Generative View Synthesis: From Single-view Semantics to Novel-view Images

Tewodros Habtegebrial[1,4] **Varun Jampani**[2] **Orazio Gallo**[3] **Didier Stricker**[1,4]
[1]TU Kaiserslautern [2]Google Research [3]NVIDIA [4]DFKI

Abstract

Content creation, central to applications such as virtual reality, can be tedious and time-consuming. Recent image synthesis methods simplify this task by offering tools to generate new views from as little as a single input image, or by converting a semantic map into a photorealistic image. We propose to push the envelope further, and introduce *Generative View Synthesis* (GVS) that can synthesize multiple photorealistic views of a scene given a single semantic map. We show that the sequential application of existing techniques, e.g., semantics-to-image translation followed by monocular view synthesis, fail at capturing the scene's structure. In contrast, we solve the semantics-to-image translation in concert with the estimation of the 3D layout of the scene, thus producing geometrically consistent novel views that preserve semantic structures. We first lift the input 2D semantic map onto a 3D layered representation of the scene in feature space, thereby preserving the semantic labels of 3D geometric structures. We then project the layered features onto the target views to generate the final novel-view images. We verify the strengths of our method and compare it with several advanced baselines on three different datasets. Our approach also allows for style manipulation and image editing operations, such as the addition or removal of objects, with simple manipulations of the input style images and semantic maps respectively. For code and additional results, visit the project page at https://gvsnet.github.io

1 Introduction

The rising demand for digital content, together with the widespread availability of high-quality digital cameras, has fueled the need for tools and algorithms to democratize content creation. A prominent example of one such technology is novel view synthesis (NVS), which allows the artist to render a scene from new viewpoints using as few as two images [11, 35], or even just one [33]. Photorealistic images can also be generated by editing a simplified representation of the scene, such as a semantic map, followed by image-to-image translation [24], but the viewpoint cannot be manipulated.

In this work, we propose *Generative View Synthesis* (GVS), which combines the advantages of both approaches. Given a single semantic map, which is easy to edit and requires no image capture, GVS can generate RGB images of the same layout, but from new, arbitrary viewpoints. Not surprisingly, GVS also inherits the challenges of both: generating RGB values from a bare semantic map is an ill-posed problem that is further complicated by the need for the different output views to be photometrically and geometrically consistent. One could tackle this problem with the sequential application of existing techniques. That is, we can first convert the single-view semantic map into an RGB image using image-to-image translation techniques [19], and generate novel RGB views using monocular novel view synthesis techniques [33]. However, we observe that this may fail at preserving the scene's structure accurately, as shown in the animation in Figure 1.

Our key insight is that semantic maps are particularly informative about the structure of a scene, despite offering no information about its photometric properties. Semantic segments, in fact, carry more unambiguous information about occlusion boundaries. This is in contrast with RGB images, where edges can also result from texture. We leverage this observation to preserve geometric

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

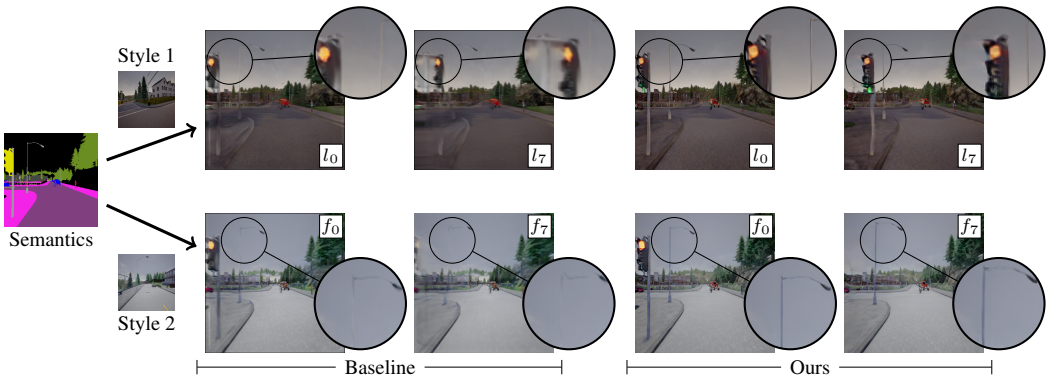

Figure 1: **Generative View Synthesis** is a method to generate photorealistic images from novel viewpoints, given just a semantic map and a style image. Here we show lateral (l_{0-7}) and forward (f_{0-7}) camera motion. Because no methods exist to solve this problem, we propose to use SPADE [24] followed by single-image MPI rendering [35] as a baseline. Our method better preserves thin structures and produces geometrically consistent views. *Animated figure. Please view in Adobe Reader and click on the image to see the animation. Other PDF viewers may have issues, in which case please refer to the supplementary materials.*

consistency between multiple output views. Specifically, instead of converting the semantic map to an RGB image, we propose to first uplift the 2D semantics into layered 3D semantics with a structure similar to multi-plane images (MPI) for RGB images [29, 35]. We call this structure *lifted semantics*. Unlike MPIs, to relax the memory requirements and for translation efficiency, our lifted semantics use a hybrid representation with a small set of semantic layers and a larger set of transparency layers. We convert the lifted semantics to layered features, which we refer to as *layered appearance*, and combine them with the transparency layers. Finally, we project the resulting appearance features onto the target views and convert them to RGB images with a small network. The late fusion of the lifted semantics is key to the quality of our results. In summary, our work benefits from, and leverages three main observations:

1. Semantic maps naturally disambiguate between texture edges and geometric discontinuities, thus offering a representation well-suited to generate novel views.
2. As a result, preserving semantic information for as long as possible in the pipeline, rather than converting to RGB early, encourages geometric consistency between the output views.
3. However, projecting the semantic information directly in the output views and converting to RGB causes photometric inconsistencies, which can be addressed by converting the semantics to an MPI-based representation of appearance first.

We perform extensive experimental analysis on three different multi-view datasets: CARLA [14], Cityscapes [12], and Virtual-KITTI-2 [3]. We show both qualitatively and quantitatively that our approach, which compares favorably with strong baseline techniques, produces novel-view images that are geometrically and semantically consistent. In addition, we also demonstrate that we can estimate high-quality depth information from single-view semantics.

2 Related Works

Novel view synthesis (NVS) has a rich history that predates the deep learning era [8, 38]. Traditional methods tackle the challenge of generating pixels for unseen viewpoints with proxy geometry [2, 13], or with a significant number of input images [5, 4]. Thanks to learned priors, impressive results with as little as two input images and no additional information are also possible [11, 35, 28, 15]. A key regularization technique at the core of many of these methods is a scene representation consisting of a set of fronto-parallel layers that can be merged down into the target view, after appropriate warping. This approach, which relates to representations proposed by earlier methods [29], is dubbed Multi-Plane Images (MPI) [35] and is also central to the success of our method, albeit with significant modifications. Recent NVS works go even further and show single-image NVS by learning to predict single-image depth [33, 10, 31], voxel grids [23], or by predicting MPI from single image [30].

While powerful, all these NVS methods require an input RGB image. To better leverage their creative agency, users can also edit a simplified representation of a scene or object, which can then be converted to an RGB image. For instance, sketches can be turned into photorealistic pictures [9, 19]. A scene-level representation that is particularly flexible is afforded by semantic maps. Indeed, a number of works using both traditional tools [20] and deep learning [7, 32, 19] produce impressive semantic-to-RGB results. An example of the engagement this type of technology can enable is offered by the method by Park et al. [24], which takes a hand-drawn semantic map and produces an RGB image. Within months of publication, more than 500,000 images were created by web users [1].

Image-to-image translation offers more control over the content generated, but unlike NVS methods, it does not allow to modify the viewpoint. We propose to combine the advantages of the two lines of work. Our method takes a semantic image as input and produces photorealistic images from novel viewpoints. However, compared with the sequential combination of existing methods, our strategy better leverages the information the semantic maps offer: a robust representation of the scene structure. We discuss this in more detail in the following section.

3 Approach

We present *Generative View Synthesis* (GVS), a method that takes a single 2D semantic map as input and generates photorealistic images from novel viewpoints. One way to tackle this problem is by a straightforward combination of existing techniques, that is, 2D semantics to 2D image conversion in the reference view [24], followed by monocular novel view synthesis [10, 36, 33] to generate the target views. However, this naïve approach fails at preserving some of the structures observed in the semantic input, as can be observed in Figure 1. This is because the semantic map's strong cues about the layout of the scene are lost in the early conversion to RGB. In contrast, GVS carries the semantic information forward, and only converts it to RGB after its projection onto the target viewpoint. This results in photorealistic target views that are geometrically consistent and that better preserve the structures in the input semantics.

Approach Overview. Formally, GVS takes a 2D semantic map $\mathcal{S}^r \in \mathbb{R}^{n \times l}$ in the reference view r, where n is the number of pixels and l the number of labels. It also takes the relative camera pose transformation from source to target view $\theta_{r \to t} \in SE(3)$. The output is a image $I^t \in \mathbb{R}^{n \times 3}$ in the target view t. Additionally, we use an image $Q \in \mathbb{R}^{n \times 3}$ to control the style of the generated images.

We train a network to convert the 2D semantics $\mathcal{S}^r$ to $\hat{\mathcal{S}}^r$, a layered semantics representation that we call *MPI semantics*. MPI semantics are inspired by multi-plane images (MPIs) [29, 35], which represent a 3D scene with a stack of 2D layers positioned at m depth levels $(d_1, d_2, ..., d_m)$. We represent the MPI semantics with m layers: $\hat{\mathcal{S}}^r \in \mathbb{R}^{n \times l \times m}$. Each layer contains semantic labels at each pixel, at that layer's depth, and the pixel transparency $\alpha \in \mathbb{R}^{n \times m}$. Multi-plane images [35, 29] are a widely used representation for novel view synthesis because their 2D nature allows for well-studied and powerful processing techniques such as convolutional neural networks (CNN).

