Extreme View Synthesis

Inchang Choi[1,2] Orazio Gallo[1] Alejandro Troccoli[1] Min H. Kim[2] Jan Kautz[1]
[1]NVIDIA [2]KAIST

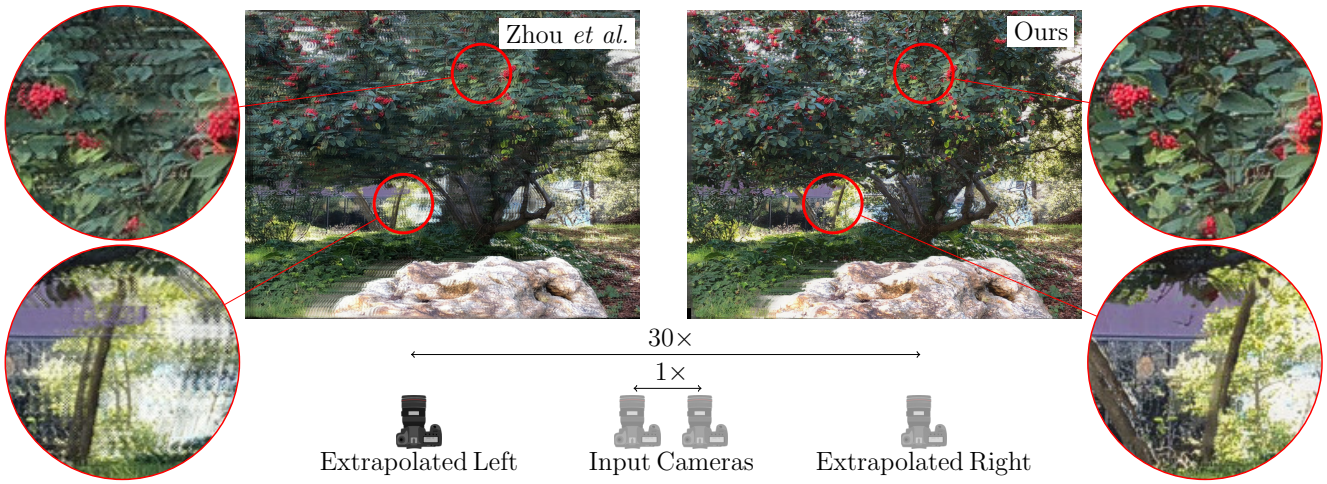

Figure 1: We propose a novel view synthesis method that can generate *extreme* views, *i.e.*, images synthesized from a small number of cameras (two in this example) and from significantly different viewpoints. In this comparison with the method by Zhou *et al.* [34], we show the left view from the camera setup depicted above. Even at a 30× baseline magnification our method produces sharper results.

Abstract

We present Extreme View Synthesis, a solution for novel view extrapolation that works even when the number of input images is small—as few as two. In this context, occlusions and depth uncertainty are two of the most pressing issues, and worsen as the degree of extrapolation increases. We follow the traditional paradigm of performing depth-based warping and refinement, with a few key improvements. First, we estimate a depth probability volume, rather than just a single depth value for each pixel of the novel view. This allows us to leverage depth uncertainty in challenging regions, such as depth discontinuities. After using it to get an initial estimate of the novel view, we explicitly combine learned image priors and the depth uncertainty to synthesize a refined image with less artifacts. Our method is the first to show visually pleasing results for baseline magnifications of up to 30×*. The code is available at* https://github.com/NVlabs/extreme-view-synth

1. Introduction

The ability to capture visual content and render it from a different perspective, usually referred to as *novel view synthesis*, is a long-standing problem in computer graphics. When appropriately solved, it enables telepresence applications such as head-mounted virtual and mixed reality, and navigation of remote environments on a 2D screen—an experience popularized by Google Street View. The increasing amount of content that is uploaded daily to sharing services offers a rich source of data for novel view synthesis. Nevertheless, a seamless navigation of the virtual world requires a more dense sampling than these sparse observations offer. Synthesis from sparse views is challenging, in particular when generating views creating disocclusions, a common situation when the viewpoint is extrapolated, rather than interpolated, from the input cameras.

Early novel view synthesis methods can generate new images by interpolation either in pixel space [4], or in ray space [20]. Novel views can also be synthesized with methods that use 3D information explicitly. A typical approach would use it to warp the input views to the virtual camera and merge them based on a measure of quality [1]. The advantage of such methods is that they explicitly leverage geometric constraints. Depth, however, does not come without disadvantages. First and foremost is the problem of occlusions. Second, depth estimation is always subject to a de-

gree of uncertainty. Both of these issues are further exacerbated when the novel view is pushed farther from the input camera, as shown in Figure 2. Existing methods deal with uncertainty by propagating reliable depth values to similar pixels [2], or by modeling it explicitly [24]. But these approaches cannot leverage depth to refine the synthesized images, nor do they use image priors to deal with the unavoidable issues of occlusions and artifacts.

