Digging Into Self-Supervised Monocular Depth Estimation

Clément Godard Oisin Mac Aodha Gabriel J. Brostow

Abstract. Depth-sensing is important for both navigation and scene understanding. However, procuring RGB images with corresponding depth data for training deep models is challenging; large-scale, varied, datasets with ground truth training data are scarce. Consequently, several recent methods have proposed treating the training of monocular color-to-depth estimation networks as an image reconstruction problem, thus forgoing the need for ground truth depth.

There are multiple concepts and design decisions for these networks that seem sensible, but give mixed or surprising results when tested. For example, binocular stereo as the source of self-supervision seems cumbersome and hard to scale, yet results are less blurry compared to training with monocular videos. Such decisions also interplay with questions about architectures, loss functions, image scales, and motion handling. In this paper, we propose a simple yet effective model, with several general architectural and loss innovations, that surpasses all other self-supervised depth estimation approaches on KITTI.

Keywords: monocular depth, self-supervision, CNNs

1 Introduction

We seek to automatically infer a dense depth map from a single color input image. Just estimating the *relative* depths of objects in a single image is an ill-posed problem. However, even without stereo cues, the human visual system is remarkably adept at performing this task. While there are possibly many distinct depth explanations for a given scene, humans learn from navigating and interacting in the real-world, enabling us to hypothesize plausible solutions in novel environments.

Accurate monocular depth estimation has many applications, from image editing through to image understanding. For specific medical applications, where there may be form factor restrictions, it offers the possibility of using single cameras instead of stereo pairs. Unlike expensive LIDAR sensors commonly used in self-driving solutions, traditional cameras are a much cheaper alternative, and simplify car design. Finally, solving for depth is a powerful way to use large quantities of unlabeled data for pretraining deep networks, which can then be used for other discriminative tasks, requiring less supervision [1].

One way to train deep depth estimation models is to use **ground truth depth images** paired with their corresponding intensity images as a supervision signal, e.g. [2,3]. However, collecting large and varied training datasets with

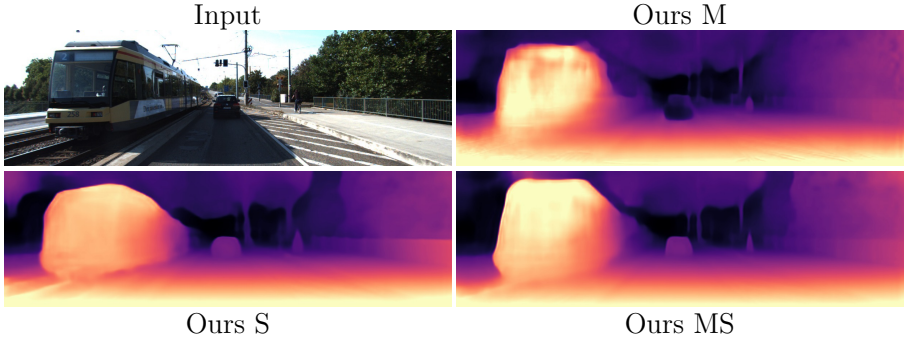

Fig. 1. Estimated depth. Our method significantly outperforms all previously published self-supervised depth estimation methods, whether training with monocular supervision (M), stereo supervision (S), or both (MS).

ground truth depth is itself a formidable challenge. Recently, several approaches have shown that it is instead possible to train monocular depth estimation models using only synchronized **stereo pairs** at training time [4,5,6]. While easier than laser-scanning, this still requires the collection of binocular stereo images. As an alternative, [7] successfully showed that **monocular video** sequences can be used for training, but with a drop in the quality of test time depth predictions.

Among the two self-supervised approaches, monocular video is an attractive alternative to stereo based supervision, but it introduces its own set of challenges. In addition to estimating depth, the model also needs to estimating the egomotion between temporal image pairs during training. This typically involves training a *separate* pose network that takes a sequence of frames as input, and outputs the corresponding camera transformations. Using stereo data for training makes the camera-pose estimation a one-time offline step, but can cause issues related to occlusion and texture-copy artifacts [6]. To address these issues, we propose a new architecture that shares weights between the pose and depth networks in the monocular setting, and also drastically reduces texture-copy artifacts by performing image sampling at the input scale (see Fig. 1).

