Unsupervised Learning of Depth and Ego-Motion from Monocular Video Using 3D Geometric Constraints

Reza Mahjourian  
University of Texas at Austin, Google Brain

Martin Wicke  
Google Brain

Anelia Angelova  
Google Brain

Abstract

We present a novel approach for unsupervised learning of depth and ego-motion from monocular video. Unsupervised learning removes the need for separate supervisory signals (depth or ego-motion ground truth, or multi-view video). Prior work in unsupervised depth learning uses pixel-wise or gradient-based losses, which only consider pixels in small local neighborhoods. Our main contribution is to explicitly consider the inferred 3D geometry of the whole scene, and enforce consistency of the estimated 3D point clouds and ego-motion across consecutive frames. This is a challenging task and is solved by a novel (approximate) backpropagation algorithm for aligning 3D structures.

We combine this novel 3D-based loss with 2D losses based on photometric quality of frame reconstructions using estimated depth and ego-motion from adjacent frames. We also incorporate validity masks to avoid penalizing areas in which no useful information exists.

We test our algorithm on the KITTI dataset and on a video dataset captured on an uncalibrated mobile phone camera. Our proposed approach consistently improves depth estimates on both datasets, and outperforms the state-of-the-art for both depth and ego-motion. Because we only require a simple video, learning depth and ego-motion on large and varied datasets becomes possible. We demonstrate this by training on the low quality uncalibrated video dataset and evaluating on KITTI, ranking among top performing prior methods which are trained on KITTI itself. ¹

1. Introduction

Inferring the depth of a scene and one’s ego-motion is one of the key challenges in fields such as robotics and autonomous driving. Being able to estimate the exact position of objects in 3D and the scene geometry is essential for motion planning and decision making.

¹Code and data available at http://sites.google.com/view/vid2depth
This paper proposes a method for unsupervised learning of depth and ego-motion from monocular (single-camera) videos. The only form of supervision that we use comes from assumptions about consistency and temporal coherence between consecutive frames in a monocular video (camera intrinsics are also used).

Cameras are by far the best understood and most ubiquitous sensor available to us. High quality cameras are inexpensive and easy to deploy. The ability to train on arbitrary monocular video opens up virtually infinite amounts of training data, without sensing artifacts or inter-sensor calibration issues.

In order to learn depth in a completely unsupervised fashion, we rely on existence of ego-motion in the video. Given two consecutive frames from the video, a neural network produces single-view depth estimates from each frame, and an ego-motion estimate from the frame pair. Requiring that the depth and ego-motion estimates from adjacent frames are consistent serves as supervision for training the model. This method allows learning depth because the transformation from depth and ego-motion to a new frame is well understood and a good approximation can be written down as a fixed differentiable function.

Our main contributions are the following:

**Imposing 3D constraints.** We propose a loss which directly penalizes inconsistencies in the estimated depth without relying on backpropagation through the image reconstruction process. We compare depth extracted from adjacent frames by directly comparing 3D point clouds in a common reference frame. Intuitively, assuming there is no significant object motion in the scene, one can transform the estimated point cloud for each frame into the predicted point cloud for the other frame by applying ego-motion or its inverse (Fig. 1 and Fig. 2).

To the best of our knowledge, our approach is the first depth-from-video algorithm to use 3D information in a differentiable loss function. Our experiments show that adding losses computed directly on the 3D geometry improves results significantly.

**Principled masking.** When transforming a frame and projecting it, some parts of the scene are not covered in the new view (either due to parallax effects or because objects left or entered the field of view). Depth and image pixels in those areas are not useful for learning; using their values in the loss degrades results. Previous methods have approached this problem by adding a general-purpose learned mask to the model [32], or applying post-processing to remove edge artifacts [11]. However, learning the mask is not very effective. Computing the masks analytically leaves a simpler learning problem for the model.

**Learning from an uncalibrated video stream.** We demonstrate that our proposed approach can consume and learn from any monocular video source with camera mo-

![Figure 2. The 3D loss: ICP is applied symmetrically in forward and backward directions to bring the depth and ego-motion estimates from two consecutive frames into agreement. The products of ICP generate gradients which are used to improve the depth and ego-motion estimates.](image)

2. Related Work

Classical solutions to depth and ego-motion estimation involve stereo and feature matching techniques [25], whereas recent methods have shown success using deep learning [7].