More recent approaches use large data collections and learn the new views directly [7, 34]. The power of learning-based approaches lies in their ability to leverage image priors to fill missing regions, or correct for poorly reconstructed ones. However, they still cause artifacts when the position of the virtual camera differs significantly from that of the inputs, in particular when the inputs are few.

In their *Stereo Magnification* work, Zhou *et al*. cleverly extract a layered representation of the scene [34]. The layers, which they learn to combine into the novel view, offer a regularization that allows for an impressive stereo baseline extrapolation of up to $4.5\times$. Our goal is similar, in that we want to use as few as two input cameras and extrapolate a novel view. Moreover, we want to push the baseline extrapolation much further, up $30\times$, as shown in Figure 1. In addition, we allow the virtual camera to move and rotate freely, instead of limiting to translations along the baseline.

At a high level, we follow the depth-warp-refine paradigm, but we leverage two key insights to achieve such large extrapolation. First, depth estimation is not always reliable: instead of exact depth estimates, we use depth probability volumes. Second, while image refinement networks are great at learning generic image priors, we also use explicit information about the scene by sampling patches according to the depth probability volumes. By combining these two concepts, our method works for both view interpolation and extreme extrapolation. We show results on a large number of examples in which the virtual camera significantly departs from the original views, even when only two input images are given. To the best of our knowledge, ours is the first method to produce visually pleasing results for such extreme view synthesis from unstructured cameras.

2. Related Work

Early methods for novel view synthesis date back several decades [9]. Image interpolation methods, among the first approaches to appear, work by interpolating between corresponding pixels from the input images [4], or between rays in space [20]. The novel view can also be synthesized as a weighted combination of the input cameras, when information about the scene geometry is available [1, 6]. All of these methods generally assume additional information—correspondences, depth, or geometry—to be given.

Recent methods produce excellent results taking only images as an input. This can be done, for instance, by using an appropriate representation of the scene, such as plane sweep volumes, and by learning weights to merge them down into a single image [7]. Further building on the concept layered depth images [10], Zitnick *et al*. developed a high-quality video-based rendering system for dynamic scenes that can interpolate between views [36]. Zhou *et al*. propose a learned layer-based representation of the scene, dubbed MPI [34]. Their results are impressive, but quickly degrade beyond limited translations of the novel view. The works of Mildenhall *et al*. [23] and Srinivasan *et al*. [27] build on the MPI representation further improving the quality of the synthesized view, even for larger camera translations[1].

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

Figure 2: (a) A point cloud and three cameras. (b)-(d) The images "captured" from the red, green, and the blue cameras. The point cloud was generated from the depth map of the red camera. Depth uncertainty causes larger artifacts as the viewpoint moves farther from the red camera.

A different approach is to explicitly use depth information, which can be estimated from the input images directly, and used to warp the input images into the novel view. Kalantari *et al*., for instance, learn to estimate both disparity and the novel view from the sub-aperture images of a light-field camera [14]. For larger displacements of the virtual camera, however, depth uncertainty results in noticeable artifacts. Chaurasia *et al*. take accurate but sparse depth and propagate it using super-pixels based on their similarity in image space [2]. Penner and Zhang explicitly model the confidence that a voxel corresponds to empty space or to a physical surface, and use it while performing back-to-front synthesis of the novel view [24].

The ability of deep learning techniques to learn priors has also paved the way to single-image methods. Srinivasan *et al*. learn a light field and depth along each ray from a single image [28]. Zhou *et al*. cast this problem as a prediction of appearance flows, which allows them to synthesize novel views of a 3D object or scene from a single observation [35]. From a single image, Xie *et al*. produce stereoscopic images [32], while Tulsiani *et al*. infer a layered representation of the scene [29].

Our approach differs from published works for its ability to generate extrapolated images under large viewpoint changes and from as few as two cameras.

[1]These works were published after the submission of this paper and are included here for a more complete coverage of the state-of-the-art.

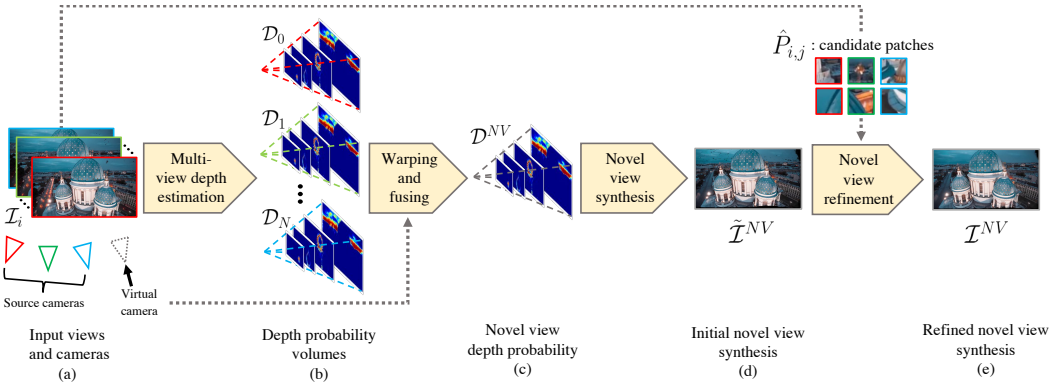

Figure 3: Method overview: from a set of posed input views (a), we generate a set of depth probability volumes for each view (b). Given the novel view camera pose, we create its depth probability volume via warping and fusion of the input depth volumes (c). Next, we synthesize an initial novel view (d), which we refine with a neural network to synthesize the final image (e). Our image refinement is done in a patch-based manner guided by the depth distribution.

