

Self-Guided Novel View Synthesis via Elastic Displacement Network

Yicun Liu^{1,2†} Jiawei Zhang^{1*} SenseTime Research¹ Ye Ma¹ Jimmy S. Ren¹ Columbia University²

yicun.l@columbia.edu {zhangjiawei,maye,rensijie}@sensetime.com

Abstract

Synthesizing a novel view from different viewpoints has been an essential problem in 3D vision. Among a variety of view synthesis tasks, single image based view synthesis is particularly challenging. Recent works address this problem by a fixed number of image planes of discrete disparities, which tend to generate structurally inconsistent results on wide-baseline, scene-complicated datasets such as KITTI. In this paper, we propose the Self-Guided Elastic Displacement Network (SG-EDN), which explicitly models the geometric transformation by a novel non-discrete scene representation called layered displacement maps (LDM). To generate realistic views, we exploit the positional characteristics of the displacement maps and design a multi-scale structural pyramid for self-guided filtering on the displacement maps. To optimize efficiency and scene-adaptivity, we allow the effective range of each displacement map to be 'elastic', with fully learnable parameters. Experimental results confirm that our framework outperforms existing methods in both quantitative and qualitative tests.

1. Introduction

The perception of the 3D shape in our world is a long pursuing problem for computer vision research. Among a variety of 3D vision tasks, reasoning from limited data or partial observations is considered to be quite challenging, including novel view synthesis: given one or more views of a 3D scene, predict the projected image of the same scene, but from a previously unseen viewing angle.

Although human beings can do well in hallucinating the view from another perspective, computers still find it a challenging task. Over the past decades, researchers propose to conduct view interpolation with a predefined 3D model from multiple views [37, 3]. They usually demand a relatively large amount of multi-view images (which are generally more than two images) as input, additional information



Figure 1. Challenging examples in KITTI [20]. For thin objects (e.g. poles) and smooth textures (e.g. car surface), structural consistency could be a major issue in novel view synthesis task. Our proposed framework is designed to handle the complex scenes for wide-baseline stereo pairs, with effective scene representation and self-guided filtering to enforce the structural consistency.

such as depth maps as an auxiliary hint. For a long time, there have been very few practically usable frameworks for single image based view synthesis. With recent rise of deep learning, several works attempt to addressed this problem by learning-based appearance warping [43], shifting planes [39], spatially-variant kernels [23], and homographies [18].

Despite the progress, the single-image novel view synthesis problem is far from being solved. The first issue is structure coherence. As shown in Figure 1, when applied on high-resolution stereo pairs of wide baselines and complex scenes, these methods usually give structurally incon-

[†]Work was done during internship at SenseTime Research. *Corresponding author.



Figure 2. For novel view synthesis problem, pixel correspondence should not be restricted to one-to-one (e.g., disparity warping). The toy example shows a scenario in which one red pixel vanishes, and one blue pixel emerges, which is not one-to-one.

sistent results. Small and thin objects usually suffer severe deformation, and ambiguity emerges in texture-less areas. The main reason behind this is that it directly depends on photometric reconstruction loss as supervision thus cannot enforce structural consistency.

The second issue is inflexibility in pixel correspondence. As shown in Figure 2, unlike other 3D vision tasks, 'pixel shift' in view synthesis requires a more flexible pixel correspondence setting. In transformation between two viewpoints, pixels can vanish, surfaces can shrink, objects can be de-occluded. Only using pixel-to-pixel model or wrapping by disparity cannot well formulate the process; instead, we need a flexible pixel correspondence model that naturally supports many-to-many pixel correspondence.

The third issue is continuity versus efficiency. On the one hand, plane-sweep based methods like Deep3D [39] and MPI [42, 32] approximate the transformation by a small fixed number of shifting planes, which is computationally efficient but does not support fine-grained disparities in two consecutive planes. On the other hand, spatially-variant kernels [5] have continuous shifting range but come with high computation and memory cost. Therefore, we need an effective 3D representation for learning the view synthesis task, which should be able to model any shifting operation with a continuous range, and of low cost.

To address the three issues, we proposed the Self-Guided Elastic Displacement Network (SG-EDN). We design a new 3D representation in the spatial domain by modeling the pixel displacement between two views. It supports the many-to-many pixel correspondence desired in the view synthesis task. Modeling the pixel displacement also guarantees that objects on the displacement map lie in the same position as the objects in the input image. Taking advantage of that property, we embed a guided filtering processs in the network to enforce structural consistency between the input image and the displacement maps. To leverage efficiency and continuity, we introduce a new3D representation for view synthesis task, which is Layered Displacement Maps (LDMs), which is inspired by multi-plane image (MPIs) [42] and layered depth images (LDIs) [10].

