Light Field Interpolation using a Variational Autoencoder

Denis Zavadski, Ben Dangelmayr, Nabila Hoque, and Christian Laun

Abstract

We attempt to change the point of view of existing images of a scene. As input data we use a sparsely populated light field and build a pipeline to obtain the missing perspectives to complete the field. We achieve this using a Variational Autoencoder that we train to replicate single rows and columns of the light field. Later we can then interpolate between the encoded probability distributions of the row and the column stacks to generate new perspectives and weight the samples to obtain different diagonal views. This can drastically reduce the effort and equipment costs to create light fields by utilizing the highly redundant nature of the data.

I. Introduction

In photography, a 3D scene is projected onto a 2D image plane using a camera. The same scene can have many different projections, depending on the location of our camera centre. While we can accurately compute these 2D projections if we know the scene and the location of the camera, the reverse is not true. Using real-world assumptions we can, however, make guesses how such a new view might look. This can be done by rendering such guesses or to execute certain geometric operations or just interpolate between existing points of view.

In this project, we look at 5x5 light fields. These are created using a 5x5 camera array, such that each camera captures a slightly different view of the same scene. This is a large number of cameras that end up capturing a lot of redundant information. If we use only one row and one column of this camera array, the number of cameras needed decreases from 25 to 9 while still capturing almost all of the relevant scene information. An epipolar plane image (EPI) is a 2D slice which arises out of a 4D space when we fix a vertical or horizontal stack of a light field, our row and column, and move through the different viewpoints in one direction. We attempt to then create the missing views in post-processing. This is done by using a 3D convolutions based network to learn a latent space representation of the data and decode it back to a light field representation. A good way to do this is using an Autoencoder for the light field reproduction. The Autoencoder is then extended by a variational part, by mapping the latent space on a parametric distribution, to be able to produce intermediate light fields. Variational Autoencoders (VAEs) have one fundamentally unique property that separates them from autoencoders, and it is this property that makes them so useful for generative modelling: their latent spaces are continuous, allowing easy random sampling and interpolation. This is made possible by encoding the image not into a simple feature vector, but into a probability distribution of feature vectors from which we then sample. Due to the noisy nature of the sampling, the network is forced to not react too strongly to small changes, creating the continuous latent space.

We also utilized an auxiliary U-Net and observed its impact on the validation data linking different layers of the U-Net with our main VAE (Variational Autoencoder).

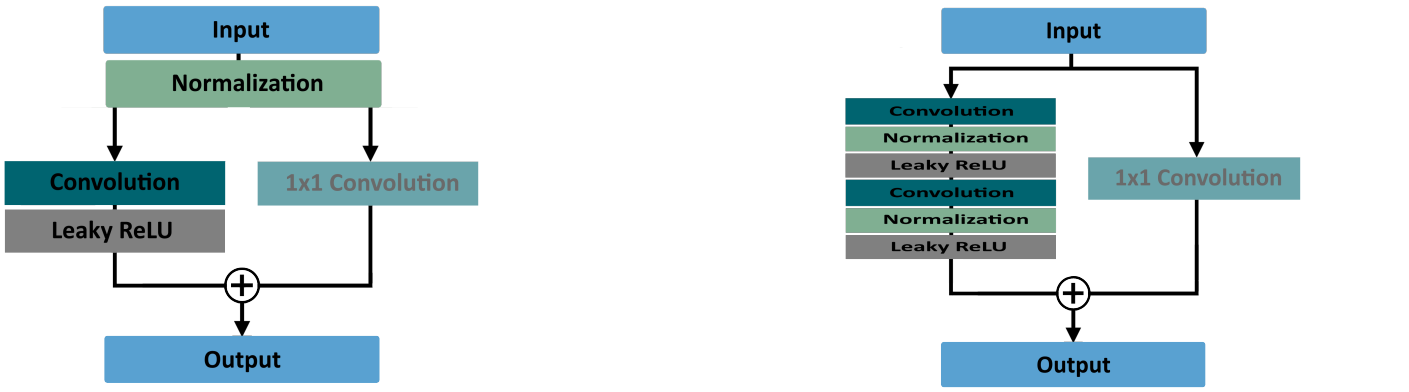
.