3. Overview

Our goal is to synthesizes a novel view, $\mathcal{I}^{\text{NV}}$, from N input views, $\mathcal{I}_i$. A common solution to this problem is to estimate depth and use it to warp and fuse the inputs into the novel view. However, depth estimation algorithms struggle in difficult situations, such as regions around depth discontinuities; this causes warping errors and, in turn, artifacts in the final image. These issues further worsen when N is small, or $\mathcal{I}^{\text{NV}}$ is extrapolated, *i.e.*, when the virtual camera is not on the line connecting the centers of any two input cameras. Rather than using a single depth estimate for a given pixel, our method accounts for the depth's probability distribution, which is similar in spirit to the work of Liu *et al.* [22]. We first estimate N distributions $\mathcal{D}_i$, one per input view, and combine them to estimate the distribution for the virtual camera, $\mathcal{D}^{\text{NV}}$, Section 4. Based on the combined distribution $\mathcal{D}^{\text{NV}}$, we render the novel view back to front, Section 5. Finally, we refine $\mathcal{I}^{\text{NV}}$ at the patch level informed by relevant patches from the input views, which we select based on the depth distribution and its uncertainty, Section 6. Figure 3 shows an overview of the method.

4. Estimating the Depth Probability Volume

Several methods exist that estimate depth from multiple images [15, 8], stereo pairs [16, 17], and even single image [21, 25]. Inspired by the work of Huang *et al.*, we treat depth estimation as a learning-based, multi-class classification problem [12]. Specifically, depth can be discretized into n_d values and each depth value can be treated as a class. Depth estimation, then, becomes a classification problem: each pixel (x_i, y_i) in $\mathcal{I}_i$ can be associated with a probability distribution over the n_d depth values along $\mathcal{R}_i(x_i, y_i)$, the ray leaving the camera at (x_i, y_i) and traversing the scene. We refer to the collection of all the rays for camera i as a *depth probability volume*, $\mathcal{D}_i \in \mathbb{R}^{h \times w \times n_d}$, where $h \times w$ is the resolution of $\mathcal{I}_i$. The network to estimate the $\mathcal{D}_i$'s, can be trained with a cross-entropy loss against ground truth one-hot vectors that are 1 for the correct class and 0 elsewhere, as in Huang *et al.* [12]. We follow the common practice of uniformly sampling disparity instead of depth[2] to improve the estimation accuracy of closer objects.

Empirically, we observe that the resulting depth volumes exhibit desirable behaviors. For most regions, the method is fairly certain about disparity and the probability along $\mathcal{R}_i(x, y)$ presents a single, strong peak around the correct value. Around depth discontinuities, where the point-spread-function of the lens causes pixels to effectively belong to both foreground and background, the method tends to produce a multi-modal distribution, with each peak corresponding to the disparity levels of the background and foreground, see for instance Figure 4. This is particularly important because depth discontinuities are the most challenging regions when it comes to view synthesis.

Solving for the depth probability volumes requires that we know the location and the camera's intrinsic parameters for each input view. We estimate these using Colmap [26]. For a given scene, we set the closest and farthest disparity levels as the bottom 2 and top 98 depth percentiles respectively, and use $n_d = 100$ uniformly spaced disparity steps. Similarly to the method of Huang *et al.*, we also cross-bilateral filter the depth probability volume guided by an input RGB image [19]. However, we find $\theta_\alpha = 25$, $\theta_\beta = 10$, and $\mu = 5$ to work better for our case and iterate the filter for 5 times. We refer the reader to Krähenhühl and Koltun for the role of each parameter [19].

Finally, we can estimate the probability volume $\mathcal{D}^{\text{NV}}$ for

[2]Technically, "disparity" is only defined in the case of a stereo pair. Here we use the term loosely to indicate a variable that is inversely proportional to depth.

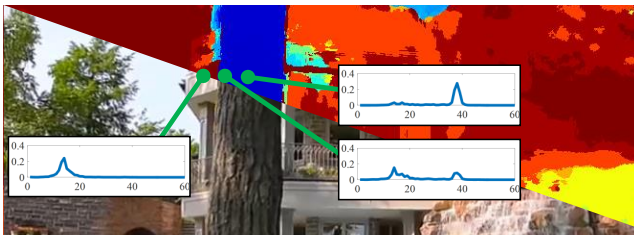

Figure 4: Depth probability distributions along three rays in $\mathcal{D}$. The disparity shows clear peaks for points that are sufficiently distant from a depth discontinuity. Closer to the edge, the inherent uncertainty is captured by the presence of two lower peaks: one corresponding to the foreground, and one to the background.