Different from prior representations where each plane

only corresponds to a fixed depth or disparities, each displacement map in LDM corresponds to a continuous range of displacements. Furthermore, instead of manually setting the displacement range for each interval, we create a small module inside the network to learn the optimized choice for each interval. This design ensures that each interval is maximally utilized, where the change from 'static' to 'elastic' grants more flexible displacement predictions with less computational costs.

In short, our contributions are threefold:

- We propose a new framework SG-EDN for the challenging task of novel view synthesis from a single image. The network explicitly enforces structural and geometric consistency by using the input image as selfguidance for spatial transformation estimation.
- We derive a new 3D representation layered displacement maps (LDM) for the novel view synthesis task. Different from previous presentations that use discrete approximated depths, our representation supports any continuous prediction for displacements.
- We introduce an elastic module that can learn the effective range of displacement maps based on input data distribution. This design grants more flexibility in displacement predictions with less computational cost.

2. Related Work

Rendering-Based NVS View synthesis has been extensively studied in the traditional computer graphics. Many previous works have been proposed to consider novel view synthesis as a rendering problem, including interpolating rays from dense imagery and light fields [8, 17, 21], reconstructing the 3D scene representation based on multiple inputs images [19, 11]. These methods mainly require multiple views (usually more than 3) as inputs. Besides methods setting a high standard for input data, there are other approaches that require a bank of known 3D structures or textures as an example for view synthesis, including instance-based 3D model matching [27], and texture matching [22].

Geometry-Based NVS In computer vision, NVS receives more attention on the correctness of the predicted geometry [30, 31]. Instead of depending on dense reconstruction in the rendering process [8, 17], many works consider the occluded areas in multi-view stereopsis, where some pixels are missing in other views. Traditional approaches first reconstruct a partial scene with non-occluded areas, then fill the holes of occlusion by additional information like depth [44]. While such methods have relatively robust geometry modeling, they need multi-view images as input, as long as depth information as an additional guide.

Learning-Based NVS, The powerful capability of deep neural network inspires later works to learn from diverse scene data. DeepStereo [6] is proposed as one of the pioneering works to learn novel view synthesis from the world's imagery. Learning-based rendering systems' performance has also been improved by using more delicate deep architectures [15, 33, 25]. In the meanwhile, recent research has also addressed the importance of conducting synthesis based on fewer input images. A learning-based rendering system is proposed in [42], which takes two views from a narrow-baseline stereo camera and use view extrapolation to magnify the baseline to fit wider baselines. There is only few research attempt to tackle the view synthesis problem from a single image. Appearance flow [43] estimates the photometric difference to model the transformation between the views. Deep3D [39] proposes to generate 3D movies from 2D videos. Beside scene-level, objectivelevel view synthesis is also studied in [33, 24, 40].

Structural-Geometric Coherence For novel view synthesis from a single image, preserving structural and geometric coherence is especially hard. One of the main reasons is that a single image as input does not have rich geometric cues to enforce structural constraints. Few prior works have explicitly modeled such constraints for single-image NVS. [28] proposed a collection of 3D models and used the most similar exemplar as a guide for an input image. New methods use plane-sweep with region selection mechanism to model the geometry [18, 39, 42], but the depth corresponding to each selected planes is discrete, and no further structural constraints for thin objects is enforced.

3. Our Approach

The task of single image-based novel view synthesis (NVS) can be formulated as follows: Given an input image I, we aim to predict the latent novel view \hat{I} from a different viewing angle. In a rectified stereo setting, this scenario can be simplified into the following: For a rectified stereo pair with a specific baseline,¹ given image I_L from the left camera, our goal is to predict image I_R for the right camera.

3.1. LDM Scene Representation

As shown in Table 1, there are potentially many ways to represent the spatial transformation between the input image I_L and output image I_R . A proper scene representation for novel view synthesis should not only encode the geometric information in its design but also be efficient and continuous, meaning that fine-grained spatial transformation of any range can be well modeled by this representation.

