MultiViewStereoNet: Fast Multi-View Stereo Depth Estimation using Incremental Viewpoint-Compensated Feature Extraction

W. Nicholas Greene Nicholas Roy

***Abstract*— We propose a novel learning-based method for multi-view stereo (MVS) depth estimation capable of recovering depth from images taken from known, but unconstrained, views. Existing MVS methods extract features from each image independently before projecting them onto a set of planes at candidate depths to compute matching costs. By projecting features after extraction, networks must learn rotation and scale invariant representations even though the relative poses of the cameras are known. In our approach, we compensate for viewpoint changes directly in the extraction layers, allowing the network to learn features that are projected by construction and reducing the need for rotation and scale invariance.**

Compensating for viewpoint changes naively, however, can be computationally expensive as the feature layers must either be applied multiple times (once per depth hypothesis), or replaced by 3D convolutions. We overcome this limitation in two ways. First, we only compute our matching cost volume at a coarse image scale before upsampling and refining the outputs. Second, we incrementally compute our projected features such that the bulk of the layers need only be executed a single time across all depth hypotheses. The combination of these two techniques allows our method to perform competitively with the state-of-the-art, while being significantly faster. We call our method MultiViewStereoNet and release our source code publicly for the benefit of the robotics community.

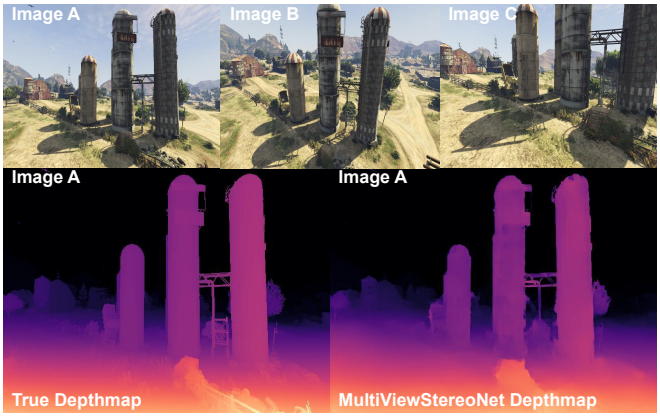

Fig. 1: *MultiViewStereoNet* – We propose a novel, learning-based method for multi-view stereo (MVS) depth estimation that we call MultiViewStereoNet. By combining coarse stereo matching costs, guided refinement, and incrementally computed features that compensate for known viewpoint changes, our method is able to achieve reconstruction accuracy comparable to the state-of-the-art, while being significantly faster at runtime. The top row of the figure above shows input images that are used to generate the depthmap in the bottom right. The groundtruth depthmap is shown in the bottom left for comparison.

I. Introduction

Multi-view stereo (MVS) is a fundamental problem in computer vision where the geometry of a scene is estimated from a set of images taken from known, but otherwise unconstrained, viewpoints. While the scene geometry may be represented in a variety of ways, a common design choice is to designate one of the images as a privileged reference and estimate a depthmap with respect to that image. Classical methods [1], [2] generally start by defining a volume in the reference image's coordinate frame by sampling a set of depths for each reference pixel. Matching costs that record how consistent a depth hypothesis is with the neighboring (or *comparison*) images are then computed by projecting each pixel at each candidate depth into the comparison views and comparing intensities. After filtering the volume to reduce noise, the reference depthmap that minimizes the matching costs can be extracted.

Plane Sweep stereo techniques [3]–[5] compute the matching cost volume more efficiently by interpreting the volume as a set of planes, one for each depth hypothesis. The comparison images can then be projected (or *warped*) onto each plane, creating a set of transformed images (one for each depth hypothesis) that are then compared to the reference image directly to compute matching costs.

In recent years, deep learning approaches have shown great promise at solving the MVS problem by exploiting prior information learned from large training datasets [6]–[12]. Instead of using raw pixel intensities or hand-crafted feature extractors and filtering schemes, these systems learn the MVS components from data by training stacked layers of convolutional neural networks (CNNs). The weights of these convolutional layers can encode additional global context and semantic information that can improve estimation performance in the presence of lighting changes, low texture, and other imperfections common in natural scenes.

Despite the rapid progress enabled by learned feature descriptors, MVS depth estimation is still a challenging problem in the wild, primarily due to the difficulty in robustly matching dense image features across the viewpoint changes common with freely moving cameras. Objects in a scene can appear radically different, or be occluded entirely, when viewed from disparate viewing angles or lighting. Any learning-based system must also generalize beyond the data used to train the network. MVS is a particularly difficult problem in this sense, as supporting wholly unconstrained camera motion at test time requires extensive training sam-

Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 {wng,nickroy}@csail.mit.edu

This material is based upon work supported by the Army Research Laboratory under Cooperative Agreement Number W911NF-17-2-0181. Their support is gratefully acknowledged.

ples to ensure adequate coverage of the operating regimes.