II. Related Work

The fundamental elements of our work are light fields. Light fields have their origin in computer graphics and were firstly subject of image-based rendering [HK07] and light field rendering [LH96] to render additional viewpoints.There is a broad interest in light fields and the applications associated with 3D scenes as they require a lot of computing power and data storage. Therefore light field compression is an important topic and discussed by several papers, e.g. to make use of the redundancy of data in different viewpoints[MG00]. Also, Kaushik et al mentioned the importance of the imminent need for light field processing [Kau12] tackling several tasks like denoising or superresolution using a gaussian mixture model.

Finally, also neuronal networks can be applied to work on such tasks mentioned above. Deep learning architectures were used to perform material recognition on 4D light fields [Wan+16]. Another attempt by S. Heber et al [Ste16] was to use a convolutional neuronal network(CNN) to train it to gain a better synthetic lightfield. This approach obtains a higher depth ground truth accuracy resulting in a more accurate dataset. This is necessary as due to the fact that the ground truth depth of existing datasets is inaccurate and/or the datasets are limited to a small number of LFs.

The basic idea of this paper is to follow up on the idea of using an encoder-decoder part to reproduce light fields like A. Alperovich et al [AG18]. They used the Autoencoder for a low-dimensional representation of EPIs. We, however, add a variational part to generate new viewpoints. We also trained an auxiliary U-Net to concatenate and gain its input for further information as the U-Net is a well-explored option to make use of contracted images [Ola15].



(a) 3 Dimensional Residual Block (b) 2 Dimensional Residual Block

Fig. 1: Residual Blocks. Depending on if the output dimension is changed in any way, the convoluted, non-linear input is added to the raw (or normalized) input or the raw (or normalized) input is also convoluted by a 1×1 kernel to equalize the dimension.

III. Approach

Let x be a stack of horizontal or vertical, consecutive image perspectives. The idea is to build a Variational Autoencoder (VAE) to be able to not just reconstruct the provided perspective stack, but also obtain new perspectives from overlayed probability samples of existing perspectives.

A. Network Architecture

For the VAE we mainly utilize the structure of the encoder-decoder network from [AG18] using 3-dimensional residual blocks pictured in 1. Because of their ability to provide the network with a better representation of the identity function, residual blocks are widely used in several tasks related to image-processing. When processed by a residual block, the input is normalized first, before the normalized input is processed by a convolutional layer leaky ReLU. If there is no change in channel-size or spatial size between the input and the output of the residual block, the convoluted input is added to the normalized input. Otherwise, the normalized input is convoluted with a 1×1 kernel to adjust the input to the desired size. The VAE encoder consists of overall 19 residual blocks with the kernel size of 3 that narrow the input size down to 25%. The encoding is then put through a convolution layer for the means and a convolution layer for the variances. The sample from the latent distribution is then decoded by a decoder with an encoder-mirroring structure.

B. Auxiliary U-Net

All perspective stacks have one constant centred image for the whole perspective circle. We can utilize this constant image to aid the network's shape and colour reconstruction because of the strong spatial similarity throughout all perspectives. Because the features of the central image get blended with the network's representation of all the other images from the input, we train an auxiliary U-Net [Ola15] network that reconstructs solely the constant, centred perspective. The pre-trained encoder of the auxiliary U-Net then provides additional structural information about the constant perspective to the main decoder through U-Net skip-connections by concatenation as pictured in 2. The auxiliary network does not train during the training of the VAE. The auxiliary U-Net consists of 2-dimensional residual blocks 1 and the upsampling is done with transposed convolutions.

C. Loss Functions

To train the various parts of our model, we combine several loss functions. We use the Kullback-Leibler-Divergence (KL loss) to regularize the latent space of the VAE and the mean squared error (MSE) loss with hard-flow-example-mining (HFEM) [Xu+19] as reconstruction loss. The auxiliary U-Net uses the same MSE loss and HFEM. We also tested the performance of L_1 loss either with or instead of the MSE loss, but did not end up using it in the final training as it did result in any considerable improvements.

