

May 16, 2018
DRAFT

Learning Depth from Monocular Videos using Direct Methods

Chaoyang Wang

CMU-RI-TR-18-09

April 2018

School of Computer Science Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Simon Lucey

David Held

Chen-Hsuan Lin

*Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Robotics.*

May 16, 2018
DRAFT

Keywords: Monocular depth estimation, visual odometry, direct method, deep learning, 3D vision

May 16, 2018
DRAFT

For my parents, my friends, and their support for spending my twenties in school.

May 16, 2018
DRAFT

Abstract

The ability to predict depth from a single image - using recent advances in CNNs - is of increasing interest to the vision community. Unsupervised strategies to learning are particularly appealing as they can utilize much larger and varied monocular video datasets during learning without the need for ground truth depth or stereo. In previous works, separate pose and depth CNN predictors had to be determined such that their joint outputs minimized the photometric error. Inspired by recent advances in direct visual odometry (DVO), we argue that the depth CNN predictor can be learned without a pose CNN predictor. Further, we demonstrate empirically that incorporation of a differentiable implementation of DVO, along with a novel depth normalization strategy - substantially improves performance over state of the art that use monocular videos for training.

May 16, 2018
DRAFT

Acknowledgments

My greatest gratitude goes to my advisor – Prof. Lucey. He taught me how to do research, how to write a good paper, and how to give a great presentation. He showed extraordinary tolerance to my sometimes broken oral English, and occasional pessimistic attitude. And he gave the greatest support he could to my study and career.

I also have to thank my fellow lab members & collaborators: Chen-Hsuan Lin, Rui Zhu, Dr. Hamed Kiani, Prof. José M. Buenaposada and Prof. Hondong Li. We worked together, sometimes had lunch together, and of course we coauthored a number of papers. My life in CMU is enjoyable and fruitful because of you.

Finally, I'd like to thank my thesis committee members, especially Prof. Held, who kindly gave me great support and reference, and also offered great opportunity to present my work to his group.

May 16, 2018
DRAFT

Contents

1	Introduction	1
2	Learning depth estimation from videos	3
2.1	Scale ambiguity	4
2.2	Modeling pose predictor	5
2.3	Training loss	6
2.3.1	Appearance dissimilarity loss	6
2.3.2	Inverse depth smoothness loss	8
3	Differentiable direct visual odometry	9
3.1	Direct visual odometry	9
3.2	Differentiable implementation	10
3.3	Differentiable vs non-differentiable direct visual odometry module	10
4	Experiments	13
4.1	Implementation details	13
4.2	Training configurations	14
4.3	Results on KITTI	14
4.4	Estimate pose from depth prediction	15
4.5	Results on Make3D	17
5	Conclusion	21
	Bibliography	23

May 16, 2018
DRAFT

List of Figures

1.1	Our depth prediction (3rd row) compared against Zhou [34] (2nd row) on KITTI dataset. Our method recovers more fine details such as tree trunks and advertising board.	2
2.1	Effect of inverse depth map normalization. top right: Without normalization, the scale (blue curve, quantified by the mean of inverse depth) decreases throughout each epochs during training, and finally saturates to zero; with normalization, the scale stabilizes within a reasonable range(red curve); bottom: Due to the decrease in scale, the prior loss penalizes less on the smoothness of the depth estimation. As a result, the model makes increasingly more structure mistakes. And the training finally breaks before the 15th epoch, where the prediction degenerates to zero; top mid: With normalization, the depth prediction is more precise.	5
2.2	Illustration of our unsupervised learning pipeline. The learning algorithm takes three sequential images at a time. The Depth-CNN produces three inverse depth maps for the inputs, and the pose predictor (lower part) estimates two relative camera pose between the second image and the other two. The appearance dissimilarity loss is measured between the second image \mathcal{I}_2 and the inversely warped images of $\mathcal{I}_1, \mathcal{I}_3$; In addition, the loss is evaluated in a reverse direction (marked by orange arrows) - it is also measured between $\mathcal{I}_1, \mathcal{I}_3$ and two warped images of \mathcal{I}_2 . Lower part of the figure illustrates three architectures we evaluated for pose prediction: 1) Pose-CNN, 2) use our proposed differentiable Direct Visual Odometry (DDVO), the initialization of pose is set as zero (identity transformation), and 3) a hybrid of the above two - use pretrained Pose-CNN to give a better initial pose for DDVO.	7
3.1	Compare training with <i>differentiable</i> DVO (“DDVO”) to <i>non-differentiable</i> DVO module (“DVO”) on one short clip of video. right: “DDVO”(red) converges faster and reaches better local minimum compared to “DVO”(blue); left-mid: “DDVO” gives a good reconstruction of the scene; left-bottom: “DVO” results in wrong depth estimation.	11

4.1	Qualitative results on KITTI test set (Eigen split). Here, groundtruth depth maps are interpolated from sparse point clouds captured by Lidar (2nd column), thus only serves for visualization purpose. Compared to Zhou [34] (3rd column), our depth map prediction (last column) preserves more details such as tree trunks, roadway stakes, and gives more precise reconstruction of objects near view, such as the van and tree in the 3rd and 4th rows.	17
4.2	Qualitative results on Make3D dataset. We directly apply our model trained from KITTI without any tuning for this dataset.	18
4.3	Failure cases of our method. Our method is sensitive to 1) dynamic scenes like pedestrians and cyclists (1st row); 2) over-exposure of the road, which cause large texture-less regions (2nd row); 3) vast open areas with large texture-less regions (3rd row).	19

List of Tables

4.1	Evaluation of depth map accuracy on KITTI test set. The methods trained over KITTI dataset are denoted by K , and also pretrained on Cityscapes are denoted by CS+K . We use (D) to represent methods trained with depth supervision, (B) to refer methods trained using rectified binocular image pairs and (M) to denote methods trained on monocular video sequences. We show in bold the overall best results trained only on KITTI, and with a blue box to highlight the best results within the (M) methods. Gordard and Zhou’s results are taken directly from [34].	16
4.2	Absolute Trajectory Error (ATE) on the KITTI odometry split averaged over all 5-frame snippets. ORB-SLAM results are copied from [34]. “full” means ORB-SLAM is performed on the full video while “short” means it’s only performed on the 5-frame snippets. For both Zhou and ours, we use the Depth-CNNs trained from the combined dataset of Cityscapes and KITTI.	16
4.3	Results on the Make3D dataset [27]. Following Zhou [34], we do not use any of the Make3D data for training. We directly apply the model trained on KITTI to the test set. Following the evaluation protocol of [34], we first match the scale of our depth prediction to the groundtruth, then the errors are computed only for pixels in a central image crop with groundtruth depth less than 70 meters.	18

May 16, 2018
DRAFT

Chapter 1

Introduction

Depth prediction from a single image using CNNs has had a surge of interest in recent years [7, 20, 22]. As from the cognitive point of view, it attempts to mimic human’s ability to estimate depth through cues such as perspective, occlusion, scaling relative to the known size of familiar objects, lighting and shading [27]. Moreover, it has various applications such as, 2D-3D view synthesis [33], depth perception for grasping in robotics [21], and improving robustness and accuracy in SLAM systems [29].

Recently, unsupervised methods that rely solely on monocular video for training (without depth or stereo groundtruth) have captured the attention of the community. Of particular note in this regard is the work of Zhou et al. [34] who proposed a strategy that learned separate pose and depth CNN predictors by minimizing the photometric consistency across monocular video datasets during training. Although achieving impressive results this strategy falls noticeably behind those that have been trained using rectified stereo image pairs [13, 19]. These rectified stereo methods have shown comparable accuracy to supervised methods [7, 24, 27] over datasets where only sparse depth annotation is available. However, the assumption of using calibrated binocular image pairs excludes itself from utilizing monocular video which is easier to obtain and richer in variability. This performance gap between stereo [13, 19] and monocular [34] learning strategies is of central focus in this thesis.