Designing networks that can learn distinctive feature representations from limited data is therefore of primary importance to solving MVS. Current learning-based methods, however, do not leverage all information available to aid this process. In particular, modern networks extract learned features from each input image independently before projecting them onto the planes that comprise the cost volume. By applying the projection after feature extraction, the learned features must implicitly compensate for this projection and exhibit scale and rotation invariance despite never being exposed to the projection parameters. The projection parameters (the camera intrinsics and extrinsics) are assumed known, however, which suggests more structure can be imposed on the feature extraction layers.

Our key insight is that by compensating for the known viewpoint changes during the feature extraction process itself, the network can learn features that are *specific* to the desired reference frame and *projected by construction*. This technique lessens the burden on the network to achieve scale and rotation invariance and therefore increases robustness to viewpoint changes during matching.

Compensating for viewpoint changes in this way can be computationally expensive, however, if care is not taken. In principle, we must extract features not from a single comparison image, but from the *set* of warped comparison images produced by projecting the image data onto the planes that comprise the cost volume. One can naively apply a conventional feature extractor CNN to each warped image, but this approach quickly grows unmanageable as the number of planes (i.e. depth samples) increases and the feature extractor must be run repeatedly. Alternatively, layers of 3D convolutions could be used to extract features from the volume generated by concatenating the warped images, but these more complex layers are similarly expensive and prevent the use of commonly accepted network architectures built on stacks of 2D convolutions.

In this work, we overcome these limitations in two key ways. First, we generalize the approach of Khamis et al. (StereoNet) [13] from the two-view, rectified stereo domain to the multi-view, unrectified setting using differentiable Spatial Transformer Networks (STNs) [14]. Like StereoNet, we compute features and matching costs at a reduced image scale to produce coarse depthmaps that are then iteratively upsampled and refined with the image data as guidance. This type of architecture retains the benefits of learned stereopsis, but drastically reduces the amount of costly high-resolution feature matching for improved speed.

Second, we incrementally compute our projected features such that the bulk of the feature extraction layers need only be executed a single time across all depth hypotheses. We initially generate a single feature map corresponding to the furthest depth plane in the volume with a conventional feature network. Then we apply a series of inexpensive homographies, coupled with simple refinement layers, that incrementally warp this feature map to other depth planes in the reference volume.

The combination of these two techniques allows our method to achieve reconstruction accuracy comparable to the state-of-the-art, while being significantly more efficient. We call our method MultiViewStereoNet and release our source code publicly for the benefit of the robotics community.

II. Related Work

A. Two-View Stereo

Two-view stereo generally refers to the scenario where two cameras are rigidly mounted along a narrow baseline such that the corresponding images can be rectified onto a common image plane to estimate disparities [15], [16]. Early attempts to apply machine learning to this problem replaced one or more classical building blocks with learned components before completely end-to-end solutions were proposed.

Zbontar and LeCun, for example, proposed a network to compute matching costs from small image patches, before using the costs in a classical pipeline [17]. Mayer et al. developed a network that directly regresses disparity using stacks of convolutions and deconvolutions [18]. Kendall et al. aggregate global context in the stereo cost volume using 3D convolutions [19]. Khamis et al. similarly use 3D convolutions to aggregate information in the cost volume, but significantly reduce the spatial resolution of the volume for speed before applying image-guided refiners to upsample the resulting disparities [13]. Our solution takes inspiration from this network structure and generalizes it to the multi-view setting.

B. Multi-View Stereo

Learning-based approaches to MVS often follow Plane Sweep [3], where matching costs from multiple images are aggregated in a single volume after geometric warping [3]–[5]. MVSNet from Yao et al., for example, extracts features per image, transforms them into the reference volume using a differentiable warp, then regularizes the costs using multi-scale 3D convolutions [8], [10]. Gu et al. and Luo et al. build upon MVSNet by improving the efficiency and accuracy of cost volume generation [9], [11]. Similarly, Im et al. refine the costs for each depth hypothesis using the reference image features [6]. Wang and Shen compute a multi-view cost volume using classical techniques, but then regress the depths using an encoder-decoder network [20]. Huang et al., on the other hand, estimate depthmaps on 64×64 pixel patches before tiling the results to the input resolution [7].

III. MultiViewStereoNet

The MultiViewStereoNet architecture is divided into four primary components as shown in Figure 2 and Figure 3. The reference image is first passed through a conventional feature extraction network composed of strided 2D convolutional layers with residual connections [21] to generate a set of reference features. Each comparison image, on the other hand, is passed through our novel, viewpoint-compensated feature network that incrementally computes projected features for each candidate depth. These feature maps are then

