Generative Appearance Flow: A Hybrid Approach for Outdoor View Synthesis

Muhammad Usman Rafique[1]
usman.rafique@uky.edu
Hunter Blanton[1]
hunter.blanton@uky.edu
Noah Snavely[2]
snavely@cs.cornell.edu
Nathan Jacobs[1]
nathan.jacobs@uky.edu

[1] Multimodal Vision Research Lab,
University of Kentucky,
Lexington, Kentucky

[2] Computer Science
Cornell Tech, Cornell University
New York City, New York

Abstract

We address the problem of view synthesis in complex outdoor scenes. We propose a novel convolutional neural network architecture that includes flow-based and direct synthesis sub-networks. Both sub-networks introduce novel elements that greatly improve the quality of the synthesized images. These images are then adaptively fused to create the final output image. Our approach achieves state-of-the-art performance on the KITTI dataset, which is commonly used to evaluate view-synthesis methods. Unlike many recently proposed methods, ours is trained without the need for additional geometric constraints, such as a ground-truth depth map, making it more broadly applicable. Our approach also achieved the best performance on the Brooklyn Panorama Synthesis dataset, which we introduce as a new, challenging benchmark for view synthesis. Our dataset, code, and pretrained models are available at https://mvrl.github.io/GAF.

1 Introduction

View synthesis is the task of generating novel views of a scene given only a set of known images. Inferring the appearance of a scene from different viewpoints requires a rich understanding of its geometric and radiometric structure. As such, view synthesis has long been a topic of interest in the computer vision and graphics communities. Early work focused on view synthesis in laboratory settings. Recent work has explored view synthesis in natural, outdoor scenes using convolutional neural networks (CNNs) that take as input a single source image and a camera motion vector [13, 25]. There are two predominant approaches: flow-based synthesis [25] and direct synthesis [19]. Flow-based methods use a CNN to predict a flow field that is used to warp the input image using existing pixel content only. The main advantage of flow-based methods is that the synthesized images are typically sharp and colors are preserved. However, there are issues in dealing with disocclusion, because it is not possible to copy occluded regions from the input image. Direct synthesis methods are not limited to warping the input since the CNN outputs the raw pixel intensity values. Unfortunately, training

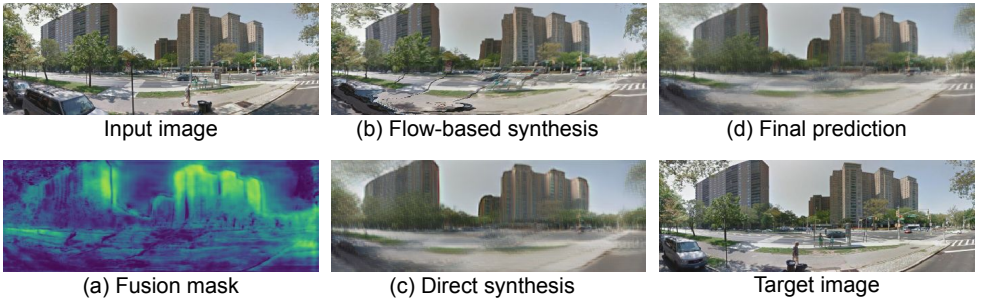

Figure 1: Given an input image and camera transformation, our system synthesizes a flow-based prediction (b) and a direct prediction (c). An adaptive fusion mask (a) is predicted to fuse flow-based and direct predictions to make the final prediction (d). By fusing the results of these two predictions, we produce a new image with the benefits of both.

such models is difficult, especially if the scene structure is unknown. This has motivated recent approaches that use auxiliary geometric information, such as ground-truth depth during training [8] or the semantic layout of the target image at inference time [18]. Without these additional cues, synthesis approaches often generate sub-par results. Our approach addresses this challenge without requiring additional information.

We propose a convolutional neural network (CNN) architecture that uses an adaptive fusion process to combine flow-based and direct synthesis methods. See Figure 2 for an overview of the full architecture. We use a fully convolutional flow-prediction sub-network which uses a distributed encoding of the camera motion parameters that improves training stability. Also, we propose using an adaptive image scale during training that allows for progressive sharpening of generated images as training progresses. We use the output flow from the flow sub-network to warp the intermediate features of a direct synthesis sub-network. This warping significantly improves the quality of the predictions. Finally, we train a fusion module that learns to combine direct and flow-based images to produce the final output.

