures a slice of the original image as an input. We achieve this by introducing an auto-encoder module into the deep neural network architecture.

Model Description

The goal of our method is to generate 3D porous media $\hat{x}$ of size $h \times w \times d$ from a given 2D input slice s of size $h \times w$. We require that the central slice in the $\hat{x}$ is close to s with respect to the Euclidean distance.

Our model consists of the following neural networks:

- The *encoder* $E_\tau(s)$ with parameters τ. It transforms the input slice 2D s to a vector representation h;
- The *generator* $G_\theta(h, z)$ with parameters θ. It transforms the input noise vector z and the encoded slice h to a 3D image x;

- The *discriminator* $D_\phi(x)$ with parameters ϕ. It predicts the class of input 3D image x. This is a standard GANs discriminator.

To obtain the central slice from the 3D image, we introduce a mask $\mathbf{M}$. This is a function that takes a 3D image as an input and returns its central slice. All three neural networks are trained using the combination of Euclidean slice loss (1) and adversarial loss (2). Our model is presented in Fig. 1.

$$L(s) = \| s - \mathbf{M} \odot G_\theta(E_\tau(s), z) \|_2^2 \rightarrow \min_{\tau,\theta} \tag{1}$$

$$L(D_\phi, G_\theta) = \mathbb{E}_{x \sim p_{data}(x)}[\log D_\phi(x)] + \mathbb{E}_{z \sim p_{noise}(z)}[\log(1 - D_\phi(G_\theta(E_\tau(s), z)))] \rightarrow \min_\theta \max_\phi . \tag{2}$$

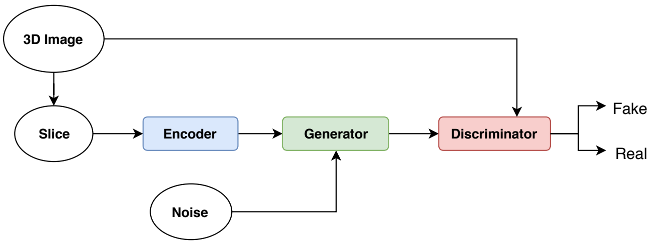

Figure 1: Architecture of the proposed method. It takes as an input a 2D porous media slice and a vector of noise. As an output, generator produces 3D porous media. The discriminator is responsible for the adversarial loss.

Empirical Evaluation

In order to evaluate our method, we compare our generated 3D samples (Fig. 2) with real 3D samples from the dataset. For this purpose, in our paper, we show that the distribution of porosity, permeability, Minkowski functionals [20], and two-point correlation functions [3] of real samples are close to the distribution of these metrics on generated samples.

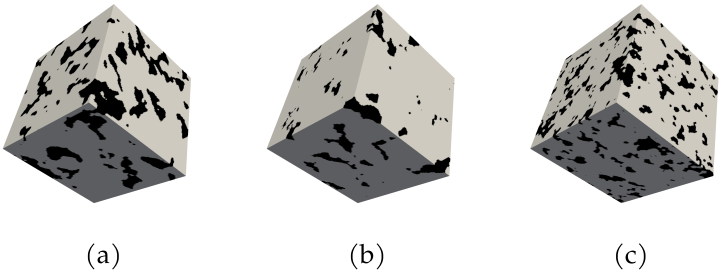

Figure 2: Generated 3D samples of three different types: Berea (a), Ketton (b), South-Russian sandstone (c)

Table 1: Comparison of the Kullback–Leibler divergence between the distribution of porosity and permeability of real, baseline [23] and our samples.

Sample name	Characteristic	$KL(p_{real}, p_{baseline}) \downarrow$	$KL(p_{real}, p_{ours}) \downarrow$
Berea	porosity	0.4974	**0.2667**
Ketton	porosity	0.5251	**0.1425**
Berea	permeability	1.0035	**0.2487**
Ketton	permeability	0.2301	**0.0646**

We also compare our method with the baseline [23]. We use the Kullback–Leibler (KL) divergence between real and baseline distributions $KL(p_{real}, p_{baseline})$, and between real and our distributions $KL(p_{real}, p_{ours})$. The results of comparison of the KL divergence are presented in Table 1. The distribution of both porosity and permeability of our samples is closer to the real porosity distribution than that provided by the baseline.

Conclusion

We proposed a new method for generation of three-dimensional porous media from two-dimensional central slices. Our method is shown to outperform the baseline. Moreover, it generates cubes preserving properties of real cubes. Potentially, a model of such kind can be applied to seismic and geological data [24; 6]. Another possible application is texture generation.

4.2 User-Controllable Multi-Texture Synthesis with Generative Adversarial Networks

In order to obtain reliable results in analyzing properties of generated image sequences, one should be able to generate them with high resolution. However, the existing models, when used for high-resolution objects, face two issues [23; 30]. First, they cannot fit into the computational memory. Second, there is not enough high-resolution data available for training. Thus the development of generative models for high-resolution image sequences is an important problem.