In this thesis, we attempt to close this gap by drawing inspiration from recent advances in direct visual odometry. Specifically, we note that the major difference between stereo and monocular strategies stems from: (i) unknown camera pose between frames, and (ii) ambiguity in scale. Existing methods [30, 34] for learning depth from monocular video address these differences only partially by adding an extra CNN pose prediction module. In this thesis we argue that these previous strategies do not adequately address the scale ambiguity issue - causing divergence during training. Further, we advocate that the incorporation of an additional CNN pose prediction module is unnecessary. Instead we argue that one could employ a differentiable and deterministic objective for pose prediction which is now commonly employed within the SLAM community for direct visual odometry [2, 6, 8, 9].

Contributions: We make the following contributions. First, we characterize theoretically and demonstrate empirically why scale ambiguity in current monocular methods is problematic. Specifically, the problem lies in the scale sensitivity of the depth regularization terms employed during training. Inspired by related work in direct visual odometry [8] we propose a simple nor-

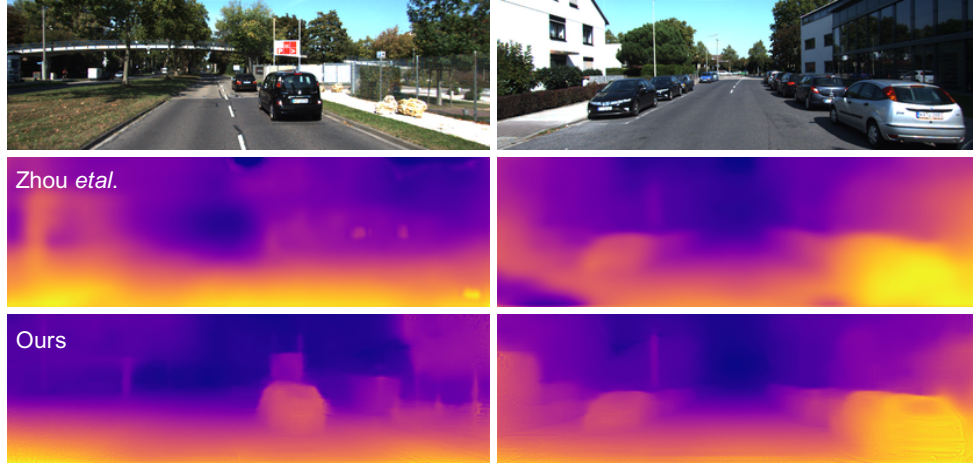


Figure 1.1: Our depth prediction (3rd row) compared against Zhou [34] (2nd row) on KITTI dataset. Our method recovers more fine details such as tree trunks and advertising board.

malization strategy that circumvents many of these problems and leads to noticeably improved performance over Zhou [34](see Fig. 1.1 for qualitative comparison).

Second, we suggest that learning an additional pose predicting CNN (which we shall refer to herein as the Pose-CNN) is not the most effective strategy for estimating a depth predicting CNN from monocular video. The Pose-CNN employed by Zhou does not fully exploit the relation between camera pose and depth prediction, and ignores the fact that pose estimation from depth is a well-studied problem with known geometric properties and well performing algorithms. We instead propose the incorporation of a Direct Visual Odometry (DVO) [28] pose predictor into our framework as: (i) it requires no learnable parameters, (ii) establishes a direct relationship between the input dense depth map and the output pose prediction, and (iii) it is derived from the same image reconstruction loss used for minimizing our entire unsupervised training pipeline. To incorporate DVO into end-to-end training, we propose a differentiable implementation of the DVO (DDVO module), so that the back-propagation signals reaching the camera pose predictor can be propagated to the depth estimator.

Finally, since DVO is a second order gradient descent based method, a good initialization point can lead to better solution. So instead of starting our DDVO module from the identity pose, we propose a mixed training procedure - use a pretrained Pose-CNN to provide pose initialization for DDVO during training. We demonstrate empirically that this hybrid method provides better performance compared to training with Pose-CNN or DDVO alone, and achieves comparable results to Gordard [13], which is the state of the art method trained with calibrated binocular pairs on KITTI dataset [12].

Notation: lowercase boldface symbols (\mathbf{x}) denote vectors, uppercase boldface symbols (\mathbf{W}) denote matrices, and uppercase calligraphic symbols (\mathcal{I}) denote images. We also use the notations $\mathcal{I}(\mathbf{x}) : \mathbb{R}^2 \rightarrow \mathbb{R}^K$ to indicate sampling of the K -channel image representation at subpixel location $\mathbf{x} = [x, y]^\top$.

Chapter 2

Learning depth estimation from videos

Our goal is to learn a function f_d (modeled as a CNN)- parametrized by θ_d - which predicts the inverse depth map \mathcal{D} from a single image \mathcal{I} . Instead of doing supervised learning, we want to learn from more widely available data sources without groundtruth depth. Moreover, instead of restricting ourselves to calibrated binocular images [11, 13, 19], we go for more general case of monocular video sequences.

Before introducing our end-to-end training pipeline, it is worth to mention an alternative approach. Given the temporal cues between sequential video frames, it is possible to first get auxiliary depth annotation from some structure-from-motion (SfM) algorithm, and then use its output as supervision to learn the depth estimator. Suppose that the SfM algorithm can be oversimplified as doing photometric bundle adjustment, which minimizes a cost function combining appearance error \mathcal{L}_{ap} , measuring dissimilarity of pixel-wise correspondences, and some prior cost $\mathcal{L}_{\text{prior}}$, encouraging the smoothness of the depth map. This procedure can be summarized as a two-step optimization:

1. Compute inverse depth maps and camera poses with SfM:

$$\mathcal{D}^*, \mathbf{p}^* = \arg \min_{\mathcal{D}, \mathbf{p}} \mathcal{L}_{\text{ap}}(\mathcal{D}, \mathbf{p}) + \mathcal{L}_{\text{prior}}(\mathcal{D}) \quad (2.1)$$

2. Use inverse depth maps from Eq. 2.1 as supervision to learn the depth estimator:

$$\min_{\theta_d} \mathcal{L}(f_d(\mathcal{I}; \theta_d), \mathcal{D}^*) \quad (2.2)$$

We argue that this two-step optimization is sub-optimal in theory. Since we are assuming that the SfM’s cost function in Eq. 2.1 reflects the quality of the depth map prediction, what we really want is to find θ_d which leads to the minimum of this cost. However, minimizing Eq. 2.2 in the second step does not necessarily lead to minimizing Eq. 2.1.

Therefore, end-to-end learning of a depth predictor by directly minimizing the cost function in Eq. 2.1 is, in principle, the best approach. In practise, however, since Eq. 2.1 is an oversimplification of a real SfM pipeline, it is still possible to get better supervision from the above two-step optimization approach compared to an end-to-end training with gradient descent. Hence, as covered in section 3, one of our contribution is to bring one piece of the sophistication of modern SLAM algorithms - direct visual odometry into the training framework.