In this work, we propose several architectural and loss innovations that, when combined, lead to large improvements in monocular depth estimation, using either monocular video, stereo pairs, or a combination as training data. We present the following three contributions: (1) A novel architecture and loss function for the problem of self-supervised monocular depth estimation. Our models achieve state-of-the-art results on the KITTI dataset [8] in the self-supervised setting, and simplify many of the common components found in existing networks that use either monocular or stereo training data. (2) A detailed evaluation, and list of best practices, that highlights how each component of our model lends itself to improved test time performance. (3) Finally, we show that KITTI, the most commonly used evaluation dataset for monocular depth estimation, is not an informative benchmark for real-world monocular training moving forward, as it contains only a limited number of independently moving objects. These three contributions are a result of findings that may be non-obvious or even surprising,

e.g. training with high resolution images is better than with low resolutions, and training with monocular video is improved by expressly excluding footage with moving objects.

2 Related Work

Given a single color image at test time, our goal is to predict the depth of each pixel in the input image. Here we review work that takes a single color image as input test time. This is in contrast to multi-view stereo based approaches, which have access to multiple input frames or temporal sequences at test time.

Supervised Depth Estimation
Depth from a single image is an inherently ill-posed problem as the same input image can project to multiple plausible depths. To address this issue, learning based methods have shown themselves capable of fitting predictive models that exploit the relationship between input images and their corresponding depth maps. Various methods, such as combining local predictions [9,2], non-parametric scene sampling [10], through to convolution neural networks [3,11] have been explored. Learning based algorithms are also among some of the best performing for conventional stereo estimation [12,13,14,15] and optical flow [16,17,18].

The majority of the above approaches are fully supervised, requiring access to ground truth depth data at training time. However, ground truth depth is challenging to acquire in varied real-world settings. There is a growing body of work that exploits weakly supervised training data in the form of known object sizes [19], sparse supervision [20,21], supervised appearance matching terms [12,22], or unpaired depth supervision [23], but they still require the collection of additional depth or other annotations. Synthetic data is a promising alternative [24], but it is not trivial to generate large amounts of synthetic data that includes all the variety of real-world appearance and motion.

Self-supervised Stereo Training
An alternative to depth based supervision is to use self-supervision in the form of synchronized stereo pairs. The key observation is to pose depth estimation as an image reconstruction problem at training time. A model that can perform monocular depth prediction at test-time can be trained by predicting the warping function that maps one image from a stereo pair onto the other. With known focal length and camera baseline the predicted disparity map, i.e. the inverse depth map, can then be converted into scaled metric depth. This type of approach has also been used for novel view synthesis [25,4], where the estimated depth map is used to select pixels from the input views to generate novel viewpoints.

Initial work in this space used discretized representations [4] or simple image reconstruction losses [5] limiting the quality of the output depth maps. By including a more sophisticated reconstruction loss and a left-right depth consistency term [6] was able to generate results that were superior to existing supervised approaches at the time. This approach was extended by training with semi-supervised data [26,27] and by including additional temporal information [28,22].

Using rectified stereo data at training time has the benefit of reducing the depth estimation problem to that of a 1D disparity search problem. Importantly, when the stereo cameras are synchronized they are not affected by scene motion. However, occlusion and dis-occlusion are still an issue as parts of the scene may not be visible given a fixed camera baseline. Another challenge is that large corpora of stereo data are not as readily available compared to traditional monocular videos. In this work we show that with careful design choices training with only a single view can reach the performance of stereo training on existing datasets and results can be improved even further by including temporal information.

Self-supervised Monocular Training
A more unconstrained form of self-supervision is to use only monocular videos during training. Here, a similar type of image reconstruction loss as in the stereo supervision case is used. However, instead of using stereo pairs, neighboring temporal frames from the input video provide the training signal. The added challenge during training is that the depth estimation model has to also estimate the camera pose between the input frames in addition to coping with the motion of objects in the scene. Unlike the stereo training regime that can cope with static cameras, in the monocular training setting there must be some non-trivial camera motion between the input images or there will be no training signal.