Most pioneering works that learn depth from images rely on supervision from depth sensors [6, 15, 18]. Several subsequent approaches [16, 17, 3] also treat depth estimation as a dense prediction problem and use popular fully-convolutional architectures such as FCN [19] or U-Net [22].

Garg *et al.* [8] propose to use a calibrated stereo camera pair setup in which depth is produced as an intermediate output and the supervision comes from reconstruction of one image in a stereo pair from the input of the other. Since the images on the stereo rig have a fixed and known transformation, the depth can be learned from that functional relationship (plus some regularization). Other novel learning approaches, that also need more than one image for depth estimation are [29, 21, 15, 14, 30].

Godard *et al.* [11] offer an approach to learn single-view depth estimation using rectified stereo input during training. The disparity matching problem in a rectified stereo pair re-
quires only a one-dimensional search. The work by Um-
menhofer et al. [26] called DeMoN also addresses learning of
depth from stereo data. Their method produces high-
quality depth estimates from two unconstrained frames as
input. This work uses various forms of supervision includ-
ing depth and optical flow.

Zhou et al. [32] propose a novel approach for unsuper-
vised learning of depth and ego-motion using only monoc-
ular video. This setup is most aligned with our work as we
similarly learn depth and ego-motion from monocular video
in an unsupervised setting. Vijayanarasimhan et al. [27] use
a similar approach which additionally tries to learn the mo-
tion of a handful of objects in the scene. Their work also
allows for optional supervision by ground-truth depth or op-
tical flow to improve performance.

Our work differs in taking the training process to three
dimensions. We present differentiable 3D loss functions
which can establish consistency between the geometry of
adjacent frames, and thereby improve depth and ego-motion
estimation.

3. Method

Our method learns depth and ego-motion from mono-
cular video without supervision. Fig. 1 illustrates its dif-
ferent components. At the core of our approach there is
a novel loss function which is based on aligning the 3D
genometry (point clouds) generated from adjacent frames
(Sec. 3.4). Unlike 2D losses that enforce local photomet-
ric consistency, the 3D loss considers the entire scene and
its geometry. We show how to efficiently backpropagate
through this loss.

This section starts with discussing the geometry of the
problem and how it is used to obtain differentiable losses. It
then describes each individual loss term.

3.1. Problem Geometry

At training time, the goal is to learn depth and ego-
motion from a single monocular video stream. This prob-
lem can be formalized as follows: Given a pair of consec-
tutive frames $X_{t-1}$ and $X_t$, estimate depth $D_{t-1}$ at time
t $t - 1$, depth $D_t$ at time $t$, and the ego-motion $T_t$ represen-
ting the camera’s movement (position and orientation) from
time $t - 1$ to $t$.

Once a depth estimate $D_t$ is available, it can be projected
into a point cloud $Q_t$. More specifically, the image pixel at
coordinates $(i,j)$ with estimated depth $D_t^{ij}$ can be projected
into a structured 3D point cloud

$$Q_t^{ij} = D_t^{ij} \cdot K^{-1}[i,j,1]^T,$$

where $K$ is the camera intrinsic matrix, and the coordinates
are homogeneous.

Given an estimate for $T_t$, the camera’s movement from
t $t - 1$ to $t$, $Q_t$ can be transformed to get an estimate for the
previous frame’s point cloud: $\hat{Q}_{t-1} = T_tQ_t$. Note that the
transformation applied to the point cloud is the inverse of
the camera movement from $t$ to $t - 1$. $Q_t$ can then be pro-
jected onto the camera at frame $t - 1$ as $K\hat{Q}_{t-1}$. Combining
this transformation and projection with Eq. 1 establishes a
mapping from image coordinates at time $t$ to image coordi-
nates at time $t - 1$. This mapping allows us to reconstruct
frame $\hat{X}_t$ by warping $X_{t-1}$ based on $D_t, T_t$:

$$\hat{X}_t^{ij} = X_{t-1}^{ij}, [\hat{i}, \hat{j}, 1]^T = K T_t (D_t^{ij} \cdot K^{-1}[i,j,1]^T)$$  (2)

Following the approach in [32, 13], we compute $\hat{X}_t^{ij}$ by
performing a soft sampling from the four pixels in $X_{t-1}$
whose coordinates overlap with $(i,j)$.