Overall, the end-to-end training objective is written as:

$$\min_{\theta_d, \mathbf{p}} \mathcal{L}_{\text{ap.}}(f_d(\mathcal{I}; \theta_d), \mathbf{p}) + \mathcal{L}_{\text{prior}}(f_d(\mathcal{I}; \theta_d)) \quad (2.3)$$

Taking apart the joint minimization, we have the following equivalent formulation:

$$\min_{\theta_d} \min_{\mathbf{p}} \mathcal{L}_{\text{ap.}}(f_d(\mathcal{I}; \theta_d), \mathbf{p}) + \mathcal{L}_{\text{prior}}(f_d(\mathcal{I}; \theta_d)) \quad (2.4)$$

Minimizing $\mathcal{L}_{\text{ap.}}$ over camera pose \mathbf{p} in Eq. 2.4 can be viewed as a function which takes inputs from sequential images $\mathcal{I}_1, \dots, \mathcal{I}_n$ and current depth estimation:

$$f_p(\mathcal{D}, \mathcal{I}_1, \dots, \mathcal{I}_n) \triangleq \arg \min_{\mathbf{p}} \mathcal{L}_{\text{ap.}}(\mathcal{D}, \mathbf{p}). \quad (2.5)$$

Here, we omit \mathcal{I}_i from the input of $\mathcal{L}_{\text{ap.}}$ for conciseness, and we refer f_p as an auxiliary pose predictor. Substitute Eq. 2.5 to the end-to-end training objective function (Eq. 2.4), we have our final formulation of the training objective:

$$\min_{\theta_d} \mathcal{L}_{\text{ap.}}(f_d(\mathcal{I}; \theta_d), f_p(f_d(\mathcal{I}; \theta_d))) + \mathcal{L}_{\text{prior}}(f_d(\mathcal{I}; \theta_d)). \quad (2.6)$$

Detailed definition of our appearance and prior smoothness loss is given in section 2.3.

Optimizing this loss function meets two crucial issues, which we will address over the following subsections.

2.1 Scale ambiguity

Unlike training from calibrated binocular pairs with known baseline, estimating depth and pose from monocular sequence bears scale ambiguity. Any two inverse depth maps differing only in scale (with the pose scaled correspondingly) are equivalent in the projective space. Hence, they have the same appearance dissimilarity loss, or in other words, $\mathcal{L}_{\text{ap.}}$ is scale-invariant in this sense. However, for the regularization term $\mathcal{L}_{\text{prior}}$, usually formulated with derivatives of the inverse depth map [19, 34], smaller inverse depth scale always results in smaller loss value. This scale-sensitive property of the regularization term leads to a catastrophic effect, the loss function in Eq. 2.6 does not even have local minima, let alone global minima. The proof for this is simple: given any inverse depth estimation, we can construct a new inverse depth map through decreasing its scale (and updating the pose accordingly), which always results in lower loss.

Surprisingly, none of the previous works [30, 34] address this issue. Empirically, we find that this critical defect of the loss function not only leads to inferior results compared to training with binocular pairs, but also results in divergence during training. As shown in Fig. 2.1, with our re-implementation of Zhou’s architecture, the scale of inverse depth prediction is decreasing throughout each training epoch. After several epochs, the training diverges and finally breaks due to the inverse depth prediction saturates to close to zero.

To solve this problem, we propose a simple yet effective approach - apply a non-linear operator to normalize the output of the depth CNN before feeding it to the loss layer. The operator

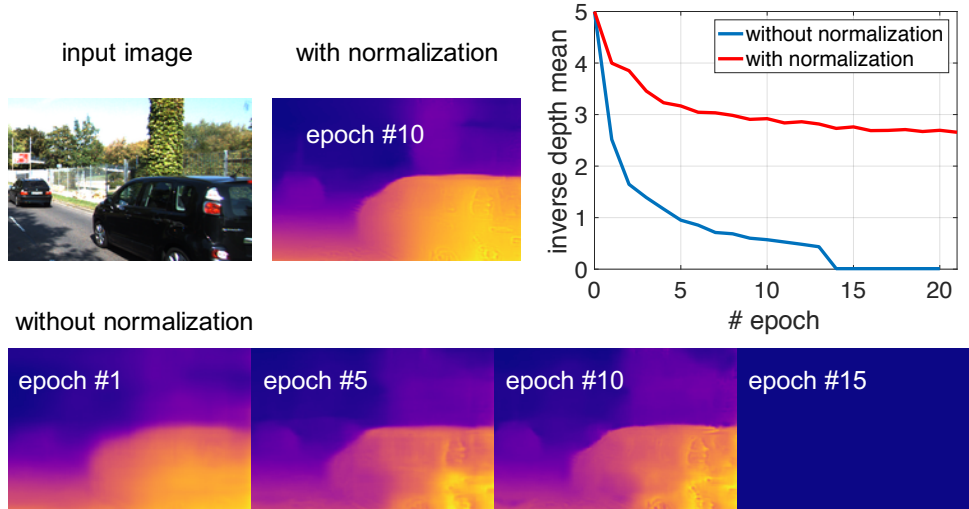


Figure 2.1: Effect of inverse depth map normalization. top right: Without normalization, the scale (blue curve, quantified by the mean of inverse depth) decreases throughout each epochs during training, and finally saturates to zero; with normalization, the scale stabilizes within a reasonable range (red curve); bottom: Due to the decrease in scale, the prior loss penalizes less on the smoothness of the depth estimation. As a result, the model makes increasingly more structure mistakes. And the training finally breaks before the 15th epoch, where the prediction degenerates to zero; top mid: With normalization, the depth prediction is more precise.

$\eta(\cdot)$ we use here is dividing the input tensor by its mean:

$$\eta(d_i) \triangleq \frac{Nd_i}{\sum_{j=1}^N d_j} \quad (2.7)$$

Although this normalization trick has also been used in LSD-SLAM [8] for keyframe selection, it has not been used in the context of learning depth estimation from monocular videos. Empirically we find that this simple normalization trick significantly improves results (see Fig. 2.1) as it removes the scale sensitive problem of the loss function.

2.2 Modeling pose predictor

In previous works, the pose predictor is approximated as a feed-forward CNN, which takes input only from a sequence of frames. Though this approach enjoys the simplicity of using a black box, and is easy to integrate into end-to-end training, it ignores the geometric relation between camera pose and depth as summarized in Eq. 2.5 which could be exploited for better training.

We propose an alternative approach which attempts to close this discrepancy between theory and practice through noticing the fact that Eq. 2.5 relates to a class of well-studied geometry-based methods, direct visual odometry (DVO). Given the current depth estimation from f_d , DVO solves for camera pose through minimizing the photometric reprojection error between a reference image and a source image. DVO relates to a more general class of image registration

method – the Lucas-Kanade(LK) algorithm. It applies the same Gauss-Newton minimization as in LK for faster convergence, and use the inverse composition trick for computational efficiency.

However, to incorporate DVO as the pose prediction module f_p in the end-to-end training, we need to be able to evaluate the derivatives of DVO. Inspired by recent differentiable implementation of inverse compositional LK for object tracking [31] and planar alignment [4], we propose a (sub)differentiable DVO implementation which is described in detail in section 3. With it, during training, the gradient of estimated depth with respect to loss $\mathcal{L}_{\text{ap.}}$ comes from two sources: partial derivative of loss over depth and partial derivative of loss over pose:

$$\frac{d\mathcal{L}_{\text{ap.}}}{df_d} = \frac{\partial\mathcal{L}_{\text{ap.}}}{\partial f_d} + \frac{\partial\mathcal{L}_{\text{ap.}}}{\partial f_p} \frac{\partial f_p}{\partial f_d}. \quad (2.8)$$

It is worth to mention that Eq. 2.8 demonstrates one of our major theoretical difference to Zhou [34] - our depth estimator gets additional back-propagation signals from the pose prediction, while they ignore the role depth estimation plays in getting camera pose.