Kullback-Leibler (KL) loss is the Kullback-Leibler divergence ([KL51]) between the predicted probability distribution and the centered isotropic multivariate Gaussian $N(z; 0, 1)$ [KW14], which is

$$L_{KL} = \sum_i -2 \cdot \log(\sigma_i) + \mu_i^2 + \sigma_i^2, \tag{1}$$

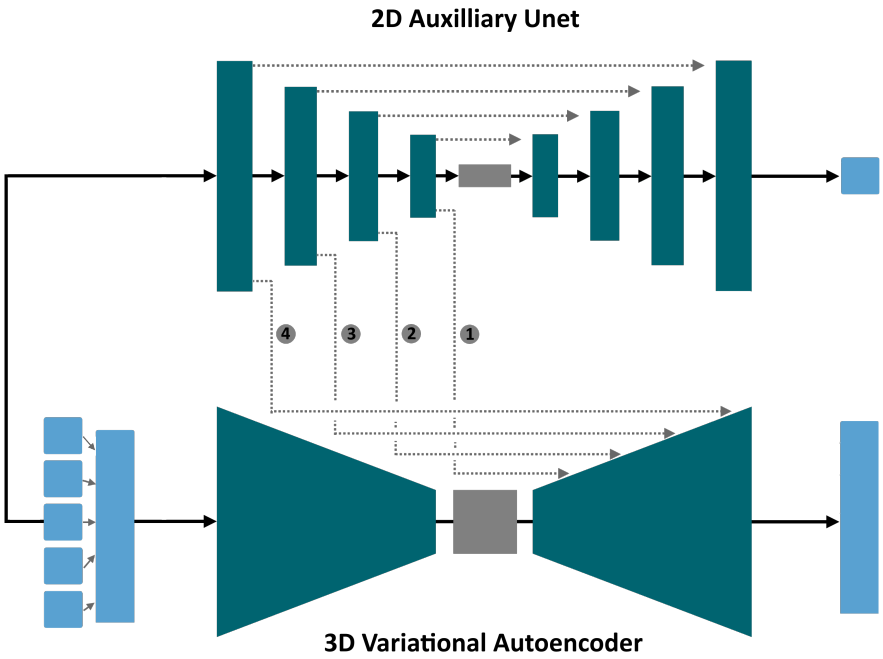


Fig. 2: Network Architecture. The network consists of a Variational Autoencoder (VAE) (bottom) and an auxiliary U-Net (top). The VAE receives a stack of 5 images and maps them onto a parametrical distribution, before decoding them back. The auxiliary U-Net receives the central image and reconstructs it. The intermediate encodings of the U-Net are provided to the intermediate layers of the decoder of the VAE.

with σ_i, μ_i being the predictions of the encoder network. The term forces the network to predict a meaningful probability distribution: Without it, the means of the Gaussian may become large compared to their standard deviations, effectively predicting only a single point in latent space.

The **Mean Squared Error (MSE) loss** is used as reconstruction loss both for the VAE as well as the auxiliary U-Net. It is the squared absolute difference between the predicted pixel colour values x_i and the ground truth values $\hat{x}_i$:

$$L_{MSE} = \frac{1}{N}\sum_i (x_i - \hat{x}_i)^2, \tag{2}$$

N being the total number of image colour values.

The L_1 **loss** is very similar: It is the absolute difference between prediction and target.

$$L_1 = \frac{1}{N}\sum_i |x_i - \hat{x}_i|. \tag{3}$$

To better focus the reproduction loss on difficult locations like boundaries, we use **Hard Flow Example Mining (HFEM)**. A good prediction of crisp boundaries is both a challenging task and a result we are very interested in. To achieve this, we rank the predicted pixel colour values in descending order by their loss magnitude, from highest to lowest. The top p per cent is then used, the rest is discarded. This can be expressed using a binary mask M that is 1 for bad predictions and 0 for good predictions. The loss term then reads as

$$L_{HFEM} = \frac{1}{|M|}\sum_i M_i \cdot (x_i - \hat{x}_i)^2. \tag{4}$$

To train the VAE, a weighted sum of the loss terms is used:

$$L_{total} = L_{KL} \cdot \frac{|L_{MSE}|}{|L_{KL}|} + L_{MSE} + \lambda L_{HFEM} \tag{5}$$

The KL loss is normed to the same magnitude as the MSE loss to avoid one term dominating. The HFEM loss is weighed with λ, for which we chose 5.

