Self-Supervised CascadeMVSNeRF

Chang, Di
di.chang@tum.de

Kocas, Halil Eralp
halileralp.kocas@tum.de

Ding, Ran
ran.ding@tum.de

Abstract

Unsupervised Multi-View Stereo depth estimation methods have recently achieved significant progress. However previous methods are built upon the assumption of corresponding pixels share similar photometric features and they may suffer from ambiguous supervision. In this project we propose to utilize neural rendering and radiance field in unsupervised MVS, replenishing what previous models ignore. To address these limitations, we propose a novel rendering consistency framework (SSCasMVSNeRF) for unsupervised Multi-View-Stereo. To alleviate the ambiguous supervision from photometric consistency, we use volumetric rendering to generate consistent RGB-level supervision. Extensive experiments on DTU and Tanks&Temples benchmark prove that our SSCasMVSNeRF framework achieves the best performance among end-to-end unsupervised MVS approaches, even with competitive performance to many supervised methods.

1. Introduction

Multi-View Stereo (MVS) is a classic computer vision problem where the inputs are the RGB images of the scene and the corresponding camera parameters, and the output is a 3D model of the scene. The recent introduction of deep learning into multi-view stereo (MVS) has achieved remarkable results. Trained on a large-scale indoor DTU dataset, MVSNet [28] significantly outperforms previous state-of-the-art techniques in terms of accuracy and completeness metrics and is several times faster in runtime. Moreover, it has strong generalization capability, ranking first on the complex outdoor Tanks&Temples dataset without any fine-tuning until April 18, 2018. The follow-up work CasMVSNet [9] introduces the cascade cost volume to the representative MVSNet. It has both achieved improvement in accuracy and also a reduction in GPU memory and run-time on the DTU benchmark, and also owns strong generalization capability on the Tanks&Temples dataset.

Although the supervised learning methods have achieved good results in depth map inference tasks, they depend on a large number of labeled datasets containing RGB images, depth maps, camera parameters, etc, which is costly to benchmark. Recently, there is an increasing number of self-supervised multi-view stereo methods proposed to solve the dataset deficiency. However, most of the self-supervised works mainly rely on the same color constancy hypothesis, which assumes that the corresponding points among different views have the same color, which is impractical since various factors may interfere with the color distribution in real scenes, such as lighting conditions, reflections, noise, etc. Hongbing Xu's work JDACS [24] addresses this problem using prior semantic correspondence and prior data augmentation consistency, achieving remarkable results both on the DTU benchmark and strong generalization performance on the Tanks&Temples dataset, which is comparable to the supervised methods.

Recently, Neural Radiance Field [18] has achieved great success in neural rendering and view synthesis. It describes how to efficiently optimize neural radiance fields to render realistic new views of scenes with complex geometry and appearance, and demonstrates results that outperform previous work. However, the per-field optimization process of the initial NeRF is highly expensive. The follow-up work MVSNeRF [3] can generalize across scenes (even indoor scenes, completely different from our training scenes of objects) and generate realistic view synthesis results using only three input images, significantly outperforming concurrent works on generalizable radiance field reconstruction.

Inspired by the great success of Neural Radiance Field [18] in the novel view synthesis field, we introduce NeRF to the depth map inference task and utilize it to enhance feature extraction and matching. What's more, we apply more than photometric consistency: Cross-View Rendering Consistency and propose an occlusion-aware method. To be specific, we combine MVSNeRF [3] and CasMVSNet [9] and train the network in a self-supervised manner. In the CasMVSNet [9] branch, the inputs are the source and reference images, camera parameters. We generate cost volumes after feature extraction, differentiable homography warping, and variance cost metric. At last, we obtain depth maps after 3D CNN and Soft Argmax. In the MVSNeRF branch, it inputs the same except the reference image in

case the neural network may find shortcuts. It obtains 2D Features, Cost Volumes, and finally the render depth pixels and render RGB pixels after the MLP layer. The 2D Feature Net is the same in both branches, but the 3D U-Net is different in both branches. Our algorithm converts the depth maps generated by the CasMVSNet and the source images to the warped images, making a photometric consistency loss with the reference image. We define the render loss between the reference image and the render RGB point generated by MVSNeRF. In addition, our algorithm design a smooth L1 loss between the generated depth maps and render depth pixels. Our work has realized state-of-the-art accuracy on DTU Dataset and strong generalization performance on Tanks&Temples Dataset without finetuning. In short, our contributions are summarized as below: 1. Propose a novel end-to-end learning-based self-supervised MVS depth inference. 2. Define a render consistency loss.

2. Related Work

2.1. Neural Radiance Field

Recently, neural radiance fields have been introduced into the field of view synthesis and obtained remarkable performance. As first proposed by Ben [18] , it represents scenes as a neural radiance field, combines MLPs with volume rendering, and achieves photo-realistic view synthesis. However, initial NeRF requires an expensive per-field optimization process. Follow-up work MVSNeRF proposed by Anpei Chen [3] achieves high-performance cross-scene neural radiation field estimation using deep MVS techniques. This performs geometry-aware scene inference using plane-swept cost volumes and combines it with physics-based volume rendering for neural radiance field reconstruction.