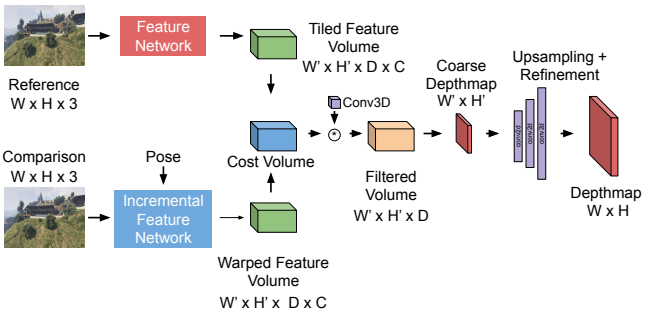

Fig. 2: *MultiViewStereoNet Block Diagram* – In our network, coarse-resolution features are extracted from the input images using two subnetworks: a traditional CNN for the reference image and a novel CNN that compensates for the known viewpoint change for the comparison images. The two sets of feature volumes are then combined to form a cost volume, from which a coarse depthmap is extracted. A series of image-guided refiners is then used to upsample the depthmap to the input image size.

concatenated to form a warped feature *volume*. After tiling the reference feature map to create an identically sized feature volume, the absolute difference of the two volumes forms our coarse cost volume. We then apply a series of 3D convolutions and normalization steps to filter the costs, before extracting depths using a `softargmin` operator. The coarse depthmap is then passed through a series of upsampling and image-guided refinement layers to produce the final depthmap.

A. Reference Feature Network

Our reference feature network is derived from that detailed by Khamis et al. [13]. The reference image is first passed through 4 2D convolutional layers with kernel size 5, stride 2, and 32 output channels. The resulting feature map is then passed through 6 residual blocks with kernel size 3, stride 1, and the same 32 output channels. We make two modifications to the residual connections described in [13]. First, we use only a single convolution in the skip connection, instead of the normal two as described by He et al. [21], which achieves similar performance with fewer parameters. Second, we replace the batch normalization [22] layer with a group normalization layer [23] to better support small batch sizes during training. For an image of size $H \times W \times 3$, this network produces a feature map of size $H' \times W' \times C$ for $W' = W/16$, $H' = H/16$, and $C = 32$. Given a set of D candidate depth samples $\{d_i\}_{i=1}^{D}$, we then tile the feature map to produce our reference feature volume of size $H' \times W' \times D \times C$.

B. Incremental Viewpoint-Compensated Feature Network

Before describing our incremental, viewpoint-compensated feature network, we first review some key concepts from multi-view geometry [24], [25]. Assume we have two cameras: a reference camera r and a comparison camera c, with the same intrinsic parameters $K \in \mathbb{R}^{3\times3}$. Let $\Omega_r, \Omega_c \subset \mathbb{R}^2$ denote the image domain of each camera, respectively. Let $I_r : \Omega_r \to \mathbb{R}^3$ and $I_c : \Omega_r \to \mathbb{R}^3$ designate the images from the two cameras with 3 channels (e.g. representing RGB values). Finally, let $R_c^r \in \mathbb{SO}(3)$ and

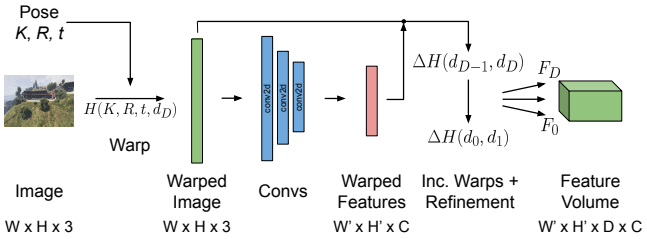

Fig. 3: *Incremental Viewpoint-Compensated Feature Network* – Our novel feature network compensates for known viewpoint changes by projecting the comparison image before (rather than after) extraction. We incrementally compute each plane F_i of the comparison feature volume (corresponding to depth hypothesis d_i) from the previous plane using the relative homography ΔH between the planes. This allows for the feature maps to be computed for each depth hypothesis, while only requiring the bulk of the convolutional layers to be executed once, increasing the network's speed.

$t_c^r \in \mathbb{R}^3$ represent rotation and translation of the comparison camera with respect to the reference camera, which we assume are known.

If the scene geometry can be represented by a single plane with normal vector $n \in \mathbb{R}^3$ and depth $d > 0$ with respect to the reference coordinate system, the transform that projects (homogeneous) pixels from Ω_r to Ω_c is given by the function $H(d) : \Omega_r \to \Omega_c$, which can be represented by a 3×3 homography matrix:

$$H(d) = K(R_r^c - t_r^c n^T/d)K^{-1}. \tag{1}$$

The image $\tilde{I}_d : \Omega_r \to \mathbb{R}^3$ represents the projection of the comparison image onto the reference plane at depth d and is given by

$$\tilde{I}_d(\mathbf{u}) = I_c(\pi(H(d)\bar{\mathbf{u}})), \tag{2}$$

where $\bar{\mathbf{x}} = (\mathbf{x}, 1)$ signifies a homogeneous pixel coordinate and $\pi(x, y, z) = (x/z, y/z)$ denotes the perspective projection function. When implemented, the pixel domain Ω_r is uniformly sampled to generate discrete pixels before applying $H(d)$ and the indexing into I_c is accomplished using bilinear interpolation, which describes a type of Spatial Transformer Network (STN) [14].