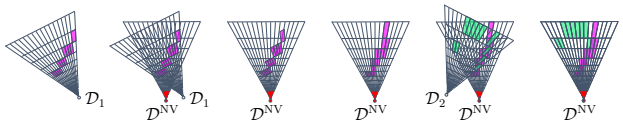

Figure 5: To compute the depth probability volume with respect to the novel view, we resample the input volumes and accumulate them. Here we only look at a planar slice of the depth probability volumes, and we make the simplifying assumption that the input volumes have $p = 1$ for one disparity and $p = 0$ for all the others. Note that the probability along the rays in the final result do not sum to 1 and, therefore require an additional normalization.

the novel view by resampling these probability volumes. Conceptually, the probability of each disparity d for each pixel (x, y), $\mathcal{D}^{\text{NV}}(x, y, d)$, can be estimated by finding the intersecting rays $\mathcal{R}_i$'s from the input cameras and average their probability. This, however, is computationally demanding. We note that this can be done efficiently by resampling the $\mathcal{D}_i$'s with respect to $\mathcal{D}^{\text{NV}}$, accumulating each of the $\mathcal{D}_i$ volumes into the novel view volume, and normalizing by the number of contributing views. This accumulation is sensible because the probability along $\mathcal{R}_i$ is a proper distribution. This is in contrast with traditional cost volumes [11] for which costs are not comparable across views: the same value for the cost in two different views may not indicate that the corresponding disparities are equally likely to be correct. Depth probability volumes also resemble the soft visibility volumes by Penner and Zhang [24]. However, their representation is geared towards identifying empty space in front of the first surface. Therefore, they behave differently in regions of uncertainty, such as depth discontinuities, where depth probability volumes carry information even beyond the closest surface.

Figure 5 shows an example of the resampling procedure, where we consider only a planar slice of the volumes and, for simplicity, that the probability along the input rays is binary. We use nearest neighbor sampling, which, based on our experiments, yields quality comparable with tri-linear interpolation at a fraction of the cost. After merging all views, we normalize the values along each ray in $\mathcal{D}^{\text{NV}}$ to enforce a probability distribution.

5. Synthesis of a Novel View

Using the depth probability volume $\mathcal{D}^{\text{NV}}$, we backward warp pixels from the inputs $\mathcal{I}_i$ and render in a back-to-front fashion an initial estimate of the novel view, $\widetilde{\mathcal{I}}^{\text{NV}}$. Specifically, we start from the farthest plane, where $d = 0$, and compute a pixel in the novel view as

$$\widetilde{\mathcal{I}}^{\text{NV}}(x, y)\Big|_{d=0} = \text{R}\left(\left\{\mathcal{I}_i(x_i, y_i) \cdot \mathbb{1}_{\{\mathcal{D}^{\text{NV}}(x,y,0)>t\}}\right\}_{i=1:N}\right), \tag{1}$$

where $\mathbb{1}$ is the indicator function, and (x_i, y_i) are the coordinates in $\mathcal{I}_i$ that correspond to (x, y) in $\widetilde{\mathcal{I}}^{\text{NV}}$. Note that these are completely defined by the cameras' centers and the plane at d. R is a function that merges pixels from $\mathcal{I}_i$ weighting them based on the distance of the cameras' centers, and the angles between the cameras' principal axes. Details about the threshold t and the weights are in the Supplementary. As we sweep the depth towards a larger disparity d, *i.e.*, closer to the camera, we overwrite those pixels for which $\mathcal{D}^{\text{NV}}(x, y, d)$ is above threshold[3].

The resulting image $\widetilde{\mathcal{I}}^{\text{NV}}$ will, in general, presents artifacts and holes, see Figure 6(a). This is expected, since we are rejecting depth estimates that are too uncertain, and we overwrite pixels as we sweep the depth plane from back to front. However, at this stage we are only concerned with generating an initial estimate of the novel view that obeys the geometric constraints captured by the depth probability volumes.

6. Image Refinement

The image $\widetilde{\mathcal{I}}^{\text{NV}}$ synthesized as described in Section 5 is generally affected by apparent artifacts, as shown in Figures 6(a) and (c). Most notably, these include regions that are not rendered, either because of occlusions or missing depth information, and the typical "fattening" of edges at depth discontinuities. Moreover, since we render each pixel independently, structures may be locally deformed. We address these artifacts by training a refinement network that works at the patch level. For a pixel p in $\widetilde{\mathcal{I}}^{\text{NV}}$, we first extract a 64×64 patch $\widetilde{P}^{\text{NV}}$ around it (for clarity of notation, we omit its dependence on p). The goal of the refinement network is to produced a higher quality patch with

[3] An alternative to overwriting the pixels, is to weigh their RGB values with the corresponding depth probabilities. However, in our experiments, this resulted in softer edges or ghosting that were harder to fix for the refinement network (Section 6.1). We speculate that the reason to be that such artifacts are more "plausible" to the refinement network than abrupt and incoherent RGB changes.