2.2. Depth Map-based Multi-View Stereo

Multi-view stereo (MVS) is a classical 3D reconstruction problem, which leverages images from multiple viewpoints to achieve dense reconstruction. Depending on the different 3D representations, multi-view stereo can be broadly classified into volumetric-based methods [14] [20] [11], point cloud-based methods [15] [7] and depth map-based methods [2] [8] [19] [27]. In contrast, the depth map-based approach is the most flexible, robust and efficient. There are many cutting-edge depth map-based MVS methods. Yaos work MVSNet [28] achieves good results on both indoor datasets like the DTU dataset, and complex outdoor datasets like the Tanks&Temples dataset. They build 3D cost volume by implementing homography warping on the 2D features, then applying 3D UNets and Soft Argmax to obtain the Depth Maps. The follow-up work CascadeMVSNet proposed by Xiaodong Gu [9] performs better in both accuracy and time, memory complexity. They propose a cost volume based on a feature pyramid, narrow the range of depth (or variance) at each stage by the prediction of the previous stage, and recover the output in a coarse to fine manner.

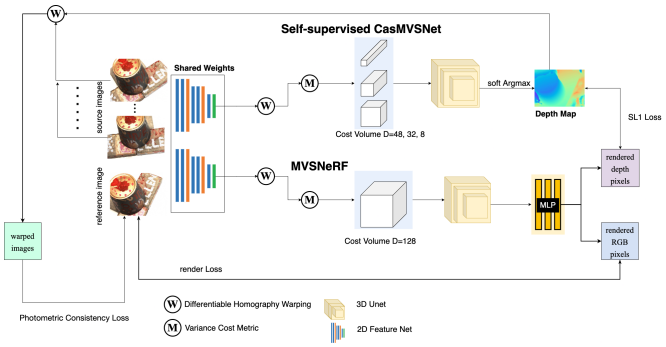

Figure 1. Overview of SSCasMVSNeRF. Input images are processed by 2D Feature Extraction Network and the differentiable homography warping to construct 3D Cost Volumes. Then, 3D U-Net architectures process these 3D Cost Volumes to obtain regularized 3D Cost Volumes. CascadeMVSNet directly regresses a depth map from the regularized cost volume. MVSNeRF renders depth and RGB values of some pixels using the regularized cost volume and the depth map obtained from CascadeMVSNet.

2.3. Self-supervised Multi-View Stereo

Recently, self-supervised learning has been developed in the task of multi-view stereo. Compared with supervised learning, this method addresses the data deficiency of multi-view depth data. There are many state-of-the-art self-supervised approaches that have achieved qualitative and quantitative results compared with the supervised methods. In Tejas Khots work UnsupMVS [12], they utilize photometric consistency between multiple views as the supervised signal and define a robust loss formulation that overcomes constraints caused by occlusion and lighting changes across views. This work achieves a significant improvement in the self-supervised Multi-View Stereo approaches. However, it has deficiencies of having an assumption of constant color and encountering the problem of ambiguous supervision in complex scenes. The follow-up work JDACS proposed by Hongbing Xu [24] addresses this problem and achieves better performance with more reliable supervision. In Xus work, they excavate mutual semantics from multi-view images to guide semantic consistency and apply effective data augmentation that ensures transformation robustness.

3. Self-Supervised Cascade MVSNeRF

Self-Supervision is crucial in AI since it eliminates the need for ground-truth data. Ground-truth data is costly to obtain and brings the ground-truth error risk. Self-supervision is an intermediate form of unsupervised and

supervised learning where we generally utilize input data to supervise our model training, yet, there is no external ground-truth data. Hence, we adapt MVSNeRF [3] into the CascadeMVSNet [9] pipeline to enable self-supervision using six different loss functions in our work. Our proposed model is in Figure 1.

Our proposed model consists of CascadeMVSNet and MVSNeRF models in which they share the feature extraction stage. Following feature extraction from reference and source images, we have two different branches in which we build cost volumes using homography warping and variance-based cost metric. Then, 3D U-Net models are used to obtain regularized cost volumes in both CascadeMVSNet and MVSNeRF branches. Finally, we estimate the depth map of reference view using Soft Argmax operation in the CascadeMVSNet branch. Also, we feed the regularized cost volume through MLP layers of MVSNeRF so that we obtain pixel-wise depth and RGB renderings in training. We do not render pixel-wise, yet, we directly regress the full depth map and RGB image during the testing. To supervise our model, we have six different loss functions. We warp source images with the depth map from CascadeMVSNet and compute photometric consistency loss between the warped source images and the reference image. Also, we compute structural similarity loss between the warped source images and the reference image. Additionally, we compute smoothness loss using the depth map from CascadeMVSNet branch and the reference image. We compute render loss between the pixel colors and the reference image pixels' colors. In addition, we compute depth loss between the depth values rendered from MVSNeRF branch and the corresponding depth pixels from the CascadeMVSNet's depth estimation. Finally, we compute data augmentation loss between depth estimations of original and augmented input images. Since our supervision signals are generated, our method is self-supervised.