A standard benchmark dataset for single-image view synthesis is KITTI [6], which consists of perspective images and corresponding camera poses. We show that our method achieves state-of-the-art performance on KITTI. Through an ablation study, we also show that our flow-based network alone also improves upon previous work. However, the motion involved in KITTI is limited, with little lateral or vertical camera movement. To address this issue, we created the Brooklyn Panorama Synthesis (BPS) dataset. It consists of pairs of panoramic images with corresponding relative camera motion.

Main Contributions We propose a novel view synthesis method that combines elements of flow-based and direct synthesis approaches, achieving state-of-the-art performance. Our flow-based sub-network includes three novel elements: a) a fully convolutional flow-prediction network, b) a distributed motion encoding scheme, and c) an adaptive scale space training method which is critical when image motion is large. This sub-network, by itself, improves upon the state of the art. We also propose a novel direct synthesis method that integrates the flow-field estimated by the flow-based sub-network. We evaluate our approach on the standard KITTI benchmark and introduce a more diverse, large-scale dataset suitable for evaluating outdoor, single-image view synthesis methods.

2 Related Work

Given the long history of novel view synthesis, there exists a variety of methods and use cases. Many modern view synthesis and next frame prediction methods rely on multiple input images [2, 4, 5, 28]. These methods require several nearby views and tend to perform poorly on low frame-rate video or when only a single reference view is given. NeRF [11] uses several images of a scene to learn a radiance field that can be used to synthesize novel views close to poses of training images. While this method achieves high quality results, it requires per-environment training. We address the challenging problem of single image view synthesis in complex outdoor environments with large camera transformations.

Traditional geometric methods of synthesizing novel views require estimating the 3D layout of the scene [1, 23, 24]. Once the 3D information is available, the image can be warped and rendered from the desired viewpoint. These methods typically cannot deal with the difficult problem of occlusion that manifests through independent object motion and view point transformation. Furthermore, estimating 3D geometry from color imagery is itself an active research area. As opposed to geometric methods, learning-based approaches implicitly learn to simultaneously understand and manipulate the 3D structure of the scene.

Direct Methods Image synthesis through CNNs has become extremely popular due to the success of generative adversarial networks (GANs) and autoencoders. While most work is focused on simply generating realistic images, several approaches perform an explicit view transform. Tatarchenko *et al*. [19] propose a CNN to generate images from specific view points through an encoder-decoder architecture. However, this method performs poorly on real world data, producing blurry images that lack detail. Xu *et al*. [21] proposed a GAN for generating images with view-invariant features. While this method performs well, applications are limited to synthesis of single objects viewed from different angles.

Flow-Based Methods Zhou *et al*. introduced Appearance Flow [25], in which a CNN outputs a dense, full-resolution pixel flow field. These 2D flow vectors specify the sampling location in the source image for all coordinates of the output image. The underlying assumption of this approach is that nearby images share much of the same structure and color information. This method produces sharp images, but fails when the target image contains content not seen in the input.

Refinement Based Methods There are existing methods that use direct synthesis networks to improve the quality of synthesis from other methods. For example, Park *et al*. [13], uses a refinement network, an encoder-decoder network, that improves the prediction of a flow-based network. The method by Sun *et al*. [17] uses multiple views to synthesize a single target image. The target view is synthesized by a recurrent direct synthesis method which does not directly share information with the flow-based network.

Incorporating Geometry A common technique for improving performance for view synthesis is by using additional details such as scene depth. These approaches take inspiration from traditional view synthesis which uses explicit scene geometry to perform image warping. Yin *et al*. [22] improve results on natural images through the use of inverse depth maps, explicit camera geometry, and an adversarial loss. Similarly, the method of Liu *et al*. [8] depends on separate depth, normal, and plane estimation networks and uses homography