One could project the MPI semantics onto the target views and independently translate each of the target-view semantics into RGB images. This would enforce geometric consistency, but it would not guarantee the consistent appearance of the output images. Therefore, we propose a translation scheme that converts the MPI semantics to layered 3D appearance features. We then project the layered appearance features onto the target views and convert them to the target RGB images. We demonstrate that this approach results in multi-view-consistent images and preserves the input semantic structures. Figure 2 (top) shows our approach overview: we uplift the 2D semantics using a *semantics uplifting network* (SUN) and translate the lifted semantics to layered appearance features using a *layered translation network* (LTN). We then project the layered appearance features to novel views and convert projected features using an *appearance decoder network* (ADN) to generate target images.

We find that the use of MPI semantics, however, is intractable because of the large memory footprint of the layered translation network (LTN). A faithful approximation of any content in 3D space, in fact, requires $m > 32$ MPI layers, making the semantics-to-appearance MPI translation infeasible—recall that we have l labels for each layer, and each pixel. Therefore, we propose a hybrid layered representation. Specifically, we learn only $k < m$ semantic layers and the full set of m transparency layers, which we then combine by learning an association function. We call our overall network for generative view synthesis *GVSNet*.

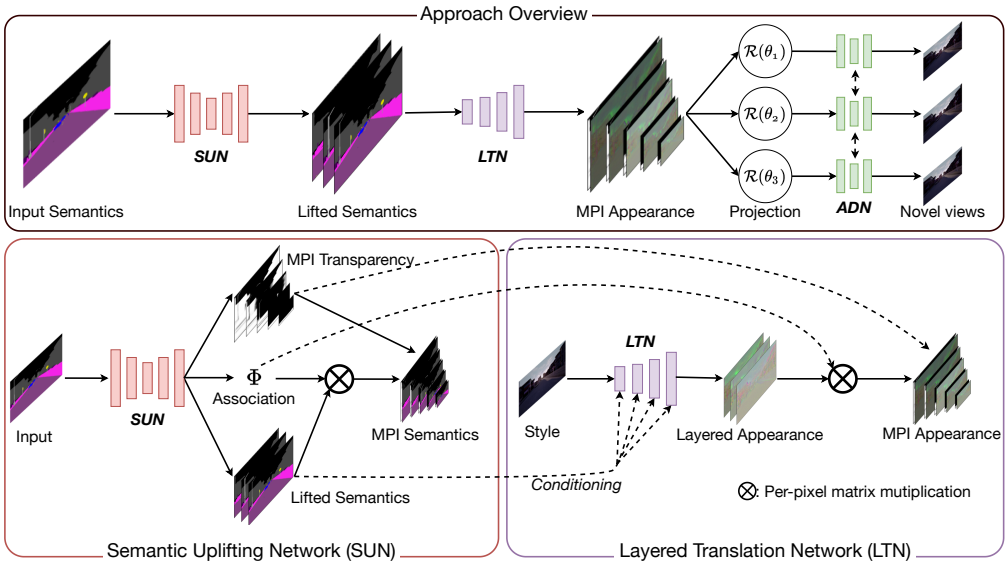

Figure 2: **GVSNet Overview.** [Top] Approach overview illustrating the lifting of semantics to layered 3D representation using Semantic Uplifting Network (SUN) and then translating to MPI appearance with Layered Translation Network (LTN). We then project the MPI appearance onto target views and use Appearance Decoder Network (ADN) to generate images. [Bottom] Illustration of different representations and operations in SUN and LTN.

3.1 Semantics Uplifting Network

We first uplift the input 2D semantics to layered 3D semantics using a 2D CNN, which we refer to as *Semantics Uplifting Network* (SUN). As outlined above, instead of converting the semantics to MPI-semantics, we propose to use a hybrid representation.

Hybrid Layered 3D Semantics Representation. Representing semantic information at each of the m depth layers of the MPI has a memory footprint in the order of $O(n \times l \times m)$, where n is the total number of pixels, and l the number of labels. In practice, however, the 3D scene is mostly empty, causing each of the layers in the MPI to be sparse. Therefore, we propose to represent the layered 3D semantics with fewer layers, $k < m$, which we call *lifted semantics*, $\tilde{\mathcal{S}}^r \in \mathbb{R}^{n \times l \times k}$. For this representation, we use the input 2D semantics as the first layer and predict the remaining layers using the SUN network. In practice, we observe that $k = 3$ layers (including the input semantics) suffice. On the other hand, we do not compress the transparency, α, to fewer layers, as it serves as a proxy to the scene geometry and it only requires a scalar value for each pixel in each layer. That is, we represent transparency α with the original number of MPI layers m. The transparency layers are also shared with the layered 3D appearance features.