In the present work, we develop a new model for image sequence generation. Our model can generate high-resolution images while trained on low-resolution images only. In order to achieve such an effect, we propose a new conditional generation setup. This approach allows us to use datasets with image sequences of different classes, which increases the dataset size and leads to a better generation quality.

Model Description

We look for a multi-texture synthesis pipeline that can generate image sequence objects in a conditional manner, ensure full dataset coverage, and is scalable with respect to dataset size. We use an encoder network $E_\varphi(x)$, which maps objects to a latent space and gives low-dimensional representations. A generator $G_\theta(z)$ is used to generate samples from the latent space.

We use three types of adversarial losses:

- the *generator matching loss* (3) $\mathcal{L}_x$ for matching the distribution of both samples $G_\theta(z)$ and reproductions $G_\theta(z_\varphi)$ to the distribution of real objects $p^*(x)$:

$$\mathcal{L}_x(\theta) = -\frac{1}{st}\sum_{i,j}^{s,t}\Big[\mathbb{E}_{p(z)}\log D_\psi^{ij}(G_\theta(z)) + \mathbb{E}_{q_\varphi(z)}\log D_\psi^{ij}(G_\theta(z_\varphi))\Big] \to \min_\theta \tag{3}$$

- the *pair matching loss* (4) $\mathcal{L}_{xx}$ for matching the distribution of pairs (x, x') to that of pairs $(x, G_\theta(z_\varphi(x)))$, where x and x' are samples of the same object class. It will ensure that $G_\theta(z_\varphi(x))$ has the same object class as x. For a detailed structure of pair matching Discriminator, see Fig. 4:

$$\mathcal{L}_{xx}(\theta,\varphi) = -\frac{1}{pq}\sum_{i,j}^{p,q}\mathbb{E}_{p_{\theta,\varphi}(x,y)}\log D_\tau^{ij}(x,y) \to \min_{\theta,\varphi} \tag{4}$$

- the *encoder matching loss* (5) $\mathcal{L}_z$ for matching the aggregated distribution $q_\varphi(z)$ to the prior distribution $p(z)$:

$$\mathcal{L}_z(\varphi) = -\mathbb{E}_{q_\varphi(z)}\log D_\zeta(z) \to \min_\varphi \tag{5}$$

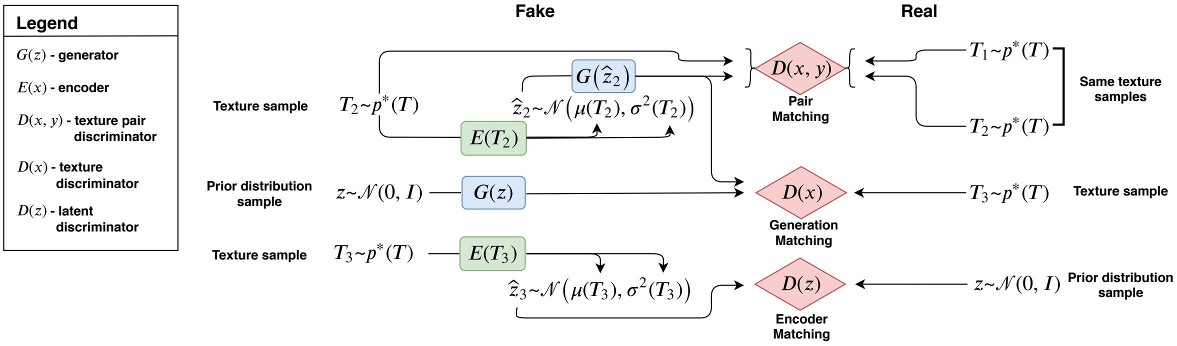
Legend
$G(z)$ - generator
$E(x)$ - encoder
$D(x, y)$ - texture pair discriminator
$D(x)$ - texture discriminator
$D(z)$ - latent discriminator
Fake
Real
Texture sample
$T_2 \sim p^*(T)$
$G(\hat{z}_2)$
$\hat{z}_2 \sim \mathcal{N}(\mu(T_2), \sigma^2(T_2))$
$E(T_2)$
$D(x, y)$
Pair Matching
$T_1 \sim p^*(T)$
$T_2 \sim p^*(T)$
Same texture samples
Prior distribution sample
$z \sim \mathcal{N}(0, I)$
$G(z)$
$D(x)$
Generation Matching
$T_3 \sim p^*(T)$
Texture sample
Texture sample
$T_3 \sim p^*(T)$
$E(T_3)$
$\hat{z}_3 \sim \mathcal{N}(\mu(T_3), \sigma^2(T_3))$
$D(z)$
Encoder Matching
$z \sim \mathcal{N}(0, I)$
Prior distribution sample

Figure 3: Training pipeline of the proposed method.

