Unsupervised Learning of Depth and Ego-Motion from Video

Tinghui Zhou* UC Berkeley
Matthew Brown Google
Noah Snavely Google
David G. Lowe Google

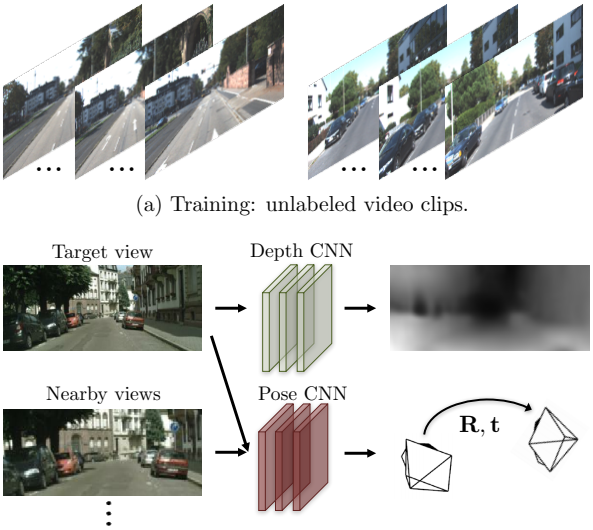
Abstract

We present an unsupervised learning framework for the task of monocular depth and camera motion estimation from unstructured video sequences. In common with recent work [10, 14, 16], we use an end-to-end learning approach with view synthesis as the supervisory signal. In contrast to the previous work, our method is completely unsupervised, requiring only monocular video sequences for training. Our method uses single-view depth and multi-view pose networks, with a loss based on warping nearby views to the target using the computed depth and pose. The networks are thus coupled by the loss during training, but can be applied independently at test time. Empirical evaluation on the KITTI dataset demonstrates the effectiveness of our approach: 1) monocular depth performs comparably with supervised methods that use either ground-truth pose or depth for training, and 2) pose estimation performs favorably compared to established SLAM systems under comparable input settings.

1. Introduction

Humans are remarkably capable of inferring ego-motion and the 3D structure of a scene even over short timescales. For instance, in navigating along a street, we can easily locate obstacles and react quickly to avoid them. Years of research in geometric computer vision has failed to recreate similar modeling capabilities for real-world scenes (e.g., where non-rigidity, occlusion and lack of texture are present). So why do humans excel at this task? One hypothesis is that we develop a rich, structural understanding of the world through our past visual experience that has largely consisted of moving around and observing vast numbers of scenes and developing *consistent* modeling of our observations. From millions of such observations, we have learned about the regularities of the world—roads are flat, buildings are straight, cars are supported by roads *etc*., and we can apply this knowledge when perceiving a new scene, even from a single monocular image.

*The majority of the work was done while interning at Google.



(a) Training: unlabeled video clips.

(b) Testing: single-view depth and multi-view pose estimation.

Figure 1. The training data to our system consists solely of unlabeled image sequences capturing scene appearance from different viewpoints, where the poses of the images are not provided. Our training procedure produces two models that operate independently, one for single-view depth prediction, and one for multi-view camera pose estimation.

In this work, we mimic this approach by training a model that observes sequences of images and aims to explain its observations by predicting likely camera motion and the scene structure (as shown in Fig. 1). We take an end-to-end approach in allowing the model to map directly from input pixels to an estimate of ego-motion (parameterized as 6-DoF transformation matrices) and the underlying scene structure (parameterized as per-pixel depth maps under a reference view). We are particularly inspired by prior work that has suggested view synthesis as a metric [44] and recent work that tackles the calibrated, multi-view 3D case in an end-to-end framework [10]. Our method is unsupervised, and can be trained simply using sequences of images with no manual labeling or even camera motion information.

Our approach builds upon the insight that a geometric view synthesis system only performs *consistently* well when its intermediate predictions of the scene geometry and the camera poses correspond to the physical ground-

truth. While imperfect geometry and/or pose estimation can cheat with reasonable synthesized views for certain types of scenes (e.g., textureless), the same model would fail miserably when presented with another set of scenes with more diverse layout and appearance structures. Thus, our goal is to formulate the entire view synthesis pipeline as the inference procedure of a convolutional neural network, so that by training the network on large-scale video data for the 'meta'-task of view synthesis the network is forced to learn about intermediate tasks of depth and camera pose estimation in order to come up with a consistent explanation of the visual world. Empirical evaluation on the KITTI [15] benchmark demonstrates the effectiveness of our approach on both single-view depth and camera pose estimation. Our code will be made available at https://github.com/tinghuiz/SfMLearner.

2. Related work

Structure from motion The simultaneous estimation of structure and motion is a well studied problem with an established toolchain of techniques [12, 50, 38]. Whilst the traditional toolchain is effective and efficient in many cases, its reliance on accurate image correspondence can cause problems in areas of low texture, complex geometry/photometry, thin structures, and occlusions. To address these issues, several of the pipeline stages have been recently tackled using deep learning, e.g., feature matching [18], pose estimation [26], and stereo [10, 27, 53]. These learning-based techniques are attractive in that they are able to leverage external supervision during training, and potentially overcome the above issues when applied to test data.

Warping-based view synthesis One important application of geometric scene understanding is the task of novel view synthesis, where the goal is to synthesize the appearance of the scene seen from novel camera viewpoints. A classic paradigm for view synthesis is to first either estimate the underlying 3D geometry explicitly or establish pixel correspondence among input views, and then synthesize the novel views by compositing image patches from the input views (e.g., [4, 55, 43, 6, 9]). Recently, end-to-end learning has been applied to reconstruct novel views by transforming the input based on depth or flow, e.g., DeepStereo [10], Deep3D [51] and Appearance Flows [54]. In these methods, the underlying geometry is represented by quantized depth planes (DeepStereo), probabilistic disparity maps (Deep3D) and view-dependent flow fields (Appearance Flows), respectively. Unlike methods that directly map from input views to the target view (e.g., [45]), warping-based methods are forced to learn intermediate predictions of geometry and/or correspondence. In this work, we aim to distill such geometric reasoning capability from CNNs trained to perform warping-based view synthesis.