In one of the first works that used monocular self-supervision, [7] trained a depth estimation network along with a separate pose estimation network. The goal of the pose network is to predict the relative camera transformation between each subsequent temporal image pair. This estimated camera pose is only needed at training time to help constrain the depth estimation network. To deal with non-rigid scene motion, an additional motion explanation mask was learned allowing the model to ignore specific regions that violated the rigid scene assumption when computing the image reconstruction loss during training. However, later iterations of their model available online disables this term and achieves superior performance. Inspired by [29], [30] proposed a more sophisticated motion model, where each image pixel is explained by a combination of multiple rigid transformations defined by multiple motion masks. However, this motion decomposition is not fully evaluated making it difficult to know its utility. [31] also decompose motion into rigid and non-rigid components using the projected depth and a predicted residual optical flow to explain object motion. This improves the final optical flow estimation, but the authors report no improvement when jointly training the residual flow estimator and the depth network. Thus, their depth estimation is not improved by this additional non-rigid motion term.

Recent approaches are beginning to close the performance gap between monocular and stereo based training supervision by making use of several different constraints during training. [32] constrained the output depth to be consistent with predicted surface normals. [33] used an approximate ICP based geometry matching loss to enforce temporal depth consistency. [34] observed that the commonly used depth smoothness term has a preference towards smaller depth maps making the training of these models more unstable. To overcome this limitation, they normalize the predicted depth maps before computing the smoothness term, re-

sulting in better performance. However, most of these methods do not explicitly deal with scene motion and as a result fail on moving objects during training.

We propose several architectural and loss innovations that simplify many of these recent approaches yet still produces superior performance. We show that object motion is a challenge for monocular based methods and can be alleviated by either ignoring those scenes during training or by the inclusion of stereo data at training time when available.

3 Method

Here we describe our monocular depth prediction network, which requires only a single color image at test time. We first review the key idea behind self-supervised training for monocular depth estimation, and then describe our depth and pose estimation networks and combined training loss.

3.1 Self-supervised Training

The key idea behind self-supervised depth estimation is to frame the learning problem as one of novel view-synthesis, essentially teaching the network to predict the appearance of a target image from the viewpoint of *another* image. By constraining the network to perform image synthesis using an intermediary variable, in our case depth or disparity, we can then extract out this interpretable depth from the model. This is an ill-posed problem as there are an extremely large number of possible wrong depths per pixel which can correctly reconstruct the novel view given a relative transformation between those two views. This is essentially the same problem faced by binocular and multi-view stereo methods. These methods typically address this ambiguity through enforcing smoothness in the depth maps by computing photo-consistency on patches and solving for the optimal depths for each pixel in a global optimization framework [35].

Like [5,6,7], we also formulate our problem as the minimization of a photometric reprojection error at training time. For a target color image I_t, multiple source views $I_{t'}$, and the relative rigid transformation between those views and the target view $T_{t\to t'}$ which minimizes the photometric reprojection error L_p we predict a dense depth map D_t such that

$$\underset{D_t}{\operatorname{argmin}}\, L_p, \tag{1}$$

$$\text{with} \quad L_p = \sum_{t'} pe(I_t, I_{t'\to t}), \tag{2}$$

$$\text{and} \quad I_{t'\to t} = I_{t'}\Big\langle proj(D_t, T_{t\to t'}, K)\Big\rangle. \tag{3}$$

Here pe is a photometric reconstruction error, e.g. the L1 or L2 distance in pixel space, $proj$ are the resulting 2d coordinates of the projected depths D_t in $I_{t'}$ and $\langle\rangle$ is the sampling operator. For simplicity of notation we assume the intrinsics K of all the views to be identical, they can however be different.

Following [36] we use bilinear sampling to sample the source images, which is *locally* sub-differentiable. This locality is a limitation which we overcome by making use of a multi-scale reconstruction approach, further improved by our *upsampled* multi-scale sampling.

Until now, we have assumed that we know the relative transformations $T_{t\to t'}$ between our target view I_t and source view $I_{t'}$. This is generally not the case for monocular sequences. However, if the target image and its source image are from a rectified stereo pair, the transformation between the pairs is purely horizontal. Stereo based training approaches like [5,6] make use of this constraint when training single frame depth estimation models. For more general monocular training, [7] showed that it is possible to train a second pose estimation network jointly with the depth estimation network. Where the goal is to predict the relative poses $T_{t\to t'}$ used in the projection function $proj$. Also solving for the camera transformations, in addition to the depth, our objective becomes

$$\underset{D_t, T_{t\to t'}}{\operatorname{argmin}} L_p. \tag{4}$$

3.2 Improved Monocular Depth Estimation

Here, we describe several improvements to existing self-supervised depth estimation models. We step through the details of our approach, working through the design decisions taken regarding pose, loss functions, and scale.