For both the generator G_θ and the encoder E_φ we optimize the following objectives:

- the generator G_θ loss

$$\mathcal{L}(\theta) = \alpha_1\mathcal{L}_x(\theta) + \alpha_2\mathcal{L}_{xx}(\theta,\varphi) \to \min_\theta \tag{6}$$

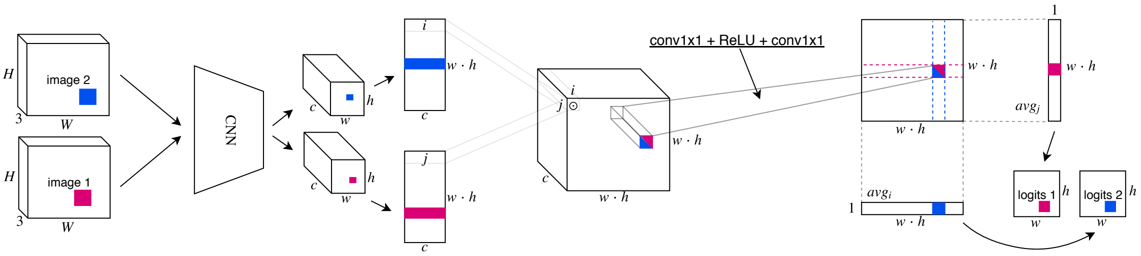

Figure 4: The architecture of the discriminator on pairs $D_\tau(x, y)$.

- the encoder E_φ loss

$$\mathcal{L}(\varphi) = \beta_1 \mathcal{L}_z(\varphi) + \beta_2 \mathcal{L}_{xx}(\theta, \varphi) \to \min_{\varphi} \tag{7}$$

The total pipeline of our model is presented in Fig. 3.

Empirical Evaluation

In this section, we demonstrate the applicability of our model to digital rock physics. We trained our model on 3D Porous Media structures[1] (i.e., see Fig. 5a) of five different types: Ketton, Berea, Doddington, Estaillades and Bentheimer. Each type of rock has an initial size of 1000^3 binary voxels. As a baseline, we considered Porous Media GANs [23], which are deep convolutional GANs with 3D convolutional layers.

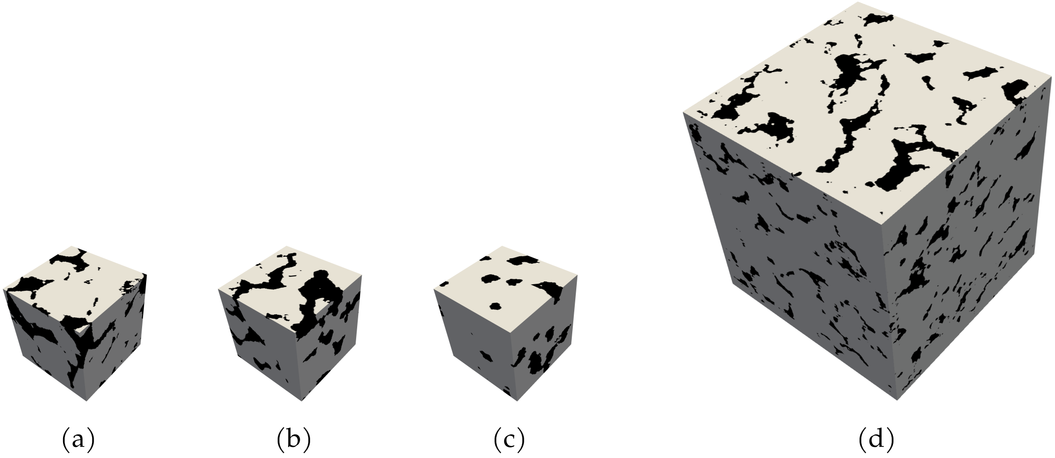
(a) (b) (c) (d)

Figure 5: Berea sample. Real (a, size 150^3), sampled with our model (b, size 150^3), samples with the baseline model (c, size 150^3), sampled with our model (d, size 428^3)

For comparison of our model with real samples and baseline samples, we use permeability statistics and two Minkowski functionals [20]. The permeability is a measure of the ability of

[1] All samples were taken from Imperial College database

a porous material to allow fluids to pass through it. Minkowski functionals describe the morphology and topology of 3D binary structures. In our experiments, we used two functionals: the Surface area and the Euler characteristic. If the considered measures on synthetic samples are close to those on real ones, it will guarantee that the synthetic samples are valid for Digital Rock Physics applications.

We used the following experimental setup. We trained our model on random crops of size 160^3 on all types of porous structures. We also trained five baseline models on each type separately. Then we generated 500 synthetic samples of size 160^3 of each type using our model and the baseline model. We also cropped 500 samples of size 160^3 from the real data. As a result, for each type of structure, we obtained three sets of objects: real, synthetic and baseline.

The visual result of the synthesis is presented in Fig. 5 for Berea sandstone. In the figure, there are three samples: real ones (i.e., cropped from the original big sample), ours, and a sample of the baseline model [23]. Since our model is fully convolutional, we can increase the generated sample size by expanding the spatial dimensions of the latent embedding z on inference without retraining the neural network. This is feasible because we know the latent distribution and can sample from it using the random number generator. We demonstrate the synthesized 3D porous media of size 428^3 in Figure 5d. Then,

1. For each real, synthetic and baseline objects we calculated three statistics: permeability, Surface Area and Euler characteristics.