In Plane Sweep stereo [3]–[5], one computes a series of such transformed images (one for each candidate depth d_i), which are then compared to the reference image to compute matching costs. In existing learned MVS systems, the original image channels are simply replaced by learned features $F : \Omega \to \mathbb{R}^C$, where C denotes the number of feature channels. Note, however, that the feature extraction occurs *before* the projection, which means the features must implicitly compensate for any scale, rotation, or perspective changes between the cameras. One could extract features for each warped image $\tilde{I}_d$ independently or concatenate the $\tilde{I}_d$ into a volume and apply 3D convolutions, but both options are computationally expensive. Instead, we will take an *incremental* approach to feature extraction, building the feature map F_i for candidate depth d_i from the neighboring feature map F_{i+1} for depth d_{i+1}.

We compute the initial feature map F_D corresponding to the maximum candidate depth d_D by transforming the

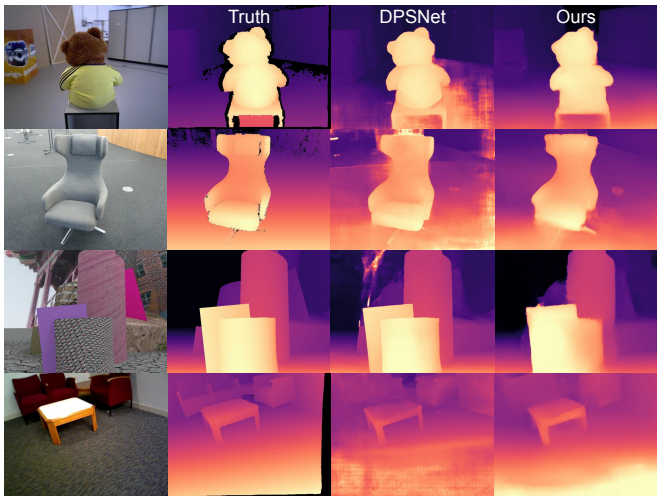

Fig. 4: *DeMoN Depthmaps*: Here we show the qualitative reconstruction performance of MultiViewStereoNet on the DeMoN benchmark test set [26]. MultiViewStereoNet is capable of producing depthmaps comparable to DPSNet, while being significantly faster.

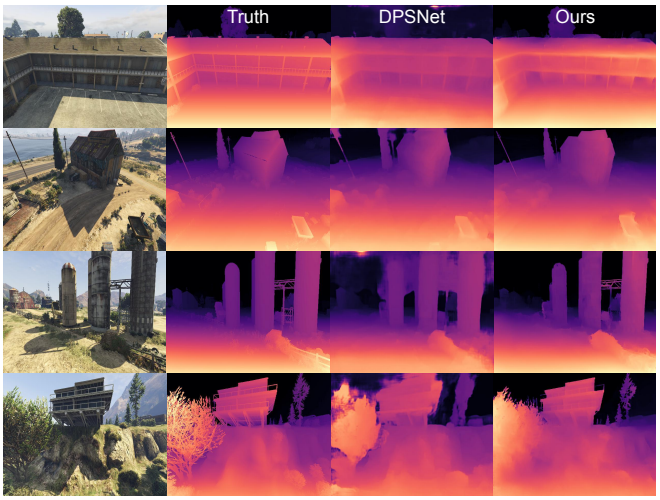

Fig. 5: *GTA-SfM Depthmaps*: MultiViewStereoNet is capable of producing compelling depth estimates on challenging imagery, such as that from the GTA-SfM dataset. Here, example depthmaps from the test set are shown.

comparison image I_c by $H(d_D)$ to form $\tilde{I}_{d_D}$ and applying the feature extraction network described in Section III-A. Computing F_{D-1} from F_D is accomplishing by applying the *relative* homography $\Delta H(d_{D-1}, d_D)$ from depth plane d_{D-1} to d_D given by

$$\Delta H(d_{D-1}, d_D) = H(d_D)^{-1} H(d_{D-1}). \tag{3}$$

Note that these homographies use a scaled intrinsic matrix to reflect the downsampling of the image domain. The features F_{D-1} can then be computed as

$$F_{D-1}(\mathbf{u}) = F_D(\pi(\Delta H(d_{D-1}, d_D)\mathbf{u}). \tag{4}$$

This incremental technique allows for the feature maps for all candidate depths to be computed using only a single invocation of the feature extraction network, while still appropriately compensating for the known viewpoint changes. It is possible, however, that the relative homographies $\Delta H(d_{i-1}, d_i)$ generate pixel locations that lie outside (or on the boundary) of the valid domain in the parent feature map F_i. To account for these edge cases, we apply another instance of refinement using the warped image as guidance. We concatenate F_i and $\tilde{I}_i$ along the channel dimension and apply 3 convolutional layers with kernel size 3, stride 1, with a single skip connection. The outputs of these layers are then added to the original feature map.