The auxiliary U-Net uses the almost identical combination of $L_{total} = L_{MSE} + \lambda L_{HFEM}$ with $\lambda = 5$, the only difference being the lack of the KL loss.

Model	L2 Loss ×100, training	L2 loss ×100, validation	L1 Loss ×100, training	L1 loss ×100, validation
Auxiliary U-Net	0.023	0.145	1.214	2.248
VAE (1Layer)	0.206	0.778	3.077	6.103
VAE (2Layer)	0.140	0.819	2.580	6.220
VAE (3Layer)	0.083	0.648	1.922	5.351
VAE (4Layer)	0.023	0.163	0.960	2.211

TABLE I: Network losses for different configurations. nLayer states the amount (n) of intermediate information streams from the auxiliary U-Net provided to the VAE. The training set contains of the training + the additional set of the 4D Light Field Dataset [Hon+16; Joh+17]

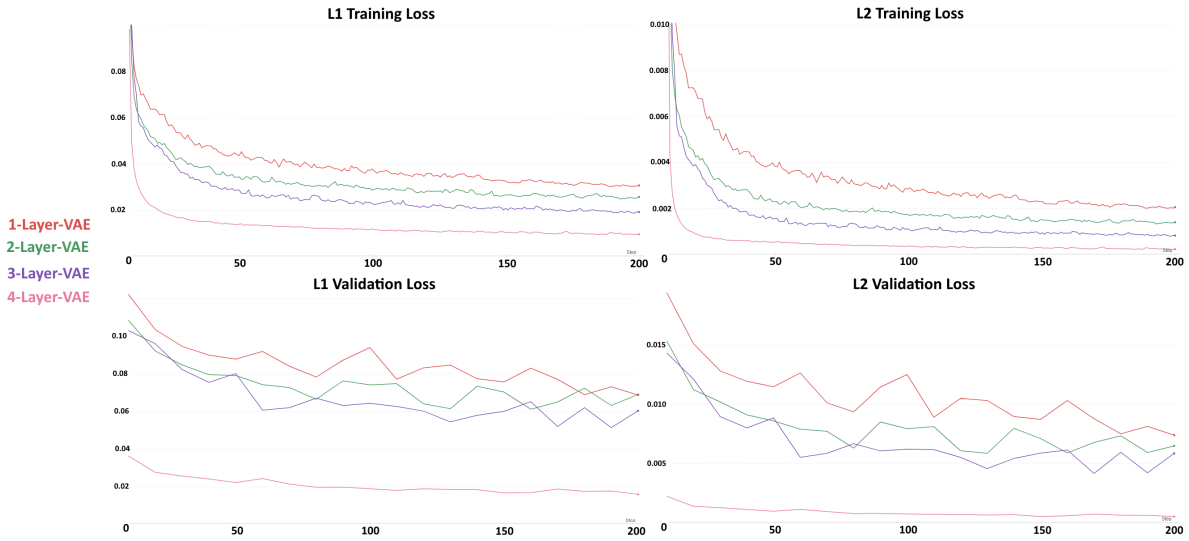


Fig. 3: Loss Graphs for different configurations. nLayer states the amount (n) of intermediate information streams from the auxiliary U-Net provided to the VAE.

D. Interpolation

After encoding the image stack in a probability distribution, we draw a sample z_1 from that distribution. Due to the continuous nature of the latent space, we can draw a second sample z_2 and then interpolate between the samples. z_2 does not need to be drawn from the same probability distribution for that to work, instead we can use this to produce an interpolation between two image stacks in the latent space. The samples are combined to a new $z = 0.5z_1 + 0.5z_2$, which is then decoded using the decoder part of the network. The auxiliary central perspective encoding from the auxiliary U-Net is also fed to the decoder as it remains constant throughout all the horizontal, vertical and all intermediate perspective stacks.

IV. Experiments

We use the publicly available 4D Light Field Dataset [Hon+16; Joh+17]. The data set consists of 24 9x9x512x512x3 light fields and is split into a training set of size 20 and a validation set of size 4. Using a sliding window, we turn each 9x9 light field into 25 5x5 light fields. We train the VAE to replicate a row or a column from these light fields. For that purpose, we randomly choose either the center column or the center row as input for the VAE.