This process is repeated in the other direction to project
$D_{t-1}$ into a point cloud $Q_{t-1}$, and reconstruct frame $\hat{X}_{t-1}$
by warping $X_t$ based on $D_{t-1}$ and $T_{t-1}$.

3.2. Principled Masks

Computing $\hat{X}_t$ involves creating a mapping from image
coordinates in $X_t$ to $X_{t-1}$. However, due to the camera’s
motion, some pixel coordinates in $X_t$ may be mapped to
coordinates that are outside the image boundaries in $X_{t-1}$.
With forward ego-motion, this problem is usually more pro-
nounced when computing $\hat{X}_{t-1}$ from $X_t$. Our experiments
show that including such pixels in the loss degrades perfor-
ance. Previous approaches have either ignored this prob-
lem, or tried to tackle it by adding a general-purpose mask
to the network [8, 32, 27], which is expected to exclude
regions that are unexplainable due to any reason. However,
this approach does not seem to be effective and often results
in edge artifacts in depth images (see Sec. 4).

As Fig. 3 demonstrates, validity masks can be com-
puted analytically from depth and ego-motion estimates.
For every pair of frames $X_{t-1}, X_t$, one can create a pair of
masks $M_{t-1}, M_t$, which indicate the pixel coordinates
where $\hat{X}_{t-1}$ and $\hat{X}_t$ are valid.

3.3. Image Reconstruction Loss

Comparing the reconstructed images $\hat{X}_t$, $\hat{X}_{t-1}$ to the
input frames $X_t$, $X_{t-1}$ respectively produces a differentiable
image reconstruction loss that is based on photometric con-
sistency [32, 8], and needs to be minimized:

$$L_{\text{rec}} = \sum_{ij} \| (X_t^{ij} - \hat{X}_t^{ij})M_t^{ij} \|$$  (3)

The main problem with this type of loss is that the pro-
cess used to create $\hat{X}_t$ is an approximation—and, because
derifferentiability is required, a relatively crude one. This
process is not able to account for effects such as light-
ning, shadows, translucence, or reflections. As a result, this

\footnote{Note: All losses mentioned in this section are repeated for times $t$ and
t $- 1$. For brevity, we have left out the terms involving $t - 1$.}
loss is noisy and results in artifacts. Strong regularization is required to reduce the artifacts, which in turn leads to smoothed out predictions (see Sec. 4). Learning to predict the adjacent frame directly would avoid this problem, but such techniques cannot generate depth and ego-motion predictions.

3.4. A 3D Point Cloud Alignment Loss

Instead of using \( \hat{Q}_{t-1} \) or \( \tilde{Q}_t \) just to establish a mapping between coordinates of adjacent frames, we construct a loss function that directly compares point clouds \( \hat{Q}_{t-1} \) to \( Q_{t-1} \), or \( \tilde{Q}_t \) to \( Q_t \). This 3D loss uses a well-known rigid registration method, Iterative Closest Point (ICP) [4, 2, 23], which computes a transformation that minimizes point-to-point distances between corresponding points in the two point clouds.

ICP alternates between computing correspondences between two 3D point clouds (using a simple closest point heuristic), and computing a best-fit transformation between the two point clouds, given the correspondence. The next iteration then recomputes the correspondence with the previous iteration’s transformation applied. Our loss function uses both the computed transformation and the final residual registration error after ICP’s minimization.

Because of the combinatorial nature of the correspondence computation, ICP is not differentiable. As shown below, we can approximate its gradients using the products it computes as part of the algorithm, allowing us to backpropagate errors for both the ego-motion and depth estimates.

ICP takes as input two point clouds \( A \) and \( B \) (e.g. \( \hat{Q}_{t-1} \) and \( Q_{t-1} \)). Its main output is a best-fit transformation \( T' \) which minimizes the distance between the transformed points in \( A \) and their corresponding points in \( B \):

\[
\arg\min_{T'} \frac{1}{2} \sum_{ij} \|T' \cdot A^{ij} - B^{c(ij)}\|^2
\]

where \( c(\cdot) \) denotes the point to point correspondence found by ICP. The secondary output of ICP is the residual \( r^{ij} = A^{ij} - T'^{-1} \cdot B^{c(ij)} \), which reflects the residual distances between corresponding points after ICP’s distance minimizing transform has been applied.