Learning single-view 3D from registered 2D views Our work is closely related to a line of recent research on learning single-view 3D inference from registered 2D observations. Garg *et al.* [14] propose to learn a single-view depth estimation CNN using projection errors to a calibrated stereo twin for supervision. Concurrently, Deep3D [51] predicts a second stereo viewpoint from an input image using stereoscopic film footage as training data. A similar approach was taken by Godard *et al.* [16], with the addition of a left-right consistency constraint, and a better architecture design that led to impressive performance. Like our approach, these techniques only learn from image observations of the world, unlike methods that require explicit depth for training, e.g., [20, 42, 7, 27, 30].

These techniques bear some resemblance to direct methods for structure and motion estimation [22], where the camera parameters and scene depth are adjusted to minimize a pixel-based error function. However, rather than directly minimizing the error to obtain the estimation, the CNN-based methods only take a gradient step for each batch of input instances, which allows the network to learn an implicit prior from a large corpus of related imagery. Several authors have explored building differentiable rendering operations into their models that are trained in this way, e.g., [19, 29, 34].

While most of the above techniques (including ours) are mainly focused on inferring depth maps as the scene geometry output, recent work (e.g., [13, 41, 46, 52]) has also shown success in learning 3D volumetric representations from 2D observations based on similar principles of projective geometry. Fouhey *et al.* [11] further show that it is even possible to learn 3D inference without 3D labels (or registered 2D views) by utilizing scene regularity.

Unsupervised/Self-supervised learning from video Another line of related work to ours is visual representation learning from video, where the general goal is to design pretext tasks for learning generic visual features from video data that can later be re-purposed for other vision tasks such as object detection and semantic segmentation. Such pretext tasks include ego-motion estimation [2, 24], tracking [49], temporal coherence [17], temporal order verification [36], and object motion mask prediction [39]. While we focus on inferring the explicit scene geometry and ego-motion in this work, intuitively, the internal representation learned by the deep network (especially the single-view depth CNN) should capture some level of semantics that could generalize to other tasks as well.

Concurrent to our work, Vijayanarasimhan *et al.* [48] independently propose a framework for joint training of depth, camera motion and scene motion from videos. While both methods are conceptually similar, ours is focused on the unsupervised aspect, whereas their framework adds the capability to incorporate supervision (e.g., depth, camera motion or scene motion). There are significant differences in how scene dynamics are modeled during training, in which they explicitly solve for object motion whereas our explainability mask discounts regions undergoing motion, occlusion and other factors.

3. Approach

Here we propose a framework for jointly training a single-view depth CNN and a camera pose estimation CNN from unlabeled video sequences. Despite being jointly trained, the depth model and the pose estimation model can be used independently during test-time inference. Training examples to our model consist of short image sequences of scenes captured by a moving camera. While our training procedure is robust to some degree of scene

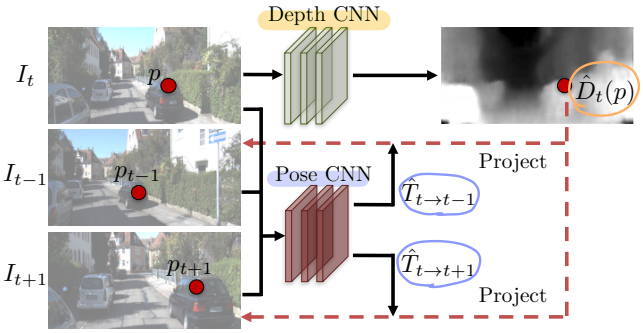


Figure 2. Overview of the supervision pipeline based on view synthesis. The depth network takes only the target view as input, and outputs a per-pixel depth map $\hat{D}_t$. The pose network takes both the target view (I_t) and the nearby/source views (e.g., I_{t-1} and I_{t+1}) as input, and outputs the relative camera poses ($\hat{T}_{t\to t-1}$, $\hat{T}_{t\to t+1}$). The outputs of both networks are then used to inverse warp the source views (see Sec. 3.2) to reconstruct the target view, and the photometric reconstruction loss is used for training the CNNs. By utilizing view synthesis as supervision, we are able to train the entire framework in an unsupervised manner from videos.

motion, we assume that the scenes we are interested in are mostly rigid, i.e., the scene appearance change across different frames is dominated by the camera motion.

3.1. View synthesis as supervision

The key supervision signal for our depth and pose prediction CNNs comes from the task of *novel view synthesis*: given one input view of a scene, synthesize a new image of the scene seen from a different camera pose. We can synthesize a target view given a per-pixel depth in that image, plus the pose and visibility in a nearby view. As we will show next, this synthesis process can be implemented in a fully differentiable manner with CNNs as the geometry and pose estimation modules. Visibility can be handled, along with non-rigidity and other non-modeled factors, using an "explanability" mask, which we discuss later (Sec. 3.3).

Let us denote $< I_1, \ldots, I_N >$ as a training image sequence with one of the frames I_t being the target view and the rest being the source views $I_s (1 \le s \le N, s \ne t)$. The view synthesis objective can be formulated as

$$\mathcal{L}_{vs} = \sum_s \sum_p |I_t(p) - \hat{I}_s(p)| \,, \quad (1)$$

where p indexes over pixel coordinates, and $\hat{I}_s$ is the source view I_s warped to the target coordinate frame based on a depth image-based rendering module [8] (described in Sec. 3.2), taking the predicted depth $\hat{D}_t$, the predicted 4×4 camera transformation matrix[1] $\hat{T}_{t\to s}$ and the source view I_s as input.

Note that the idea of view synthesis as supervision has also been recently explored for learning single-view depth estimation [14, 16] and multi-view stereo [10]. However, to the best of our knowledge, all previous work requires posed image sets during training (and testing too in the case of DeepStereo), while our framework can be applied to standard videos without pose information. Furthermore, it predicts the poses as part of the learning framework. See Figure 2 for an illustration of our learning pipeline for depth and pose estimation.

[1] In practice, the CNN estimates the Euler angles and the 3D translation vector, which are then converted to the transformation matrix.