2. To measure the distance between distributions of statistics for real, our and baseline samples we approximated these distributions by discrete ones obtained by using the histogram method with 50 bins.

3. For each statistic, we calculated the KL divergence between the distributions of the statistic of a) real and our generated samples; b) real and baseline generated samples.

A comparison of the KL divergences for the permeability is presented in Table 2. As we can see, our model performs better for most types of porous structures.

Table 2: KL divergence between real, our, and the baseline distributions of permeability for size 160^3. Standard deviation was computed using the bootstrap method with 1000 resamples.

	$KL(p_{real}, p_{ours})$	$KL(p_{real}, p_{baseline})$
Ketton	5.06 ± 0.35	4.68 ± 0.56
Berea	0.49 ± 0.07	0.50 ± 0.12
Doddington	**0.42 ± 0.10**	3.41 ± 1.68
Estaillades	**0.80 ± 0.24**	3.41 ± 0.46
Bentheimer	**0.47 ± 0.08**	1.38 ± 0.49

Conclusion

In this chapter, we proposed a novel model for 3D voxel data synthesis. We showed its applicability to digital rock physics. Our model is capable of generating samples of high-resolution while trained on low-resolution samples only. Moreover, it outperforms the baseline in most cases which verifies its usefulness for real-world problems.

4.3 Latent Video Transformer

Video prediction and generation is an important problem with a lot of down-stream applications: self-driving, anomaly detection, timelapse generation [25], animating landscape [11] etc. The task is to generate the most probable future frames given several initial ones.

Recent advances in generative learning make it possible to generate realistic objects with high quality: images, text, and speech. However, video generation is still a very challenging task. Even for short videos (16 frames) of low resolution, neural networks require up to 512 Tensor Processing Units (TPUs) [22] for parallel training. Despite this, the quality of the generated video remains low.

In the present work, we introduce a Latent Video Transformer. We combine the idea of representation learning and recurrent video generation. Instead of working in the pixel space, we conduct the generation process in the latent space. Our model tends to significantly relax the computational requirements without significant deterioration in quality.

The key novelty in our model is the usage of a discrete latent space [29]. It allows us to represent each frame as a set of indices. Thanks to discrete representation, we can use autoregressive generative models and other approaches from natural language processing.

Method

Consider a video X to be a sequence of T frames $\{x_t\}_{t=1}^{T}$. Each frame $x_t \in \mathbb{R}^{H \times W \times 3}$ has height H, width W and 3 RGB channels. Given the first T_0 frames, the goal is to generate the remaining $T - T_0$ frames.

We use the frame autoencoder to learn a compact latent representation so that we can transfer the task of video modeling from the pixel space to the discrete latent space. The recurrent model is then used for generation of new frames. We train the frame autoencoder to transfer individual images (frames) to the latent space. The particular choice of the autoencoder is VQ-VAE [29] — a variational autoencoder with discrete latent space.

The autoencoder (see Fig. 6) learns to encode an input image $x \in \mathbb{R}^{H \times W \times 3}$ using a codebook $e \in \mathbb{R}^{K \times D}$, where K denotes the codebook size (i.e., the latent space is K-way categorical) and D represents the size of an embedding in the codebook.

Our autoencoder consists of an *encoder*, which encodes the image into more compact representation $E(x) = z_e(x) \in \mathbb{R}^{h \times w \times D}$; *a bottleneck*, that discretizes each pixel by mapping it to its nearest embedding e_i from the codebook and produces $z(x) \in [K]^{h \times w \times 1}$; *a decoder* D receives discrete latent codes $z(x)$, maps the indexes to corresponding embeddings, and decodes the result of the mapping $z_q(x) \in \mathbb{R}^{h \times w \times D}$ back to the input pixel space. The autoencoder is trained with the following objective:

$$L = \|x - D(z_q(x))\|^2 + \|z_e(x) - \text{sg}[e]\|^2, \tag{8}$$

where sg$[\cdot]$ is the stop gradient operator, which returns its argument during the forward pass and zero gradients during the backward pass. The first term is a reconstruction loss, and the second term is a regularization term to make the encodings less volatile. We use exponential moving average updates over the codebook variables.