After feature refinement, we concatenate the feature maps across all the depth samples into a warped feature volume of size $H' \times W' \times D \times C$.

C. Cost Volume Formulation and Filtering

To compute matching costs, we take the absolute difference of the reference and warped comparison feature volumes described in Section III-A and III-B. Although there is some evidence that asymmetric distance measures [13] or concatenating feature channels [19] improves matching quality, we found simple absolute differences to work well.

The cost volume is passed through a series of 3D convolutional layers designed to pool global context information and reduce noise. We follow the architecture proposed by Khamis et al. [13] and use 4 3D convolutional layers with each followed by a groupnorm [23] operation and LeakyReLU activation. The input and output channels for these layers are of equal size. A final 3D convolutional layer is then applied which reduces the feature dimension to a scalar, resulting in a volume of size $H' \times W' \times D$. The kernel sizes for all layers is set to $3 \times 3 \times 3$.

D. Depth Regression and Guided Refinement

We extract a coarse depthmap of size $H' \times W'$ from the cost volume before upsampling and refining the outputs to the input resolution. Relying on upsampling and refinement, rather than high-resolution feature matching, has significant advantages in terms of efficiency, as described by Khamis et al. in StereoNet [13]. StereoNet, however, assumes the input images are rectified, meaning the feature projection described in Equation 2 can be accomplished by a simple shift of indices. Here, we employ a similar upsampling and guided refinement scheme, but apply it to our multi-view features and cost volume.

For a given pixel $\mathbf{u}$, let c_i denote the matching costs for depth sample i. We form a probability distribution σ over the depths by applying the `softmin` operator:

$$\sigma(c_i) = \frac{\exp(-c_i)}{\sum_j \exp(-c_j)}. \tag{5}$$

The depth $D(\mathbf{u})$ for pixel $\mathbf{u}$ is then given by the mean of this distribution, $D(\mathbf{u}) = \sum_i d_i \sigma(c_i)$.

Next, we iteratively upsample the coarse depthmap using bilinear interpolation and then pass it through an image-guided refinement network to resolve fine structures. The refinement network concatenates the coarse depthmap and appropriately-sized reference image before passing them through an initial convolutional layer with 32 output channels. Group normalization is applied along with a

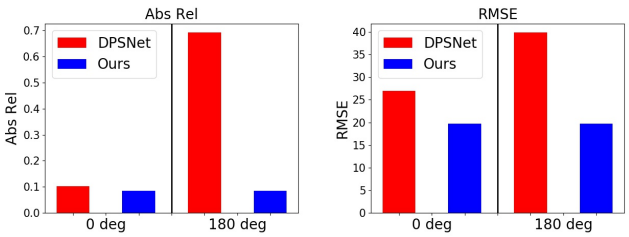

Fig. 6: *Viewpoint Compensation*: Our viewpoint-compensated features take the known camera poses into account during extraction. Here we compare reconstruction performance of MultiViewStereoNet and DPSNet on the GTA-SfM dataset using two test sets: a standard test set and a set where one of the images for each sample is rolled about the optical axis by 180 degrees. Despite being given the parameters of the roll, DPSNet sees a significant increase in absolute relative depth error (left) and RMSE depth error (right). MultiViewStereoNet sees no significant difference in performance across the two test sets.

LeakyReLU activation. After this, 6 dilated residual block layers are applied, where each block consists of a single convolution with group normalization and LeakyReLU activation, followed by a skip connection. The dilation strides are set to $(1, 2, 4, 8, 1, 1)$. The input and output channels for these layers is kept at 32. A final convolution is then applied to reduce the output to a single channel representing a depth residual. This residual is then added to the input depthmap. We apply an initial refinement round to the coarse depthmap extracted from the cost volume followed by 4 rounds of upsampling by a factor of 2 and refinement to yield the final output depthmap of size $H \times W$.

E. Multi-View Fusion

In conventional Plane Sweep Stereo, fusing depth information from multiple comparison views is achieved by simply averaging matching costs for each comparison image. We found this approach to be brittle in a learned MVS context since certain locations in the reference volume will only be observable from a subset of the cameras. Careful bookkeeping is then required to keep track of these locations and perform the averaging correctly. We found performing the fusion at the *depthmap* level, rather than the cost volume level, to be more robust. We pass each comparison image through a subset of the network to producing a set of coarse depthmaps of size $H' \times W'$. We then average these depthmaps and then proceed with the final upsampling and refinement layers described in Section III-D.

IV. Evaluation

A. Implementation Details

We implemented our network using PyTorch [28]. Training was performed on 8 Nvidia V100 GPUs with a batch size of 8 per GPU, while testing was performed on a single Nvidia GTX 1080Ti with a batch size of 1. We used the Adam optimizer [29] for training, with a learning rate of 0.001. We use the pseudo-Huber loss described by Barron [30] against groundtruth depth labels applied to the depthmaps at each image scale. The depth samples $\{d_i\}$ for our cost volumes are generated by sampling uniformly in inverse depth space between 0 (infinite depth) and a maximum inverse depth value computed for each training example based on a maximum disparity of 192 pixels. We set $D = 12$ for all experiments. To remove any dependence on the metric scale of the geometry, we normalize the camera poses to have unit baseline before computation. All training was performed using two input images per sample.