Because of the mismatch in the number of layers, we need to estimate an association map $\Phi \in \mathbb{R}^{n \times k \times m}$ to convert the lifted semantics $\tilde{\mathcal{S}}^r$ to MPI semantics $\hat{\mathcal{S}}^r$. At each pixel p, we can convert the lifted semantics $\tilde{\mathcal{S}}^r_p \in \mathbb{R}^{l \times k}$ into MPI semantics $\hat{\mathcal{S}}^r_p \in \mathbb{R}^{l \times m}$ representation with the column-normalized association matrix $\Phi^*_p \in \mathbb{R}^{k \times m}$: $\hat{\mathcal{S}}^r_p = \tilde{\mathcal{S}}^r_p \, \Phi^*_p$. Figure 2 (bottom-left) illustrates the SUN network that takes 2D semantics in the reference view as input and predicts the lifted semantics $\tilde{\mathcal{S}}^r$, the MPI transparency α, and the association map Φ. Figure 3 shows sample lifted semantics layers. We visually observe that the lifted semantics layers roughly correspond to occlusion layers, where the farther layers capture the content occluded by the closer layers.

3.2 Layered Translation Network

With the MPI semantics $\hat{\mathcal{S}}^r$, we can render the semantics into the target views $\mathcal{S}^t$, and use any of the recent conditional image generation networks [24] to generate target-view images. We empirically observe that independent translations of semantics to RGB images can result in inconsistent results—

even when the structure is consistent, the corresponding texture may vary across views. To remedy this, we estimate layered 3D appearance features and directly translate the lifted semantic layers to appearance feature layers. Differently put, we carry forward for as long as possible a view-independent representation of the scene.

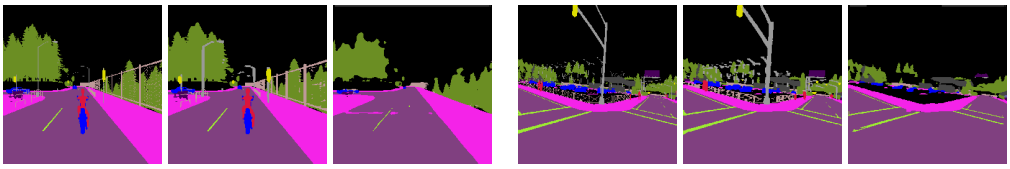

Figure 3: **Lifted Semantics.** L1, L2 and L3 correspond to the first, second and third lifted semantic layers respectively. We observe that the lifted semantic layers loosely correspond to occlusion layers, where the later layers capture the content occluded by the earlier layers, see L3 images. Another interesting observation is that, at mid levels (L2), the network *dilates* thin structures (see the poles).

State-of-the-art image-to-image translation networks use large generative adversarial networks (GAN), making it infeasible to translate MPI semantics to MPI appearance features. This is the very reason why we use SUN to predict lifted semantics $\tilde{\mathcal{S}}^r$ instead of MPI semantics $\hat{\mathcal{S}}^r$. We use a *Layered Translation Network* (LTN) to convert k lifted semantics layers $\tilde{\mathcal{S}}^r$ to k-layered appearance features $\tilde{\mathcal{A}}^r \in \mathbb{R}^{n \times f \times k}$, where f denotes the appearance feature dimensionality. LTN is a modified version of SPADE [24]. Like the original SPADE, LTN can also take a style image as input to enable easy manipulation of the appearance of the generated images. However, SPADE takes 2D semantics as input and generates 2D image, while LTN takes multi-layer inputs and produces multi-layer outputs. We then use the same MPI transparency layers α and association map Φ, estimated with SUN, to convert the k-layered appearance features to MPI appearance features with m layers. That is, at each pixel p: $\hat{\mathcal{A}}^r_p = \tilde{\mathcal{A}}^r_p \Phi^*_p$. Figure 2 (bottom-right) illustrates the LTN network.

3.3 Appearance Decoder Network

We render the MPI appearance features $\hat{\mathcal{A}}^r$ into target-view appearance $A^t \in \mathbb{R}^{n \times f}$ for a given target-view t. We then train a small CNN called *Appearance Decoder Network* (ADN) that converts target-view appearance to the final target image I^t. As an alternative, one could directly estimate color MPI using LTN and then just render the color MPI onto target-view to obtain a target image. We empirically observe that projecting high-dimensional (f-dimensional) MPI features can result in better target views in comparison to projecting color MPI. This is because high-dimensional, per-pixel features help mitigate some of the artifacts that arise from the discrete nature of the MPI planes. The high-dimensional features capture neighborhood pixel information and can provide more contextual information to ADN to deal with possible MPI rendering artifacts.