3.1. 2D Feature Extraction

Feature extraction is shared between CascadeMVSNet and MVSNeRF. We utilize 2D Convolution layers to extract image features from *N* input images. 2D CNN parameters are shared across all input images so that the model can learn efficiently. Our 2D CNN model consists of 8 Convolution layers where we build a multi-scale feature representation by having downsampling in layers 3 and 6. The output features are 32-channel feature maps and 4-times downsized input images in each dimension.

3.1.1 Difference between CascadeMVSNet and MVSNeRF Branches

The only difference between CascadeMVSNet and MVSNeRF branches is how we fed the feature maps into the later stages. In the CascadeMVSNet, the feature maps from all input images are fed into the next steps. On the other hand, considering MVSNeRF may overfit the reference image, only the feature maps of *N-1* source images fed to the MVSNeRF, not the reference image's feature map.

3.2. Cost Volume

3D Cost Volume is reconstructed using 2D feature maps and camera parameters of input cameras. Hence, we need feature maps from 2D CNN and camera intrinsic and extrinsic (rotation matrix & translation vector) matrices. Homography warping is applied to source images' feature maps to warp them to reference view. The following formula is for homography warping:

$$\mathcal{H}_i(z) = K_i \cdot \left(R_i \cdot R_{ref}^T + \frac{(t_{ref} - t_i) \cdot n_{ref}^T}{z} \right) \cdot K_{ref}^{-1} \quad (1)$$

where $H_i(z)$ is the warping matrix we apply to source feature maps. *K* is the corresponding camera intrinsic parameters and *R, t* are the corresponding camera extrinsic parameters (rotation and translation). Then, using a 3x3 homography matrix, we can warp feature maps as follows:

$$F_{i,z}(u, v) = F_i \left(\mathcal{H}_i(z)[u, v, 1]^T \right) \quad (2)$$

where $F_{i,z}(u, v)$ is the warped source image feature map at depth z and pixel location *(u, v)* at the reference image. After we warp all feature maps to the reference view, we build 3D Cost Volume by aggregating multiple cost volumes obtained from feature maps. Following the same structure as CascadeMVSNet and MVSNeRF, we apply a variance-based cost metric to adapt an arbitrary number of views as follows:

$$P(u, v, z) = \mathrm{Var}\left(F_{i,z}(u, v) \right) \quad (3)$$

The variance-based cost metric helps us to encode image appearance variations across different input views. For each voxel in Cost Volume *P* centered at *u, v, z*, the following formula is applied across all feature maps to define the mapping $\mathcal{M} : \underbrace{\mathbb{R}^V \times \cdots \times \mathbb{R}^V}_{N} \rightarrow \mathbb{R}^V$:

$$\mathbf{C} = \mathcal{M}\left(\mathbf{V}_1, \cdots, \mathbf{V}_N \right) = \frac{\sum_{i=1}^{N} \left(\mathbf{V}_i - \overline{\mathbf{V}}_i \right)^2}{N} \quad (4)$$

where $V = \frac{W}{4} \cdot \frac{H}{4} \cdot D \cdot F$ the feature volume size and $\overline{\mathbf{V}}_i$ is the average volume among all feature volumes.

3.2.1 Cost Volume in CascadeMVSNet Branch

CascadeMVSNet uses the cost volume for geometry reconstruction. The difference between the cost volume of

CascadeMVSNet and the cost volume of MVSNeRF can be observed in Figure 1. We have multi-scale cost volume with D=$\{48, 32, 8\}$ channels in the CascadeMVSNet.

3.2.2 Cost Volume in MVSNeRF Branch

MVSNeRF uses the cost volume to obtain a complete scene appearance for neural rendering. We have a single-scale cost volume with D=$\{128\}$ channels in MVSNeRF.

3.3. Cost Volume Regularization

Cost Volume Regularization aims to refine the quality of Cost Volume obtained from feature maps. This step encodes local scene geometry and appearance to decrease the noise in feature maps. We utilize 3D U-Net architectures to enrich the information our model has.

3.3.1 Cost Volume Regularization in CascadeMVSNet Branch

The cost volume regularization in CascadeMVSNet aims to predict the probability volume so that the depth map can be inferred. The last convolution layer used after 3D U-Net in CascadeMVSNet reduces the number of channels from 32 to 1 in the output volume. Then, the softmax operation normalizes the probabilities in the depth direction.

3.3.2 Cost Volume Regularization in MVSNeRF Branch

The cost volume regularization in MVSNeRF aims to transform the cost volume into a new neural encoding volume so that the high-quality rendering of scenes can be regressed directly from this neural encoding volume. Hence, the cost volume regularization in MVSNeRF outputs 8-channels Cost Volume called Neural Encoding Volume to regress volume rendering properties directly from this volume.

3.4. Depth Estimation in CascadeMVSNet Branch

Following the same structure as CascadeMVSNet, we compute Soft Argmax (*expectation* value) along the depth direction using the following formula:

$$\mathbf{D} = \sum_{d=d_{\min}}^{d_{\max}} d \times \mathbf{P}(d) \tag{5}$$

where $\mathbf{P}(d)$ is the probability estimation for all pixels at depth *d*. The Soft Argmax operations are fully differentiable so that it does not hurt the end-to-end training of our model.