Fig. 4 demonstrates how ICP is used in our method to penalize errors in the estimated ego-motion \( T_t \) and depth \( D_t \). If the estimates \( T_t \) and \( D_t \) from the neural network are perfect, \( \hat{Q}_{t-1} \) would align perfectly with \( Q_{t-1} \). When this is not the case, aligning \( \hat{Q}_{t-1} \) to \( Q_{t-1} \) with ICP produces a transform \( T'_t \) and residuals \( r_t \) which can be used to adjust \( T_t \) and \( D_t \) toward a better initial alignment. More specifically, we use \( T'_t \) as an approximation to the negative gradient of the loss with respect to the ego-motion \( T_t \). To correct the depth map \( D_t \), we note that even after the correction \( T'_t \) has been applied, moving the points in the direction \( r_t \) would decrease the loss. Of the factors that generate the points in \( Q_t \) and thereby \( \hat{Q}_{t-1} \), we can only change \( D_t \). We therefore use \( r_t \) as an approximation to the negative gradient of the loss with respect to the depth \( D_t \). Note that this approximation of the gradient ignores the impact of depth errors on

\(3\)While we describe a point-to-point distance, we use the more powerful point-to-plane distance [4] as in the Point Cloud Library [24]. The definition of the residual changes to include the gradient of the distance metric used, but it is still the gradient of the error.

\(4\)Technically, \( T' \) is not the negative gradient: it points in the direction of the minimum found by ICP, and not in the direction of steepest descent. Arguably, this makes it better than a gradient.
ego-motion and vice versa. However, ignoring these second order effects works well in practice. The complete 3D loss is then
\[ L_{\text{3D}} = \| T'_s - I \|_1 + \| r_s \|_1, \] (5)
where \( \| \cdot \|_1 \) denotes the L1-norm, \( I \) is the identity matrix.

3.5. Additional Image-Based Losses

Structured similarity (SSIM) is a commonly-used metric for evaluating the quality of image predictions. Similar to [11, 31], we use it as a loss term in the training process. It measures the similarity between two images patches \( x \) and \( y \) and is defined as
\[ \text{SSIM}(x, y) = \frac{(2\mu_x\mu_y+c_1)(2\sigma_{xy}+c_2)}{(\mu_x^2+\mu_y^2+c_1)(\sigma_x^2+\sigma_y^2+c_2)}, \] where \( \mu_x, \sigma_x \) are the local means and variances [28]. In our implementation, \( \mu \) and \( \sigma \) are computed by simple (fixed) pooling, and \( c_1 = 0.01^2 \) and \( c_2 = 0.03^2 \). Since SSIM is upper bounded to one and needs to be maximized, we instead minimize
\[ L_{\text{SSIM}} = \sum_{i,j} [1 - \text{SSIM}(\hat{X}_{ij}^t, X_{ij}^t)] M_{ij}^t. \] (6)

A depth smoothness loss is also employed to regularize the depth estimates. Similarly to [12, 11] we use a depth gradient smoothness loss that takes into account the gradients of the image:
\[ L_{\text{sm}} = \sum_{i,j} \| \partial_x D_{ij} \| e^{-\| \partial_y X_{ij} \|} + \| \partial_y D_{ij} \| e^{-\| \partial_x X_{ij} \|} \] (7)

By considering the gradients of the image, this loss function allows for sharp changes in depth at pixel coordinates where there are sharp changes in the image. This is a refinement of the depth smoothness losses used by Zhou et al. [32].

3.6. Learning Setup

All loss functions are applied at four different scales \( s \), ranging from the model’s input resolution, to an image that is \( \frac{1}{8} \) in width and height. The complete loss is defined as:
\[ L = \sum_s \alpha L_{\text{rec}}^s + \beta L_{\text{3D}}^s + \gamma L_{\text{sm}}^s + \omega L_{\text{SSIM}}^s \] (8)
where \( \alpha, \beta, \gamma, \omega \) are hyper-parameters, which we set to \( \alpha = 0.85, \beta = 0.1, \gamma = 0.05, \) and \( \omega = 0.15 \).