One intuitive approach is to use disparity to warp the left view to the right view. However, disparity warping makes an overly strong assumption: we only need one-to-one pixel correspondence to solve the problem. As shown in Figure 2, the transformation from the input view to the novel view



Figure 3. Scene representations for NVS task. All shapes are of 45 degrees intersection with the camera plane. MPI [42] representation uses a number of image planes with discrete depth (in the example is 2,4,8) for approximation, which is suboptimal for these shapes in between the planes. Our LDM representation is shown in the right, where each plane corresponds to a displacement interval ([0,2], [2,4], or [4,8]). This design supports continuous displacement and can model any displacement within its range.

usually involves a sheer transformation or occlusion. They are not one-to-one but many-to-many pixel correspondence.

To capture the many-to-many pixel correspondence, we propose a new scene representation called layered displacement maps (LDM). The idea is is inspired by layered depth images (LDI) in [10, 35] and multi-plane images (MPI) in [42], which constructs n image planes P of discrete depth d_k to describe the 3D scene for view synthesis:

$$\{P^{d_1}, ..., P^{d_k}, ..., P^{d_n}\}, P^{d_k}_{i,j} \in [0, 255]$$

where P_{d_k} is a slice of plane-sweep from the input image I_L of depth d_k . In fact, P_{d_k} shares the same size as I_L and I_R , which is $H \times W \times 3$. As the modeling assumption indicates, P_{d_k} should include objects around depth d_k in the scene. Applying *n* planes with discrete depth as approximation has its limitations: (1) not capable of capturing depth in between the planes (2) inconsistent for objects across multiple planes. Although this approximation somehow works for datasets with small baselines and short-range displacement (such as human stereopsis), it is not theoretically sound when applied on wider baseline datasets with long-range spatial transformation.

To overcome the aforementioned drawbacks, we build the scene representation in the spatial domain by modeling the displacement between the views, instead of directly predicting the appearance of the novel view. One simplified illustration of our representation is shown in Figure 3. Rather than directly predicting the multi-plane RGB images after the shift operation, our representation predicts a set of displacement maps (planes) D to describe the displacement per pixel:

$$\{D^1, ..., D^k, ..., D^n\}, D^k_{i,j} \in [\lambda_{k-1}, \lambda_k]$$

where each map D^k is of size $H \times W \times 1$ and should be re-

¹Baseline varies on datasets, from around 1cm (dual-cameras on mobile phones), to 6-8cm (human stereopsis), all the way up to 50-100cm (KITTI). More substantial baseline usually corresponds to more challenges like occlusion and long-range displacement.

Table 1. Various scene representations for NVS task. We compare the memory cost (* denotes approximation on KITTI dataset), domain of modelling (ST for spatial transformation domain, AP for appearance domain), number of inputs, continuity, and pixel correspondence.

	Disparity Warpping [7]	Selection Mask [39]	Dynamic Filters [23]	MPI [42, 32]	LDM (Ours)
Mem. Cost	$H \times W \times 1$	$H \times W \times 192$	$H \times W \times 192$	$H \times W \times 150^*$	$H \times W \times 24$
Operation Domain	ST	AP	ST	AP	ST
Num. of Inputs?	1	1	1	2	1
Continuous?	1	X	X	X	1
Many-to-Many?	×	1	✓	1	1

sponsible for estimating a specific range of the displacement $[\lambda_{k-1}, \lambda_k]$. Thus the displacement prediction of each pixel could lie in a continuous range. The upper bound λ_{k-1} and lower bound λ_k of the range can be manually determined or learned from the data distribution. For a pixel with position i, j, the pixel correspondence from the input left view I_L to the novel right view I_R can be formulated as:

$$I_R(i,j) = \sum_{k=1}^n I_L(i+D_{i,j}^k,j)$$
(1)

where the final pixel at $I_R(i, j)$ is an aggregation of at most n plausible pixels at different displacement range from I_L . As discussed before, the LDM representation could degrade to disparity map with a special case of n = 1, and $\lambda_0 = 0$, $\lambda_1 = max(disp)$. To capture the potential one-to-many pixel correspondence, we will discuss the choice of n in our experiment section. In practice, objects at the front should have a longer range of horizontal shift, which is captured by D^k with intervals lying in large displacement.