3.5. RGB Image and Depth Map Rendering in MVSNeRF Branch

MVSNeRF utilizes MLP layers and ray marching algorithm to render RGB images and depth maps. MVSNeRF regresses the volume density and view-dependent radiance from the neural encoding volume using MLP layers given 3D location, a viewing direction, and pixel colors. The following formula is the representation of this regression:

$$\sigma, r = A(x, d, f, c), \quad f = S(x) \tag{6}$$

where *x* is an arbitrary 3D location which is parametrized according to the reference view's normalized device coordinate (NDC) space, *d* is the viewing direction unit vector at reference view's coordinate and, *S* is the neural encoding volume. The function *f* is the neural feature trilinear interpolation from the volume *S* given the location *x*. The color vector *c* is the pixel colors represented as $c = [I_i\,(u_i, v_i)]$. It is sampled from the original input images I_i where the location (u_i, v_i) is mapped from 3D location *x* to corresponding 2D pixel coordinates in view *i*. *c* concatenates all the color vectors across all input views for the pixel position (u_i, v_i), hence, it is a 3*M*-channel vector. Finally, *A* represents the MLP layers.

MVSNeRF aims to model a neural radiance field of the scene so that the model can regress the volume density and view-dependent radiance in the scene. Moreover, the neural encoding volume constructed can be used with MLP decoder independently from 2D CNN feature extractor and 3D CNNs. It outputs the volume properties for volume rendering. These volume properties enable differentiable volume rendering of images' colors.

Volume rendering is performed via differentiable ray marching for view synthesis. Through the marching of a ray, the radiance is accumulated at the sampled shading points on the ray as follows:

$$\hat{\mathbf{Z}}(\mathbf{r}) = \sum_{k=1}^{K} w_k t_k \quad \hat{\mathbf{C}}(\mathbf{r}) = \sum_{k=1}^{K} w_k \mathbf{c}_k \tag{7}$$

$$\begin{aligned} w_k &= \tau_k \left(1 - \exp\left(-\sigma_k \delta_k\right)\right) \\ \tau_k &= \exp\left(-\sum_{k'=1}^{k} \sigma_{k'} \delta_{k'}\right) \\ \delta_k &= t_{k+1} - t_k \end{aligned} \tag{8}$$

where $\hat{\mathbf{Z}}(\mathbf{r})$ is the regressed depth pixel, $\hat{\mathbf{C}}(\mathbf{r})$ is the regressed pixel color, τ_k is the volume transmittance, and δ_k is the time interval between the sequential steps of ray marching. The volume density σ_k and the view-dependent radiance r_k are regressed via our MVSNeRF branch.

SSCasMVSNeRF regresses a portion of the pixels of the image in the training. However, our model regresses all pixels of RGB Image and the depth map during the testing.

3.6. Loss Functions

We have six different loss functions to supervise our training. None of them requires ground-truth data so that our proposed model is self-supervised.

Photometric Consistency Loss. Photometric consistency loss is computed between the warped source images and the reference image. We use the input camera parameters and the refined depth map obtained from CascadeMVSNet to warp the source images to the reference image using inverse differentiable homography warping. Then, we compute the photometric consistency loss using the following formula:

$$L_{PC} = \sum_{i=2}^{N} \frac{\left\|\left(I_i' - I_1\right) \odot M_i\right\|_2 + \left\|\left(\nabla I_i' - \nabla I_1\right) \odot M_i\right\|_2}{\left\|M_i\right\|_1} \quad (9)$$

where I_1 is the reference image, I_i' represents source images, M_i is the mask for valid pixels after source image warping.

Structural Similarity (SSIM) Loss. Structure similarity loss is computed between the warped source images and the reference image. SSIM is used to measure the difference between two images. The following formula is used:

$$L_{\mathrm{SSIM}} = \sum_{i=2}^{N} \left[1 - \mathrm{SSIM}\left(I_1, I_i'\right)\right] M_i \quad (10)$$

$$\mathrm{SSIM}(x, y) = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_2\right)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)} \quad (11)$$

and μ, σ^2 represent the average and variance of the corresponding images. M_i is the mask for valid pixels after source image warping.

Smooth Loss. Smoothness loss is computed using the depth map obtained from the CascadeMVSNet branch and the reference view. It aims to regularize depth estimation. The following formula is used for the computation:

$$L_{\mathrm{Smooth}} = \left\|\partial_x D_1\right\| e^{-\left\|\partial_x I_1\right\|} + \left\|\partial_y D_1\right\| e^{-\left\|\partial_y I_1\right\|} \quad (12)$$

where D_1 is the depth estimation of CascadeMVSNet branch, I_1 is the reference view.