Pose Estimation

The majority of current state-of-the-art models for monocular depth estimation that use monocular training data employ a very similar architecture e.g. [7,33,31,34]. This involves a standard U-Net model [37] for the depth estimator and a separate pose estimation network, see Fig. 2. The pose estimation network, which is not necessary for depth estimation at test time, takes as input a sequence of two or more concatenated input frames and estimates the pose transformation between them. We argue that this base design is sub-optimal. Concatenating several input frames only makes learning harder as the training set remain finite in size but the dimensionality of the input data grows with the number of frames in the input sequence. Moreover, the pose estimation model has to learn the difficult task of structure-from-motion from a short ordered sequence, with the only supervision signal being from the reprojection error. Improvements have been proposed by [34] who use direct methods in combination with the estimated depth maps, and [33] who use a 3d alignment loss between the predicted depth maps in the input sequence to improve both the pose and depth estimation. These approaches build on the idea that the combination of the predicted monocular depth maps in the training sequence is a very strong geometric signal that can be used to better estimate the relative poses. However, both approaches involve a more complicated training procedure, resulting in only relatively minor test time improvements.

We instead propose a simple modification to the base architecture, that results in a significant improvement in depth accuracy as well as a reduction in the

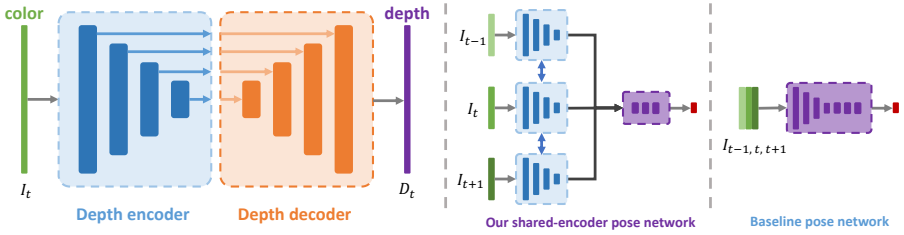

Fig. 2. Overview of our network architecture. Unlike existing approaches for monocular depth estimation that make use of a separate pose network (right), we share weights between our depth encoder (left) and pose network (middle), resulting in faster convergence and improved performance. The pose estimation network outputs a 6-dimensional vector for each source image (red block), representing the relative rigid transformation of the camera pose between frames.

number of parameters that need to be learned. We make the observation that the deepest features of our depth encoder are only a small number of convolutional layers away from producing depth. We thus concatenate the last features from our depth encoder, and feed them through a small three layer fully convolutional network i.e. the pose decoder, followed by a global average pooling, see Fig. 2 (middle). In effect, we are replacing the concatenation of the input images by the concatenation of the depth features. This results in the pose decoder receiving abstract features which have an intrinsic understanding of the geometry of each of the input images. Our pose decoder is identical to the last three layers of the standard pose network from [7], but produces significantly better results.

Appearance Matching Loss

When computing the reprojection error from multiple source images, existing self-supervised depth estimation methods use the average reprojection error described in Eqn. (4). This is problematic with pixels that are visible in the target image but are *not visible* in some of the source images. If the network predicts the correct depth for such a pixel, the corresponding color in an occluded source image will likely not match the target one, inducing a photometric error penalty. Such problematic pixels are from two categories: out-of-view pixels due to ego-motion at image boundaries and dis-occluded pixels. As we show in Fig. 5, using an average reprojection error typically results in black holes (infinite depth) around image edges and soft occlusion boundaries, for each category respectively. The effect of out-of-view pixels can be reduced by simply ignoring such pixels in the reprojection loss [33,30], this however doesn't handle dis-occluded pixels.