We experiment with two versions of applying our differentiable direct visual odometry (DDVO) module in training: 1) we train our Depth-CNN from scratch with DDVO; 2) we first pretrain the Depth-CNN with Pose-CNN, then we fine-tune the Depth-CNN by using the output of Pose-CNN as initialization for DDVO. The motivation for the second approach can be explained in both ways: compared to initializing from the identity pose, Pose-CNN provides a better starting point for DVO; on the other hand, DVO refines the output of Pose-CNN with geometric cues.

2.3 Training loss

Our training loss is a combination of appearance loss and prior smoothness loss, aggregated through 4 scales. The appearance loss measures dissimilarity of pixel-wise correspondence between a triplet of images(illustrated in Fig. 2.2); and the prior loss enforce smoothness for the inverse depths \mathcal{D}_i of the images in the triplet. The loss on the k th scale is expressed as:

$$\begin{aligned} \mathcal{L}^{(k)} = & \sum_{i \neq j \in \{1,2,3\}} \mathcal{L}_{\text{ap.}}(\mathcal{D}_i^{(k)}, \mathbf{p}_{ij}; \mathcal{I}_i^{(k)}, \mathcal{I}_j^{(k)}) \\ & + \lambda \sum_{i=1}^3 \mathcal{L}_{\text{prior}}(\mathcal{D}_i^{(k)}). \end{aligned} \quad (2.9)$$

We use weighting parameter λ to control the penalty from smoothness prior.

2.3.1 Appearance dissimilarity loss

As illustrated in Fig. 2.2, we measure appearance dissimilarity loss between three sequential images. Given inverse depth maps predicted from these images respectively, and two camera extrinsics estimated relative to the second image, we compare the inversely warped first and last images to the second image. We also do it reversely, comparing the inversely warped second image to the other two images. This bidirectional appearance loss allows us to augment the training set with minimum extra computational cost.

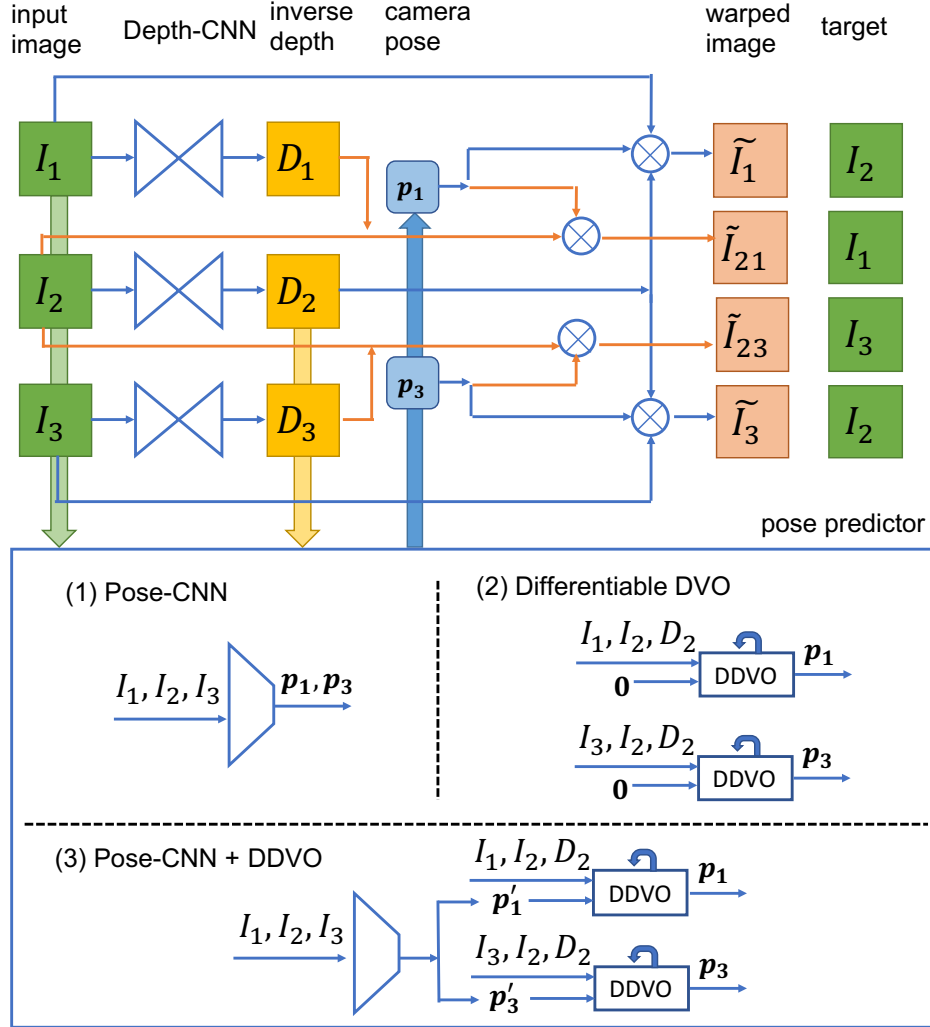


Figure 2.2: Illustration of our unsupervised learning pipeline. The learning algorithm takes three sequential images at a time. The Depth-CNN produces three inverse depth maps for the inputs, and the pose predictor (lower part) estimates two relative camera pose between the second image and the other two. The appearance dissimilarity loss is measured between the second image \mathcal{I}_2 and the inversely warped images of $\mathcal{I}_1, \mathcal{I}_3$; In addition, the loss is evaluated in a reverse direction (marked by orange arrows) - it is also measured between $\mathcal{I}_1, \mathcal{I}_3$ and two warped images of \mathcal{I}_2 . Lower part of the figure illustrates three architectures we evaluated for pose prediction: 1) Pose-CNN, 2) use our proposed differentiable Direct Visual Odometry (DDVO), the initialization of pose is set as zero (identity transformation), and 3) a hybrid of the above two - use pretrained Pose-CNN to give a better initial pose for DDVO.

Appearance dissimilarity is aggregated through four scales of the output of the Depth-CNN. For the three coarser scales, we use L1 photometric loss; For the last finest scale, we adopt a linear combination of L1 photometric loss and single scale SSIM [32] loss, which is the same as in [13].

2.3.2 Inverse depth smoothness loss

We use second order gradients as smoothness cost to encourage the depth prediction has flat slopes. To encourage sharpness of the depth prediction, we employ two strategies:

(1) We give different weightings to the smoothness cost at different pixel locations according to the Laplacian of the image intensity. If the Laplacian is higher, meaning the pixel is more likely on an edge or corner, we impose a lower smoothness cost at that location:

$$\mathcal{L}_{\text{prior}}(d_i) = e^{-\nabla^2 \mathcal{I}(\mathbf{x}_i)} (|\partial_{xx}d_i| + |\partial_{xy}d_i| + |\partial_{yy}d_i|) \quad (2.10)$$

(2) Instead of directly compute smoothness cost from the final inverse depth estimation, we downsample the inverse depth map by factor 2 to three coarser scales. We aggregate smoothness cost only from the two coarsest scales.

Chapter 3

Differentiable direct visual odometry

3.1 Direct visual odometry

Steinbrücker [28] first proposed the direct visual odometry in the form of minimizing pixel-wise photometric error between sequential RGB-D frames. It was latter improved with more robust cost functions [15, 17] and features [1], and extended to photometric bundle adjustment [2, 6, 8]. Compared to visual odometry methods using sparse feature correspondences, direct methods offer better accuracy when inter-frame motion is small, since they make use of the whole dense information contained in the image. Hence direct methods have been widely used in current state-of-the-art visual SLAM systems [8, 10].