Render Loss. Render loss is computed between the RGB values of pixels rendered in the MVSNeRF branch and the corresponding pixels' RGB values of the reference image using the following Mean Square Error formula:

$$\mathrm{L}_{\mathrm{render}} = \frac{1}{n} \sum_{i=1}^{n} \left(x_i - y_i\right)^2 \quad (13)$$

where *n* is the number of pixels we rendered, *x* represents the pixels of the reference view, and *y* represents the pixels rendered from MVSNeRF.

Depth Loss. Depth loss is computed between the depth values of pixels rendered in the MVSNeRF branch and the corresponding pixels of CascadeMVSNet's depth estimation. The following mean reducted smooth L1 loss formula is used for the computation:

$$\mathrm{L}_{\mathrm{depth}} = \frac{1}{n} \sum_{i=1}^{n} \begin{cases} 0.5\left(x_i - y_i\right)^2 / \beta & \text{if } \left|x_i - y_i\right| < \beta \\ \left|x_i - y_i\right| - 0.5 * \beta & \text{otherwise} \end{cases} \quad (14)$$

where β is 1, *n* is the number of pixels we rendered, *x* represents the pixels of the depth estimation of CascadeMVSNet, and *y* represents the pixels rendered from MVSNeRF. Only, depth values that are bigger than 0 are considered for the computation.

Data Augmentation Loss. Data augmentation loss is computed between the two different depth estimations of the CascadeMVSNet branch. When we compute data augmentation loss, we have two different input image sets through the CascadeMVSNet branch. The first set is the original input images. The second set is the augmented input images. These data augmentation techniques consist of random modification of the brightness and contrast of input images. Then, two different depth maps estimated from these input sets are used for the data augmentation loss computation as follows:

$$L_{DA} = \frac{1}{\left\|M\right\|_1} \sum \left\|\left(D_1 - D_{Aug}\right) \odot M\right\|_2 \quad (15)$$

where *D* variables represent the original estimation and augmented estimation correspondingly. *M* represents the mask we used for filtering depth estimations based on the occlusions.

Learning setup. Our complete loss function for the end-to-end model training is as follows:

$$L = \sum \alpha L_{PC} + \beta L_{SSIM} + \gamma L_{Smooth} + L_{Render} + L_{Depth} + \theta L_{DA} \quad (16)$$

where $\alpha = 0.8$, $\beta = 0.2$, $\gamma = 0.0067$, and $\theta = 0.1$ and its value is doubled in every 2 epochs during training.

Self-supervised CascadeMVSNeRF is an end-to-end trainable model since all of the operations in the model are differentiable.

Table 1. Point cloud evaluation results on DTU [1]. The lower is better for Accuracy (Acc.), Completeness (Comp.), and Overall. The best result is highlighted in bold and the second in italic bold.

	Method	Acc.↓	Comp.↓	Overall.↓
Sup. and Geo.	Camp [2]	0.835	0.554	0.695
	Furu [7]	0.613	0.941	0.777
	Tola [21]	0.342	1.190	0.766
	Gipuma [8]	**0.283**	0.873	0.578
	SurfaceNet [11]	0.450	1.04	0.745
	MVSNet [28]	0.396	0.527	0.462
	R-MVSNet [29]	0.383	0.452	0.417
	CIDER [26]	0.417	0.437	0.427
	Point-MVSNet [4]	0.342	0.411	0.376
	GBi-Net [17]	***0.315***	**0.262**	**0.289**
Semi-Sup.	U-MVSNet [25]	0.354	0.3535	***0.3537***
UnSup.	Unsup_MVSNet [12]	0.881	1.073	0.977
	MVS2 [6]	0.76	0.515	0.637
	M3VSNet [10]	0.636	0.531	0.583
	JDACS-MS [24]	0.398	0.318	0.358
	Ours	0.4209	***0.2927***	0.3568

4. Experiments

4.1. Datasets

The **DTU** dataset [1] is an indoor dataset with multi-view images and camera poses. We Follow MVSNet [28] for dividing training and testing set. There are 27097 training samples in total. The **BlendedMVS** dataset [30] is a large-scale dataset with indoor and outdoor scenes. Following [16, 23, 31], we only use this dataset for training. There are 16904 training samples in total. **Tanks and Temples** [13] is a large-scale dataset with various outdoor scenes. It contains Intermediate subset and Advanced subset. The evaluation on this benchmark is conducted online by submitting generated point clouds to the official website.

4.2. Implementation Details

Training Details. The proposed SSCasMVSNeRF is trained on the DTU dataset for DTU benchmarking and tested on the DTU testing set. The same model is directly tested on Tanks and Temples benchmarking without fine-tuning, following [16, 23, 31]. We use the high-resolution DTU data provided by the open-source code of MVSNet [28]. The original image size is 1200×1600. We first rescale the input images into 600×800 following MVSNet [28]. And we then crop images of 512×640 from images of 600×800. The motivation is that cropping smaller images from larger images could help to learn better features for larger image scales without increasing the training overhead. The number of input images is set to $N = 3$ and the maximum stage number is set to 3. For every stage, we use different multi-scale feature maps and the 3D-CNN network parameters. The whole network is optimized by Adam optimizer in Pytorch for 16 epochs with an initial learning rate of 0.0001, which is down-scaled by a factor of 2 after 10, 12, and 14 epochs. The total training batch size is 4 on four NVIDIA RTX 3090 GPUs.