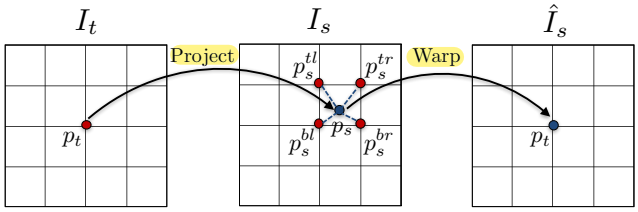


Figure 3. Illustration of the differentiable image warping process. For each point p_t in the target view, we first project it onto the source view based on the predicted depth and camera pose, and then use bilinear interpolation to obtain the value of the warped image $\hat{I}_s$ at location p_t.

3.2. Differentiable depth image-based rendering

As indicated in Eq. 1, a key component of our learning framework is a differentiable depth image-based renderer that reconstructs the target view I_t by sampling pixels from a source view I_s based on the predicted depth map $\hat{D}_t$ and the relative pose $\hat{T}_{t\to s}$.

Let p_t denote the homogeneous coordinates of a pixel in the target view, and K denote the camera intrinsics matrix. We can obtain p_t's projected coordinates onto the source view p_s by[2]

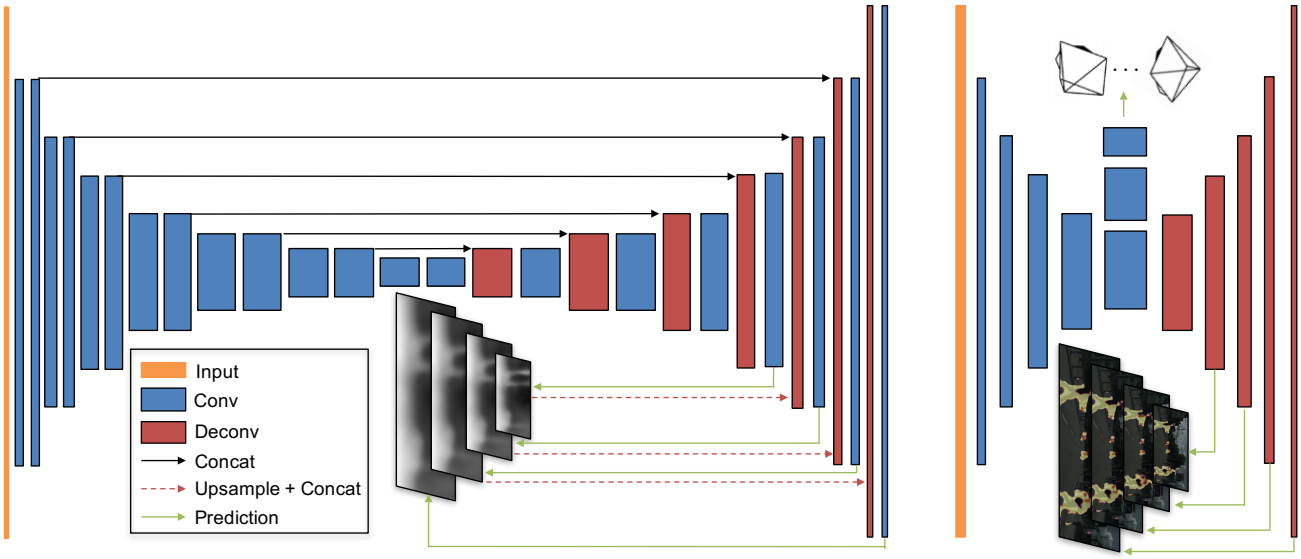
$$p_s \sim K \hat{T}_{t\to s} \hat{D}_t(p_t) K^{-1} p_t \quad (2)$$

Notice that the projected coordinates p_s are continuous values. To obtain $I_s(p_s)$ for populating the value of $\hat{I}_s(p_t)$ (see Figure 3), we then use the differentiable bilinear sampling mechanism proposed in the *spatial transformer networks* [23] that linearly interpolates the values of the 4-pixel neighbors (top-left, top-right, bottom-left, and bottom-right) of p_s to approximate $I_s(p_s)$, i.e. $\hat{I}_s(p_t) = I_s(p_s) = \sum_{i\in\{t,b\},j\in\{l,r\}} w^{ij} I_s(p_s^{ij})$, where w^{ij} is linearly proportional to the spatial proximity between p_s and p_s^{ij}, and $\sum_{i,j} w^{ij} = 1$. A similar strategy is used in [54] for learning to directly warp between different views, while here the coordinates for pixel warping are obtained through projective geometry that enables the factorization of depth and camera pose.

3.3. Modeling the model limitation

Note that when applied to monocular videos the above view synthesis formulation implicitly assumes 1) the scene is static without moving objects; 2) there is no occlusion/disocclusion between the target view and the source views; 3) the surface is Lambertian so that the photo-consistency error is meaningful. If any of these assumptions are violated in a training sequence, the gradients could be corrupted and potentially inhibit training. To improve the robustness of our learning pipeline to these factors, we additionally train a *explainability prediction* network (jointly and simultaneously with the depth and pose networks) that outputs a per-pixel soft mask $\hat{E}_s$ for each target-source pair, indicating the

[2] For notation simplicity, we omit showing the necessary conversion to homogeneous coordinates along the steps of matrix multiplication.



(a) Single-view depth network (b) Pose/explainability network

Figure 4. Network architecture for our depth/pose/explainability prediction modules. The width and height of each rectangular block indicates the output channels and the spatial dimension of the feature map at the corresponding layer respectively, and each reduction/increase in size indicates a change by the factor of 2. (a) For single-view depth, we adopt the DispNet [35] architecture with multi-scale side predictions. The kernel size is 3 for all the layers except for the first 4 conv layers with $7, 7, 5, 5$, respectively. The number of output channels for the first conv layer is 32. (b) The pose and explainabilty networks share the first few conv layers, and then branch out to predict 6-DoF relative pose and multi-scale explainability masks, respectively. The number of output channels for the first conv layer is 16, and the kernel size is 3 for all the layers except for the first two conv and the last two deconv/prediction layers where we use $7, 5, 5, 7$, respectively. See Section 3.5 for more details.

network's belief in where direct view synthesis will be successfully modeled for each target pixel. Based on the predicted $\hat{E}_s$, the view synthesis objective is weighted correspondingly by

$$\mathcal{L}_{vs} = \sum_{<I_1,\dots,I_N>\in\mathcal{S}} \sum_{p} \hat{E}_s(p) |I_t(p) - \hat{I}_s(p)| \,. \quad (3)$$

Since we do not have direct supervision for $\hat{E}_s$, training with the above loss would result in a trivial solution of the network always predicting $\hat{E}_s$ to be zero, which perfectly minimizes the loss. To resolve this, we add a regularization term $\mathcal{L}_{reg}(\hat{E}_s)$ that encourages nonzero predictions by minimizing the cross-entropy loss with constant label 1 at each pixel location. In other words, the network is encouraged to minimize the view synthesis objective, but allowed a certain amount of slack for discounting the factors not considered by the model.