We propose an improvement which deals with both issues at once. Instead of averaging the photometric error per pixel over all source images, we simply use the per-pixel minimum. As shown in Fig. 5, this significantly reduces artifacts at image borders, improves the sharpness of occlusion boundaries, and leads to better accuracy. Following [38,6], we use a combination of L1 and SSIM [39] as

our photometric error function pe. Our final photometric loss is

$$L_p = \min_{t'} pe(I_t, I_{t' \to t}), \tag{5}$$

$$\text{where} \quad pe(I_a, I_b) = \alpha \frac{1 - \text{SSIM}(I_a, I_b)}{2} + (1 - \alpha)\|I_a - I_b\|_1. \tag{6}$$

We also make use of edge aware smoothness on the predicted disparities

$$L_s = |\partial_x d_t^*| \, e^{-|\partial_x I_t|} + |\partial_y d_t^*| \, e^{-|\partial_y I_t|}, \tag{7}$$

$$\text{with} \quad d_t^* = d_t / \overline{d_t}, \tag{8}$$

where d_t^* is the mean-normalized inverse depth for I_t as used by [34].

Multi-scale Estimation

Because of the gradient locality of the bilinear sampler, existing models use multi-scale depth prediction and image reconstruction to constrain the training objective, where the total loss is typically the average of the individual losses at each scale. The original formulations from [5] and [6] compute the photometric error on downsampled images, which we observe has the tendency to create 'holes' in large low-texture regions, such as roads or the sky, in the intermediate lower resolution depth maps. This can be explained by the lack of texture information at these resolutions, thus effectively making the photometric error more ambiguous. This in turn complicates the task of the depth estimation network which is then free to predict an incorrect depth for a given pixel at the low resolution resulting in a low reprojection error at that scale, which in turn leads to a large photometric error at higher resolutions.

We propose an improvement to this multi-scale formulation. Instead of computing the photometric error on the ambiguous low-resolution images, we first upsample the lower resolution depth maps to the input image resolution and then warp and compute the photometric error pe at this higher input resolution. This effectively constrains the depth maps from each resolution to work towards the exact same objective i.e. reconstructing the high resolution input target image as accurately as possible. We found that this significantly improves the depth accuracy, while also reducing the texture-copy artifacts which are very noticeable in the previous multi-scale formulation as can be seen in Fig. 5. This is related to matching patches, a standard practice in stereo reconstruction [40], as a low-resolution disparity value will be responsible for reprojecting an entire patch of pixels in the high resolution image.

Final Training Loss

We combine our photometric and smoothness terms into a final training loss

$$L = L_p + \lambda L_s, \tag{9}$$

which is averaged per pixel, per scale, and per batch.

3.3 Implementation Details

Our depth estimation network is based on the general U-Net architecture [37], which is essentially an encoder-decoder network, with skip connections enabling

us to represent both deep abstract features as well as local information. We use a Resnet18 [41], pretrained on ImageNet [42], as our encoder. Our depth decoder is similar to [6] and uses ELU [43] activation functions except on the output depth layers which use sigmoids. We then turn this output into depth by scaling and inverting the predicted disparities. We also make use of reflection padding, in place of zero padding, in the decoder layers, and return the value of the closest border pixels in the source image during reprojection, instead of zero, when samples land outside of the image boundaries. We found that these steps significantly reduce the border artifacts commonly found in existing approaches e.g. [6]. During training, we set the weight of the smoothness term, λ, to 0.001.

We adopt the inverse depth normalization trick of [34] in all our experiments to avoid catastrophic shrinking of the estimated depth. For pose estimation, we follow [34] and predict the rotation using an axis-angle representation and scale the rotation and translation outputs by 0.01 and 0.001 respectively. When training with stereo data we use the left and right pairs as the input views, for monocular training we use a set of three frames, $t-1$, t, and $t+1$. All our networks were implemented in PyTorch [44] and trained for 15 epochs, with a batch size of 8, using Adam [45], and with an initial learning rate of 10^{-4} for the first 10 epochs which was then dropped to 10^{-5}. Training on the KITTI dataset [8] with an input image size of 128×416 pixels, which we refer to as Low Resolution (LR), takes 8 hours on a Titan X Maxwell, and twice as long for 192×640 pixels. The output depth map resolution is the same resolution as the input image one.