Loss Functions. The overall GVSNet, illustrated in Figure 2, has three main sub-networks: Semantics uplifting network (SUN), Layered translation network (LTN) and Appearance decoder network (ADN). To train GVSNet, we use a weighted sum of target segmentation loss $\mathcal{L}_{sem}$, depth loss $\mathcal{L}_{dep}$, target color loss $\mathcal{L}_{col}$ and GAN loss $\mathcal{L}_{gan}$:

$$\mathcal{L} = \lambda_0 \, \mathcal{L}_{sem} + \lambda_1 \, \mathcal{L}_{dep} + \lambda_2 \, \mathcal{L}_{col} + \lambda_3 \, \mathcal{L}_{gan}. \tag{1}$$

$\mathcal{L}_{sem}$ denotes the negative log-likelihood loss for the predicted semantics in the target view. We project the MPI semantics to target-view semantics using standard MPI rendering [35] that involves homography transformation and alpha composition. Refer to [35] for more details on MPI rendering. We then use negative log-likelihood loss $\mathcal{L}_{sem}$ between ground-truth (GT) semantics and predicted semantics in the target views. For the depth loss, we first compute per-pixel inverse depth D^r in the reference view from the predicted MPI transparency α by performing back-to-front alpha composition on the inverse-depth values of each plane at the given pixel. Refer to the supplementary for more details on the inverse depth computation from transparency. Then the depth loss $\mathcal{L}_{dep}$ is the L1 distance between the GT inverse depth and the computed inverse depth. The target color loss and GAN lossses, include the same losses as in SPADE [24], that is, perceptual loss on generated color image, discriminator feature reconstruction loss and GAN losses. The main difference to SPADE [24] is that we use these loss functions on generated *target* image with respect to the GT target image.

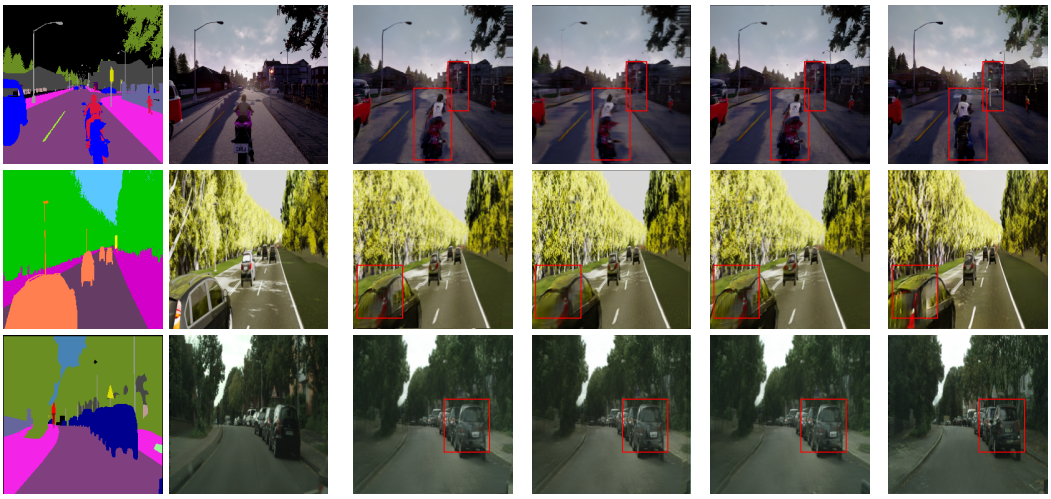

Figure 4: **Sample Visual Results** showing generated novel-view images on CARLA (top), Virtual-KITTI-2 (middle), and Cityscapes (bottom) images.

We train GVSNet in two stages. In the first stage, we pre-train SUN with the target segmentation and depth losses. In the second stage, we train LTN and ADN with the target color loss, while keeping the SUN fixed. We use this two-stage training because the entire network training does not fit on NVIDIA GTX-2080-Ti GPUs, which is what we use for training. However, we could conceptually train GVSNet end-to-end, because all the components are differentiable. The SUN network is a composed of two parts: backbone and three prediction heads (for α, $\tilde{\mathcal{S}}^r$, and Φ). The backbone is a UNet [26] style encoder-decoder network with 7 encoding and decoding stages. The prediction heads are convolutional blocks that share their first 3 layers. ADN is a light CNN with 5 encoder-decoder layers. The LTN network is a SPADE [24] network with 7 SPADE-Residual blocks and UpSampling layers. For our experiments, we used $k = 3$ lifted semantics layers, $m = 32$ MPI planes, and $f = 20$ appearance features per pixel. We implemented our model in PyTorch [25] and use the Adam [21] optimizer for training. More details about the training and network architectures are given in the supplementary material.

4 Experiments

Datasets. GVSNet is fully supervised and thus requires datasets providing two or more views for each scene. It also needs semantic segmentation annotations for at least one view. Given these constraints, we perform experiments on three different datasets: CARLA [14], Virtual-KITTI-2 [3] and Cityscapes [12]. We use a pair of cameras to train GVSNet, treating one of the images as the input view and another one as the target view. In CARLA, we sample pairs from a set of cameras arranged along x-(left-right) and z-(forward) axes. For Virtual-KITTI-2 and Cityscapes, we use stereo pairs with a horizontal baseline. For CARLA and Virtual-KITTI-2, following the practice in SPADE [24], we use the color image from the input camera as a style guidance when generating the target view. In order to make our results comparable to SPADE [24], we do not use style input images in Cityscapes experiments. For the Cityscapes dataset, the ground truth (GT) semantic segmentation is only available for the left camera. We used a pre-trained semantic segmentation network [37] to generate semantics for the right camera images. As there is no ground truth depth in Cityscapes, we computed depth maps by training the DPSNet [18] in a self-supervised manner. In the Cityscapes experiments we use instance masks and the details can be found in the supplementary material. In all of our experiments we use images at a resolution of 256×256 pixels.