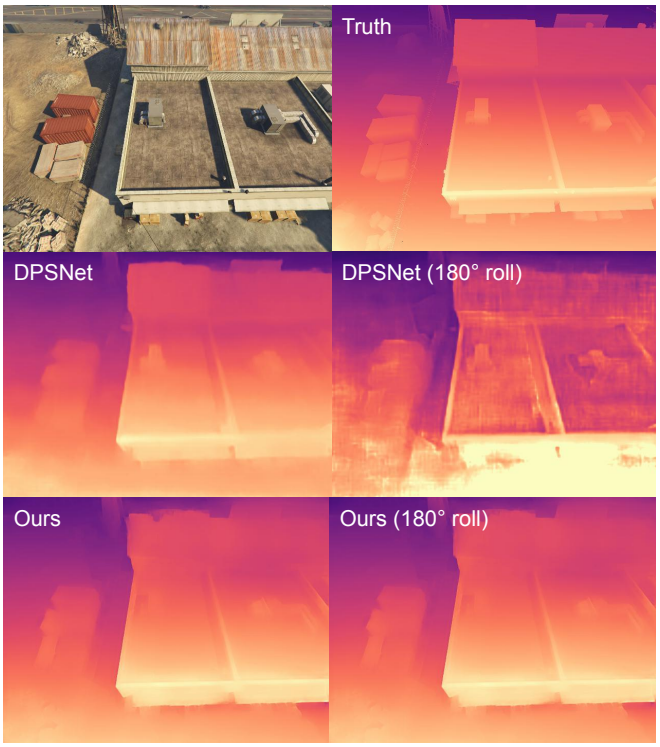

Fig. 7: *Viewpoint Compensation*: Here we show the qualitative effect of viewpoint-compensated features on the generated depthmaps. DPSNet sees a significant drop in depth quality when one of the input images is rotated about the optical axis by 180 degrees. MultiViewStereoNet sees no change in quality.

B. DeMoN Benchmark

We evaluate our approach against the DeMoN dataset [26] commonly used to benchmark MVS systems. This dataset includes 51k training scenes assembled from both real and simulated imagery. We use the same training split as [6], which yields 168k training samples and 708 test samples at VGA resolution. Groundtruth depths are provided via RGBD sensors or simulation. For this dataset, we train for 45 epochs and compute standard depth metrics against groundtruth that are summarized in Table I. We compare our proposed network against DPSNet [6], MVDepthNet [20], DeepMVS [7], and a traditional reconstruction pipeline based on COLMAP [27]. As shown in Table I, our approach achieves depth reconstruction accuracy and completeness comparable to the state-of-the-art, while being significantly more efficient. Figure 4 shows qualitative performance on the DeMoN test set, where we compare favorably with DPSNet.

C. GTA-SfM Multi-View Evaluation

We also evaluate our network on the GTA-SfM dataset presented by Wang and Shen [31]. This dataset contains