Testing Details. The model trained on the DTU training set is used for testing on DTU testing set. The input image number N is set to 5, each with a resolution of 1152×1600. It takes 0.912 seconds for each testing sample. The model trained on the DTU training dataset is used for testing on Tanks and Temples intermediate and advanced dataset. The image sizes are set to 1024×1920 or 1024×2048 to make the images divisible by 64. The input image number N is set to 7. All the testings are conducted on an NVIDIA RTX 3090 GPU. We then filter and fuse depth maps of a scene into one point cloud, by photometric and geometric consistencies. The geometric consistency is defined similarly to that of MVSNet [28]. The photometric is defined from the average of classification probabilities. Fig. 2 and Fig. 3 are visualizations of depth maps and point clouds of our method.

4.3. Benchmark Performance

Overall Evaluation on DTU Dataset. We evaluate the results of the DTU testing set by two types of metrics. The first type of metric evaluates point clouds using official evaluation scripts of DTU [1]. It compares the distance between ground-truth point clouds and the produced point clouds.

The state-of-the-art comparison results are shown in Table 1. Our model SSCasMVSNeRF is significantly improved, with the best performance on the completeness and overall score (lower is better for both metrics) among all the unsupervised methods.

Our best model improves the overall score from **0.583** of M3VSNet [10] to **0.3568**, while the completeness is improved by 0.0253 compared to JDACS [24]. The overall score is also comparable to some full-supervised state-of-the-art methods.

Overall Evaluation on Tanks and Temples. We train the proposed SSCasMVSNeRF on DTU traning set, and test on Tanks and Temples dataset. We compare our method to state-of-the-art methods. Table 2 shows results on both the Advanced subset and the Intermediate subset. Our SSCasMVSNeRF achieves a mean score of 29.46 (higher is better) on the Advanced subset. Note that the Advanced subset contains different large-scale outdoor scenes. The results can fully confirm the effectiveness of our method. Table 2 also shows the evaluation results on the Intermediate subset. Except for achieving state-of-the-art among un-supervised methods, our SSCasMVSNeRF even obtains highly comparable results to the state-of-the-art supervised methods. Notably, with significant improvement, our mean score is 8.13 higher than JDACS [24] and only 3.46 lower than U-MVSNet [25]. Moreover, we also obtain state-of-the-art scores on all the sub-scenes.

Table 2. Point cloud evaluation results on the Advanced and Intermediate subsets of Tanks and Temples dataset [13]. Higher scores are better. The Mean is the average score of all scenes. The sections are partitioned into supervised, semi-supervised and end-to-end unsupervised, respectively. The best result is highlighted in bold.

	Advanced							Intermediate								
Method	Mean	Aud.	Bal.	Cou.	Mus.	Pal.	Tem.	Mean	Fam.	Fra.	Hor.	Lig.	M60	Pan.	Pla.	Tra.
MVSNet [28]	-	-	-	-	-	-	-	43.48	55.99	28.55	25.07	50.79	53.96	50.86	47.90	34.69
Point-MVSNet [4]	-	-	-	-	-	-	-	48.27	61.79	41.15	34.20	50.79	51.97	50.85	52.38	43.06
UCSNet [5]	-	-	-	-	-	-	-	54.83	76.09	53.16	43.03	54.00	55.60	51.49	57.38	47.89
CasMVSNet [9]	31.12	19.81	38.46	29.10	43.87	27.36	28.11	56.42	76.36	58.45	46.20	55.53	56.11	54.02	58.17	46.56
PatchmatchNet [22]	32.31	23.69	37.73	30.04	41.80	28.31	32.29	53.15	66.99	52.64	43.24	54.87	52.87	49.54	54.21	50.81
GBi-Net [17]	**37.32**	**29.77**	**42.12**	**36.30**	**47.69**	**31.11**	**36.93**	**61.42**	**79.77**	**67.69**	**51.81**	**61.25**	**60.37**	**55.87**	**60.67**	**53.89**
U-MVSNet [25]	30.97	22.79	35.39	28.90	36.70	28.77	33.25	57.15	76.49	60.04	49.20	55.52	55.33	51.22	56.77	52.63
MVS2 [6]	-	-	-	-	-	-	-	37.21	47.74	21.55	19.50	44.54	44.86	46.32	43.38	29.72
M3VSNet [10]	-	-	-	-	-	-	-	37.67	47.74	24.38	18.74	44.42	43.45	44.95	47.39	30.31
JDACS-MS [24]	-	-	-	-	-	-	-	45.48	66.62	38.25	36.11	46.12	46.66	45.25	47.69	37.16
Ours	29.46	20.87	34.3	27.46	36.55	26.78	30.81	53.61	73.53	50.3	44.89	52.66	52.18	49.76	54.55	51