(a) (b) (c)

Figure 6: The novel view $\widetilde{\mathcal{I}}^{NV}$, obtained by just warping the inputs, presents several types of artifacts (a). Our refinement network uses the depth probability as well as patches from the input images to fix them (b). More examples of synthesized (top) and refined (bottom) patches are shown in (c).

less artifacts. One could consider the refinement operation akin to denoising, and train a network to take a patch $\widetilde{P}^{\text{NV}}$ and output the refined patch, using a dataset of synthesized and ground truth patches and an appropriate loss function [13, 33]. However, at inference time, this approach would only leverage generic image priors and disregard the valuable information the input images carry. Instead, we turn to the depth probability volume. Consider the case of a ray traveling close to a depth discontinuity, which is likely to generate artifacts. The probability distribution along this ray generally shows a peak corresponding to the foreground and one to the background, see Figure 4. Then, rather than fixing the artifacts only based on generic image priors, we can guide the refinement network with patches extracted from the input views at the locations reprojected from these depths. Away from depth discontinuities, the distribution usually has a single, strong peak, and the synthesized images are generally correct. Still, since we warp the pixels independently, slight depth inaccuracy may cause local deformation. Once again, patches from the input views can inform the refinement network about the underlying structure even if the depth is slightly off.

To minimize view-dependent differences in the patches without causing local deformations, we warp them with the homography induced by the depth plane. For a given disparity $d = \bar{d}$, we compute the warped patch

$$\widehat{P}_{i,j} = \text{W}(P_{i,j}, H_{i\to\text{NV}}^{d=\bar{d}}), \tag{2}$$

where $\text{W}(\cdot, H)$ is an operator that warps a patch based on homography H, and $H_{i\to\text{NV}}^{d=\bar{d}}$ is the homography induced by plane at disparity $\bar{d}$. This patch selection strategy can be seen as an educated selection of a plane sweep volume [5], where only the few patches that are useful are fed into the refinement network, while the large number of irrelevant patches, which can only confuse it, are disregarded. In the next section we describe our refinement network, as well as details about its training.

6.1. Refinement Network

Our refinement strategy, shown in Figure 7, takes a synthesized patch $\widetilde{P}^{\text{NV}}$ and J warped patches $\widehat{P}_{i,j}$ from each input view $\mathcal{I}_i$. The number of patches contributed to each $\widetilde{P}^{\text{NV}}$ can change from view to view: because of occlusions, an input image may not "see" a particular patch, or the patch could be outside of its field of view. Moreover, the depth distribution along a ray traveling close to a depth discontinuity may have one peak, or several. As a result, we need to design our refinement network to work with a variable number of patches.

Network Architecture. We use a UNet architecture for its proven performance on a large number of vision applications. Rather than training it on a stack of concatenated patches, which would lock us into a specific value of J, we apply the encoder to each of the available patches independently. We then perform max-pooling over the features generated from all the available patches and we concatenate the result with the features of the synthesized patch, see Figure 7. The encoder has seven convolutional layers, four of which downsample the data by means of strided convolution. We also use skip connections from the four downsampling layers of the encoder to the decoder. Each skip connection is a concatenation of the features of the synthesized patch for that layer and a max-pooling operation on the features of the candidate patches at the same layer.

Training. We train the refinement network using the MVS-Synth dataset [12]. We use a perceptual loss [13] as done by Zhuo *et al*. [34], and train with ADAM [18]. More details about the network and the training are in the Supplementary.

7. Evaluation and Results

In this section we offer a numerical evaluation of our method and present several visual results. We recommend to zoom into the images in the electronic version of the paper to better inspect them, and to use a media-enabled PDF viewer to play the animated figures.

Execution Time. Using two views as an input, the depth probability volumes take 40s, view synthesis (estimation of

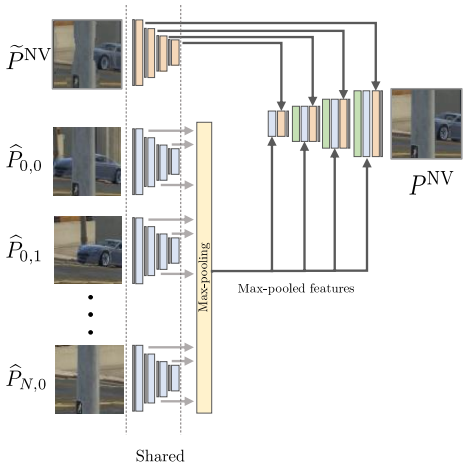

Figure 7: The refinement network takes as input a patch $\widetilde{P}^{\mathrm{NV}}$ from the synthesized image $\widetilde{\mathcal{I}}^{\mathrm{NV}}$, and a variable number of warped patches $\widehat{P}_{i,j}$ from each input view $\mathcal{I}_i$. All patches go through an encoder network. The features of the warped patches are aggregated using max-pooling. Both feature sets are concatenated and used in the decoder that synthesizes the refined patch P^{NV}.

the depth volume in the novel view and rendering) takes 30s, and the refinement network takes 28s (all averages).