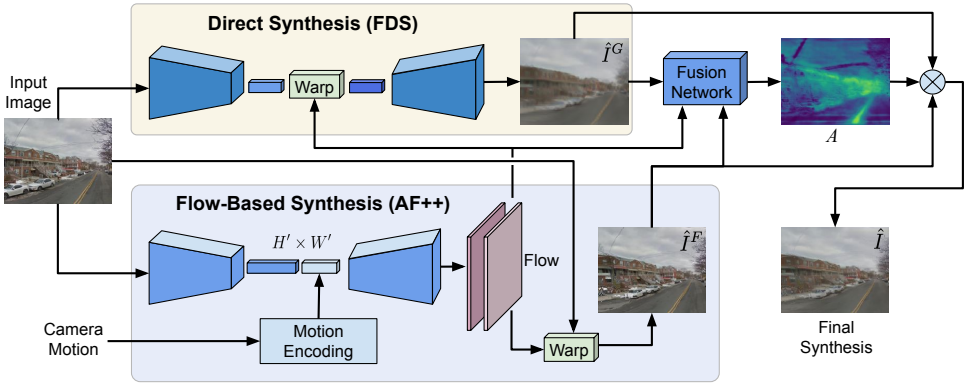

Figure 2: Our Generative Appearance Flow model consists of a flow-based synthesis network (bottom) and a direct synthesis network (top) which uses the flow from the flow-network to warp image features and then predicts the image output. A fusion network learns to combine the flow-based and directly synthesized images to produce the final output.

transformations. These constraints limit the methods to images with known camera intrinsics and environments where reasonable depth predictions can be made. SynSin [20] proposes depth estimation and differentiable rendering for single-image view synthesis. Shih *et al*. [16] propose a depth and color inpainting method for view synthesis for RGB-D images. Our method differs in two main aspects. 1) We do not need to explicitly learn depth of the scene; instead we propose to use the appearance flow between views and use that image-plane warp. 2) This enables us to train across different datasets and work in complex outdoor environments with large camera viewpoint displacement without requiring 3D supervision for training.

3 Approach

We address the task of single-image view synthesis, focusing on translational motion in outdoor scenes. We are given source and target images $I_s, I_t \in \mathbb{R}^{H\times W\times 3}$ and a motion vector v from camera pose of I_s to I_t. The goal is to use I_s and v to synthesize an output image $\hat{I}$ that is similar to the target image I_t. Flow-based methods first estimate a flow field $f \in \mathbb{R}^{H\times W\times 2}$ which specifies relative pixel motion between the source and target views. The flow-prediction network is typically a CNN, which can be modeled as an encoder-decoder architecture: $f = F(E_I(I_s), E_M(v))$ where F is a decoder that generates the flow field, E_I is an image encoder, and E_M is a motion encoder. The output image is synthesized by sampling the input image with the estimated flow. This sampling is typically performed using bilinear interpolation. Direct synthesis methods do not require intermediate outputs. Instead, the image is directly computed by a decoder CNN: $\hat{I} = D(E_I(I_s), E_M(v))$. These methods rely on the decoder D to learn to apply the image transformation.

Overview We propose Generative Appearance Flow (GAF), shown in Figure 2, which combines elements of flow-based and direct synthesis methods. The main components are 1) a flow-based synthesis sub-network, 2) a direct synthesis sub-network that uses the flow field estimated by the previous sub-network to improve output quality, and 3) an adaptive fusion

sub-network that combines the outputs from the previous two.

3.1 Improved Appearance Flow (AF++)

Our flow-based sub-network, AF++, uses the framework introduced by Zhou *et al*. [25] but makes three key improvements that result in state-of-the-art performance: fully convolutional flow prediction, distributed motion encoding, and scale-adaptive spatial sampling.

Fully Convolutional Flow Prediction: We structure our network as a fully convolutional encoder-decoder architecture. We concatenate the image and motion encodings and use a decoder to obtain a two-channel flow field. This removes the fully connected layers present in the the Appearance Flow framework, allowing the features to preserve spatial information.

We use ResNet-18 for the image encoder, which results in a feature map of $\frac{1}{8}$ the input image resolution. Our decoder contains three blocks. Each block upsamples the input feature maps and performs two 2D convolution operations. We use nearest neighbor upsampling to reduce checkerboard artifacts that are common in transposed convolutions [12]. The flow field is dependent on the pixel location but the convolutional operation is independent of the patch location. To deal with this problem in a fully convolutional network, we propose to use CoordConv layer [9], which uses pixel location as an additional input feature. It is important to include pixel location because the expected flow varies drastically across the image based on the epipolar geometry induced by the camera motion. The first and last convolutional layers in decoder are CoordConv layers. The predicted flow values are constrained to the range $[-1,1]$ using the *tanh* activation.