DeMoN Benchmark

Dataset	Method	Abs Rel ↓	Sq Rel ↓	RMSE ↓	$\text{RMSE}_{\log}$ ↓	α_1 ↑	α_2 ↑	α_3 ↑	Runtime [sec] ↓
MVS	COLMAP	0.38	1.26	1.48	0.50	0.48	0.66	0.84	-
	DeepMVS	0.23	0.62	1.15	0.30	0.67	0.89	0.94	80.9
	MVDepthNet	0.20 ± 0.02	0.47 ± 0.10	0.71 ± 0.06	0.25 ± 0.02	**0.80 ± 0.02**	0.90 ± 0.01	0.94 ± 0.01	**0.121 ± 0.005**
	DPSNet	**0.08 ± 0.01**	**0.07 ± 0.01**	**0.40 ± 0.03**	**0.15 ± 0.01**	**0.90 ± 0.01**	**0.96 ± 0.01**	**0.98 ± 0.01**	0.630 ± 0.001
	Ours	**0.18 ± 0.03**	**0.36 ± 0.18**	**0.59 ± 0.06**	**0.22 ± 0.01**	0.79 ± 0.02	**0.92 ± 0.01**	**0.96 ± 0.01**	**0.065 ± 0.001**
SUN3D	COLMAP	0.62	3.24	2.32	0.66	0.33	0.55	0.72	-
	DeepMVS	0.28	0.44	0.94	0.36	0.56	0.74	0.90	80.9
	MVDepthNet	**0.18 ± 0.01**	**0.19 ± 0.04**	**0.55 ± 0.03**	0.24 ± 0.01	0.74 ± 0.02	0.91 ± 0.01	0.96 ± 0.01	**0.121 ± 0.005**
	DPSNet	**0.16 ± 0.01**	**0.13 ± 0.01**	**0.45 ± 0.02**	**0.20 ± 0.01**	**0.79 ± 0.02**	**0.93 ± 0.01**	**0.98 ± 0.01**	0.630 ± 0.001
	Ours	0.19 ± 0.02	0.24 ± 0.06	**0.55 ± 0.04**	**0.21 ± 0.01**	**0.76 ± 0.02**	**0.92 ± 0.01**	**0.97 ± 0.01**	**0.065 ± 0.001**
Scenes11	COLMAP	0.62	3.71	3.66	0.87	0.39	0.57	0.67	-
	DeepMVS	0.21	0.37	0.89	0.27	0.69	0.89	0.97	80.9
	MVDepthNet	**0.08 ± 0.01**	**0.13 ± 0.01**	**0.63 ± 0.02**	**0.16 ± 0.01**	**0.93 ± 0.01**	**0.97 ± 0.01**	**0.98 ± 0.01**	**0.121 ± 0.005**
	DPSNet	**0.09 ± 0.01**	**0.20 ± 0.02**	**0.76 ± 0.03**	**0.15 ± 0.01**	**0.93 ± 0.01**	**0.97 ± 0.01**	**0.98 ± 0.01**	0.630 ± 0.001
	Ours	0.13 ± 0.01	0.27 ± 0.02	0.92 ± 0.02	0.22 ± 0.01	0.87 ± 0.01	0.95 ± 0.01	0.97 ± 0.01	**0.065 ± 0.001**
RGBD	COLMAP	0.54	1.76	1.51	0.72	0.27	0.50	0.72	-
	DeepMVS	0.29	0.43	0.87	0.35	0.55	0.81	0.92	80.9
	MVDepthNet	0.21 ± 0.01	0.36 ± 0.04	1.07 ± 0.06	0.34 ± 0.02	0.66 ± 0.02	0.82 ± 0.02	0.89 ± 0.01	**0.121 ± 0.005**
	DPSNet	**0.16 ± 0.01**	**0.23 ± 0.04**	**0.73 ± 0.06**	**0.24 ± 0.02**	**0.79 ± 0.02**	**0.90 ± 0.01**	**0.95 ± 0.01**	0.630 ± 0.001
	Ours	**0.17 ± 0.01**	**0.25 ± 0.04**	**0.80 ± 0.05**	**0.22 ± 0.01**	**0.76 ± 0.02**	**0.92 ± 0.01**	**0.97 ± 0.01**	**0.065 ± 0.001**

TABLE I: *DeMoN Benchmark* – Our network achieves reconstruction accuracy comparable to the state-of-the-art, while being significantly faster. Here we compare MultiViewStereoNet to existing methods COLMAP [27], DeepMVS [7], MVDepthNet [20], and DPSNet [6] on the two-view DeMoN Benchmark. The rows of the table correspond to the different splits of the datasets (MVS, Sun3D, Scenes11, RGBD), while the columns show commonly used depth accuracy metrics such as Abs Rel (the mean absolute relative depth error) and depth completion metrics such as α_1 (the fraction of pixels with less than 25 percent depth error). Each metric is computed per depthmap and then averaged across the test set. Standard errors are shown beside each mean (standard errors for COLMAP and DeepMVS are not available). The top two performing methods according to each metric are bolded. Runtime metrics were computed using VGA image resolution on an Nvidia GTX 1080Ti GPU for all algorithms.

17k training images and 2k testing images (VGA resolution) from trajectories produced inside the Grand Theft Auto V video game. For each training image, we randomly sample a single comparison view from the camera sequence to form training samples. For each test image, we randomly sample N comparison views. We train on this dataset for 150 epochs and compare reconstruction performance against DPSNet [6] as the number of comparison views varies. Table II summarizes the results, which shows our method achieves better depth accuracy and completeness than DPSNet. Furthermore, performance increases as more comparison images are used, although the effect quickly plateaus. Each additional comparison image takes an additional 15 milliseconds to process. Figure 1 and Figure 5 shows qualitative performance on this dataset, where again, we compare favorably with DPSNet.

D. GTA-SfM Viewpoint Compensation Evaluation

We also investigate the effect of viewpoint compensation on reconstruction performance using a simple modification of the GTA-SfM test set. For each test sample, we rotate the comparison image by 180 degrees about the optical axis. We similarly update the camera poses and compare depth estimation performance against the unmodified test set. Figure 6 shows clearly that not performing viewpoint compensation can have severe effects on depth quality. DPSNet, which does no compensation, sees a significant increase in both absolute relative depth error and root-mean-square depth error between the two test sets, despite the same information being presented to the network. Our solution, which does utilize viewpoint compensation, sees no drop in performance. Figure 7 shows the qualitative effect of compensation, where the DPSNet depthmap suffers a significant drop in quality when the comparison image is rotated, while that of our solution is unaffected.