3.4. Overcoming the gradient locality

One remaining issue with the above learning pipeline is that the gradients are mainly derived from the pixel intensity difference between $I(p_t)$ and the four neighbors of $I(p_s)$, which would inhibit training if the correct p_s (projected using the ground-truth depth and pose) is located in a low-texture region or far from the current estimation. This is a well known issue in motion estimation [3]. Empirically, we found two strategies to be effective for overcoming this issue: 1) using a convolutional encoder-decoder architecture with a small bottleneck for the depth network that implicitly constrains the output to be globally smooth and facilitates gradients to propagate from meaningful regions to nearby regions; 2) explicit multi-scale and smoothness loss (e.g., as in [14, 16]) that allows gradients to be derived from larger spatial regions directly. We adopt the second strategy in this work as it is less sensitive to architectural choices. For smoothness, we minimize the L_1 norm of the second-order gradients for the predicted depth maps (similar to [48]).

Our final objective becomes

$$\mathcal{L}_{final} = \sum_{l} \mathcal{L}^l_{vs} + \lambda_s \mathcal{L}^l_{smooth} + \lambda_e \sum_{s} \mathcal{L}_{reg}(\hat{E}^l_s) \,, \quad (4)$$

where l indexes over different image scales, s indexes over source images, and λ_s and λ_e are the weighting for the depth smoothness loss and the explainability regularization, respectively.

3.5. Network architecture

Single-view depth For single-view depth prediction, we adopt the DispNet architecture proposed in [35] that is mainly based on an encoder-decoder design with skip connections and multi-scale side predictions (see Figure 4). All conv layers are followed by ReLU activation except for the prediction layers, where we use $1/(\alpha * sigmoid(x) + \beta)$ with $\alpha = 10$ and $\beta = 0.01$ to constrain the predicted depth to be always positive within a reasonable range. We also experimented with using multiple views as input to the depth network, but did not find this to improve the results. This is in line with the observations in [47], where optical flow constraints need to be enforced to utilize multiple views effectively.

Pose The input to the pose estimation network is the target view concatenated with all the source views (along the color channels), and the outputs are the relative poses between the target view and each of the source views. The network consists of 7 stride-2 convolutions followed by a 1×1 convolution with $6 * (N-1)$ output channels (corresponding to 3 Euler angles and 3-D translation for each source view). Finally, global average pooling is applied to aggregate predictions at all spatial locations. All conv layers are followed by ReLU except for the last layer where no nonlinear activation is applied.

Explainability mask The explainability prediction network shares the first five feature encoding layers with the pose network, followed by 5 deconvolution layers with multi-scale side predictions. All conv/deconv layers are followed by ReLU except for the prediction layers with no nonlinear activation. The number of output channels for each prediction layer is $2 * (N-1)$, with every two channels normalized by *softmax* to obtain the explainability prediction for the corresponding source-target pair (the second channel after normalization is $\hat{E}_s$ and used in computing the loss in Eq. 3).

4. Experiments

Here we evaluate the performance of our system, and compare with prior approaches on single-view depth as well as ego-motion estimation. We mainly use the KITTI dataset [15] for benchmarking, but also use the Make3D dataset [42] for evaluating cross-dataset generalization ability.

Training Details We implemented the system using the publicly available TensorFlow [1] framework. For all the experiments, we set $\lambda_s = 0.5/l$ (l is the downscaling factor for the corresponding scale) and $\lambda_e = 0.2$. During training, we used batch normalization [21] for all the layers except for the output layers, and the Adam [28] optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$, learning rate of 0.0002 and mini-batch size of 4. The training typically converges after about $150K$ iterations. All the experiments are performed with image sequences captured with a monocular camera. We resize the images to 128×416 during training, but both the depth and pose networks can be run fully-convolutionally for images of arbitrary size at test time.

4.1. Single-view depth estimation

We train our system on the split provided by [7], and exclude all the frames from the testing scenes as well as static sequences with mean optical flow magnitude less than 1 pixel for training. We fix the length of image sequences to be 3 frames, and treat the central frame as the target view and the ± 1 frames as the source views. We use images captured by both color cameras, but treated them independently when forming training sequences. This results in a total of $44,540$ sequences, out of which we use $40,109$ for training and $4,431$ for validation.

To the best of our knowledge, no previous systems exist that learn single-view depth estimation in an unsupervised manner from monocular videos. Nonetheless, here we provide comparison with prior methods with depth supervision [7] and recent methods that use calibrated stereo images (i.e. with pose supervision) for training [14, 16]. Since the depth predicted by our method is defined up to a scale factor, for evaluation we multiply the predicted depth maps by a scalar $\hat{s}$ that matches the median with the ground-truth, i.e. $\hat{s} = median(D_{gt})/median(D_{pred})$.

Similar to [16], we also experimented with first pre-training the system on the larger Cityscapes dataset [5] (sample predictions are shown in Figure 5), and then fine-tune on KITTI, which results in slight performance improvement.

Figure 5. Our sample predictions on the Cityscapes dataset using the model trained on Cityscapes only.

KITTI Here we evaluate the single-view depth performance on the 697 images from the test split of [7]. As shown in Table 1, our unsupervised method performs comparably with several supervised methods (e.g. Eigen *et al.* [7] and Garg *et al.* [14]), but falls short of concurrent work by Godard *et al.* [16] that uses calibrated stereo images (i.e. with pose supervision) with left-right cycle consistency loss for training. For future work, it would be interesting to see if incorporating the similar cycle consistency loss into our framework could further improve the results. Figure 6 provides examples of visual comparison between our results and some supervised baselines over a variety of examples. One can see that although trained in an unsupervised manner, our results are comparable to that of the supervised baselines, and sometimes preserve the depth boundaries and thin structures such as trees and street lights better.