Distributed Motion Encoding A naïve way to incorporate the motion vector v is to append the real-valued motion parameters directly to each pixel of the image feature map. However, we found this to be unstable during training. We propose a distributed encoding for each motion component v_i which is, essentially, a soft form of one-hot encoding. We first define a 1D Gaussian distribution $\mathcal{N}(v_i, \sigma_m^2)$, centered on the component motion. Given the known maximum motion d_{max} along the axis, we linearly sample N displacements $\{-d_{max}, \ldots, d_{max}\}$ and evaluate the Gaussian distribution at each location. The result is an N-dimensional motion encoding, $E_m(v_i)$, with larger values for bins near the true motion. We compute encodings for each dimension of the motion vector and concatenate them to produce an encoding of length $L = N \times K$ where K is the number of motion components. Finally, we tile the encoding vector to the size $H' \times W' \times L$ so that it can be concatenated with the image encoding.

Scale-Adaptive Spatial Sampling Traditional optical flow estimation methods commonly use an image pyramid or smooth the input image to make pixel matching robust. Following these ideas, we model the input image in scale-space before applying the warp to generate the output image. The scale is applied to the input by convolving the image with a 2D Gaussian kernel with scale σ.

A large σ helps training in the early stage, but prevents the network from preserving fine details from the input image. To overcome this, we make σ a learnable parameter. We initialize σ to 2, and found that σ decreases as training continues, converged to just below 1 roughly half way through training. This idea is similar to multi-scale loss evaluation common in optical flow methods, but removes the need to create an explicit image pyramid.

3.2 Flow-Guided Direct Synthesis (FDS)

Our direct synthesis sub-network, FDS, uses an encoder-decoder architecture. The key element is incorporating the flow field f estimated by AF++ to warp the bottleneck feature maps. A similar idea was proposed by Zhu *et al*. [27], where optical flow between video frames was applied to feature maps to reduce the need for feature extraction. Using the image encoder E_I^G, the output is synthesized as: $\hat{I}^G = D^G(S(E_I^G(I_s), f))$ where D^G is the decoder and $S(E_I^G(I_s), f)$ is the image feature map after applying the warp. To apply the flow field to the image features, the field is down-sampled using nearest neighbor sampling to match the feature map resolution. We found that a naïve use of the direct synthesis method that predicts pixel values based only on the input image and the transformation vector to be suboptimal and observed significant performance gain by introducing the feature flow transformation.

For the FDS network, we use a ResNet-50 encoder and a decoder similar to the one in AF++, with the only difference being in the final layer. In D^G, we replace the CoordConv with a standard convolution and the three channel pixel values are predicted in the range [0, 1] by applying the *sigmoid* activation function.

3.3 Adaptive Image Fusion

While our flow-based method AF++ captures fine details and produces sharp results, the direct synthesis method FDS is able to hallucinate missing pixels and generate more coherent predictions. Motivated by this, we propose to adaptively fuse the images generated by these sub-networks to produce the final output image. We train a standard U-Net [15] architecture to predict a fusion weight for each pixel. The network takes as input the concatenation of the predicted flow and the images generated by the first two sub-networks. The output A is a single channel that predicts values in the range [0, 1] using the *sigmoid* activation function. The final output image of GAF is computed using: $\hat{I} = A \odot \hat{I}^F + (1 - A) \odot \hat{I}^G$, where $\hat{I}^F$ is the output of AF++, $\hat{I}^G$ is the output of FDS, and $\odot$ is element-wise multiplication. See Figure 4 for a visualization of the predicted per-pixel fusion mask.

3.4 Loss Functions

We train our full model, and all baseline models, using the same loss function, which combines the following loss components. The first is a reconstruction loss, which in our case is the L_1 loss between target image I_t and generated image $\hat{I}$: $\mathcal{L}_r(\hat{I}, I_t) = \|\hat{I} - I_t\|_1$. To encourage more realistic synthesized images, we add a perceptual loss [7] by extracting CNN features for the synthesized and target image and minimize the mean squared error between the features. We use a ResNet-18 pre-trained for Cityscapes [3] segmentation as the feature extractor. To deal with small artifacts, we also include an adversarial loss by adding a discriminator which aims to differentiate between real and synthesized images. For a set of image patches P, we use a patch discriminator [26] with the least squares loss, $\mathcal{L}_G(\hat{I})$ [10]. For brevity, we omit the loss function for training of the discriminator. We combine these component losses, using hyper-parameters λ_1, λ_2, and λ_3, to define our total loss:

$$\mathcal{L}(\hat{I}, I_t) = \lambda_1 \mathcal{L}_r(\hat{I}', I_t) + \lambda_2 \mathcal{L}_p(\hat{I}, I_t) + \lambda_3 \mathcal{L}_G(\hat{I}). \quad (1)$$