Evaluation Metrics. The results of a GVS system should have three properties: 1. *Semantic Preservation*: The generated image should retain the semantic structures of the input semantics; 2. *Photorealism*: The generated target images should be photo-realistic; and 3. *Multi-view Consistency*: When rendered across multiple views, appearance of a given physical object shouldn't

change drastically. To measure semantic preservation, we apply a semantic segmentation network (DeeplabV3+ [6]) on the synthesized images and compare its output with the GT semantics in the target view. Specifically, we report mean class accuracy and mean Intersection over Union (IoU) as segmentation metrics. To measure photo-realism, we report the Fréchet Inception Distance (FID) score [17] and Perceptual Distance (PD) [34] metric, which measure the distance between generated target-view images and GT images in VGG [27] feature space. We evaluate multi-view consistency qualitatively and quantitatively by warping two novel views onto a reference camera and comparing the warped images.

	CARLA [14]				Virtual-KITTI-2 [3]				Cityscapes [12]	
Method	Cls. Acc. ↑	IoU ↑	PD ↓	FID ↓	Cls. Acc. ↑	IoU ↑	PD ↓	FID ↓	PD ↓	FID ↓
GVSNet (Ours)	**74.34**	**66.43**	**1.74**	**62.06**	**77.13**	**69.62**	**2.08**	**36.21**	**2.76**	**48.72**
SPADE [24] + SM [35]	69.93	60.82	1.95	75.81	74.84	64.71	2.19	41.61	2.82	60.71
SPADE [24] + CVS [10]	66.84	57.29	1.88	69.24	76.23	67.73	2.12	37.79	2.80	57.46
SPADE [24] + AF [36]	66.15	56.45	1.92	76.89	76.81	68.66	2.15	40.95	2.83	57.15
Target GT Images	77.47	69.67	-	-	83.58	75.39	-	-	-	-

Table 1: **Comparisons to Baselines.** Semantic segmentation (Class Accuracy and IoU), FID [17] and Perceptual Distance (PD) evaluations on different datasets for GVSNet (ours) along with SPADE+X baseline techniques, that first perform semantics-to-RGB conversion followed by monocular NVS.

Comparisons to Baselines. As this is the first work to tackle GVS, there is no existing baseline technique against which we can directly compare. To properly evaluate our approach, then, we propose sensible baseline methods based on the adaptation and combination of state-of-the-art methods. Specifically, we use a pipeline that converts 2D semantics to an RGB image with SPADE [24] and applies monocular NVS techniques to render novel views. For the latter task, we pick Stereo-magnification (SM) [35] adapted to single-view NVS, Continuous View Synthesis (CVS) [10], and Appearance Flow (AF) [36]. We refer to these baselines as 'SPADE+X' where 'X' could be 'SM', 'CVS' or 'AF'. Table 1 shows the quantitative results. Results show that GVSNet is consistently superior to the SPADE+X techniques across different metrics and datasets. This demonstrates that GVSNet can better preserve the semantic structures of the input while generating more realistic images. Figure 4 shows sample visual results. The visual results in Figure 1 (play the animation), Figure 4, and Figure 5 further validate that GVSNet better preserves semantic structures and geometric consistency. We show more visual results in the supplementary material.

Method	Class Acc. ↑	IoU ↑	PD ↓	FID ↓
GVSNet Variations				
SUN + SPADE [24]	72.92	65.52	1.75	68.96
SUN + LTN	71.90	63.12	1.83	69.46
SUN + LTN + ADN (Full model)	**74.34**	**66.43**	**1.74**	**62.06**

Table 2: **Ablative Studies on GVSNet.** Semantic segmentation (Class Accuracy and IoU), FID [17] and Perceptual Distance (PD) evaluations on CARLA [14] with different variations of our GVSNet. Results show that all the component networks of SUN, LTN and ADN are important.

Method	CARLA [14]	Virtual-KITTI-2 [3]
GVSNet	0.0154	**0.0328**
SPADE [24]+SM [35]	**0.0130**	0.0330
SUN+SPADE [24]	0.0253	0.0511

Table 3: **Multi-view Consistency** measured by warping two novel views onto a reference camera and calculating their difference as *mean-absolute-brightness* error. GVSNet is more consistent than SUN+SPADE. GVSNet is almost as consistent as SPADE+SM, despite the latter computing novel views by resampling the input view directly.