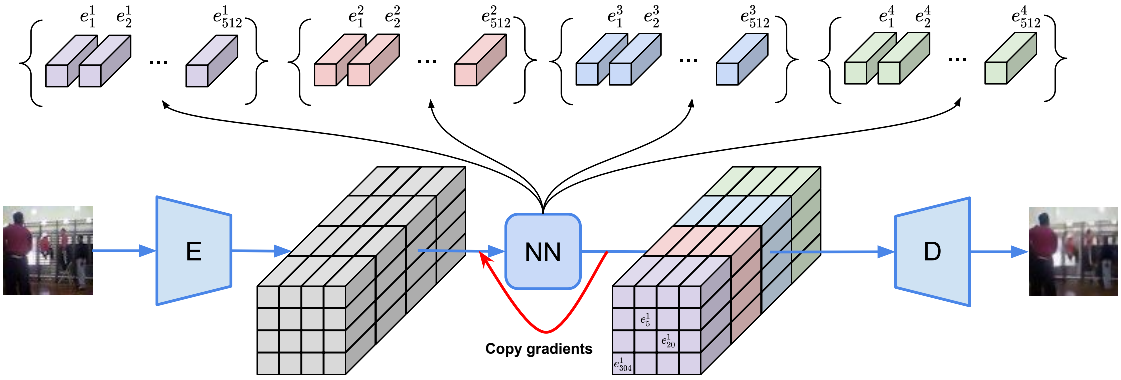
e_1^1 e_2^1 ... e_{512}^1
e_1^2 e_2^2 ... e_{512}^2
e_1^3 e_2^3 ... e_{512}^3
e_1^4 e_2^4 ... e_{512}^4
E
NN
D
Copy gradients
e_5^1
e_{20}^1
e_{304}^1

Figure 6: Frame autoencoder architecture. An input image is passed through the encoder and split along the channel dimension into $n_c = 4$ parts. Then we map pixels in each part to the corresponding nearest embeddings in the codebook. These nearest embeddings are then passed as an input to the decoder.

The frame encoder transforms the first T_0 frames to a discrete representation $Z_0 \in [K]^{T_0 \times h \times w \times n_c}$. The autoregressive model is used to generate new $T - T_0$ frames conditioned on Z_0. As such model, we use the Video Transformer [31], an autoregressive video generative model, but apply it in the latent space in contrast to the pixel space in the original paper. Next, we describe the architecture of a video transformer. We refer to a latent representation of a video as a *latent video* and individual elements of it as *latent frames* and *pixels*.

The model takes as input a tensor $Z \in [K]^{T \times h \times w \times n_c}$ and primes the generation process on first T_0 given latent frames, i.e., $Z_{:T_0,:,:,:} = Z_0$. The generation process proceeds from a slice to a slice, from a pixel to a pixel inside one slice, and from a channel to a channel for one pixel:

$$p(Z) = \prod_{i=0}^{Thw-1} \prod_{k=0}^{n_c-1} p\left(Z_{\pi(i)}^k | Z_{\pi(<i)}, Z_{\pi(i)}^{<k}\right) \tag{9}$$

The model consists of an encoder and a decoder. To generate a new pixel value inside a slice $Z_{(a,b,c)}$, the encoder outputs the representation of already generated slices $Z_{<(a,b,c)}$. This representation goes to the decoder, which mixes it with a representation of already generated pixels

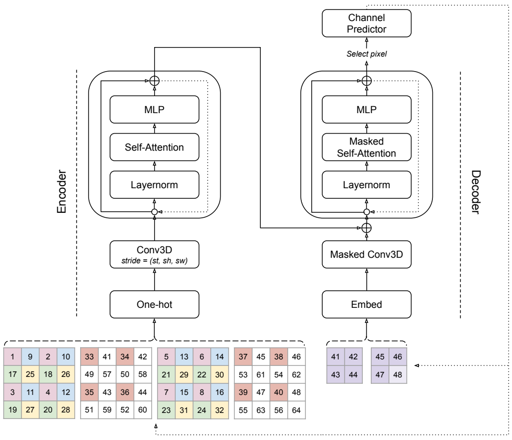

Figure 7: Video Transformer adapted to latent codes. Numbers represent generation order. Pixels are colored if they are already generated. White-colored pixels are zero-padded. Pixels with the same color belong to the same slice. The example represents the generation of the last pixel of slice $Z_{(1,0,1)}$ for a latent video of size $(t, h, w) = (4, 4, 4)$ and $(s_t, s_h, s_w) = (2, 2, 2)$.

inside the current slice $Z_{(a,b,c)}$. This autoregressive order is preserved by padding input latent video inside the encoder, and masking used in convolutions and attention inside the decoder. After generating a new pixel value, we replace the respective padding with the generated output and repeat the generation process recursively. The generation process for a spatiotemporal $(s_t > 0, s_h > 0, s_w > 0)$ subscaling is illustrated in Fig. 7.

Finally, when the generation process is done, the latent frame decoder takes as an input $Z \in [K]^{T \times h \times w \times n_c}$ (now all values are valid), maps it to the already learned embeddings $Z_q \in \mathbb{R}^{T \times h \times w \times D}$ and decodes it back frame by frame to an original pixel space $X \in \mathbb{R}^{T \times H \times W \times 3}$.

Empirical Results

We model the videos of length $T = 16$ and spatial size 64×64 similarly to the setup of prior works in this field [5; 31]. We compare the video generation quality on the BAIR Robot Pushing [9] dataset and the Kinetics-600 [4] dataset.

As main measures of quality, we use the Fréchet Video Distance (FVD) [14] and the bits per dimension (bits/dim), which is the negative $\log_2$-probability averaged across all generated (latent) pixels and channels.