Synthetic Scenes. Non-blind image quality metrics such as SSIM [31] and PSNR require ground truth images. For a quantitative evaluation of our proposed method we use the MVS-Synth [12] dataset. The MVS-Synth dataset provides a set of high-quality renderings obtained from the game GTA-V, broken up into a hundred sequences. For each sequence, color images, depth images, and the camera parameters for each are provided. The location of the cameras in each sequence is unstructured. In our evaluation we select two adjacent cameras as the input views to our method and generate a number of nearby views also in the sequence. We then compute the PSNR and SSIM metrics between the synthesized and ground-truth images.

In addition, we can use the same protocol to compare against Stereo Magnification (SM) by Zhou *et al*. [34]. Although SM is tailored towards magnifying the baseline of a stereo pair, it can also render arbitrary views that are not in the baseline between the two cameras. We chose to quantitatively compare against SM because it also addresses the problem of generating extreme views, although in a more constrained setting.

Table 1 shows PSNR and SSIM values for our method before and after refinement, and for SM. The results show that the refinement network does indeed improve the quality of the final result. In addition, the metrics measured on our method output are higher than those of SM.

Metric	Ours Warped	Ours Refined	SM
Mean SSIM	0.851	**0.877**	0.842
Mean PSNR	24.6dB	**27.38dB**	25.49dB

Table 1: Quantitative analysis of our proposed method and SM. "Ours warped" refers to the images produced by backward warping before refinement,"Ours refined" refers to the images created by the refinement network, and "SM" refers to the images created by the method of Zhou *et al*. [34]

Real Scenes. While sequences of real images cannot be used to evaluate our algorithm numerically, we can at least use them for visual comparisons of the results.

We perform a qualitative evaluation and compare against SM on their own data. In their paper, Zhou *et al*. show results when magnifying a stereo baseline by a factor of $4.5\times$. While their results are impressive at that magnification, in this paper we push the envelop to *extreme* and show results for $30\times$ magnification of the input baseline.

Figure 1 and 11 show $30\times$ magnification on stereo pairs of scenes with complicated structure and occlusions. At this magnification level, the results of Zhou *et al*. are affected by strong artifacts. Even in the areas that appear to be correctly reconstructed, such as the head of Mark Twain's statue in Figure 11(left), a closer inspection reveals a significant amount of blur. Our method generates results that are sharper and present fewer artifacts. We also compare against their method at the magnification level they show, and observe similar results, see Supplementary.

The method by Penner and Zhang arguably produces state-of-the-art results for novel view synthesis. However, their code is not available and their problem setting is quite different in that they focus on interpolation and rely on a larger number of input cameras than our method. For completeness, however, we show a comparison against their method in Figure 12. Our reconstruction, despite using many fewer inputs, shows a quality that is comparable to theirs, though it degrades for larger extrapolation.

To validate our method more extensively, inspired by the collection strategy implemented by Zhou *et al*. [34], we capture a number of frame sequences from YouTube videos.

A few of the results are shown in Figure 10. The leftmost column shows the camera locations for the images shown on the right. The color of the cameras matches the color of the frame around the corresponding image, and gray indicates input cameras. We present results for a number of different camera displacements and scenes, showcasing the strength of our solution. In particular, the first three rows show results using only two cameras as inputs, with the virtual cameras being displaced by several times the baseline between the inputs cameras. The third row shows a dolly-in trajectory (*i.e*., the camera moves towards the scene), which is a particularly difficult case. Unfortunately, it may be