As the data set is not very large we use data augmentation. The following transformations are used:

Normalization: The data is channel-wise normalized. This is reverted in the output of the net.

Random flips: The images are randomly flipped in vertical and/or horizontal direction.

Random Resized Crop: The images are cropped to a random size and then resized to 96×96 pixels.

The training is then done using the ADAM [KB17] optimizer and a learning rate scheduler that decreases the learning rate once the MSE loss plateaus. We used mini-batches of size 32 and trained on a NVIDIA GTX 1080ti for 200 epochs, taking roughly 6 hours per network configuration until convergence. The training is logged using Weights & Biases [Bie20] and validated every 10 epochs.

V. Results

We trained several configurations of our model until convergence. Each configuration had a different amount of U-Net skip-connections from the auxiliary U-Net to the main VAE. We had 4 final (n-layer) configurations in which the network

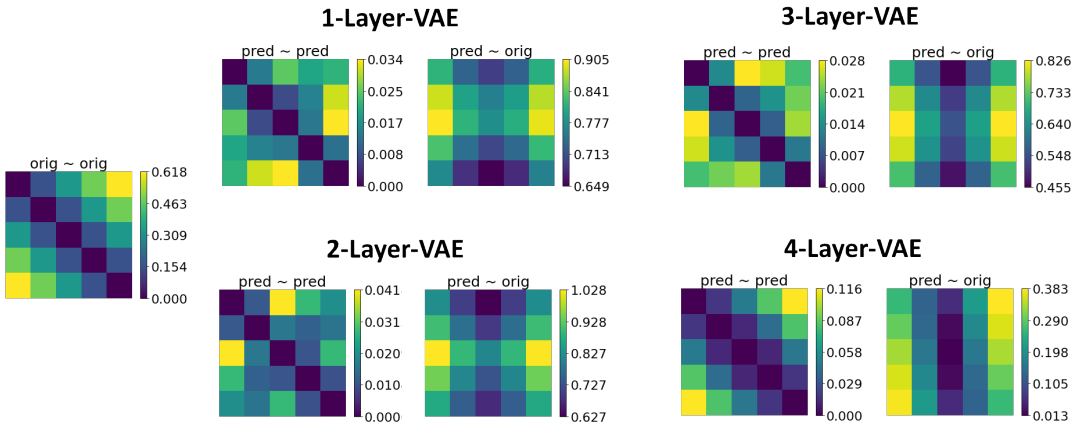


Fig. 4: Perspective Distances measured with the L2 Loss $\times 100$ on the validation dataset.

Fig. 5: Reconstructions of horizontal perspectives.

received up to 4 information streams from the U-Net according to the numbering in 2. The models were evaluated on the validation set of [Hon+16; Joh+17] with an image size of 512×512. The according L_1 and L_2 losses are stated in I.

The loss graphs for the L_1 and the L_2 loss are shown in 3 for the training and the validation during training time. Because of the random sized crops at random locations, the network does not seem to overfit even after 200 epochs on such a small dataset.

Because our model receives explicit information about the encoding of the overall constant middle perspective, a strong bias towards the middle seems to be an axiomatic assumption. To evaluate the dependence of the predicted perspectives to the provided centred representation we calculate the distance matrices of every perspective to each other for the validation dataset and the corresponding predictions, as well as the distances from the predictions to the original image 4. It is evident that the more additional information streams the VAE receives, the bigger the bias towards the centred perspective becomes as shown in the prediction-to-original distance matrix. The maximum distance between perspectives also decreases significantly with additional information. Because the target-inter-perspective-distances are, in the worst case of the 3-Layer-VAE, up to 22 times higher than the predicted ones, the model produces smaller perspective changes than there actually are.

Figure 5 shows some random examples of the model representation of the different model configurations. It is evident that with more additional information from the auxiliary U-Net a much better colour decoding is retained.