We also tried several other components which we found to not help performance. Including using a feature reconstruction loss in the appearance matching term, as in [46,47,22], by computing the L1 distance on the reprojected and normalized `relu1_1` features from an ImageNet pretrained VGG16 [48] as our *pe* function, but observed a slight decrease in accuracy compared to SSIM on KITTI for a significant increase in training time. We explored using explanation masks [7], discrete motion models [30], and temporal depth consistency [30,28], none of which made any significant improvements in our implementation. Finally, we tried adding batch normalization [49] to the decoder but observed persistent ghosting artifacts in the predicted depth.

4 Experiments

In this section we compare the performance of our models to existing state-of-the-art on the KITTI [8] and Make3D [2] datasets. We also show qualitative results the Cityscapes [50] and a wandering video dataset which we compiled from YouTube.

4.1 KITTI

We use the original data split from Eigen et al. [51] and follow Zhou's et al. [7] preprocessing to remove static frames and set the input sequence length to 3.

This results in 39,810 and 4,424 monocular triplets for training and validation. We used the same camera intrinsics for all images, where we set the principal point of the camera to the image center and the focal length to the average of all the focal lengths in KITTI. For stereo and mixed training (monocular and stereo) we fix the transformation between the two stereo frames to be a pure horizontal translation of fixed length. We perform horizontal flips and the following data augmentations during training with 50% chance: random brightness, contrast, saturation, and hue jitter with respective ranges of ± 0.2, ± 0.2, ± 0.2 and ± 0.1. Results are presented using the standard cap of 80 meters. For our monocular models, we report results using the same median ground truth scaling as [7]. With stereo training data, scale can be inferred from the known camera baseline, and for fairness we do not use median scaling for our models that use any stereo supervision. In practice, we observe that this adds a small, but noticeable, improvement to the stereo models if included.

Results

We compare the results of several variants of our model trained with different types of self-supervision: monocular only, stereo only, and both. In Table 1 we see that we outperform all existing state-of-the-art approaches with the exception of models that make use of extensive depth supervision at training time i.e. [26,27]. Our best performing variant uses a combination of monocular and stereo training data where the results are most noticeable on metrics that are sensitive to large depth errors i.e. RMSE. We also see that our monocular supervised model, whether trained at 128×416 or 192×640, outperforms all previously published self-supervised methods, whether they used monocular supervision [7,33,34,32], stereo supervision [5,6] or both [28,22]. Qualitative results can be seen in Fig. 3.

To better understand how each component of our model contributes to the overall performance, in Table 2 we perform an ablation study by turning off different parts of our model, one at a time, in the monocular setting. 'PoseCNN' corresponds to our implementation of the strong baseline used in [34], with the standard separate pose encoder from [7], but without their direct visual odometry. We see that the inclusion of our shared pose encoder in 'Ours LR' improves the results. 'Avg. projection' is the average projection used by [7], as opposed to our minimum based projection from Eqn. 5. 'Low-res multi-scale' is the multi-scale reconstruction evaluation performed by [6], in contrast to our reconstruction which is performed at the input resolution. 'No pretraining' is our model without using pretrained on ImageNet weights. As previously discussed in [33], we see that SSIM plays an important role in improving results. When combined in 'Ours LR', all these components lead to a significant improvement.

Monocular vs. Stereo Supervision

From Table 1, we see that monocular trained models perform worse than stereo based approaches, even when including the significant boost provided by median scaling at test time. Moving objects are an issue for these approaches, but the KITTI dataset does not feature a large number of such objects. This problem

Table 1. Comparison to existing methods on KITTI 2015 [8] using the Eigen split. D refers to methods that use KITTI depth supervision at training time, D* use auxiliary depth supervision, S use stereo, and M use monocular supervision. †indicates newer results from the respective online implementations. LR is our model trained to predict at 128×416 resolution, otherwise we use 192×640.