We report the FVD and bits/dim for the BAIR Robot Pushing dataset in Table 3 and for the Kinetics-600 dataset in Table 4. We also provide samples from our model for qualitative assessment (see Fig. 8).

We achieve comparable performance in comparison with other methods for the BAIR robot pushing dataset. Our results are inferior to others on Kinetics-600. We conclude that it is caused by error accumulation inside the Transformer model. We link it to the high complexity and diversity of the Kinetics-600 dataset.

Table 3: Comparison of different methods for video prediction on BAIR Robot Pushing dataset.

Method	**bits/dim**($\downarrow$)	**FVD**($\downarrow$)
Baseline	-	320.90
VideoFlow [18]	1.87	-
SVP-FP [7]	-	315.5
CDNA [12]	-	296.5
LVT (ours, $n_c = 1$)	1.25	275.71 ± 5.41
SV2P [8]	-	262.5
LVT (ours, $n_c = 4$)	1.53	125.8 ± 2.9
SAVP [19]	-	116.4
DVD-GAN-FP [5]	-	109.8
TriVD-GAN-FP [22]	-	103.3
Axial Transformer [15]	1.29	-
Video Transformer [31]	1.35	94 ± 2

Table 4: Comparison of different methods for video prediction on Kinetics-600 dataset.

Method	**Bits/dim**($\downarrow$)	**FVD**($\downarrow$)
Baseline	-	271.00
LVT (ours)	2.14	224.73
Video Transformer [31]	1.19	170 ± 5
DVD-GAN-FP [5]	-	69.15 ± 1.16
TriVD-GAN-FP [22]	-	25.74 ± 0.66

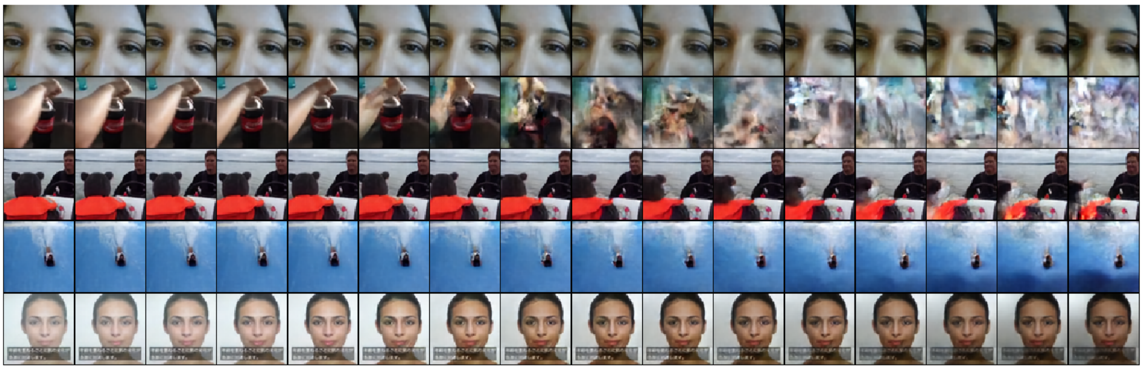

Figure 8: Samples from the Kinetics-600 dataset. Each row represents a single video with first five frames being real and others generated.

Conclusion

In this chapter, we tackled the video generation problem. Given several first frames, the goal was to predict the continuation of a video. Modern methods for video generation require up to 512 Tensor Processing Units for parallel training. We were focused on the reduction of the computational requirements for model training. We showed that one could achieve comparable results on video prediction by training a model using the usual research setup — 8 V100 GPUs. We demonstrated decent results on the BAIR Robot Pushing dataset. In the meantime, in some cases, we observe visual artifacts on the Kinetics-600 dataset.

4.4 Deep Vectorization of Technical Drawings

Vector images have a number of advantages over raster images. They include lossless upscaling and small representation size. Furthermore, vector images allow one to independently modify its primitives. However, many technical images are available in raster format only. Thus, it is important to be able to vectorize them.

In [10] the whole new vectorization pipeline was proposed. It consists of the following steps. First, the original raster image is cleaned from noise and artifacts (*cleaning* part). Then the semantically important parts are approximated with primitives. Finally, the number of primitives is minimized where possible.

In the present dissertation, we are mainly focused on the *cleaning* part. The goal of the present work is to propose a method, that would transform a raster image with artifacts and missing parts of line segments into a clean raster image with infilled holes. To be more precise, the goal of the cleaning part is to solve the following problems:

- clean the input raster image from artifacts;
- make the background to be white if its color is different (i.e., gray or brown);
- detect and inpaint holes in lines if they exist.

Method

The goal of the cleaning step is to convert the raw input data into a raster image with clear line structure by eliminating noise, infilling missing parts of lines, and setting all background/non-ink areas to white. This task can be viewed as semantic image segmentation in that the pixels are assigned the background or foreground class.

Inspired by [21; 28], our cleaning network has the U-NET [26] structure and consists of two parts: downsampling and upsampling. The Downsampling part is a sequence of convolutional layers with non-linearities between them. The Upsampling part utilize transposed convolutions, which increases the size of the input image. After each convolutional and transposed convolutional layer, we use Batch Normalization [16] and Rectified Linear Units layers. The distinguishing feature of our architecture is skip-connections, which connect layers from the downsampling part to layers of the upsampling part. The full model structure is presented in Fig. 9.