We show sample predictions made by our initial Cityscapes model and the final model (pre-trained on Cityscapes and then fine-tuned on KITTI) in Figure 7. Due to the domain gap between the two datasets, our Cityscapes model sometimes has difficulty in recovering the complete shape of the car/bushes, and mistakes them with distant objects.

We also performed an ablation study of the explainability modeling (see Table 1), which turns out only offering a modest performance boost. This is likely because 1) most of the KITTI scenes are static without significant scene motions, and 2) the occlusion/visibility effects only occur in small regions in sequences

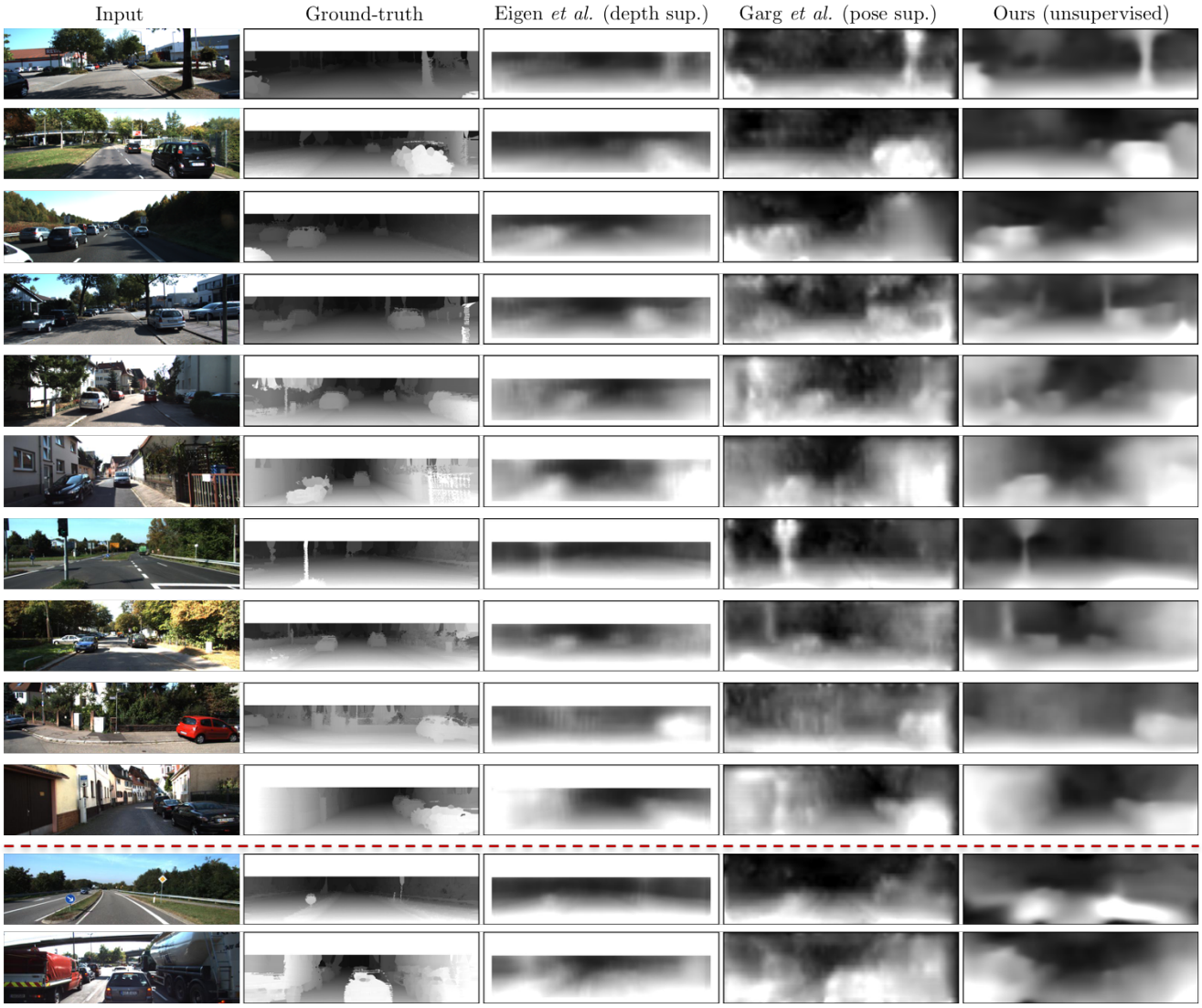


Figure 6. Comparison of single-view depth estimation between Eigen *et al.* [7] (with ground-truth depth supervision), Garg *et al.* [14] (with ground-truth pose supervision), and ours (unsupervised). The ground-truth depth map is interpolated from sparse measurements for visualization purpose. The last two rows show typical failure cases of our model, which sometimes struggles in vast open scenes and objects close to the front of the camera.

across a short time span (3-frames), which make the explainability modeling less essential to the success of training. Nonetheless, our explainability prediction network does seem to capture the factors like scene motion and visibility well (see Sec. 4.3), and could potentially be more important for other more challenging datasets.

Make3D To evaluate the generalization ability of our single-view depth model, we directly apply our model trained on Cityscapes + KITTI to the Make3D dataset unseen during training. While there still remains a significant performance gap between our method and others supervised using Make3D ground-truth depth (see Table 2), our predictions are able to capture the global scene layout reasonably well without any training on the Make3D images (see Figure 8).