GTA-SfM Dataset

Images	Method	Abs Rel ↓	RMSE ↓	α_1 ↑	Runtime [ms] ↓
2	DPSNet	0.103	26.97	0.94	662
2	Ours	0.084	19.69	0.923	65.1
3		0.077	19.47	0.932	79.7
4		0.075	19.41	0.934	92.7
5		0.075	19.40	0.935	106.5

TABLE II: *Multi-View Evaluation* – Our network can fuse information from multiple images to produce depth estimates. Here we show depth estimation performance as the number of comparison images increases (one image is designated as the reference).

V. Conclusion

We present a learning-based method for MVS depth estimation capable of recovering depth from images taken from known, but otherwise arbitrary, viewpoints. Our key insight is that by compensating for the known viewpoint changes during feature extraction, our network can learn features that are projected by construction. This technique lessens the burden on the network to learn invariant features, thereby increasing robustness to viewpoint changes during matching. We employ low-resolution techniques from Khamis et al. [13] and present a novel incremental extraction network to perform this viewpoint compensation efficiently. We show reconstruction performance on benchmark datasets comparable to the state-of-the-art, while being significantly more efficient.

REFERENCES

[1] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski, "A comparison and evaluation of multi-view stereo reconstruction algorithms," in *Proc. CVPR*, 2006.

[2] M. Goesele, B. Curless, and S. M. Seitz, "Multi-view stereo revisited," in *Proc. CVPR*, 2006.

[3] R. T. Collins, "A space-sweep approach to true multi-image matching," in *Proc. CVPR*, 1996.

[4] H. Saito and T. Kanade, "Shape reconstruction in projective grid space from large number of images," in *Proc. CVPR*, 1999.

[5] R. Szeliski and P. Golland, "Stereo matching with transparency and matting," *IJCV*, 1999.

[6] S. Im, H.-G. Jeon, S. Lin, and I. S. Kweon, "DPSNet: End-to-end Deep Plane Sweep Stereo," in *Proc. ICLR*, 2019.

[7] P.-H. Huang, K. Matzen, J. Kopf, N. Ahuja, and J.-B. Huang, "DeepMVS: Learning Multi-View Stereopsis," in *Proc. CVPR*, 2018.

[8] Y. Yao, Z. Luo, S. Li, T. Fang, and L. Quan, "MVSNet: Depth inference for unstructured multi-view stereo," in *Proc. ECCV*, 2018.

[9] K. Luo, T. Guan, L. Ju, H. Huang, and Y. Luo, "P-MVSNet: Learning Patch-wise Matching Confidence Aggregation for Multi-View Stereo," in *Proc. ICCV*, 2019.

[10] Y. Yao, Z. Luo, S. Li, T. Shen, T. Fang, and L. Quan, "Recurrent MVSNet for High-Resolution Multi-View Stereo Depth Inference," in *Proc. CVPR*, 2019.

[11] X. Gu, Z. Fan, S. Zhu, Z. Dai, F. Tan, and P. Tan, "Cascade Cost Volume for High-Resolution Multi-View Stereo and Stereo Matching," in *Proc. CVPR*, 2020.

[12] S. Cheng, Z. Xu, S. Zhu, Z. Li, L. E. Li, R. Ramamoorthi, and H. Su, "Deep Stereo using Adaptive Thin Volume Representation with Uncertainty Awareness," in *Proc. CVPR*, 2020.

[13] S. Khamis, S. Fanello, C. Rhemann, A. Kowdle, J. Valentin, and S. Izadi, "StereoNet: Guided hierarchical refinement for real-time edge-aware depth prediction," in *Proc. ECCV*, 2018.

[14] M. Jaderberg, K. Simonyan, A. Zisserman, *et al.*, "Spatial transformer networks," in *NeurIPS*, 2015.

[15] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *Trans. PAMI*, 2007.

[16] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *IJCV*, 2002.

[17] J. Zbontar and Y. LeCun, "Computing the stereo matching cost with a convolutional neural network," in *Proc. CVPR*, 2015.

[18] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," in *Proc. CVPR*, 2016.

[19] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry, "End-to-end learning of geometry and context for deep stereo regression," in *Proc. ICCV*, 2017.

[20] K. Wang and S. Shen, "MVDepthNet: Real-time multiview depth estimation neural network," in *Proc. 3DV*, 2018.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016.

[22] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[23] Y. Wu and K. He, "Group normalization," in *Proc. ECCV*, 2018.

[24] R. Hartley and A. Zisserman, *Multiple View Geometry*. Cambridge University Press, 2003.

[25] R. Szeliski, *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[26] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox, "DeMoN: Depth and Motion Network for Learning Monocular Stereo," in *Proc. CVPR*, 2017.

[27] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm, "Pixelwise view selection for unstructured multi-view stereo," in *Proc. ECCV*, 2016.

[28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *NeurIPS*, 2019.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014.

[30] J. T. Barron, "A more general robust loss function," *arXiv preprint arXiv:1701.03077*, 2017.

[31] K. Wang and S. Shen, "Flow-motion and depth network for monocular stereo and beyond," in *Proc. ICRA*, 2020.