Direct visual odometry takes input from a reference image \mathcal{I} , its corresponding depth map \mathcal{D} , and a source image \mathcal{I}' . Let the intensity and inverse depth of a pixel coordinate $\mathbf{x}_i \in \mathbb{R}^2$ at the reference image be respectively stated as $\mathcal{I}(\mathbf{x}_i)$ and d_i ; camera pose be represented by the concatenation of translation $\mathbf{t} \in \mathbb{R}^3$ and exponential coordinates $\omega \in \mathbb{R}^3$; and the camera intrinsics be known, the projection from the reference image coordinates \mathbf{x}_i to source image coordinates \mathbf{x}'_i is computed as:

$$\mathbf{x}'_i = \mathcal{W}(\mathbf{x}_i; \mathbf{p}, d_i) \triangleq \langle \mathbf{R}\tilde{\mathbf{x}}_i + d_i\mathbf{t} \rangle, \quad (3.1)$$

where rotation matrix \mathbf{R} is computed from exponential coordinates ω by the *Rodrigues' formula* [26]; $\tilde{\mathbf{x}}_i$ is the homogeneous coordinate for \mathbf{x}_i ; and $\langle \cdot \rangle$ projects 3D points to the image plane:

$$\langle [x \ y \ z]^T \rangle = \begin{bmatrix} x & y \\ z & z \end{bmatrix}^T \quad (3.2)$$

The objective of direct visual odometry is to find an optimum camera pose \mathbf{p} which minimizes the photometric error between the warped source image and the reference image which is assumed to be always at the identity pose $\mathbf{0}$:

$$\min_{\mathbf{p}} \sum_i \|\mathcal{I}'(\mathcal{W}(\mathbf{x}_i; \mathbf{p}, d_i)) - \mathcal{I}(\mathcal{W}(\mathbf{x}_i; \mathbf{0}, d_i))\|^2. \quad (3.3)$$

This nonlinear least square problem can be solved efficiently through Gauss-Newton method. For computational efficiency, instead of linearly approximating the source image as in vanilla Gauss-Newton method, the Inverse Compositional algorithm [3] reverses the roll of source image and reference image and compute the parameters update for the reference image. As a result, the Jacobian and Hessian matrix do not need to be re-evaluated per iteration. We summarize the algorithm as following:

1. Precompute the Jacobian matrix, and its pseudo inverse:

$$\mathbf{J} = \begin{bmatrix} \vdots \\ \nabla \mathcal{I}(\mathbf{x}_i) \cdot \left. \frac{\partial \mathcal{W}(\mathbf{x}_i; \mathbf{p}, d_i)}{\partial \mathbf{p}} \right|_0 \\ \vdots \end{bmatrix}, \quad \mathbf{J}^\dagger = (\mathbf{J}^T \mathbf{J})^{-1} \mathbf{J}^T. \quad (3.4)$$

2. Warp the source image by the current estimate of camera pose \mathbf{p} , and convert it to a vector denoted as $\mathbf{I}'_{\mathbf{p}}$. Form a binary diagonal matrix \mathbf{W} whose i_{th} diagonal element represents if projected coordinate \mathbf{x}'_i is in view. We use this weight matrix to exclude out-of-view pixels from taking part in computing pose update.
3. Compute pose update:

$$\Delta \mathbf{p} = \mathbf{J}^\dagger \mathbf{W} (\mathbf{I} - \mathbf{I}'_{\mathbf{p}}), \quad (3.5)$$

and compose it to get a new pose estimation:

$$\mathbf{p} \leftarrow \Delta \mathbf{p} \circ \mathbf{p}, \quad (3.6)$$

here, we define the operator \circ such that it applies a left multiplicative update to the pose.

4. Return to step 2, and repeat till converge.

The main restriction of direct methods is that inter-frame motion must be small. To handle larger motions, we form pyramids of images and depths map through downsampling, and update the pose estimate through coarse-to-fine approach.

3.2 Differentiable implementation

Our differentiable implementation of DVO is similar to the inverse compositional spatial transformer network [23]. Both methods iteratively update the geometric transformation parameters and inversely warp the source image through bilinear sampling. The difference is that, instead of using regression network layer with learnable parameters, our “regressor” is deterministically formed with Eq. 3.5. Computing derivatives with respect to Eq. 3.5 involves differentiating matrix pseudo-inverse, which we follow similar implementation as in [31].

3.3 Differentiable vs non-differentiable direct visual odometry module

As an alternative approach, instead of end-to-end optimizing the objective defined in Eq. 2.3, we could employ an EM style optimization procedure – first estimate the camera pose \mathbf{p}^* given

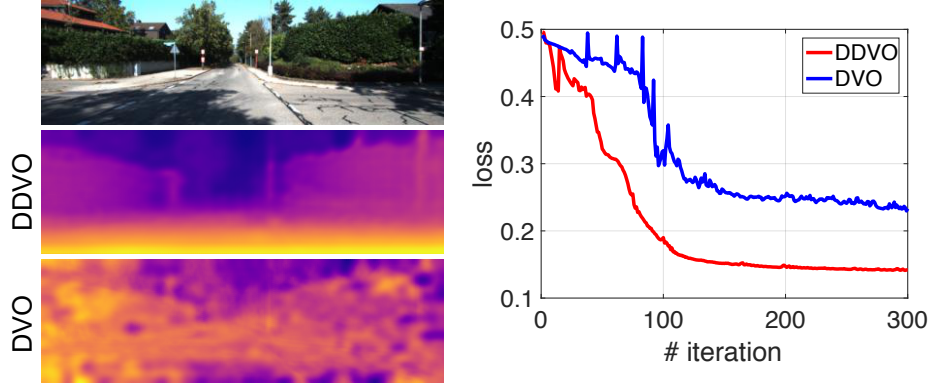


Figure 3.1: Compare training with *differentiable* DVO (“DDVO”) to *non-differentiable* DVO module (“DVO”) on one short clip of video. right: “DDVO”(red) converges faster and reaches better local minimum compared to “DVO”(blue); left-mid: “DDVO” gives a good reconstruction of the scene; left-bottom: “DVO” results in wrong depth estimation.

current depth prediction; then with the pose estimation fixed, we minimize the loss with respect to depth estimation. This is summarized as iteratively performing the following two steps:

$$\mathbf{p}^* = f_p(f_d(\mathcal{I}; \theta_d), \mathcal{I}_1, \dots, \mathcal{I}_n). \quad (3.7)$$

$$\min_{\theta_d} \mathcal{L}_{\text{ap.}}(f_d(\mathcal{I}; \theta_d), \mathbf{p}^*) + \mathcal{L}_{\text{prior}}(f_d(\mathcal{I}; \theta_d)). \quad (3.8)$$

The implication of this approach is that we no longer need to do back propagation through the direct visual odometry module, since in the second step we pretend \mathbf{p}^* to be some constant. Hereby, we refer this approach as using *non-differentiable* direct visual odometry module (denoted as “DVO” in Fig. 3.1).

To test this approach, we did a toy experiment which trains the depth estimator to overfit a short video clip. As shown in Fig. 3.1, this approach (referred as “DVO”) converges slower (see the blue curve in right) than the *differentiable* direct visual odometry module (“DDVO”, red curve) proposed in the paper, and it sticks in a poor local minimum(visualized in bottom left).

Therefore, we conclude that using *differentiable* DVO module is necessary for training a depth estimator from monocular videos.

May 16, 2018
DRAFT

Chapter 4

Experiments

This section extensively evaluates our method of unsupervised training depth estimator from monocular videos. Following the other relevant works [11, 13, 19, 30, 34] we employ KITTI dataset [12] to train our approach. We use the same train/test split used in [7], meaning that we use 28 sequences for training and 28 sequences for testing in the categories of "city", "residential" and "road". Since each of the sequences are captured with a stereo pair, we have 2 monocular sequences giving 56 monocular sequences for training.