We adopt the SiM Learner architecture [32], which is in turn based on DispNet [20]. The neural network consists of two disconnected towers: A depth tower receives a single image with resolution 128 \( \times \) 416 as input and produces a dense depth estimate mapping each pixel of the input to a depth value. An ego-motion tower receives a stack of video frames as input, and produces an ego-motion estimate—represented by six numbers corresponding to relative 3D rotation and translation—between every two adjacent frames. Both towers are fully convolutional.

At training time, a stack of video frames is fed to the model. Following [32], in our experiments we use 3-frame training sequences, where our losses are applied over pairs of adjacent frames. Unlike prior work, our 3D loss requires depth estimates from all frames. However, at test time, the depth tower can produce a depth estimate from an individual video frame, while the ego-motion tower can produce ego-motion estimates from a stack of frames.

We use TensorFlow [1] and the Adam optimizer with \( \beta_1 = 0.9, \beta_2 = 0.999, \) and \( \alpha = 0.0002 \). In all experiments, models are trained for 20 epochs and checkpoints are saved at the end of each epoch. The checkpoint which performs best on the validation set is then evaluated on the test set.

4. Experiments

4.1. Datasets

KITTI. We use the KITTI dataset [9] as the main training and evaluation dataset. This dataset is the most common benchmark used in prior work for evaluating depth and ego-motion accuracy [8, 32, 11, 26]. The KITTI dataset includes a full suite of data sources such as stereo video, 3D point clouds from LIDAR, and the vehicle trajectory. We use only a single (monocular) video stream for training. The point clouds and vehicle poses are used only for evaluation of trained models. We use the same training/validation/test split as [32]: about 40k frames for training, 4k for validation, and 697 test frames from the Eigen [6] split.

Uncalibrated Bike Video Dataset. We created a new dataset by recording some videos using a hand-held phone camera while riding a bicycle. This particular camera offers no stabilization. The videos were recorded at 30fps, with a resolution of 720 \( \times \) 1280. Training sequences were created by selecting frames at 5fps to roughly match the motion in KITTI. We used all 91,866 frames from the videos without excluding any particular segments. We constructed an intrinsic matrix for this dataset based on a Google search for “iPhone 6 video horizontal field of view” (50.9°) and without accounting for lens distortion. This dataset is available on the project website.

4.2. Evaluation of Depth Estimation

Fig. 5 compares sample depth estimates produced by our trained model to other unsupervised learning methods, including the state-of-the-art results by [32]. Table 1 quantitatively compares our depth estimation results against prior work (some of which use supervision). The metrics are computed over the Eigen [6] test set. The table reports separate results for a depth cap of 50m, as this is the only evaluation reported by Garg et al. [8]. When trained only on the KITTI dataset, our model lowers the mean absolute relative depth prediction error from 0.208
Table 1. Depth evaluation metrics over the KITTI Eigen [6] test set. Under the Dataset column, K denotes training on KITTI [10] and CS denotes training on Cityscapes [5]. δ denotes the ratio between estimates and ground truth. All results, except [6], use the crop from [8].
the end of different training epochs on the validation set—and not the test set, which is reported in Table 1. As the plot shows, using the 3D loss improves performance notably across all stages of training. It also shows that the 3D loss has a regularizing effect, which reduces overfitting. In contrast, just pre-training on the larger Cityscapes dataset is not sufficient to reduce overfitting or improve depth quality.

Table 2. Absolute Trajectory Error (ATE) on the KITTI odometry dataset averaged over all multi-frame snippets (lower is better). Our method significantly outperforms the supervised method of ORB-SLAM, which uses the entire video sequence.

| Method                  | Seq. 09   | Seq. 10   |
|-------------------------|-----------|-----------|
| ORB-SLAM (full)         | 0.014 ± 0.008 | 0.012 ± 0.011 |
| ORB-SLAM (short)        | 0.064 ± 0.141 | 0.064 ± 0.130 |
| Mean Odom.              | 0.032 ± 0.026 | 0.028 ± 0.023 |
| Zhou et al. [32]        | 0.021 ± 0.017 | 0.020 ± 0.015 |
| Ours, no ICP (3-frame)  | 0.014 ± 0.010 | 0.013 ± 0.011 |
| Ours, with ICP (3-frame)| 0.013 ± 0.010 | 0.012 ± 0.011 |

Figure 8. Evolution of depth validation error over time when training our model with and without the 3D loss. Training on KITTI and on Cityscapes + KITTI are shown. Using the 3D loss lowers the error and also reduces overfitting.