4.2. Pose estimation

To evaluate the performance of our pose estimation network, we applied our system to the official KITTI odometry split (containing 11 driving sequences with ground truth odometry obtained through the IMU/GPS readings, which we use for evaluation purpose only), and used sequences 00-08 for training and 09-10 for testing. In this experiment, we fix the length of input image sequences to our system to 5 frames. We compare our ego-motion estimation with two variants of monocular ORB-SLAM [37] (a well-established SLAM system): 1) `ORB-SLAM (full)`, which recovers odometry using all frames of the driving sequence (i.e. allowing loop closure and re-localization), and 2) `ORB-SLAM (short)`, which runs on 5-frame snippets (same as our input setting). Another baseline we compare with is the dataset mean

Method	Dataset	Supervision		Error metric				Accuracy metric		
		Depth	Pose	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Train set mean	K	✓		0.403	5.530	8.709	0.403	0.593	0.776	0.878
Eigen *et al*. [7] Coarse	K	✓		0.214	1.605	6.563	0.292	0.673	0.884	0.957
Eigen *et al*. [7] Fine	K	✓		0.203	1.548	6.307	0.282	0.702	0.890	0.958
Liu *et al*. [32]	K	✓		0.202	1.614	6.523	0.275	0.678	0.895	0.965
Godard *et al*. [16]	K		✓	0.148	1.344	5.927	0.247	0.803	0.922	0.964
Godard *et al*. [16]	CS + K		✓	0.124	1.076	5.311	0.219	0.847	0.942	0.973
Ours (w/o explainability)	K			0.221	2.226	7.527	0.294	0.676	0.885	0.954
Ours	K			0.208	1.768	6.856	0.283	0.678	0.885	0.957
Ours	CS			0.267	2.686	7.580	0.334	0.577	0.840	0.937
Ours	CS + K			0.198	1.836	6.565	0.275	0.718	0.901	0.960
Garg *et al*. [14] cap 50m	K		✓	0.169	1.080	5.104	0.273	0.740	0.904	0.962
Ours (w/o explainability) cap 50m	K			0.208	1.551	5.452	0.273	0.695	0.900	0.964
Ours cap 50m	K			0.201	1.391	5.181	0.264	0.696	0.900	0.966
Ours cap 50m	CS			0.260	2.232	6.148	0.321	0.590	0.852	0.945
Ours cap 50m	CS + K			0.190	1.436	4.975	0.258	0.735	0.915	0.968

Table 1. Single-view depth results on the KITTI dataset [15] using the split of Eigen *et al*. [7] (Baseline numbers taken from [16]). For training, K = KITTI, and CS = Cityscapes [5]. All methods we compare with use some form of supervision (either ground-truth depth or calibrated camera pose) during training. Note: results from Garg et al. [14] are capped at 50m depth, so we break these out separately in the lower part of the table.

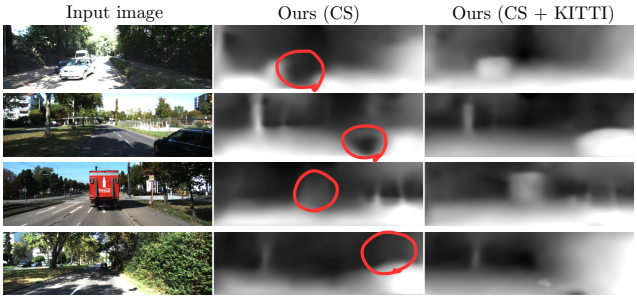


Figure 7. Comparison of single-view depth predictions on the KITTI dataset by our initial Cityscapes model and the final model (pre-trained on Cityscapes and then fine-tuned on KITTI). The Cityscapes model sometimes makes structural mistakes (e.g. holes on car body) likely due to the domain gap between the two datasets.

Method	Supervision		Error metric			
	Depth	Pose	Abs Rel	Sq Rel	RMSE	RMSE log
Train set mean	✓		0.876	13.98	12.27	0.307
Karsch *et al*. [25]	✓		0.428	5.079	8.389	0.149
Liu *et al*. [33]	✓		0.475	6.562	10.05	0.165
Laina *et al*. [31]	✓		0.204	1.840	5.683	0.084
Godard *et al*. [16]		✓	0.544	10.94	11.76	0.193
Ours			0.383	5.321	10.47	0.478

Table 2. Results on the Make3D dataset [42]. Similar to ours, Godard *et al*. [16] do not utilize any of the Make3D data during training, and directly apply the model trained on KITTI+Cityscapes to the test set. Following the evaluation protocol of [16], the errors are only computed where depth is less than 70 meters in a central image crop.

of car motion (using ground-truth odometry) for 5-frame snippets. To resolve scale ambiguity during evaluation, we first optimize

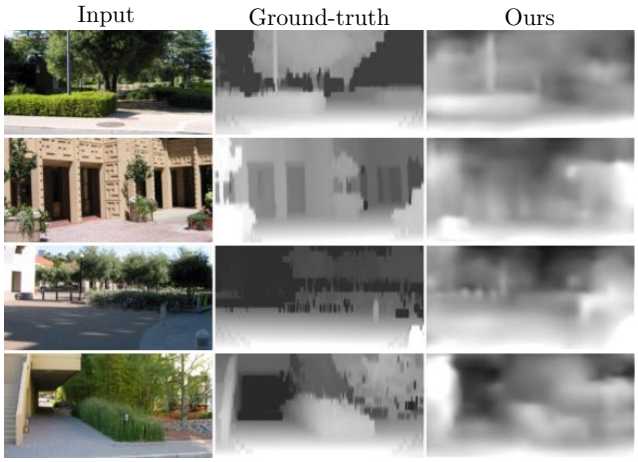


Figure 8. Our sample predictions on the Make3D dataset. Note that our model is trained on KITTI + Cityscapes only, and directly tested on Make3D.

the scaling factor for the predictions made by each method to best align with the ground truth, and then measure the Absolute Trajectory Error (ATE) [37] as the metric. ATE is computed on 5-frame snippets and averaged over the full sequence.[3] As shown in Table 3 and Fig. 9, our method outperforms both baselines (mean odometry and `ORB-SLAM (short)`) that share the same input setting as ours, but falls short of `ORB-SLAM (full)`, which leverages whole sequences (1591 for seq. 09 and 1201 for seq. 10) for loop closure and re-localization.

For better understanding of our pose estimation results, we show in Figure 9 the ATE curve with varying amount of side-

[3]For evaluating `ORB-SLAM (full)` we break down the trajectory of the full sequence into 5-frame snippets with the reference coordinate frame adjusted to the central frame of each snippet.

Method	Seq. 09	Seq. 10
ORB-SLAM (full)	$\mathbf{0.014 \pm 0.008}$	$\mathbf{0.012 \pm 0.011}$
ORB-SLAM (short)	0.064 ± 0.141	0.064 ± 0.130
Mean Odom.	0.032 ± 0.026	0.028 ± 0.023
Ours	$\mathbf{0.021 \pm 0.017}$	$\mathbf{0.020 \pm 0.015}$

Table 3. Absolute Trajectory Error (ATE) on the KITTI odometry split averaged over all 5-frame snippets (lower is better). Our method outperforms baselines with the same input setting, but falls short of `ORB-SLAM (full)` that uses strictly more data.