4.1 Implementation details

Training set construction We follow [34] to preprocess the training sequences. To do so, we first remove the static frames by inspecting if the mean magnitude of optical flow is less than 1 pixel. Each training sample is a short clip of 3 sequential video frames resized to 128×416 pixels.

Depth-CNN architecture We use the same network architecture as in [13]. The network resembles a U-Net structure. It consists of an encoder-decoder design with skip connections of intermediate features between the encoder and decoder network, and uses VGG as its encoder network architecture. We also output multi-scale inverse depth predictions at the end of the decoder. Since inverse depth is bounded between 0 (at infinity) and the inverse of minimum depth to the camera, we apply sigmoid non-linearity to the network output. Moreover, we multiply the final inverse depth output by 10 to constrain it inside a reasonable range, and add a small value 0.01 to improve numerical stability.

Pose-CNN architecture We use a similar CNN architecture as in [34], which takes input from a concatenation of three sequential frames. The differences are 1) instead of outputting rotation as Euler angles, we train our Pose-CNN to predict exponential coordinates; 2) we divide the predicted translation by the mean value of estimated depth, so that the scale of the translation is aligned to the depth prediction.

Training hyper-parameters Our training loss is a weighted sum of appearance dissimilarity loss

and smoothness loss. We set the weighting for the smoothness loss as 0.01. We empirically found that this weighting parameter offers a desired balance between sharpness and overall structure correctness of the depth prediction.

We train the network through Adam optimizer [16], with a small learning rate of 0.0001, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. Restricted by the non-conventional operation used in our DDVO module, and for the sake of computational efficiency, we set the batch size for training to 1. This compromise on batch size is solely due to implementation issues, not an implication of theory. Empirically, we observe no negative effect on overall performance.

4.2 Training configurations

For a complete study of our proposed method, we conduct experiments with four different settings:

- **Ours (Baseline)** To understand the effect of our proposed normalization trick in Sec. 2.1, we setup a baseline configuration. We use Pose-CNN as the pose predictor and turn off the inverse depth map normalization. This setup is equivalent to Zhou [34]’s method modified with our loss function definition (see Sec. 2.3) and implementation. Due to the training diverges after 10 epochs (see Fig. 2.1), we report the result at the 10th epoch.
- **Ours (Pose-CNN)** We perform inverse depth normalization, and still use Pose-CNN as the pose predictor. We report result at the 10th epoch.
- **Ours (DDVO)**. We replace the Pose-CNN with our proposed DDVO module. The DDVO is initialized with the identity pose. To account for large motion in the training set, we set the module to run through five scales in a coarse-to-fine manner. We train the model from scratch, and result is reported at the 10th epoch.
- **Ours (Pose-CNN + DDVO)**. We use the Depth-CNN and Pose-CNN from “ours (Pose-CNN)” as pretrained model. Then we fix the Pose-CNN, use its output as pose initialization for our DDVO module, and fine-tune the Depth-CNN for 2 epochs. Since Pose-CNN already gives us a roughly acceptable estimation, DDVO does not need to account for large motion in this case. Hence, to speed up training, instead of performing coarse-to-fine, we directly run DDVO over the finest scale.

4.3 Results on KITTI

In this section, we evaluate performance on the same 697 KITTI images from the Eigen’s test split [7]. We compute all the errors from the same cropped region using the same performance measures as in [11, 13, 34] and without capping to a maximum depth. Our method cannot recover the true scale of the scene due to the monocular settings of the problem. Therefore, we align the scale of our prediction to the groundtruth through multiplying the former by $\tilde{s} =$

$\text{median}(D_{\text{gt}})/\text{median}(D_{\text{predict}})$ as in [34].

Effect of depth map normalization. We compare the result of “ours (Baseline)” to “ours (Pose-CNN)” since these two configurations only differ in whether inverse depth normalization is performed or not (see section 2.1). As shown in Table 4.1, the simple normalization approach gets a significant improvement in all the measures using the same architecture. Moreover, with the results of depth normalization, “Ours (Pose-CNN)” not only beats Zhou [34]’s work by a large margin, and also gets very close results to the one trained on rectified stereo pairs by Godard *et al.* [13].

DDVO vs Pose-CNN We also test the effect of changing the Pose-CNN module by the DDVO module proposed in section 3. In this case, we have no parameter to learn in the pose estimation module. In Table 4.1 we show the results with label “ours (DDVO)”. As the DDVO module performs Gauss-Newton optimization, we initialize the pose between each pair of training images as the identity. On the other hand, Gauss-Newton approaches are sensible to the initialization point. This is the reason for getting slightly worse results with DDVO pose estimation (“ours (DDVO)”) than with a learned Pose-CNN pose estimator (“ours (Pose-CNN)”). In any case the results are still very close to the stereo pairs training method of Godard [13].

Finally, in order to improve the pose initialization point in the DDVO module we have experimented a new way of training. The pose between a pair of images in a video sequence is initialized by a pretrained Pose-CNN module and then refined with the DDVO module. In Table 4.1 we show that this strategy (“ours (Pose-CNN+DDVO)”) improves both “ours (Pose-CNN)” and “ours (DDVO)”, and gives the best results of the unsupervised trained methods trained on KITTI dataset. Qualitative results from this approach are visualized in Fig. 4.1.

Pretrain on Cityscapes As shown in [13, 34] pretraining the model on a larger dataset, Cityscapes [5], and then finetuning on KITTI improve the results. Hence, we also perform a similar experiment. First, we train on Cityscapes with “ours (Pose-CNN)” architecture for 10 epochs, then finetune the model on KITTI with “ours (Pose-CNN)” for the first 5 epochs and “ours (Pose-CNN + DDVO)” for the next 3 epochs. We observe a slight performance improvement over training only on KITTI but still outperformed by Godard [13]. This states a bottleneck of our approach, where our method is not modeling the non-rigid part of the scene. This prevents us to get more benefit from a dataset with a lot of objects in motion.

4.4 Estimate pose from depth prediction

Tateno [29] demonstrates that one plausible usage of monocular depth estimator is to initialize the depth estimation for the visual SLAM systems. Hence, to test the implication of the improvement in depth map accuracy under this usage scenario, we run the direct visual odometry algorithm(described in section 3.1) over depth maps predicted by Zhou [34] and our method. Through inspecting the pose errors (measured by absolute trajectory error), we find that the improvement in our depth map accuracy results in significantly better pose estimation (see Table 4.4).

Method	training set	abs rel	sq rel	RMSE	RMSE(log)	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Eigen [7]	K (D)	0.203	1.548	6.307	0.282	0.702	0.890	0.958
Liu [24]	K (D)	0.202	1.614	6.523	0.275	0.678	0.895	0.965
Godard [13]	K (B)	0.148	1.344	5.927	0.247	0.803	0.922	0.964
Zhou [34]	K (M)	0.208	1.768	6.856	0.283	0.678	0.885	0.957
ours (baseline)	K (M)	0.213	3.770	6.925	0.294	0.758	0.909	0.958
ours (Pose-CNN)	K (M)	0.155	1.193	5.613	0.229	0.797	0.935	0.975
ours (DDVO)	K (M)	0.159	1.347	5.789	0.234	0.796	0.933	0.973
ours (Pose-CNN+DDVO)	K (M)	0.151	1.257	5.583	0.228	0.810	0.936	0.974
Godard [13]	CS+K (B)	0.124	1.076	5.311	0.219	0.847	0.942	0.973
Zhou [34]	CS+K (M)	0.198	1.836	6.565	0.275	0.718	0.901	0.960
ours (Pose-CNN+DDVO)	CS+K (M)	0.148	1.187	5.496	0.226	0.812	0.938	0.975

Table 4.1: Evaluation of depth map accuracy on KITTI test set. The methods trained over KITTI dataset are denoted by K, and also pretrained on Cityscapes are denoted by CS+K. We use (D) to represent methods trained with depth supervision, (B) to refer methods trained using rectified binocular image pairs and (M) to denote methods trained on monocular video sequences. We show in **bold** the overall best results trained only on KITTI, and with a **blue box** to highlight the best results within the (M) methods. Godard and Zhou’s results are taken directly from [34].