Nevertheless, directly combining n plausible pixels from the source view does not agree with the internal 3D geometry. The source of rare-occurring pixels (e.g., thin objects in KITTI dataset) is strictly restricted and might not be as many as n. Besides, occlusion is not well-formulated in the representation, which means after the displacement, the pixels for objects at the back should be replaced by pixels of the objects in the front. To alleviate such a problem, we additionally introduce a set of confidence maps C in our LDM representation:

$$\{C^1, ..., C^k, ..., C^n\}, C^k_{i,j} \in [0, 1]$$

where C^k is of size $H \times W \times 1$, and $C_{i,j}^k$ corresponds to the transparency of the pixel i, j when aggregating n plausible pixels. So the pixel correspondence now becomes:

$$I_R(i,j) = \sum_{k=1}^n C^k(i + D_{i,j}^k, j) * I_L(i + D_{i,j}^k, j)$$
(2)

where * denotes the dot product. In our SG-EDN design, the displacement map D^k , the confidence map C^k , and the upper bound λ_k and lower bound λ_{k-1} can all be learned inside our SG-EDN framework.

3.2. Self-Guided Structure Filtering

The benefits of adopting our LDM representations are not limited to its capability depicting inherent geometry in a continuous manner. Moreover, predicting the spatial transformation instead of the directly predicting the shifted RGB images grants a crucial characteristic: Differs from shifted RGB images with already different positions, objects on the displacement map lie in the same position as in objects in the input image plane. In other words, where there is an edge or structure in the image plane, there should be a similar edge at the displacement maps; the only unknown issue is which map. Taking advantage of that characteristic, we could exploit the structural hints for objects in the input image to refine the object shape of the corresponding (activated) displacement map.

Edge-aware filters are particularly proficient in revealing the structures and edges of an object but ignoring smooth textures in the object surface. When predicting the novel view, regions with smooth textures and similar color to the background usually result in ambiguities for matching, which is a hard-to-solve problem for view synthesis and stereo matching. However, edge-aware filtering can be operated on the spatial domain of our LDM representations, where the objects in the displacement map are usually smooth inside with an edge, which can be refined by structural hints from the input view.

To enforce the structural consistency, we proposed to embed deep guided filter [38] in our network so that the input image I_L serves as a self-guidance for structure hints. Guided image filter was originally proposed in [9]. It assumes that the filter output O_i is a linear transform of guidance G_i in any window ω_k centered at pixel k:

$$O_i = a_k G_i + b_k, \forall i \in \omega_k, \tag{3}$$

where *i* is the index of the pixel. The linear coefficients (a_k, b_k) is assumed to be constant in the window w_k . The local linear model ensures that *O* has an edge only if *G* has an edge because $\nabla O = a \cdot \nabla G$. In addition, the filter output O_i should be similar to the input P_i with the constraint:

$$E(a_k, b_k) = \sum_{k:i\in\omega_k} ((O_i - P_i)^2 + \epsilon a_k^2), \qquad (4)$$

where ϵ is a regularization parameter. By minimizing $E(a_k, b_k)$, we can get the filtered output O. We use n_i to denote the difference between O_i and P_i . In our task, we use the left view I_L as the guide G, and the *s*-th displacement map D^s of the LDM as the output O. One realworld example showing this effect of the filtering process is shown in Figure 4. According to the property of guided filter, the gradient in the *s*-th displacement map ∇D^s should



Figure 4. Illustration of the guided filtering process. We use a patch from KITTI dataset [20] to demonstrate the filtering effect. Since the image in the unfiltered displacement map shares the same position with the original input image, we are able to use left image as structure guide for filtering. After filtering, edges in the output displacement map is closer to the original shape.

be structurally consistent with the gradient of the guide image ∇I_L . We take advantage of the edge-preserving property of the deep guided filters, which help to refine the displacement maps based on the shape knowledge, thus lead to structurally robust predictions.

3.3. Elastic Displacement Network

The overview of our SG-EDN framework is shown in Figure 6, which contains several components: a feature extractor to extract features from the input view and encode the spatial-domain information; a structure pyramid to extract crucial structural hints as self-guidance; a transformation estimator linked with the previous feature extractor to refine the shape and edge of the prediction using the self-guidance; finally a blending components to turn the predicted displacement maps, confidence maps, and interval bounds into the LDM representation and render the final novel view. Additionally, we insert two deep guided filter layers after the last two upsampling operations, which take guidance maps of half-size/full-size respectively and refine the structure of the prediction in a coarse-to-fine manner.