Ablation Study. GVSNet comprises several computational blocks, each critical to its success. To confirm this, we perform an ablation study and evaluate the impact of swapping out parts of it. One

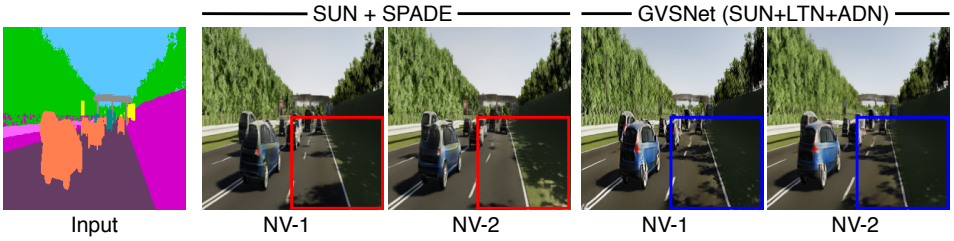

Figure 5: **Consistency across views.** We show sample generated images in two novel views (NV-1 and NV-2) using our full GVSNet model along with those obtained with SUN+SPADE. SUN+SPADE model first estimates novel-view semantics followed by semantics-to-RGB translation with SPADE [24]. Results show that view-independent translation results in inconsistency across the views and the layered translation (LTN) is important to generate view-consistent images.

basic test is to only use the semantic uplifting network (SUN) to synthesize 2D semantics in the target view, and use SPADE [24] to convert it to RGB image. We refer to this model as 'SUN+SPADE'. To evaluate the importance of f-dimensional layered appearance features (f=20) as opposed to just estimating 3-dimensional color, we experiment with a model that translates layered semantics to layered color images. We refer to this variant as 'SUN+LTN', as it does not use ADN. Ablation results shown in Table 2 indicate that all the three component networks (SUN, LTN and ADN) are important. Performance drops considerably if we use LTN to directly translate to color MPI (SUN+LTN vs. SUN+LTN+ADN).

We observe that SUN+SPADE, that applies SPADE [24] on projected 2D semantics in novel views, can generate photo-realistic novel-view images. However, we also notice that SUN+SPADE produces inconsistent images across multiple target views as image translation is applied independently on each target view semantics. Figure 5 shows a sample visual result indicating much better view-consistency with GVSNet when compared to SUN+SPADE model. Numerical evaluations in Table 3 also show that GVSNet is more consistent than SUN+SPADE. We argue that geometric or appearance inconsistency among different views is more detrimental than a slight perturbation of the semantic information, when this is common to all the synthesized views.

Depth from Semantics. Since we estimate transparency α at each MPI layer from a single 2D semantics, we can convert the MPI transparency into depth by performing alpha composition on the depth values of the MPI planes (see supplementary for details). Figure 6 shows a sample depth estimation with our GVSNet that *only* takes 2D semantics as input. For comparison, we also show depth estimation from the RGB image using MonoDepth [16] (MD), which is trained in a fully-supervised manner using GT depths. Results indicate that our depth estimates are more accurate at thin structures. This is further confirmation of the strong structural cues offered by the semantics. We present more results in the supplementary material.

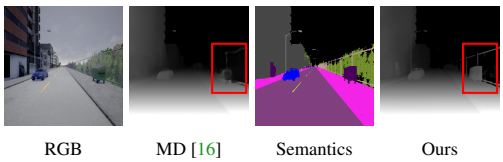

Figure 6: **Depth Estimation** We can obtain relative depth using the estimated MPI transparency from 2D semantics. Also shown is the result obtained with MonoDepth [16] (MD) that uses RGB image as input. Results show that GVSNet (ours) can estimate better geometric structures.

Semantic Editing. One of the main advantages of GVS is to simplify multi-view content generation. Editing a semantic map by simply pasting an object from another, for instance, is arguably easier than directly editing RGB pixels, which requires accounting for lighting conditions explicitly, adding realistic textures, etc. Similarly, despite the success of recent inpainting methods [22], removing objects from RGB images while accounting for the scene context requires advanced skills. Thanks to our method, simple object addition to/removal from the 2D semantic maps seamlessly translates to photorealistic multi-view images, as shown in Figure 7.

5 Conclusion

In this work, we propose Generative View Synthesis Network (GVSNet), which produces photo-realistic novel-view images from only a single 2D semantic map. We demonstrate that the simple