Figure 9. Composite of two consecutive frames from the Bike dataset. Since the phone is hand-held, the motion is less stable compared to existing driving datasets. Best viewed in color.

4.4. Evaluation of Ego-Motion

During the training process, depth and ego-motion are learned jointly and their accuracy is inter-dependent. Table 2 reports the ego-motion accuracy of our models over two sample sequences from the KITTI odometry dataset. Our proposed method significantly outperforms the unsupervised method by [32]. Moreover, it matches or outperforms the supervised method of ORB-SLAM, which uses the entire video sequence.

4.5. Learning from Bike Videos

To demonstrate that our proposed method can use any video with ego-motion as training data, we recorded a number of videos using a hand-held phone camera while riding a bicycle. Fig. 9 shows sample frames from this dataset.

We trained our depth and ego-motion model only on the Bike videos. We then evaluated the trained model on KITTI. Note that no fine-tuning is performed. Fig. 10 shows sample depth estimates for KITTI frames produced by the model trained on Bike videos. The Bike dataset is quite different from the KITTI dataset (∼ 51° vs. ∼ 81° FOV, no distortion correction vs. fully rectified images, US vs. European architecture/street layout, hand-held camera vs. stable motion). Yet, as Table 3 and Fig. 10 show, the model trained on Bike videos is close in quality to the best unsupervised model of [32], which is trained on KITTI itself.

Fig. 11 shows the KITTI validation error for models trained on Bike videos. It verifies that the 3D loss improves learning and reduces overfitting on this dataset as well.

4.6. Ablation Experiments

In order to study the importance of each component in our method, we trained and evaluated a series of models, each missing one component of the loss function. The experiment results in Table 3 and Fig. 12 show that the 3D loss and SSIM components are essential. They also show that removing the masks hurts the performance.

We trained our depth and ego-motion model only on the Bike videos. We then evaluated the trained model on KITTI. Note that no fine-tuning is performed. Fig. 10 shows sample depth estimates for KITTI frames produced by the model trained on Bike videos. The Bike dataset is quite different from the KITTI dataset (∼ 51° vs. ∼ 81° FOV, no distortion correction vs. fully rectified images, US vs. European architecture/street layout, hand-held camera vs. stable motion). Yet, as Table 3 and Fig. 10 show, the model trained on Bike videos is close in quality to the best unsupervised model of [32], which is trained on KITTI itself.

Fig. 11 shows the KITTI validation error for models trained on Bike videos. It verifies that the 3D loss improves learning and reduces overfitting on this dataset as well.
| Method                   | Dataset | Cap | Abs Rel | Sq Rel | RMSE | RMSE log | $\delta < 1.25$ | $\delta < 1.25^2$ | $\delta < 1.25^3$ |
|-------------------------|---------|-----|---------|-------|------|-----------|-----------------|-----------------|-----------------|
| All losses              | CS + K  | 80m | 0.159   | 1.231 | 5.912| 0.243     | 0.764           | 0.923           | 0.970           |
| All losses              | K       | 80m | 0.175   | 1.617 | 6.267| 0.252     | 0.759           | 0.917           | 0.967           |
| No ICP loss             | K       | 80m | 0.183   | 1.410 | 6.813| 0.271     | 0.716           | 0.899           | 0.961           |
| No Principled Masks     | K       | 80m | 0.176   | 1.386 | 6.529| 0.263     | 0.740           | 0.907           | 0.963           |
| Zhou et al. [32]        | K       | 80m | 0.208   | 1.768 | 6.856| 0.283     | 0.678           | 0.885           | 0.957           |
| All losses              | CS + K  | 80m | 0.211   | 1.771 | 7.741| 0.309     | 0.652           | 0.862           | 0.942           |
| No ICP loss             | Bike    | 80m | 0.226   | 2.525 | 7.750| 0.305     | 0.666           | 0.871           | 0.946           |

Table 3. Depth evaluation metrics over the KITTI Eigen [6] test set for various versions of our model. Top: Our best model. Middle: Ablation results where individual loss components are excluded. Bottom: Models trained only on the bike dataset.

Figure 10. Sample depth estimates produced from KITTI frames by the model trained only on the Bike video dataset.