Backbone Network We choose design similar to U-Net [29] as the backbone for our feature extractor and transformation estimator. We made several modifications to it for our view synthesis task: for the convolution layers in the feature extractor (the encoder part), we use dilated convolution of dilation 1,3,5 consecutively, which is found to be helpful in generating fine-grained predictions for small objects in the semantic segmentation task [4]. Adding dilation into the feature extractor aims at increasing the receptive fields, which could be critical in long-range displacements for wide baseline novel view synthesis. Another modi-



(a) Unfiltered Displacement Map (b) Filtered Displacement Map

(c) Final Displacement Map (aggregated with confidence map)

Figure 5. Intermediate displacement maps. The three rows are displacement maps corresponding to the foreground, middleground, and background. We use the heat map to visualize the value of the displacement map (purple denotes low and yellow denotes high). With the structural guide, some of the noise in the unfiltered displacement maps can be removed. It can be observed that the final displacement maps are with most refined edges.

fication is that we use the dense upsampling convolution (DUC) [36] to substitute the deterministic bilinear upsampling layer. Doing that could ensure the upsampling process to be learnable, with a light-weighted extra cost in the trade of better scene adaptivity.

Structure Pyramid As for the design of the structure pyramid, we use several convolution layers with an average pooling layer to generate self-guidance with full image size and half image size. Considering each displacement map D^k corresponds to its range $[\lambda_{k-1}, \lambda_k]$, where ideally only the shapes of displacement within that range should appear on the map. The convolution layer generates a $H \times W \times n$ self-guidance, where each channel corresponds to one displacement map.

Blending There are three branches of our network for three types of output: the displacement maps D of size $H \times$ $W \times n$; the corresponding confidence maps $H \times W \times n$, with a sigmoid layer to restrict its value between 0 and 1; and λ_k to determine the range of each displacement map. Instead of setting the range to be uniform, we find that for each image in a view synthesis dataset, the distribution of the pixel displacements could be divergent. For some popular displacement ranges where most pixels lie within the range, it should have denser displacement maps for fine-grained estimation. For some range with scant pixel distribution, one or two displacement maps should be enough to cover. From that intuition, we design a small module to predict the range of each displacement map and let the range to be 'elastic' for different images and datasets. For the range prediction branch, we first apply average pooling to shrink the spatial size of the feature maps, considering the range prediction only cares about the overall distribution. After several convolutions, we flatten the feature maps channel-



Figure 6. Illustration of our SG-EDN framework. The upper left part of this figure describes the high-level functionality of each module. The rest of the figure illustrates the detailed design of each module. There are two branches for the input view: the main branch is a feature extractor connected with a spatial transformation estimator; the supplementary branch includes a convolutional pyramid to generate self-guidance in full and half resolution, which is later used in the guided filtering process to preserve structural consistency. The backbone network predicts the displacement maps D, confidence maps C, and displacement range parameters λ , which later constructs the LDM representation. The last step is the blending process based on the LDM representation, which essentially shifts the pixels in the original input view by differential sampling. After that, the novel view is obtained as output.

wisely and feed the feature vector into a fully connected layer and a softmax layer, which output a probability vector v of size $n \times 1$. The final range λ_k is calculated by:

$$\lambda_k = \frac{v_k}{\sum_i^{k-1} v_k} \lambda_{max} \tag{5}$$

where λ_{max} is a predefined parameter and usually correlated with the maximum disparity of the dataset. With λ_k , we can re-normalize the values in the displacement map:

$$\bar{D^{k}} = \frac{D^{k} - \min(D^{k})}{\max(D^{k}) - \min(D^{k})} * (\lambda_{k} - \lambda_{k-1}) + \lambda_{k-1}$$
(6)

Here we use \overline{D}^k to denote displacements maps after normalization. To perform the final displacement, we use the differentiable image sampling [13] to obtain the final novel view image I_R :

$$I_R(i,j) = \sum_{k=1}^n C^k(i,j) * I_L(i + \bar{D}^k_{i,j},j) * \Phi(i + \bar{D}^k_{i,j},j)$$
(7)

where $C^k(i, j)$ is the corresponding confidence map, and Φ denotes the sampling kernel, here we use the bilinear kernel for 2D differentiable sampling.

3.4. Optimization

As discussed in Figure 2, different from fixed regression tasks like depth estimation, the nature of novel view synthe-

sis problem involves complicated 'one-to-many' pixel correspondences. Since there is no direct supervision for the inherent scene representation, we aims to learn the inherent scene representations by indirect supervision between the predicted novel view I_R and the ground-truth $\hat{I_R}$

Photo Loss The first loss is the photometric loss consisting of the l_1 -loss and MS-SSIM, where the l_1 loss supervise the pixel-level predictions and MS-SSIM should reflect the structural differences for scale-variant objects:

$$\mathcal{L}_{photo} = \alpha \cdot \|\hat{I}_R - I_R\|_1 + (1 - \alpha) \cdot (\hat{I}_R, I_R)_{\text{SSIM}} \quad (8)$$

and the value of α is the balancing hyperparameter, which is set to be 0.84 in our experiment as suggested in [41].