4 Evaluation

We present quantitative and qualitative results of the proposed methods, including an ablation study to assess the contribution of various components. Source code, pre-trained models, and the BPS dataset are available on our project page: https://mvrl.github.io/GAF. See the supplemental material for dataset details, network architectures, and additional visualizations.

4.1 Datasets

We evaluated our methods on two datasets that each consist of pairs of images of outdoor scenes (I_s, I_t) with corresponding motion vectors v. The KITTI dataset [6], containing images from 11 sequences recorded in urban road scenes, is a standard benchmark for outdoor view synthesis. Image pairs are captured with a front-facing camera and were sampled with a maximum interval of one second. This means that the motions are mostly forward or backward, simplifying the view synthesis task. While the raw images are around 1220×370, the training size is reduced significantly be resizing or cropping [17]. Also, the horizontal field of view of KITTI image is around $\sim 82^\circ$, limiting the available information for view synthesis under extreme view change. Following [8, 25], we use first 9 sequences of KITTI for training and last 2 sequences for testing.

To overcome these limitations in KITTI, we created a dataset of outdoor panoramic images, which we name the Brooklyn Panorama Synthesis (BPS) dataset. Images were randomly sampled from Google StreetView such that each pair is within 10 meters. The image size is 960×160 pixels with the horizontal field of view of 360°. In total, it contains 44 092 image pairs. We randomly split the dataset into training (40 592 pairs) and testing (3500 pairs). Note that BPS has more images, wider field of view, and larger average motion compared to KITTI. We believe that BPS is a challenging dataset that will be useful for future work on outdoor view synthesis. Please see supplemental material for detailed comparison of BPS and KITTI dataset and distribution of viewpoint changes.

4.2 Baseline Methods

We compare our method to several state-of-the-art single image methods: Appearance Flow (*AF*) [25], Geometry Aware (*GA*) [8], and Multi-View (*MV*) [17] trained for single image view synthesis. We also include a trivial baseline, *Identity*, that always predicts the source image. This method works surprisingly well, especially when image motion is small. For fairness, we prepare a variant of *AF*, named *AF-ResNet*, that has the same ResNet encoder that we are using in our flow-based method AF++.

4.3 Implementation Details

We implemented our approach using the PyTorch [14] framework. We train our networks using the Adam optimizer with parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We used a learning rate of $1e{-}5$ and L_2 regularization of $1e{-}6$ with batch size of 16. All pixel values were scaled to the range [0,1]. In panoramic images, it is possible that relevant information might be on the wrong side of the image because of the wrap-around effect of equirectangular projection. To address this, we add 48 pixels of wrap-around padding to both the left and right borders. Only the original (unpadded) image pixels are used for evaluation purposes. We encode each

Method	L_1	SSIM	PSNR
Identity	0.4923	0.4159	12.0084
AF [25]	0.4643	0.4595	13.6917
GA [8]	0.340	-	-
MV [17]	0.3971	0.5597	14.2942
AF++ (Ours)	0.3452	0.5395	16.0868
FDS (Ours)	0.3069	0.6079	16.0814
GAF (Ours)	**0.2991**	**0.6102**	**17.1469**

Table 1: Results on the KITTI dataset.

Method	L_1	SSIM	PSNR
Identity	0.4890	0.3587	12.8998
AF [25]	0.4584	0.3540	13.4134
AF-ResNet	0.4399	0.3934	13.8985
AF++ w/o motion enc.	0.4207	0.4001	14.1073
AF++ w/o scaling	0.4341	0.4140	13.8688
AF++ (Ours)	0.3702	0.4534	15.0695
FDS (Ours)	0.3276	0.5257	16.3203
GAF (Ours)	**0.3255**	**0.5276**	**16.3210**

Table 2: Results on the BPS dataset.