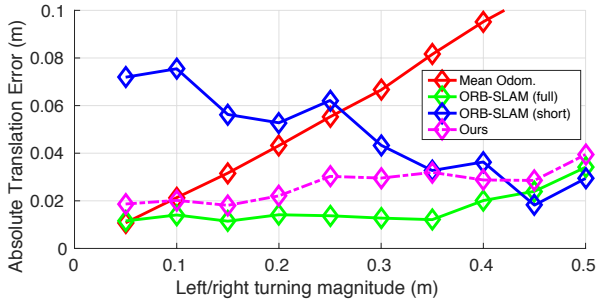


Figure 9. Absolute Trajectory Error (ATE) at different left/right turning magnitude (coordinate difference in the side-direction between the start and ending frame of a testing sequence). Our method performs significantly better than `ORB-SLAM (short)` when side rotation is small, and is comparable with `ORB-SLAM (full)` across the entire spectrum.

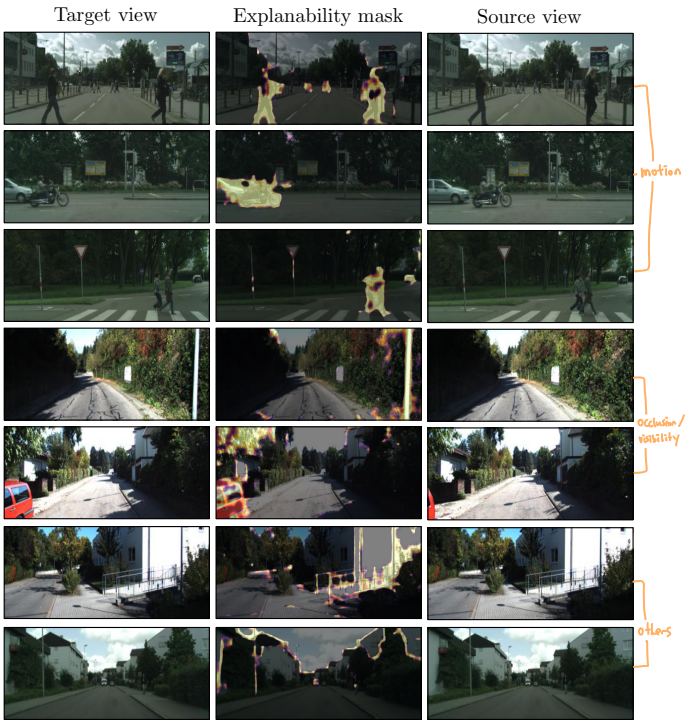


Figure 10. Sample visualizations of the explainability masks. Highlighted pixels are predicted to be unexplainable by the network due to motion (rows 1–3), occlusion/visibility (rows 4–5), or other factors (rows 7–8).

rotation by the car between the beginning and the end of a sequence. Figure 9 suggests that our method is significantly better than `ORB-SLAM (short)` when the side-rotation is small (i.e. car mostly driving forward), and comparable to `ORB-SLAM (full)` across the entire spectrum. The large performance gap between ours and `ORB-SLAM (short)` suggests that our learned ego-motion could potentially be used as an alternative to the local estimation modules in monocular SLAM systems.

4.3. Visualizing the explainability prediction

We visualize example explainability masks predicted by our network in Figure 10. The first three rows suggest that the network has learned to identify dynamic objects in the scene as unexplainable by our model, and similarly, rows 4–5 are examples of objects that disappear from the frame in subsequent views. The last two rows demonstrate the potential downside of explainability-weighted loss: the depth CNN has low confidence in predicting thin structures well, and tends to mask them as unexplainable.

5. Discussion

We have presented an end-to-end learning pipeline that utilizes the task of view synthesis for supervision of single-view depth and camera pose estimation. The system is trained on unlabeled videos, and yet performs comparably with approaches that require ground-truth depth or pose for training. Despite good performance on the benchmark evaluation, our method is by no means close to solving the general problem of unsupervised learning of 3D scene structure inference. A number of major challenges are yet to be addressed: 1) our current framework does not explicitly estimate scene dynamics and occlusions (although they are implicitly taken into account by the explainability masks), both of which are critical factors in 3D scene understanding. Direct modeling of scene dynamics through motion segmentation (e.g. [48, 40]) could be a potential solution; 2) our framework assumes the camera intrinsics are given, which forbids the use of random Internet videos with unknown camera types/calibration – we plan to address this in future work; 3) depth maps are a simplified representation of the underlying 3D scene. It would be interesting to extend our framework to learn full 3D volumetric representations (e.g. [46]).

Another interesting area for future work would be to investigate in more detail the representation learned by our system. In particular, the pose network likely uses some form of image correspondence in estimating the camera motion, whereas the depth estimation network likely recognizes common structural features of scenes and objects. It would be interesting to probe these, and investigate the extent to which our network already performs, or could be re-purposed to perform, tasks such as object detection and semantic segmentation.

Acknowledgments: We thank our colleagues, Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki for their help. We also thank the anonymous reviewers for their valuable comments. TZ would like to thank Shubham Tulsiani for helpful discussions, and Clement Godard for sharing the evaluation code. This work is also partially funded by Intel/NSF VEC award IIS-1539099.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. TensorFlow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2016. 5

[2] P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. In *Int. Conf. Computer Vision*, 2015. 2

[3] J. Bergen, P. Anandan, K. Hanna, and R. Hingorani. Hierarchical model-based motion estimation. In *Computer VisionECCV'92*, pages 237–252. Springer, 1992. 4

[4] S. E. Chen and L. Williams. View interpolation for image synthesis. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 279–288. ACM, 1993. 2

[5] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The Cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3223, 2016. 5, 7

[6] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20. ACM, 1996. 2

[7] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems*, 2014. 2, 5, 6, 7

[8] C. Fehn. Depth-image-based rendering (dibr), compression, and transmission for a new approach on 3d-tv. In *Electronic Imaging 2004*, pages 93–104. International Society for Optics and Photonics, 2004. 3