Content Loss To consider the context of large local regions and their high-level semantic differences. We propose to use the perceptual loss proposed in [14]. It is defined as the l_2 -norm between feature representations of the restored right view \hat{I}_R and the latent right view I_R :

$$\mathcal{L}_{feat} = \frac{1}{\mathcal{C}_j \mathcal{H}_j \mathcal{W}_j} \left\| \psi_j(\hat{I}_R) - \psi_j(I_R) \right\| \tag{9}$$

where $\psi_j()$ denotes the feature map from the *j*-th VGG-19 convolutional layer and C_j , \mathcal{H}_j , \mathcal{W}_j are the number, height and width of the feature maps, respectively.

Adversarial Loss Although I_R can well supervise most regions of the scenes, there are always some regions needing generative modeling. For the pixels that are previously



Figure 7. Qualitative evaluation on KITTI dataset. For each scene, from the top to the bottom are SepConv [23], Deep3D [39], LRDepth [7], and SG-EDN (ours), ground-truth right image. Each of the four scene contains typically challenging regions for novel view synthesis: small objects like poles and signs, and ambiguous texture-less areas like the the white walls.

occluded in the input view or objects with larger projection areas in the novel view, there exist many plausible solutions other than I_R . Simply regressing on I_R would lead to blurry results for such regions. Inspired by the insights in image inpainting task [26], we apply the adversarial loss \mathcal{L}_{adv} to determine whether the generated region is realistic or not. We configure our discriminator with architecture similar to Pix2Pix [12], but only inputs the 3-channel novel view I_R to the discriminator. The overall loss function for the guided view synthesis framework is:

$$\mathcal{L}_{total} = \beta \cdot \mathcal{L}_{photo} + (1 - \beta) \cdot \mathcal{L}_{feat} + \sigma \cdot \mathcal{L}_{adv} \quad (10)$$

where β and σ are hyper-parameters to balance loss terms.

4. Experiments

4.1. Experimental Settings

Datasets We first set up the comparison on KITTI dataset [20], which contains various types of scenes with a large baseline, with a large range of displacements. We train all the models on the same dataset if it is not originally trained on KITTI, using the official release code. We use KITTI RAW for training, which contains a total of 42382 rectified stereo pairs captured from 61 scenes. We benchmark all the models on the images provided for testing in KITTI 2015 dataset. Additionally, we include the MPI Sintel Stereo [2] dataset and an additional 3D movie dataset for ablation study and visual quality examination.

Evaluation Metrics We use the same metrics in previous works [18, 39]: RMSE, PSNR, and SSIM, where the first two metrics focus on evaluating the pixel-wise differences, and SSIM aims at assessing the structural differences, which is closer to human visual perception and more crucial to 3D vision task. Additionally, considering our task

Table 2. Benchmark on KITTI dataset. Our model outperforms the previous methods. We include SSIM and Inception score as addition to pixel-level metrics, producing geometrically correct and photometrically pleasing view is more crucial.

	RMSE↓	PSNR ↑	SSIM↑	Inception
SepConv	28.03	19.17	0.685	3.469
Deep3D	27.46	19.65	0.692	3.591
LRDepth	30.49	18.62	0.663	3.624
Ours(n=16, static)	26.54	19.93	0.713	3.573
Ours(n=16, elastic)	25.97	20.01	0.716	3.621
Ours(n=24, static)	26.05	20.35	0.720	3.782
Ours(n=24, elastic)	25.36	20.79	0.725	3.789

is a synthesis task, we also include the learning-based inception score [1] as additional image quality and realism metric. We random sample small patches and feed into InceptionV3 [34] network to get the inception score.

Implementation Details In the training step, we configure the deep guided filtering layer with radius r of 3 pixels and the searching threshold ϵ of $1e^{-2}$. We set the three balancing factors α , β and σ to be 0.84 (as suggested in [41]), 0.5, and 0.1 respectively. For KITTI dataset, we set the λ_{max} to be 192. For MPI Sintel dataset, we set the λ_{max} to be 96. For KITTI dataset, we use the size of 1280 * 320 for training, which is very close to the original size of KITTI images. For MPI Sintel dataset, we use the size of 800 * 320 for training. We adopt Adam [16] optimizer to for our network with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. For the training strategy, the initial learning rate is set to be $1e^{-5}$ and is set to be multiplied by 0.9 after every epoch.