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
Zhou [34]	0.063±0.126	0.085±0.115
Ours (Pose-CNN + DDVO)	0.045±0.108	0.033±0.074

Table 4.2: Absolute Trajectory Error (ATE) on the KITTI odometry split averaged over all 5-frame snippets. ORB-SLAM results are copied from [34]. “full” means ORB-SLAM is performed on the full video while “short” means it’s only performed on the 5-frame snippets. For both Zhou and ours, we use the Depth-CNNs trained from the combined dataset of Cityscapes and KITTI.

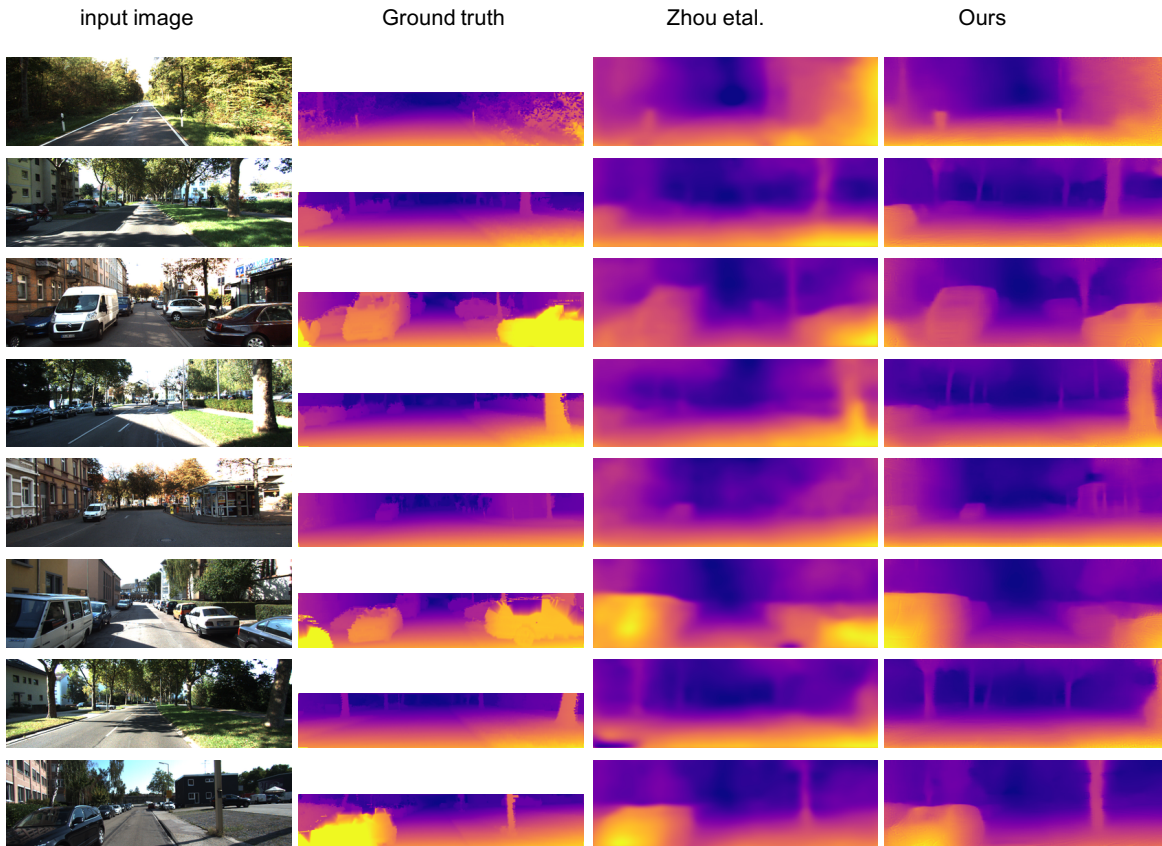


Figure 4.1: Qualitative results on KITTI test set (Eigen split). Here, groundtruth depth maps are interpolated from sparse point clouds captured by Lidar (2nd column), thus only serves for visualization purpose. Compared to Zhou [34] (3rd column), our depth map prediction (last column) preserves more details such as tree trunks, roadway stakes, and gives more precise reconstruction of objects near view, such as the van and tree in the 3rd and 4th rows.

4.5 Results on Make3D

We directly apply our model trained from KITTI (Ours (Pose-CNN + DDVO)) to the Make3D test set [27]. Due to the difference in camera aspect ratio, we only evaluate our method on the central crop of Make3D’s test images. We find that our method generalizes moderately well on this dataset. As shown in Fig. 4.2, our method can recover details such as the road sign in distance, and wooden pole in close view. Table 4.3 shows that our method achieves similar or better results compared to other state of the art approaches.

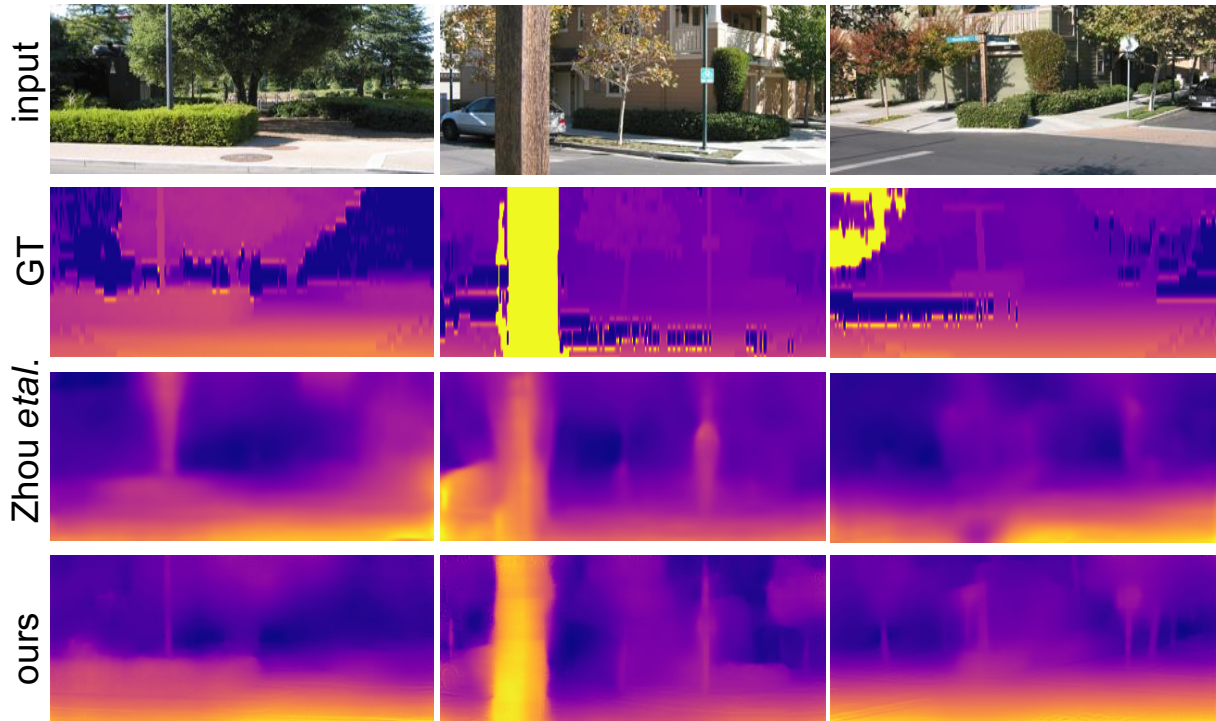


Figure 4.2: Qualitative results on Make3D dataset. We directly apply our model trained from KITTI without any tuning for this dataset.