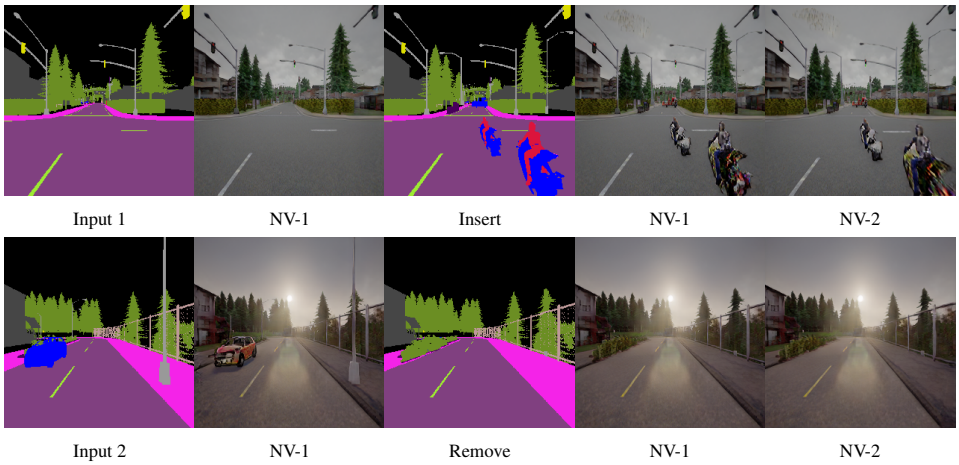

Figure 7: **Semantic Editing.** Sample GVSNet generated novel-view (NV-1 and NV-2) images before and after editing objects in a given semantics. Results with both object insertion (left) and object removal (right) show that GVSNet can generate realistic images also on manually edited semantics.

application of existing techniques for this problem yields inadequate results. Our key insight is to leverage the structural information in the input semantics by uplifting the 2D semantics to layered 3D semantics. Further, we carry the structural 3D semantic information forward with a layered semantics-to-RGB translation network. Comprehensive experimental analysis on three different datasets demonstrate the potential of GVSNet in generating geometrically consistent novel-view images, while preserving the structures in the input semantics.

Broader Impact

This work makes digital content creation easier by introducing a new Generative View Synthesis method that combines the benefits of image-to-image translation and novel view synthesis, both of which are active research areas. We hope this work inspires further research in digital content creation.

Acknowledgments and Disclosure of Funding

This project was partially funded by the BMBF project VIDETE(01IW1800). We thank the SDS department at DFKI Kaiserslautern, for their support with GPU infrastructure.

References

[1] A Pigment of Your Imagination: GauGAN AI Art Tool Receives 'Best of Show,' 'Audience Choice' Awards at SIGGRAPH, 2019 (Last accessed on 06/01/2020). https://blogs.nvidia.com/blog/2019/07/30/gaugan-ai-painting/. 3

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

[3] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual KITTI 2. In *arXiv Preprint*, 2020. 2, 6, 7

[4] 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

[5] Gaurav Chaurasia, Olga Sorkine, and George Drettakis. Silhouette-aware warping for image-based rendering. In *Eurographics Symposium on Rendering*, 2011. 2

[6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *European Conference on Computer Vision (ECCV)*, 2018. 7

[7] Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 3

[8] Shenchang Eric Chen and Lance Williams. View interpolation for image synthesis. In *ACM SIGGRAPH*, 1993. 2

[9] Tao Chen, Ming-Ming Cheng, Ping Tan, Ariel Shamir, and Shi-Min Hu. Sketch2photo: Internet image montage. In *ACM Transactions on Graphics (SIGGRAPH ASIA)*, 2009. 3

[10] Xu Chen, Jie Song, and Otmar Hilliges. Monocular neural image based rendering with continuous view control. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 2, 3, 6, 7

[11] Inchang Choi, Orazio Gallo, Alejandro Troccoli, Min H Kim, and Jan Kautz. Extreme view synthesis. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 1, 2

[12] 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. 2, 6, 7

[13] 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

[14] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *arXiv Preprint*, 2017. 2, 6, 7

[15] 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. 2

[16] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 8

[17] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems (NIPS)*, 2017. 7

[18] Sunghoon Im, Hae-Gon Jeon, Stephen Lin, and In So Kweon. DpsNet: End-to-end deep plane sweep stereo. In *arXiv Preprint*, 2019. 6

[19] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 3

[20] Matthew Johnson, Gabriel J Brostow, Jamie Shotton, Ognjen Arandjelovic, Vivek Kwatra, and Roberto Cipolla. Semantic photo synthesis. In *Computer Graphics Forum (CGF)*, 2006. 3

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *arXiv Preprint*, 2014. 6

[22] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *European Conference on Computer Vision (ECCV)*, 2018. 8

[23] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[24] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2, 3, 4, 5, 6, 7, 8

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NIPS)*, 2019. 6

[26] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015. 6

[27] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 7

[28] 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

[29] Richard Szeliski and Polina Golland. Stereo matching with transparency and matting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1998. 2, 3

[30] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *arXiv Preprint*, 2020. 2

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

[32] 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)*, 2018. 3

[33] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. SynSin: End-to-end view synthesis from a single image. In *arXiv Preprint*, 2020. 1, 2, 3

[34] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 7

[35] 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, 3, 5, 6, 7

[36] 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. 3, 6, 7

[37] Yi Zhu, Karan Sapra, Fitsum A Reda, Kevin J Shih, Shawn Newsam, Andrew Tao, and Bryan Catanzaro. Improving semantic segmentation via video propagation and label relaxation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 6

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