4.2. Comparison and Benchmarks

Quantitative Results The results are shown in Table 2, where we include the SepConv [22], Deep3D [39] and LRDepth [7] as comparison. We particularly emphasize on SSIM, which is critical for visual perception and geome-



Figure 8. Comparison of results w. and w/o. structural guidance. In each column, from top to bottom are: the left view as input, novel view generated without the structure guide, novel view generated with structure guide, the ground-truth right view. It can be observed that structure consistency in the ambiguous regions and thin objects has been significantly improved with our design.

try reasoning. It can be observed that with our baseline setting of 16 displacement maps and no elastic modules, our framework already archives better performance, which mainly contributed by the design of continuous scene representation and the structure-guided filtering process.

Qualitative Results For qualitative evaluation, we carefully examine the KITTI testing dataset and choose several images as the most challenging cases for our assessment. The scene in the images should contain small, thin, or rarely-occurring objects, ambiguous texture-less regions which are hard for pixel correspondence search. An overall qualitative comparison is shown in Figure 7, where our framework specializes in challenging cases and generate more structurally convincing results.

4.3. Performance Analysis

Framework Visualization We visualize the intermediate displacement maps before and after the guided filtering process. The visualization is shown in Figure 5. One interesting fact is that each displacement layer contains noisy regions before the filtering, and each structure guide map specifically emphasizes different structural components of the scene. After the guided filtering process, the contents in the displacement maps are more focused on the objects in their assigned ranges. A close-up illustration of the guided filtering process is also in Figure 4, which uses a simple object to demonstrate the mechanism behind the filtering layer, as well as why shared object position in input image and displacement maps matters.

LDM Configuration We include the comparison of numbers of displacement maps and the range assigning mechanism. There are two choices or mechanism, elastic



Figure 9. Qualitative results on 3D movie dataset. Since SepConv [23] aims at video interpolation in the original paper, we train our model and SepConv on our 3D movie dataset and test its visual quality. For each scene, from top to bottom are: results from SepConv [23], results from our framework, the groundtruth right view. Although the scene is not as challenging as KITTI, we can still observe sharper and more realistic results in our framework.

or static, where elastic means our design with learned λ_k range parameter, and static means that we uniformly assign the range according to the number of displacement maps. As shown in Table 2, we observe improvements in evaluation metrics when we increase the number of n from 16 to 24, also when changing from static to elastic.

Guided Structure Filtering We use a controlled-variant setting to check the real effect of the guided filtering process on MPI Sintel dataset[2]. The metric for evaluation is shown in Table 3. Challenging cases containing varying ranges of displacements are shown in Figure 8.

Table 3. Effect of structure guide on MPI Sintel datset. We train two networks w. and w/o. guided filtering and structure pyramids.

	RMSE↓	PSNR ↑	SSIM↑
Ours(w/o. structure guide)	26.78	20.21	0.718
Ours(w. structure guide)	24.29	21.65	0.745

5. Conclusion

Our work presents the self-guided elastic displacement network (SG-EDN) and a new scene representation called the layered depth map(LDM). To tackle the difficult task of novel view synthesis for datasets with complex scenes, long-range displacements, we propose to use the structural hints from the input view as the structural guide and create a new scene representation embedded in our framework. To leverage both efficiency and performance, we design an elastic module for learning the range distribution from input images. Experiments show that our framework excels in structural and geometric consistency.