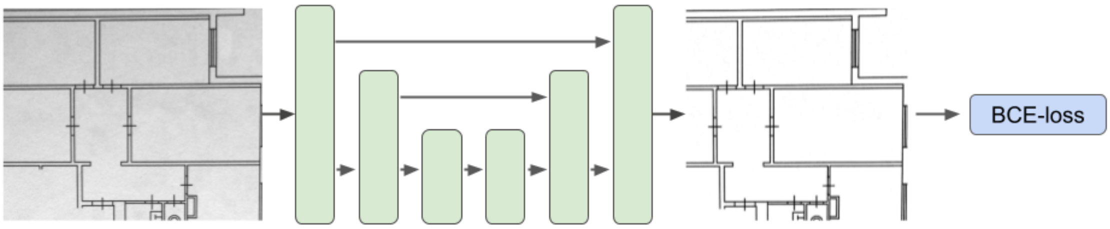

Figure 9: Cleaning pipeline. Input raster-scan image is processed via our model. The output of the model is a binary image. Pixel-wise binary cross-entropy is used as a loss function.

	IoU,%	PSNR
MS [28]	49	15.7
Our	**92**	**25.5**

Table 5: Quantitative evaluation of the preprocessing step.

In order to train such a neural network, we minimize the binary cross-entropy loss using Adam [17] algorithm. For a pixel $x_{i,j}$ at a location i, j and a ground truth value at the same location $y_{i,j}$ the loss for this particular pixel is computed as $l_{i,j} = y_{i,j} \cdot \log x_{i_j} + (1 - y_{i,j}) \cdot \log(1 - x_{i_j})$. The full loss for the image with height H and width W is the average of all pixel losses:

$$L = \frac{1}{H \cdot W} \sum_{i=1}^{H} \sum_{j=1}^{W} l_{i,j}.$$

Empirical Evaluation

We evaluate our cleaning network by comparing it with public pre-trained implementation of Mastering Sketching (MS, [28]). We show the quantitative results of this evaluation in Table 5 and the qualitative results in Figure 10.

Our preprocessing network keeps straight and repeated lines commonly found in technical drawings, while MS produces wavy strokes and tends to join repeated straight lines, thereby harming the structure of the drawing.

Conclusion

In this chapter, we propose a method for cleaning raster images from artifacts and removing holes in primitives. This method is a part of the whole vectorization pipeline in [10]. It allows one to increase further the vectorization quality and reduce the number of final vectorized primitives.

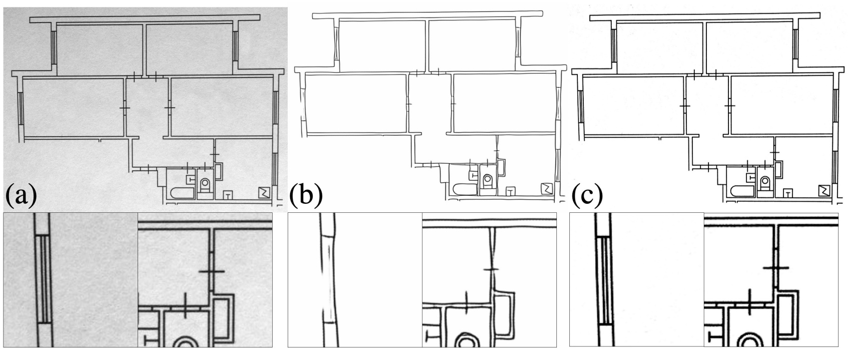

Figure 10: Example of preprocessing results: (a) raw input image, (b) output of MS [28], (c) output of our preprocessing network. Note the tendency of MS to combine close parallel lines.

5 Conclusion

In the final section, we summarize the main contributions of the work.

1. We developed a new method for generation of three-dimensional voxel data from a two-dimensional slice. We applied this method to porous media generation. With numerous numerical experiments with porous media, we demonstrated the robustness of the proposed model. We showed that our approach preserves physical characteristics such as porosity, permeability, and the two-point correlation function better than existing methods. Our model generates 3D voxel data of good visual quality.

2. We developed a method for the generation of three-dimensional voxel data, which is capable of generating samples of a high resolution being trained on low-resolution images only. With numerical experiments on porous media data, we demonstrated its ability to preserve physical characteristics better than the existing methods.

3. We developed a new video generation method, that reduces computational requirements from state-of-the-art 512 Tensor Processing Units to 8 Graphical Processing Units with preservation of similar quality.

4. We developed a generative model for image cleaning. It is capable of removing background and artifacts from an image with preservation of semantically important information. Our method outperforms the existing methods and inpaints missing holes in the image. This method was developed as a part of the raster-scan vectorization pipeline.

References

[1] Antreas Antoniou, Amos Storkey, and Harrison Edwards. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017.