Figure 3: Qualitative results on the KITTI dataset.

element of the vector v using our proposed motion encoding with a vector of size 25 with $\sigma_m = 0.75$ and concatenate them to form a single vector of size 50. For KITTI, the odometry provides complete motion in 3D. We use motion encoding for x and y axes with 21 elements each. Since motion is typically along the z-axis, we encode z with 41 bins. This results in an encoding of size 83. For each dimension we use $\sigma_m = 0.75$.

For training AF++, FDS, and GAF, we give higher weight to reconstruction and perceptual loss, $\lambda_1 = 1$, $\lambda_2 = 1$, and lower weight for the GAN loss, $\lambda_3 = 0.01$. We begin by training the flow-based synthesis sub-network, AF++ for 25 epochs. The other flow-based methods, AF and AF-ResNet, are also trained for 35 epochs. We then train our direct synthesis sub-network, FDS. We pretrain FDS as an autoencoder, without using the flows from AF++, by setting the motion encoding and flow to zero and using the same image for source and target. This is done to initialize the image decoder for reasonable image generation. Next, we train FDS for 10 epochs, leaving AF++ frozen. For the final step, we freeze AF++ and FDS and train the adaptive image fusion network for 5 epochs.

4.4 Results

We evaluate our method using standard metrics for testing image generation quality: L_1 error, peak signal-to-noise ratio (PSNR), and structural similarity (SSIM). Results on the KITTI

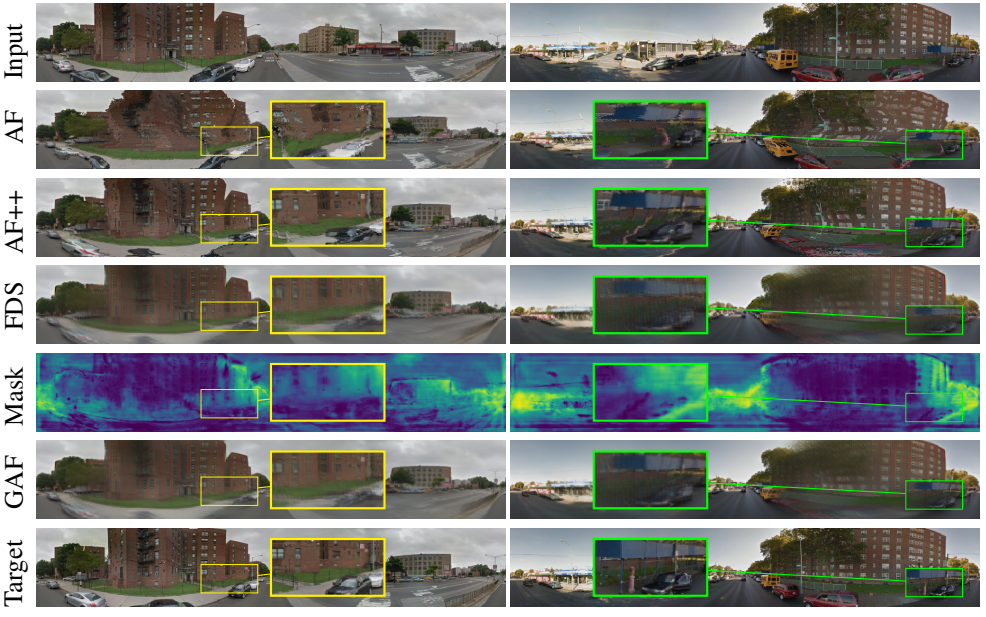

Figure 4: Example outputs from our proposed methods. Fusion masks correctly capture the best regions from AF++ and FDS predictions to synthesize the GAF prediction.

dataset, Table 1, show that our method outperforms *AF* and *GA*. Notice that *Identity* gets reasonable metrics, highlighting the visual similarity between inputs and targets. Note that our flow-based sub-network AF++ alone performs much better than the baseline methods. We see consistent performance gains as we use our direct synthesis method FDS, and GAF achieves the best results. Table 2 shows the performance of our models and the baselines on the BPS dataset. We can see that AF++ gets better metrics than existing methods. Moreover, our FDS and GAF models further improve the metrics. Since *GA* requires 3D supervision, we are unable to evaluate it on BPS.