References

- [1] S. Barratt and R. Sharma. A note on the inception score. arXiv preprint arXiv:1801.01973, 2018.
- [2] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, 2012.
- [3] G. Chaurasia, S. Duchene, O. Sorkine-Hornung, and G. Drettakis. Depth synthesis and local warps for plausible image-based navigation. *TOG*, 2013.
- [4] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587, 2017.
- [5] B. De Brabandere, X. Jia, T. Tuytelaars, and L. Van Gool. Dynamic filter networks. In *NIPS*, 2016.
- [6] J. Flynn, I. Neulander, J. Philbin, and N. Snavely. Deepstereo: Learning to predict new views from the world's imagery. In *CVPR*, 2016.
- [7] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017.
- [8] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The lumigraph. In SIGGRAPH, 1996.
- [9] K. He, J. Sun, and X. Tang. Guided image filtering. *TPAMI*, 2013.
- [10] L.-w. He, J. Shade, S. Gortler, and R. Szeliski. Layered depth images. 1998.
- [11] P. Hedman, S. Alsisan, R. Szeliski, and J. Kopf. Casual 3d photography. *TOG*, 2017.
- [12] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.
- [13] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *NIPS*, 2015.
- [14] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In ECCV, 2016.
- [15] N. K. Kalantari, T.-C. Wang, and R. Ramamoorthi. Learning-based view synthesis for light field cameras. *SIG-GRAPH Asia*, 2016.
- [16] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
- [17] M. Levoy and P. Hanrahan. Light field rendering. In Annual Conference on Computer Graphics and Interactive Techniques, 1996.
- [18] M. Liu, X. He, and M. Salzmann. Geometry-aware deep network for single-image novel view synthesis. In CVPR, 2018.
- [19] P. D. T. Malik, P. E. Debevec, and C. J. Taylor. Modeling and rendering architecture from photographs: A hybrid geometry and image-based approach. In *SIGGRAPH*, 1996.
- [20] M. Menze and A. Geiger. Object scene flow for autonomous vehicles. In CVPR, 2015.
- [21] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. ACM Transactions on Graphics (TOG)., 2019.

- [22] Y. Nakashima, F. Okura, N. Kawai, H. Kawasaki, A. Blanco, and K. Ikeuchi. Realtime novel view synthesis with eigentexture regression. In *BMVC*, 2017.
- [23] S. Niklaus, L. Mai, and F. Liu. Video frame interpolation via adaptive separable convolution. In *ICCV*, 2017.
- [24] K. Olszewski, S. Tulyakov, O. Woodford, H. Li, and L. Luo. Transformable bottleneck networks. arXiv:1904.06458, 2019.
- [25] E. Park, J. Yang, E. Yumer, D. Ceylan, and A. C. Berg. Transformation-grounded image generation network for novel 3d view synthesis. In *CVPR*, 2017.
- [26] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*.
- [27] K. Rematas, C. Nguyen, T. Ritschel, M. Fritz, and T. Tuytelaars. Novel views of objects from a single image. *TPAMI*, 2016.
- [28] K. Rematas, C. H. Nguyen, T. Ritschel, M. Fritz, and T. Tuytelaars. Novel views of objects from a single image. *TPAMI*.
- [29] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-assisted Intervention*, 2015.
- [30] D. Scharstein. Stereo vision for view dynthesis. In CVPR, 1996.
- [31] D. Scharstein. View synthesis using stereo vision. Springer-Verlag, 1999.
- [32] P. P. Srinivasan, R. Tucker, J. T. Barron, R. Ramamoorthi, R. Ng, and N. Snavely. Pushing the boundaries of view extrapolation with multiplane images. *CVPR*, 2019.
- [33] S.-H. Sun, M. Huh, Y.-H. Liao, N. Zhang, and J. J. Lim. Multi-view to novel view: Synthesizing novel views via selflearned confidence. In *ECCV*, 2018.
- [34] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.
- [35] S. Tulsiani, R. Tucker, and N. Snavely. Layer-structured 3d scene inference via view synthesis. In ECCV, 2018.
- [36] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell. Understanding convolution for semantic segmentation. In WACV, 2018.
- [37] O. J. Woodford, I. D. Reid, P. H. Torr, and A. W. Fitzgibbon. On new view synthesis using multiview stereo. In *BMVC*, 2007.
- [38] H. Wu, S. Zheng, J. Zhang, and K. Huang. Fast end-to-end trainable guided filter. In CVPR, 2018.
- [39] J. Xie, R. Girshick, and A. Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In *ECCV*, 2016.
- [40] Z. Xu, S. Bi, K. Sunkavalli, S. Hadap, H. Su, and R. Ramamoorthi. Deep view synthesis from sparse photometric images. ACM Transactions on Graphics (TOG), 2019.
- [41] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss functions for image restoration with neural networks. *TCI*, 2017.
- [42] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely. Stereo magnification: Learning view synthesis using multiplane images. In SIGGRAPH, 2018.

- [43] T. Zhou, S. Tulsiani, W. Sun, J. Malik, and A. A. Efros. View synthesis by appearance flow. In *ECCV*, 2016.
- [44] Z. Zhou, H. Jin, and Y. Ma. Plane-based content preserving warps for video stabilization. In *CVPR*, 2013.