When the additional information streams from the auxiliary U-Net are omitted, the model struggles to achieve proper colour reconstruction and tends to focus on brightness and contrast of several shades of brown. Usually, this can be counteracted by increasing the size of the encoder or the latent space to enable the network to retain additional information which contributes to a small degree to the overall loss, which might have been the proper colour. We trained the same VAE model with an increased

Fig. 6: Interpolated view between the horizontal and the vertical stack

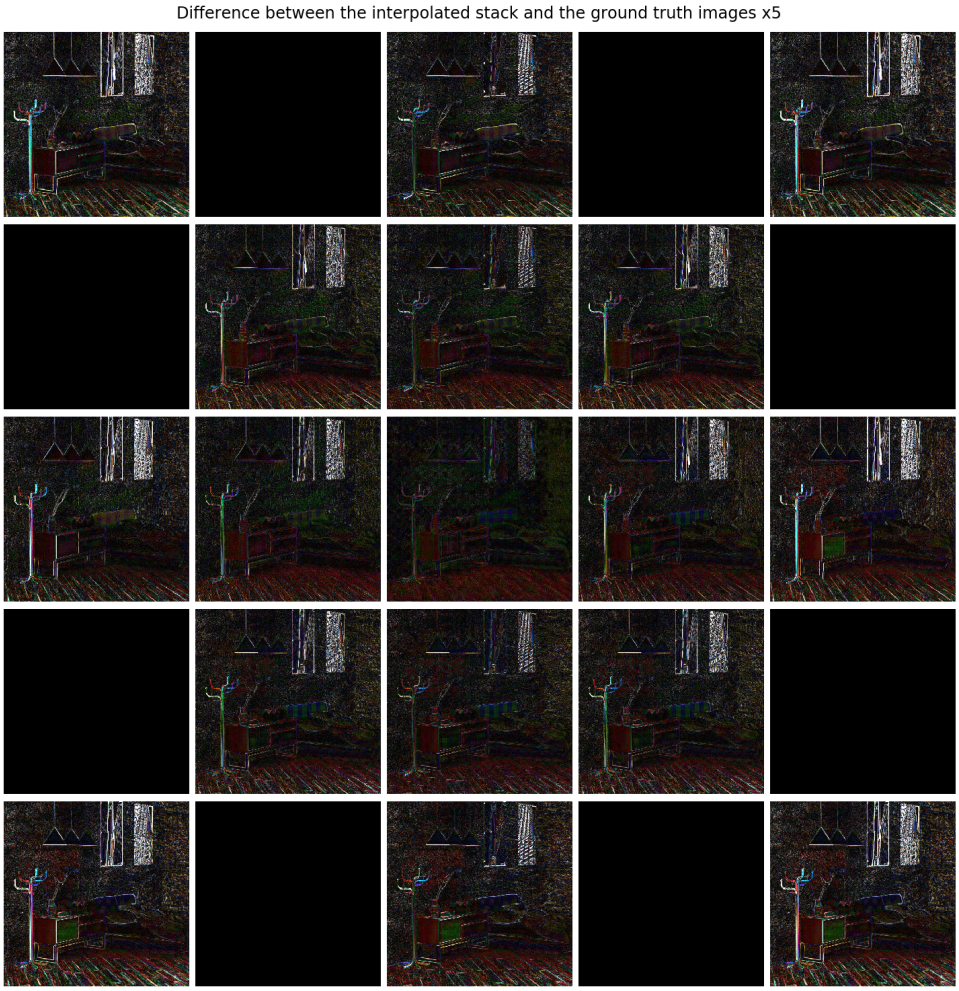


Fig. 7: Difference image: We present the pixelwise absolute difference between ground truth and interpolated image stack

amount of residual blocks in the encoder, in the decoder and both. Neither of those constellations led to any significant improvements in the colour reconstruction nor the overall loss. Increasing the latent space from 25% to 33% of the original size also did not have any significant impact on the model's performance.

A. Interpolation

We were able to interpolate between the latent representations of a vertical and a horizontal stack, see fig. 6. One can see that the produced output looks reasonable by itself and we have no artifacts produced by the interpolation. It is, however, not immediately clear in what direction the stack is oriented.

Figure 7 shows a comparison between the interpolated stack and the diagonal stack, as well as the horizontal and vertical stack. The error is dominated by our bias towards the middle image, and the pixelwise errors look otherwise very similar in all orientations.