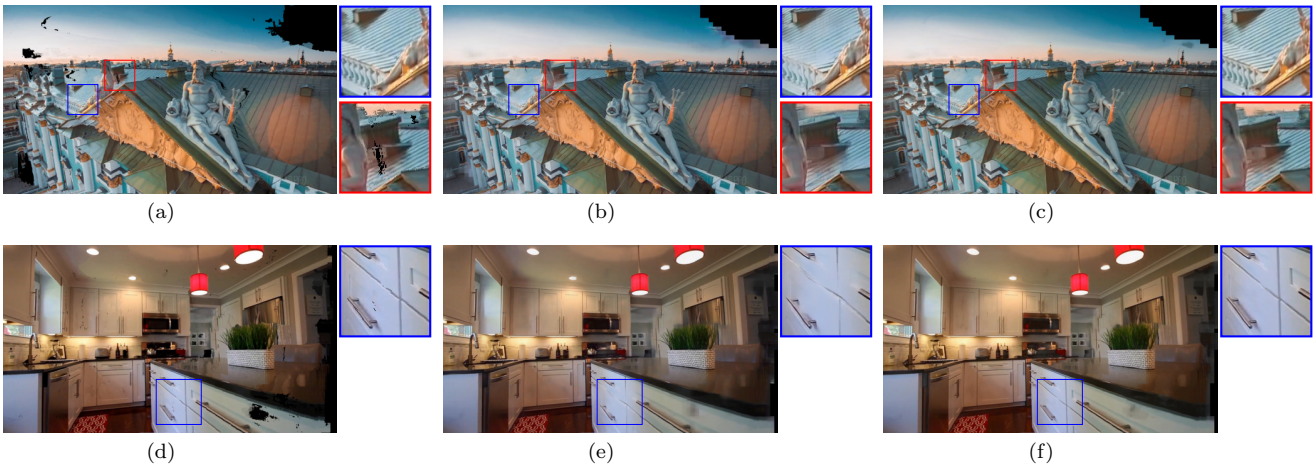

Figure 8: Our refinement network leverages information from relevant patches from the input images. Here (a) and (d) are $\widetilde{\mathcal{I}}^{\mathrm{NV}}$, (b) and (e) are images created by training the network to refine patch only based on image priors, (c) and (f) are our results, which use the patches $\widehat{P}_{i,j}$. The structure of the roof is sharper in (c) compared to (b) and (a). The kitchen cabinet is correctly rendered in (f) compared to (d) and (e).

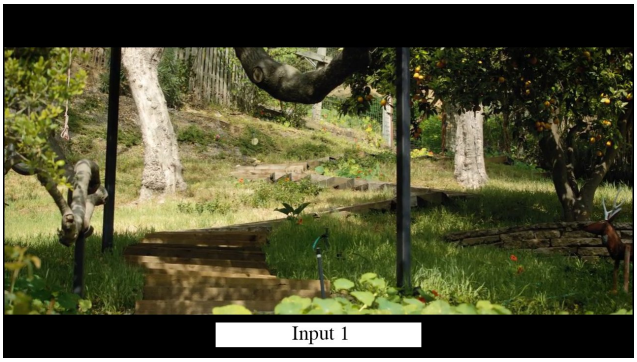

Figure 9: Animation showing the first three scenes in Figure 10. Requires a media-enabled viewer such as Adobe Reader. **Click on the image to start the animation.**

challenging to appreciate the level of extrapolation when comparing images side by side, even when zooming in. However, we also show an animated sequence in Figure 9. To play the sequence, click on the image using a media-enabled reader, such as Adobe Reader. In the Supplementary we show additional video sequences and an animation that highlights the extent of parallax in one of the scenes.

Furthermore, our method can take any number of input images. The last two rows of Figure 10 show two scenes for which we used four input cameras.

Refinement Network. We also conduct an evaluation to show that the use of patches as input to the refinement network does indeed guide the network to produce a better output. Figure 8 shows a comparison between our network and a network with the same exact number of parameters—the architecture differs only in the fact that it does not have additional patches. It can be observed that the proposed architecture (Figure 8(c) and Figure 8(f)) can reconstruct local structure even when the single-patch network (Figure 8(b) and Figure 8(e)) cannot. Indeed, the refinement network guided by patches can synthesize pixels in areas that had previously been occluded.

7.1. Limitations

While the refinement network can fix artifacts and fill in holes at the disocclusion boundaries, it can not hallucinate pixels in areas that were outside of the frusta of the input cameras—that is a different problem requiring a different solution, such as GAN-based synthesis [30]. The refinement network also struggles to fix artifacts that look natural, such as an entire region reconstructed in the wrong location.

Finally, because the depth values are discrete, certain novel views may be affected by depth quantization artifacts. A straightforward solution is to increase the number of disparity levels (at the cost of a larger memory footprint and execution time) or adjust the range of disparities to better fit the specific scene.

8. Conclusions

We presented a method to synthesize novel views from a set of input cameras. We specifically target *extreme* cases, which are characterized by two factors: small numbers of input cameras, as few as two, and large extrapolation, up to 30× for stereo pairs. To achieve this, we combine traditional geometric constraints with the learned priors. We show results on several real scenes and camera motions, and for different numbers of input cameras.

Figure 10: Extreme view synthesis on two-camera inputs and on four-camera inputs. For each row the cameras on the left show the position of the input views (light gray) and virtual views. The color of the pictures' frames matches the color of the corresponding camera on the left. The cameras on the left are rendered at the same scale to facilitate a comparison between the amounts of extrapolation in each case.

Figure 11: Comparison with Stereo Magnification for a 30× baseline magnification. While some unavoidable artifacts are visible in both methods, our results have fewer, less noticeable artifacts, and are generally sharper. Please zoom in for the best viewing experience.

Acknowledgments

The authors would like to thank Abhishek Badki for his help with Figure 10, and the anonymous reviewers for their thoughtful feedback.

References

[1] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *ACM SIGGRAPH*, 2001. 1, 2

(a) (b)

Figure 12: Comparison with Soft3D [24]. (a) A frame generated by Soft3D, which uses all the cameras in the sequence from Chaurasia *et al*. [3], and a frame generated by our method using only two input cameras around the middle of the sequence (b).