Figure 12. KITTI depth validation error for ablation experiments comparing a model trained with all losses against models missing specific loss components.

5. Conclusions and Future Work

We proposed a novel unsupervised algorithm for learning depth and ego-motion from monocular video. Our main contribution is to explicitly take the 3D structure of the world into consideration. We do so using a novel loss function which can align 3D structures across different frames. The proposed algorithm needs only a single monocular video stream for training, and can produce depth from a single image at test time.

The experiments on the Bike dataset demonstrate that our approach can be applied to learn depth and ego-motion from diverse datasets. Because we require no rectification and our method is robust to lens distortions, lack of stabilization, and other features of low-end cameras, training data can be collected from a large variety of sources, such as public videos on the internet.

If an object moves between two frames, our loss functions try to explain its movement by misestimating its depth. This leads to learning biased depth estimates for that type of object. Similar to prior work [32], our approach does not explicitly handle largely dynamic scenes. Detecting and handling moving objects is our goal for future work.

Lastly, the principled masks can be extended to account for occlusions and disocclusions resulting from change of viewpoint between adjacent frames.

Acknowledgments

We thank Tinghui Zhou and Clément Godard for sharing their code and results, the Google Brain team for discussions and support, and Parvin Taheri and Oscar Ezhdeha for their help with recording videos.
References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems. 2015. Software available from tensorflow.org.

[2] P. J. Besl and N. McKay. A method for registration of 3-d shapes. In IEEE Trans. on Pattern Analysis and Machine Intelligence, 1992.

[3] Y. Cao, Z. Wu, and C. Shen. Estimating depth from monocular images as classification using deep fully convolutional residual networks. CoRR:1605.02305, 2016.

[4] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. In Proc. IEEE Conf. on Robotics and Automation, 1991.

[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.

[6] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. NIPS, 2014.

[7] J. Flynn, I. Neulander, J. Philbin, and N. Snavely. Deep-stereo: Learning to predict new views from the worlds imagery. In CVPR, 2016.

[8] R. Garg, G. Carneiro, and I. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. ECCV, 2016.

[9] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. The International Journal of Robotics Research, 32(11):1231–1237, 2013.

[10] 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.

[11] C. Godard, O. M. Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. CVPR, 2017.

[12] P. Heise, S. Klose, B. Jensen, and A. Knoll. Pm-huber: Patchmatch with huber regularization for stereo matching. In Proceedings of the IEEE International Conference on Computer Vision, pages 2360–2367, 2013.

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

[14] A. Kar, C. Hne, and J. Malik. Learning a multi-view stereo machine. arXiv:1708.05375, 2017.

[15] L. Ladicky, J. Shi, and M. Pollefeys. Pulling things out of perspective. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 89–96, 2014.

[16] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. arXiv:1606.00373, 2016.

[17] B. Li, C. Shen, Y. Dai, A. van den Hengel, and M. He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical CRFs. CVPR, 2015.

[18] F. Liu, C. Shen, G. Lin, and I. Reid. Learning depth from single monocular images using deep convolutional neural fields. PAMI, 2015.

[19] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. CVPR, 2015.

[20] 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.

[21] R. Ranftl, V. Vineet, Q. Chen, and V. Koltun. Dense monocular depth estimation in complex dynamic scenes. CVPR, 2016.

[22] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. International Conference on Medical Image Computing and Computer-Assisted Intervention.

[23] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In 3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on, pages 145–152. IEEE, 2001.

[24] R. B. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). In Robotics and automation (ICRA), 2011 IEEE International Conference on, pages 1–4. IEEE, 2011.

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

[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. CVPR.

[27] S. Vijayanarasimhan, S. Ricco, C. Schmid, R. Sukthankar, and K. Fragkiadakis. Sfm-net: Learning of structure and motion from video. arXiv:1704.07804, 2017.

[28] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. Transactions on Image Processing, 2004.

[29] J. Xie, R. Girshick, and A. Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. ECCV, 2016.

[30] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. JMLR, 2016.

[31] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss functions for neural networks for image processing. IEEE Transactions on Computational Imaging(TCI), 2017.

[32] T. Zhou, M. Brown, N. Snavely, and D. Lowe. Unsupervised learning of depth and ego-motion from video. CVPR, 2017.