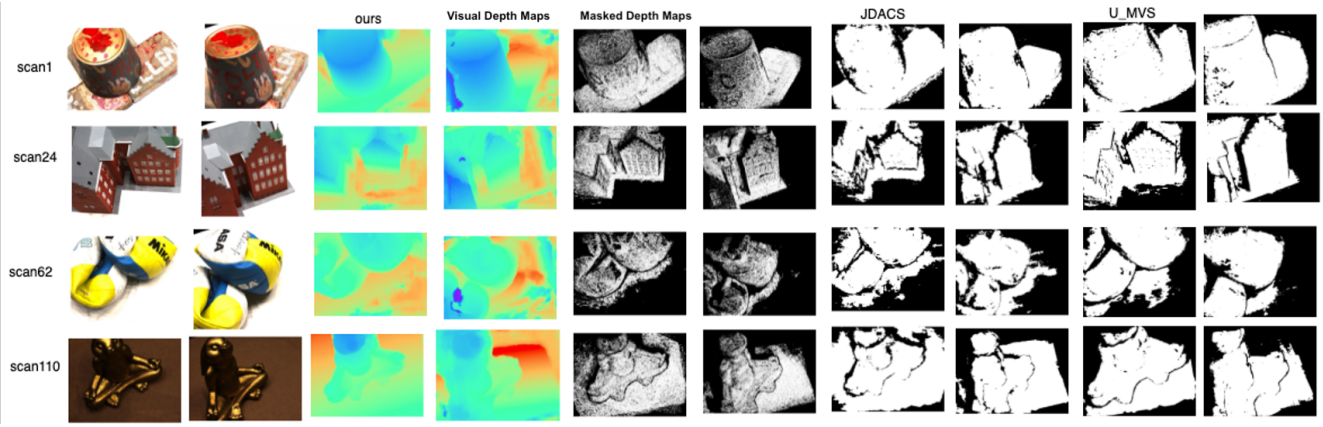

Figure 2. Qualitative comparison of the generated depth maps on the DTU benchmark. From left to right: results of our self-supervised method, results of JDACS [24], results of U-MVSNet [25], Far left is the corresponding RGB images.

Table 3. Depth map evaluation results in terms of accuracy on DTU evaluation set [1]. The unit of these thresholds are all in millimeters. The higher is better.

Method	<2↑	<4↑	<8↑
MVSNet [28]	0.704	0.778	0.815
MVSNeRF [22]	0.510	0.645	0.734
Unsup_MVSNet [12]	0.317	0.384	0.402
M3VSNet [10]	0.603	0.769	**0.857**
JDACS-MS [24]	0.553	0.705	0.786
JDACS [24]	0.610	0.724	0.779
Ours	**0.707**	**0.808**	0.845

Table 4. Ablation study of different components of our proposed unsupervision framework using CasMVSNet as backbone. The lower is better. L_{DA}: Data Augmentation Loss. L_{PC}:Photometric Consistency Loss. L_{NeRF}: NeRF Rendering Loss.

L_{DA}	L_{PC}	L_{NeRF}	Acc↓	Comp↓	Overall↓
	✓		0.462	0.328	0.395
✓	✓		0.419	0.346	0.383
✓	✓	✓	**0.4209**	**0.2927**	**0.3568**

4.4. Ablation Study

4.4.1 Comparison with backbone network

From Table 1, it's obvious that there's still a gap between previous unsupervised methods and multi-stage self-supervised methods. In order to provide a fair comparison of our contributions, we show the results of our unsupervised framework and supervised backbone in the same settings of hyper-parameters. The results are shown in Table 5. The performance in Table. 5 illustrates our method is even better than the opponent trained by ground truth depth, which shows the effectiveness of our contribution.

Figure 3. Qualitative comparison of the generated point clouds on the DTU benchmark. From left to right: Ground Truth, results of our self-supervised method, results of unsup_mvs [12], results of JDACS [24].

Table 5. Comparison between the backbone CasMVSNet [9] with the same setting trained by ground truth and our SSCasMVSNeRF.

Method	Supervised	Input Size	Depth Size	Acc↓	Comp↓	Overall↓
CasMVSNet	✓	1600 ×1152	1600 ×1152	**0.325**	0.385	0.355
Ours	×	1600 ×1152	1600 ×1152	0.4209	**0.2927**	**0.3568**

4.4.2 Effect of each component of the unsupervised framework

To evaluate the performance gain of the idea in our framework, quantitative results are listed in Table. 4. We can observe easily that our contribution yield better reconstruction results, especially completeness.

5. Conclusion

In this project, we think beyond the previous assumption of photometric consistency and propose a novel rendering consistency unsupervised MVS framework SSCasMVSNeRF. To handle the ambiguous supervision from photometric consistency, we propose to use a NeRF-like structure to create additional supervision from volumetric rendering and provide RGB-level consistency. The experiments prove the effectiveness of our SSCasMVSNeRF framework.