[2] Gaurav Chaurasia, Sylvain Duchêne, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. In *ACM Transactions on Graphics*, 2013. 2

[3] Gaurav Chaurasia, Olga Sorkine, and George Drettakis. Silhouette-aware warping for image-based rendering. In *Eurographics Symposium on Rendering*, 2011. 8

[4] Shenchang Eric Chen and Lance Williams. View interpo-

lation for image synthesis. In *ACM SIGGRAPH*, 1993. 1, 2

[5] Robert T Collins. A space-sweep approach to true multi-image matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1996. 5

[6] Paul E Debevec, Camillo J Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *ACM SIGGRAPH*, 1996. 2

[7] 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. 2

[8] Silvano Galliani, Katrin Lasinger, and Konrad Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *IEEE International Conference on Computer Vision (ICCV)*, 2015. 3

[9] Ned Greene. Environment mapping and other applications of world projections. In *IEEE Computer Graphics and Applications (CGA)*, 1986. 2

[10] Li-Wei He, Jonathan Shade, Steven Gortler, and Richard Szeliski. Layered depth images. In *ACM SIGGRAPH*, 1998. 2

[11] Asmaa Hosni, Christoph Rhemann, Michael Bleyer, Carsten Rother, and Margrit Gelautz. Fast cost-volume filtering for visual correspondence and beyond. 2012. 4

[12] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. DeepMVS: Learning multi-view stereopsis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3, 5, 6

[13] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision (ECCV)*, 2016. 5

[14] Nima Khademi Kalantari, Ting-Chun Wang, and Ravi Ramamoorthi. Learning-based view synthesis for light field cameras. In *ACM Transactions on Graphics (SIGGRAPH)*, 2016. 2

[15] Abhishek Kar, Christian Häne, and Jitendra Malik. Learning a multi-view stereo machine. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 3

[16] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 3

[17] Sameh Khamis, Sean Ryan Fanello, Christoph Rhemann, Adarsh Kowdle, Julien P. C. Valentin, and Shahram Izadi. StereoNet: Guided hierarchical refinement for real-time edge-aware depth prediction. In *European Conference on Computer Vision (ECCV)*, 2018. 3

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 5

[19] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected CRFs with Gaussian edge potentials. In *Advances in Neural Information Processing Systems (NIPS)*. 2011. 3

[20] Marc Levoy and Pat Hanrahan. Light field rendering. In *ACM SIGGRAPH*, 1996. 1, 2

[21] Zhengqi Li and Noah Snavely. MegaDepth: Learning single-view depth prediction from internet photos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[22] Chao Liu, Jinwei Gu, Kihwan Kim, Srinivasa G Narasimhan, and Jan Kautz. Neural RGB $\rightarrow$ D sensing: Depth and uncertainty from a video camera. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[23] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. In *ACM Transactions on Graphics (SIGGRAPH)*, 2019. 2

[24] Eric Penner and Li Zhang. Soft 3D reconstruction for view synthesis. In *ACM Transactions on Graphics (SIGGRAPH)*, 2017. 2, 4, 8

[25] Ashutosh Saxena, Sung H Chung, and Andrew Y Ng. Learning depth from single monocular images. In *Advances in Neural Information Processing Systems (NIPS)*, 2006. 3

[26] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 3

[27] Pratul P Srinivasan, Richard Tucker, Jonathan T Barron, Ravi Ramamoorthi, Ren Ng, and Noah Snavely. Pushing the boundaries of view extrapolation with multiplane images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[28] Pratul P Srinivasan, Tongzhou Wang, Ashwin Sreelal, Ravi Ramamoorthi, and Ren Ng. Learning to synthesize a 4D RGBD light field from a single image. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 2

[29] Shubham Tulsiani, Richard Tucker, and Noah Snavely. Layer-structured 3D scene inference via view synthesis. In *European Conference on Computer Vision (ECCV)*, 2018. 2

[30] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional GANs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8798–8807, 2018. 7

[31] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: From error visibility to structural similarity. In *IEEE Transactions on Image Processing (TIP)*, 2004. 6

[32] Junyuan Xie, Ross Girshick, and Ali Farhadi. Deep3D: Fully automatic 2D-to-3D video conversion with deep convolutional neural networks. In *European Conference on Computer Vision (ECCV)*, 2016. 2

[33] Hang Zhao, Orazio Gallo, Iuri Frosio, and Jan Kautz. Loss functions for image restoration with neural networks. In *IEEE Transactions on Computational Imaging (TCI)*, 2017. 5

[34] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo Magnification: Learning view synthesis using multiplane images. In *ACM Transactions on Graphics (SIGGRAPH)*, 2018. 1, 2, 5, 6

[35] 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. 2

[36] C. Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon A. J. Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. In *ACM Transactions on Graphics (SIGGRAPH)*, 2004. 2