Method	Supervision	Abs Rel	Sq Rel	RMSE	RMSE(log)
Train set mean	depth	0.876	13.98	12.27	0.307
Karsch [14]	depth	0.428	5.079	8.389	0.149
Liu [25]	depth	0.475	6.562	10.05	0.165
Laina [20]	depth	0.204	1.840	5.683	0.084
Godard [13]	pose	0.544	10.94	11.76	0.193
Zhou [34]	none	0.383	5.321	10.47	0.478
ours	none	0.387	4.720	8.09	0.204

Table 4.3: Results on the Make3D dataset [27]. Following Zhou [34], we do not use any of the Make3D data for training. We directly apply the model trained on KITTI to the test set. Following the evaluation protocol of [34], we first match the scale of our depth prediction to the groundtruth, then the errors are computed only for pixels in a central image crop with groundtruth depth less than 70 meters.



Figure 4.3: Failure cases of our method. Our method is sensitive to 1) dynamic scenes like pedestrians and cyclists (1st row); 2) over-exposure of the road, which cause large texture-less regions (2nd row); 3) vast open areas with large texture-less regions (3rd row).

May 16, 2018
DRAFT

Chapter 5

Conclusion

We have found that, as in monocular SLAM algorithms [8], scale ambiguity has to be taken into account when training with monocular videos. As shown in Fig. 2.1 this was a missing point in previous approaches training from monocular videos. Additionally, we have shown that Direct Visual Odometry can be used to estimate the relative camera pose between frames instead of learning a predictor with a CNN (Pose-CNN). This strategy needs less parameters to learn and potentially less images to train. We plan to investigate the number of images needed in the future. Finally, we have found that DDVO and Pose-CNN pose prediction modules can be improved with a hybrid architecture of Pose-CNN initialization and DDVO pose refinement giving the best results.

The current major bottleneck for our approach is that we're not modeling the non-rigidity of the world. As shown in Fig. 4.3, our current method does not perform well for articulated objects like bikers and pedestrians. Possible future work for this is to incorporate techniques from non-rigid SfM [18] to the pipeline.

May 16, 2018
DRAFT

Bibliography

- [1] Hatem Alismail, Brett Browning, and Simon Lucey. Direct visual odometry using bit-planes. *arXiv preprint arXiv:1604.00990*, 2016. 3.1
- [2] Hatem Alismail, Brett Browning, and Simon Lucey. Photometric bundle adjustment for vision-based slam. In *Asian Conference on Computer Vision*, pages 324–341. Springer, 2016. 1, 3.1
- [3] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, Feb 2004. 3.1
- [4] Che-Han Chang, Chun-Nan Chou, and Edward Y Chang. Clkn: Cascaded lucas-kanade networks for image alignment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2213–2221, 2017. 2.2
- [5] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 4.3
- [6] Amaël Delaunoy and Marc Pollefeys. Photometric bundle adjustment for dense multi-view 3d modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1486–1493, 2014. 1, 3.1
- [7] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Conf. Neural Information Processing Systems*, pages 2366–2374, 2014. 1, 4, 4.3, 4.3
- [8] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *Proc. European Conf. Computer Vision*, September 2014. 1, 2.1, 3.1, 5
- [9] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. In *arXiv:1607.02565*, July 2016. 1
- [10] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 3.1
- [11] Ravi Garg, Vijay Kumar B.G., Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *Proc. European Conf. Computer Vision*, pages 740–756, 2016. 2, 4, 4.3
- [12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proc. IEEE Conf. on Computer Vision and Pattern*

Recognition, 2012. 1, 4

- [13] Clément Godard, Oisín Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, July 2017. 1, 1, 2, 2.3.1, 4, 4.1, 4.3, 4.3, 4.5
- [14] Kevin Karsch, Ce Liu, and Sing Bing Kang. Depth transfer: Depth extraction from video using non-parametric sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(36):2144–2158, 2014. 4.5
- [15] Christian Kerl, Jürgen Sturm, and Daniel Cremers. Robust odometry estimation for rgb-d cameras. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3748–3754. IEEE, 2013. 3.1
- [16] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4.1
- [17] Sebastian Klose, Philipp Heise, and Alois Knoll. Efficient compositional approaches for real-time robust direct visual odometry from rgb-d data. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1100–1106. IEEE, 2013. 3.1
- [18] Suryansh Kumar, Yuchao Dai, and Hongdong Li. Monocular dense 3d reconstruction of a complex dynamic scene from two perspective frames. *arXiv preprint arXiv:1708.04398*, 2017. 5
- [19] Yevhen Kuznetsov, Jorg Stuckler, and Bastian Leibe. Semi-supervised deep learning for monocular depth map prediction. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, July 2017. 1, 2, 2.1, 4
- [20] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. In *Proc. Int’l Conf. on 3D Vision*, pages 239–248, 2016. 1, 4.5
- [21] Ian Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 34(4-5):705–724, 2015. 1
- [22] Bo Li, Chunhua Shen, Yuchao Dai, Anton van den Hengel, and Mingyi He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 1119–1127, 2015. 1
- [23] Chen-Hsuan Lin and Simon Lucey. Inverse compositional spatial transformer networks. 2017. 3.2
- [24] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian D. Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 38(10):2024–2039, 2016. 1, 4.3
- [25] Miaomiao Liu, Mathieu Salzmann, and Xuming 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. 4.5
- [26] Richard M Murray, Zexiang Li, S Shankar Sastry, and S Shankara Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 1994. 3.1

- [27] A. Saxena, M. Sun, and A. Y. Ng. Make3d: Learning 3d scene structure from a single still image. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 31(5):824–840, 2009. (document), 1, 4.5, 4.3
- [28] F. Steinbruecker, J. Sturm, and D. Cremers. Real-time visual odometry from dense rgb-d images. In *Proc. Int’l Conf. Computer Vision Workshops*, 2011. 1, 3.1
- [29] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. *arXiv preprint arXiv:1704.03489*, 2017. 1, 4.4
- [30] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. Sfm-net: Learning of structure and motion from video. *CoRR*, abs/1704.07804, 2017. URL <http://arxiv.org/abs/1704.07804>. 1, 2.1, 4
- [31] Chaoyang Wang, Hamed Kiani Galoogahi, Chen-Hsuan Lin, and Simon Lucey. Deep-1k for efficient adaptive object tracking. *arXiv preprint arXiv:1705.06839*, 2017. 2.2, 3.2
- [32] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Processing*, 13(4):600–612, April 2004. ISSN 1057-7149. 2.3.1
- [33] Junyuan Xie, Ross Girshick, and Ali Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In *European Conference on Computer Vision*, pages 842–857. Springer, 2016. 1
- [34] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, July 2017. (document), 1, 1, 1.1, 2.1, 2.2, 4, 4.1, 4.2, 4.3, 4.4, 4.3, 4.1, 4.2, 4.1, 4.5, 4.3