[9] A. Fitzgibbon, Y. Wexler, and A. Zisserman. Image-based rendering using image-based priors. *Int. Journal of Computer Vision*, 63(2):141–151, 2005. 2

[10] J. Flynn, I. Neulander, J. Philbin, and N. Snavely. DeepStereo: Learning to predict new views from the world's imagery. In *Computer Vision and Pattern Recognition*, 2016. 1, 2, 3

[11] D. F. Fouhey, W. Hussain, A. Gupta, and M. Hebert. Single image 3D without a single 3D image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1053–1061, 2015. 2

[12] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Towards internet-scale multi-view stereo. In *Computer Vision and Pattern Recognition*, pages 1434–1441. IEEE, 2010. 2

[13] M. Gadelha, S. Maji, and R. Wang. 3d shape induction from 2d views of multiple objects. *arXiv preprint arXiv:1612.05872*, 2016. 2

[14] R. Garg, V. K. BG, G. Carneiro, and I. Reid. Unsupervised CNN for single view depth estimation: Geometry to the rescue. In *European Conf. Computer Vision*, 2016. 1, 2, 3, 4, 5, 6, 7

[15] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3354–3361. IEEE, 2012. 2, 5, 7

[16] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Computer Vision and Pattern Recognition*, 2017. 1, 2, 3, 4, 5, 7

[17] R. Goroshin, J. Bruna, J. Tompson, D. Eigen, and Y. LeCun. Unsupervised learning of spatiotemporally coherent metrics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4086–4093, 2015. 2

[18] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg. MatchNet: Unifying feature and metric learning for patch-based matching. In *Computer Vision and Pattern Recognition*, pages 3279–3286, 2015. 2

[19] A. Handa, M. Bloesch, V. Patraucean, S. Stent, J. McCormac, and A. Davison. gvnn: Neural network library for geometric computer vision. *arXiv preprint arXiv:1607.07405*, 2016. 2

[20] D. Hoiem, A. A. Efros, and M. Hebert . Automatic photo pop-up. In *Proc. SIGGRAPH*, 2005. 2

[21] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 5

[22] M. Irani and P. Anandan. About direct methods. In *International Workshop on Vision Algorithms*, pages 267–277. Springer, 1999. 2

[23] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015. 3

[24] D. Jayaraman and K. Grauman. Learning image representations tied to egomotion. In *Int. Conf. Computer Vision*, 2015. 2

[25] K. Karsch, C. Liu, and S. B. Kang. Depth transfer: Depth extraction from video using non-parametric sampling. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2144–2158, 2014. 7

[26] A. Kendall, M. Grimes, and R. Cipolla. PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In *Int. Conf. Computer Vision*, pages 2938–2946, 2015. 2

[27] 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. *arXiv preprint arXiv:1703.04309*, 2017. 2

[28] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[29] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, pages 2539–2547. Curran Associates, Inc., 2015. 2

[30] Y. Kuznietsov, J. Stückler, and B. Leibe. Semi-supervised deep learning for monocular depth map prediction. *arXiv preprint arXiv:1702.02706*, 2017. 2

[31] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 239–248. IEEE, 2016. 7

[32] F. Liu, C. Shen, G. Lin, and I. Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE transactions on pattern analysis and machine intelligence*, 38(10):2024–2039, 2016. 7

[33] M. Liu, M. Salzmann, and X. He. Discrete-continuous depth estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 716–723, 2014. 7

[34] M. M. Loper and M. J. Black. OpenDR: An approximate differentiable renderer. In *European Conf. Computer Vision*, pages 154–169. Springer, 2014. 2

[35] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4048, 2016. 4

[36] I. Misra, C. L. Zitnick, and M. Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *European Conference on Computer Vision*, pages 527–544. Springer, 2016. 2

[37] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5), 2015. 6, 7

[38] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *Int. Conf. Computer Vision*, pages 2320–2327. IEEE, 2011. 2

[39] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan. Learning features by watching objects move. In *CVPR*, 2017. 2

[40] R. Ranftl, V. Vineet, Q. Chen, and V. Koltun. Dense monocular depth estimation in complex dynamic scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4058–4066, 2016. 8

[41] D. J. Rezende, S. A. Eslami, S. Mohamed, P. Battaglia, M. Jaderberg, and N. Heess. Unsupervised learning of 3d structure from images. In *Advances In Neural Information Processing Systems*, pages 4997–5005, 2016. 2

[42] A. Saxena, M. Sun, and A. Y. Ng. Make3D: Learning 3D scene structure from a single still image. *Pattern Analysis and Machine Intelligence*, 31(5):824–840, May 2009. 2, 5, 7

[43] S. M. Seitz and C. R. Dyer. View morphing. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 21–30. ACM, 1996. 2

[44] R. Szeliski. Prediction error as a quality metric for motion and stereo. In *Int. Conf. Computer Vision*, volume 2, pages 781–788. IEEE, 1999. 1

[45] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Multi-view 3d models from single images with a convolutional network. In *European Conference on Computer Vision*, pages 322–337. Springer, 2016. 2

[46] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Computer Vision and Pattern Recognition*, 2017. 2, 8

[47] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox. DeMoN: Depth and motion network for learning monocular stereo. *arXiv preprint arXiv:1612.02401*, 2016. 4

[48] S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadaki. SfM-Net: Learning of structure and motion from video. *arXiv preprint*, 2017. 2, 4, 8

[49] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2015. 2

[50] C. Wu. VisualSFM: A visual structure from motion system. http://ccwu.me/vsfm, 2011. 2

[51] J. Xie, R. B. Girshick, and A. Farhadi. Deep3D: Fully automatic 2D-to-3D video conversion with deep convolutional neural networks. In *European Conf. Computer Vision*, 2016. 2

[52] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *Advances in Neural Information Processing Systems*, pages 1696–1704, 2016. 2

[53] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *Journal of Machine Learning Research*, 17(1-32):2, 2016. 2

[54] T. Zhou, S. Tulsiani, W. Sun, J. Malik, and A. A. Efros. View synthesis by appearance flow. In *European Conference on Computer Vision*, pages 286–301. Springer, 2016. 2, 3

[55] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High-quality video view interpolation using a layered representation. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 600–608. ACM, 2004. 2