VI. Conclusion

In this work, we proposed a generative variational encoder-decoder architecture coupled to an auxiliary U-Net taking stacks of images from a light field grid and generating new viewpoints at new synthetic camera positions. Using a random resized crop together with a random flip, we can achieve a big variety of training data out of rather small dataset. This increases the flexibility and robustness of our network a lot as the model has to learn to cope with different sizes of the same object due to the random sizes of the crops.

We observed a high impact of the choice of concatenated U-Net-layers as mentioned above for the quality of recreated light fields. We were able to represent the data in a much smaller latent representation and to restore the former scenes and also generated new scenes through interpolation. The relatively small displacements between the light scenes and the bias towards the centre viewpoint caused by our network architecture makes it difficult to evaluate the generated diagonal of image stacks. Nevertheless, we were able to create images showing the difference in predictions vs targets. Images close to the centre were reconstructed quite well whereas the outer viewpoints suffer higher deviations. If animated, the reconstructed perspectives have a smaller displacement between each other than the target images. Furthermore, we observed that the displacement often tend to be diagonal which might be due to the training data, which consisted exclusively out of horizontal and vertical perspective stacks. So it seems obvious that the network might have adjusted to a spatial compromise between these two possibilities.

The overall results for recreating viewpoints are far better than the results for the interpolation. On the one hand, the U-Net is needed to gain accurate reconstruction light fields and especially for little training data it enhances the quality of the reconstructions. On the other hand, it decreases the accuracy for outer viewpoint´s interpolation since it introduces a considerable bias towards the central image.

For future work, one could take up our architecture and train it with a light field which undergoes higher displacements to observe how the bias towards the centre scene develops and to obtain better interpolations between viewpoints. The architecture could be of good use for light field reconstruction having very little training data. Perhaps it might be fruitful to adjust the information stream from the auxiliary U-Net to reduce the bias towards the constant image while still providing just enough structural information for the overall perspective reconstruction.

References

[KL51] S. Kullback and R. A. Leibler. “On Information and Sufficiency”. In: *Ann. Math. Statist.* 22.1 (Mar. 1951), pp. 79–86. DOI: 10.1214/aoms/1177729694. URL: https://doi.org/10.1214/aoms/1177729694.

[LH96] M. Levoy and P. Hanrahan. *Light field rendering*. 1996.

[MG00] Magnor M. and B. Girod. *Data compression for light field rendering*. 2000.

[HK07] S. Chan H. Shum and S. Kang. *Image-based rendering*. 2007.

[Kau12] Ashok Veeraraghavan Kaushik Mitra. *Light field denoising, light field superresolution and stereo camera based refocussing using a GMM light field patch prior*. 2012.

[KW14] Diederik P Kingma and Max Welling. *Auto-Encoding Variational Bayes*. 2014. arXiv: 1312.6114 [stat.ML].

[Ola15] Thomas Brox Olaf Ronneberger Philipp Fischer. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015.

[Hon+16] Katrin Honauer et al. “A dataset and evaluation methodology for depth estimation on 4D light fields”. In: *Asian Conference on Computer Vision*. Springer. 2016.

[Ste16] Thomas Pock Stefan Heber. *Convolutional Networks for Shape from Light Field*. 2016.

[Wan+16] Ting-Chun Wang et al. *A 4D Light-Field Dataset and CNN Architectures for Material Recognition*. 2016.

[Joh+17] Ole Johannsen et al. “A Taxonomy and Evaluation of Dense Light Field Depth Estimation Algorithms”. In: *Conference on Computer Vision and Pattern Recognition - LF4CV Workshop*. 2017.

[KB17] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[AG18] Michael Strecke Anna Alperovich Ole Johannsen and Bastian Goldluecke. *Light field intrinsics with a deep encoder-decoder network*. 2018.

[Xu+19] Rui Xu et al. *Deep Flow-Guided Video Inpainting*. 2019. arXiv: 1905.02884 [cs.CV].

[Bie20] Lukas Biewald. *Experiment Tracking with Weights and Biases*. Software available from wandb.com. 2020. URL: https://www.wandb.com/.