References

[1] Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, 120(2):153–168, 2016. 6, 7

[2] Neill DF Campbell, George Vogiatzis, Carlos Hernández, and Roberto Cipolla. Using multiple hypotheses to improve depth-maps for multi-view stereo. In *ECCV*. Springer, 2008. 2, 6

[3] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. *arXiv preprint arXiv:2103.15595*, 2021. 1, 2, 3

[4] Rui Chen, Songfang Han, Jing Xu, and Hao Su. Point-based multi-view stereo network. In *ICCV*, 2019. 6, 7

[5] Shuo Cheng, Zexiang Xu, Shilin Zhu, Zhuwen Li, Li Erran Li, Ravi Ramamoorthi, and Hao Su. Deep stereo using adaptive thin volume representation with uncertainty awareness. In *CVPR*, June 2020. 7

[6] Yuchao Dai, Zhidong Zhu, Zhibo Rao, and Bo Li. Mvs2: Deep unsupervised multi-view stereo with multi-view symmetry [24]. In *2019 International Conference on 3D Vision (3DV)*, pages 1–8. IEEE, 2019. 6, 7

[7] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *TPAMI*, 32(8):1362–1376, 2009. 2, 6

[8] Silvano Galliani, Katrin Lasinger, and Konrad Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *ICCV*, 2015. 2, 6

[9] Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2495–2504, 2020. 1, 2, 3, 7, 8

[10] Baichuan Huang, Hongwei Yi, Can Huang, Yijia He, Jingbin Liu, and Xiao Liu. M3vsnet: Unsupervised multi-metric multi-view stereo network. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 3163–3167. IEEE, 2021. 6, 7

[11] Mengqi Ji, Juergen Gall, Haitian Zheng, Yebin Liu, and Lu Fang. Surfacenet: An end-to-end 3d neural network for multiview stereopsis. In *ICCV*, 2017. 2, 6

[12] Tejas Khot, Shubham Agrawal, Shubham Tulsiani, Christoph Mertz, Simon Lucey, and Martial Hebert. Learning unsupervised multi-view stereopsis via robust photometric consistency. *arXiv preprint arXiv:1905.02706*, 2019. 2, 6, 7, 8

[13] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM ToG*, 36(4):1–13, 2017. 6, 7

[14] Kiriakos N Kutulakos and Steven M Seitz. A theory of shape by space carving. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 1, pages 307–314. IEEE, 1999. 2

[15] Maxime Lhuillier and Long Quan. A quasi-dense approach to surface reconstruction from uncalibrated images. *TPAMI*, 27(3):418–433, 2005. 2

[16] Xinjun Ma, Yue Gong, Qirui Wang, Jingwei Huang, Lei Chen, and Fan Yu. Epp-mvsnet: Epipolar-assembling based depth prediction for multi-view stereo. In *ICCV*, 2021. 6

[17] Zhenxing Mi, Di Chang, and Dan Xu. Generalized binary search network for highly-efficient multi-view stereo. *arXiv preprint arXiv:2112.02338*, 2021. 6, 7

[18] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European conference on computer vision*, pages 405–421. Springer, 2020. 1, 2

[19] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*. Springer, 2016. 2

[20] Steven M Seitz and Charles R Dyer. Photorealistic scene reconstruction by voxel coloring. *IJCV*, 35(2):151–173, 1999. 2

[21] Engin Tola, Christoph Strecha, and Pascal Fua. Efficient large-scale multi-view stereo for ultra high-resolution image sets. *Machine Vision and Applications*, 23(5):903–920, 2012. 6

[22] Fangjinhua Wang, Silvano Galliani, Christoph Vogel, Pablo Speciale, and Marc Pollefeys. Patchmatchnet: Learned multi-view patchmatch stereo. In *CVPR*, 2021. 7

[23] Zizhuang Wei, Qingtian Zhu, Chen Min, Yisong Chen, and Guoping Wang. Aa-rmvsnet: Adaptive aggregation recurrent multi-view stereo network. In *ICCV*, October 2021. 6

[24] Hongbin Xu, Zhipeng Zhou, Yu Qiao, Wenxiong Kang, and Qiuxia Wu. Self-supervised multi-view stereo via effective co-segmentation and data-augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, page 6, 2021. 1, 2, 6, 7, 8, 9

[25] Hongbin Xu, Zhipeng Zhou, Yali Wang, Wenxiong Kang, Baigui Sun, Hao Li, and Yu Qiao. Digging into uncertainty in self-supervised multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6078–6087, 2021. 6, 7

[26] Qingshan Xu and Wenbing Tao. Learning inverse depth regression for multi-view stereo with correlation cost volume. In *AAAI*, volume 34, 2020. 6

[27] Yao Yao, Shiwei Li, Siyu Zhu, Hanyu Deng, Tian Fang, and Long Quan. Relative camera refinement for accurate dense reconstruction. In *2017 International Conference on 3D Vision (3DV)*, pages 185–194. IEEE, 2017. 2

[28] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *ECCV*, 2018. 1, 2, 6, 7

[29] Yao Yao, Zixin Luo, Shiwei Li, Tianwei Shen, Tian Fang, and Long Quan. Recurrent mvsnet for high-resolution multi-view stereo depth inference. In *CVPR*, 2019. 6

[30] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *CVPR*, 2020. 6

[31] Jingyang Zhang, Yao Yao, Shiwei Li, Zixin Luo, and Tian Fang. Visibility-aware multi-view stereo network, 2020. 6