[2] Jihye Back. Fine-tuning stylegan2 for cartoon face generation. *arXiv preprint arXiv:2106.12445*, 2021.

[3] Y. Bazaikin, V. Baikov, I. Taimanov, and A. Yakovlev. Numerical analysis of topological characteristics of three-dimensional geological models of oil and gas fields. *Matematicheskoe Modelirovanie*, 25(10):19–31, 2013.

[4] Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600. *arXiv preprint arXiv:1808.01340*, 2018.

[5] A Clark, J Donahue, and K Simonyan. Adversarial video generation on complex datasets. *arXiv preprint arXiv:1907.06571*, 2019.

[6] Guillaume Coiffier, Philippe Renard, and Sylvain Lefebvre. 3d geological image synthesis from 2d examples using generative adversarial networks. *Frontiers in Water*, 2:30, 2020.

[7] Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. *arXiv preprint arXiv:1802.07687*, 2018.

[8] Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. *arXiv preprint arXiv:1802.07687*, 2018.

[9] Frederik Ebert, Chelsea Finn, Alex X Lee, and Sergey Levine. Self-supervised visual planning with temporal skip connections. *arXiv preprint arXiv:1710.05268*, 2017.

[10] Vage Egiazarian*, Oleg Voynov*, Alexey Artemov, Denis Volkhonskiy, Aleksandr Safin, Maria Taktasheva, Denis Zorin, and Evgeny Burnaev. Deep vectorization of technical drawings. In *ECCV 2020: European Conference on Computer Vision*, pages 582–598. Springer, 2020.

[11] Yuki Endo, Yoshihiro Kanamori, and Shigeru Kuriyama. Animating landscape: self-supervised learning of decoupled motion and appearance for single-image video synthesis. *arXiv preprint arXiv:1910.07192*, 2019.

[12] Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems*, pages 64–72, 2016.

[13] 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*, pages 2672–2680, 2014.

[14] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in neural information processing systems*, pages 6626–6637, 2017.

[15] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*, 2019.

[16] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[17] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[18] Manoj Kumar, Mohammad Babaeizadeh, Dumitru Erhan, Chelsea Finn, Sergey Levine, Laurent Dinh, and Durk Kingma. Videoflow: A flow-based generative model for video. *arXiv preprint arXiv:1903.01434*, 2(5), 2019.

[19] Alex X Lee, Richard Zhang, Frederik Ebert, Pieter Abbeel, Chelsea Finn, and Sergey Levine. Stochastic adversarial video prediction. *arXiv preprint arXiv:1804.01523*, 2018.

[20] D. Legland, K. Kiêu, and M.-F. Devaux. Computation of minkowski measures on 2d and 3d binary images. *Image Analysis & Stereology*, 26(2):83–92, 2011.

[21] Chengze Li, Xueting Liu, and Tien-Tsin Wong. Deep extraction of manga structural lines. *ACM Transactions on Graphics (TOG)*, 36(4):117, 2017.

[22] Pauline Luc, Aidan Clark, Sander Dieleman, Diego de Las Casas, Yotam Doron, Albin Cassirer, and Karen Simonyan. Transformation-based adversarial video prediction on large-scale data. *arXiv preprint arXiv:2003.04035*, 2020.

[23] L. Mosser, O. Dubrule, and M. Blunt. Reconstruction of three-dimensional porous media using generative adversarial neural networks. *Physical Review E*, 96(4):043309, 2017.

[24] Lukas Mosser, Olivier Dubrule, and Martin J Blunt. Stochastic seismic waveform inversion using generative adversarial networks as a geological prior. *Mathematical Geosciences*, 52(1):53–79, 2020.

[25] Seonghyeon Nam, Chongyang Ma, Menglei Chai, William Brendel, Ning Xu, and Seon Joo Kim. End-to-end time-lapse video synthesis from a single outdoor image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1409–1418, 2019.

[26] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[27] Veit Sandfort, Ke Yan, Perry J Pickhardt, and Ronald M Summers. Data augmentation using generative adversarial networks (cyclegan) to improve generalizability in ct segmentation tasks. *Scientific reports*, 9(1):1–9, 2019.

[28] Edgar Simo-Serra, Satoshi Iizuka, and Hiroshi Ishikawa. Mastering sketching: adversarial augmentation for structured prediction. *ACM Transactions on Graphics (TOG)*, 37(1):11, 2018.

[29] Aaron van den Oord, Oriol Vinyals, et al. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pages 6306–6315, 2017.

[30] Denis Volkhonskiy, Ekaterina Muravleva, Oleg Sudakov, Denis Orlov, Evgeny Burnaev, Dmitry Koroteev, Boris Belozerov, and Vladislav Krutko. Generative adversarial networks for reconstruction of three-dimensional porous media from two-dimensional slices. *Phys. Rev. E*, 105:025304, Feb 2022.

[31] Dirk Weissenborn, Oscar Täckström, and Jakob Uszkoreit. Scaling autoregressive video models. *arXiv preprint arXiv:1906.02634*, 2019.