Ablation Tables 1 and 2 show that using FDS and GAF improve metrics of the flow method AF++. We also perform an ablation study of AF++. We created two variants: one without the motion encoding and one without the scale space training strategy, keeping all other aspects unchanged. The results demonstrate that removing either component significantly decreases performance, with the removal of scale space training having a larger impact.

Qualitative Analysis Qualitative results are shown in Figures 3 and 4. We can see that AF++ retains fine details and produces sharp outputs, but there are noticeable artifacts. FDS produces smooth output that is more globally consistent. The fusion mask from GAF correctly selects the best parts from both intermediate outputs to synthesize images with fewer artifacts. Please see the supplementary material for visualization of the predicted flow fields.

5 Conclusions

We introduced a method for view synthesis that performs well on challenging outdoor scenes. Our method integrates both flow-based and direct approaches. We also introduced a view-synthesis evaluation dataset, BPS, containing panorama pairs. Our flow-based sub-network includes several novel elements: fully-convolutional flow prediction, distributed motion encoding, and an adaptive scale space training strategy. This sub-network alone achieves state-of-the-art results on the KITTI and BPS datasets. Our full method contains a direct sub-network which uses flow estimates from the flow-based sub-network to warp feature maps. The output of both sub-networks are adaptively fused, resulting in further improvements to the state of the art. All data, code, and trained models have been released publicly.

References

[1] Tao Chen, Zhe Zhu, Ariel Shamir, Shi-Min Hu, and Daniel Cohen-Or. 3-sweep: Extracting editable objects from a single photo. *ACM Transactions on Graphics (TOG)*, 2013.

[2] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H. Kim, and Jan Kautz. Extreme view synthesis. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[3] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[4] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[5] John Flynn, Michael Broxton, Paul Debevec, Matthew DuVall, Graham Fyffe, Ryan Overbeck, Noah Snavely, and Richard Tucker. Deepview: View synthesis with learned gradient descent. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[6] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[7] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, 2016.

[8] Miaomiao Liu, Xuming He, and Mathieu Salzmann. Geometry-aware deep network for single-image novel view synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[9] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[10] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[11] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[12] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016.

[13] Eunbyung Park, Jimei Yang, Ersin Yumer, Duygu Ceylan, and Alexander C Berg. Transformation-grounded image generation network for novel 3d view synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[14] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[15] 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*, 2015.

[16] Meng-Li Shih, Shih-Yang Su, Johannes Kopf, and Jia-Bin Huang. 3d photography using context-aware layered depth inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[17] Shao-Hua Sun, Minyoung Huh, Yuan-Hong Liao, Ning Zhang, and Joseph J Lim. Multi-view to novel view: Synthesizing novel views with self-learned confidence. In *European Conference on Computer Vision (ECCV)*, 2018.

[18] Hao Tang, Dan Xu, Nicu Sebe, Yanzhi Wang, Jason J Corso, and Yan Yan. Multi-channel attention selection gan with cascaded semantic guidance for cross-view image translation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[19] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Multi-view 3d models from single images with a convolutional network. In *European Conference on Computer Vision (ECCV)*, 2016.

[20] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[21] Xiaogang Xu, Ying-Cong Chen, and Jiaya Jia. View independent generative adversarial network for novel view synthesis. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[22] Xiaochuan Yin, Henglai Wei, Penghong Lin, Xiangwei Wang, and Qijun Chen. Novel view synthesis for large-scale scene using adversarial loss. *arXiv preprint arXiv:1802.07064*, 2018.

[23] Li Zhang, Guillaume Dugas-Phocion, Jean-Sebastien Samson, and Steven M Seitz. Single-view modelling of free-form scenes. *The Journal of Visualization and Computer Animation*, 2002.

[24] Youyi Zheng, Xiang Chen, Ming-Ming Cheng, Kun Zhou, Shi-Min Hu, and Niloy J Mitra. Interactive images: cuboid proxies for smart image manipulation. *ACM Transactions on Graphics (TOG)*, 2012.

[25] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *European Conference on Computer Vision (ECCV)*, 2016.

[26] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[27] Xizhou Zhu, Yuwen Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[28] Xiaoou Tang Yiming Liu Ziwei Liu, Raymond Yeh and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *IEEE International Conference on Computer Vision (ICCV)*, 2017.