Method	Train	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Train set mean	D	0.361	4.826	8.102	0.377	0.638	0.804	0.894
Eigen [3]	D	0.203	1.548	6.307	0.282	0.702	0.890	0.890
Liu [52]	D	0.201	1.584	6.471	0.273	0.680	0.898	0.967
AdaDepth [23]	D*	0.167	1.257	5.578	0.237	0.771	0.922	0.971
Kuznietsov [26]	DS	0.113	0.741	4.621	0.189	0.862	0.960	**0.986**
SVSM [27]	D*S	0.102	0.700	4.681	0.200	0.872	0.954	0.978
SVSM FT [27]	DS	**0.094**	**0.626**	**4.252**	**0.177**	**0.891**	**0.965**	0.984
UnDeepVO [28]	MS	0.183	1.730	6.57	0.268	-	-	-
Zhan Temporal [22]	MS	0.144	1.391	5.869	0.241	0.803	0.928	0.969
Zhan FullNYU [22]	D*MS	0.135	1.132	5.585	0.229	0.820	0.933	0.971
Ours LR	MS	0.124	1.148	5.273	0.213	0.849	0.947	0.975
Ours	MS	**0.114**	**0.991**	**5.029**	**0.203**	**0.864**	**0.951**	**0.978**
Monodepth [6]	S	0.148	1.344	5.927	0.247	0.803	0.922	0.964
Garg [5]†	S	0.152	1.226	5.849	0.246	0.784	0.921	0.967
Ours LR	S	0.122	1.041	5.304	0.218	0.847	0.943	0.973
Ours	S	**0.115**	**1.010**	**5.164**	**0.212**	**0.858**	**0.946**	**0.974**
Zhou [7]†	M	0.183	1.595	6.709	0.270	0.734	0.902	0.959
Yang [32]	M	0.182	1.481	6.501	0.267	0.725	0.906	0.963
Mahjourian [33]	M	0.163	1.240	6.220	0.250	0.762	0.916	0.968
GeoNet [31]	M	0.155	1.296	5.857	0.233	0.793	0.931	0.973
DDVO [34]	M	0.151	1.257	5.583	0.228	0.810	0.936	0.974
Ours LR	M	0.133	1.158	5.370	0.208	0.841	0.949	**0.978**
Ours	M	**0.129**	**1.112**	**5.180**	**0.205**	**0.851**	**0.952**	**0.978**

commonly manifests itself as 'holes' in the predicted test time depth maps for objects that are typically observed to be moving during training e.g. the missing car for the monocular methods in Fig. 4. If the moving object has the same speed and direction as the camera, then the reprojection error is low if the disparity is 0, i.e. a depth of infinity, for that object. In KITTI, this can happen when the camera is following a moving car in the same lane at the same speed. Unfortunately, monocular video alone is not sufficient to disambiguate pixels in the 'car following' scenario.

Solely to explore this hypothesis, we trained another version of the 'Ours LR' model on a subset of the KITTI dataset, where we manually removed six entire sequences that featured a moving car in front of the main camera. This resulted in 37,294 training images. In Table 2, we see this model, denoted as 'No motion' performs better on the Sq Rel and RMSE metrics, despite having less training data. Further, we observe that pretraining on Cityscapes [50], which was shown to be very useful in the case of stereo training [6], actually hurts in our monocular setting. We hypothesize that the increased proportion of motion in Cityscapes "helps" the network learn unrealistic depths that actually reproject

Table 2. Results for different variants of our model that use monocular training on KITTI 2015 [8] using Eigen's split. For training, C is Cityscapes [50] and K is the KITTI [8]. All models are trained using a resolution of 128×416.

Method	Train	Dataset	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
PoseCNN	M	K	0.147	1.227	5.653	0.219	0.811	0.943	0.977
Avg. reprojection	M	K	0.149	1.729	5.679	0.229	0.832	0.941	0.973
Low-res multi-scale	M	K	0.181	2.975	6.198	0.253	0.802	0.929	0.966
No pretraining	M	K	0.154	1.218	5.699	0.231	0.798	0.932	0.973
No SSIM	M	K	0.185	3.029	6.186	0.258	0.796	0.927	0.965
Ours LR	M	K	**0.133**	**1.158**	**5.370**	**0.208**	**0.841**	**0.949**	**0.978**
Ours LR	M	C	0.233	3.533	7.412	0.292	0.700	0.892	0.953
Ours LR	M	CK	0.138	1.430	5.609	0.215	0.843	0.948	0.975
Ours LR No motion	M	K*	0.134	1.075	5.242	0.208	0.842	0.949	0.978

correctly. For example, cars moving at the same speed as the camera are mapped to a distance of infinity.

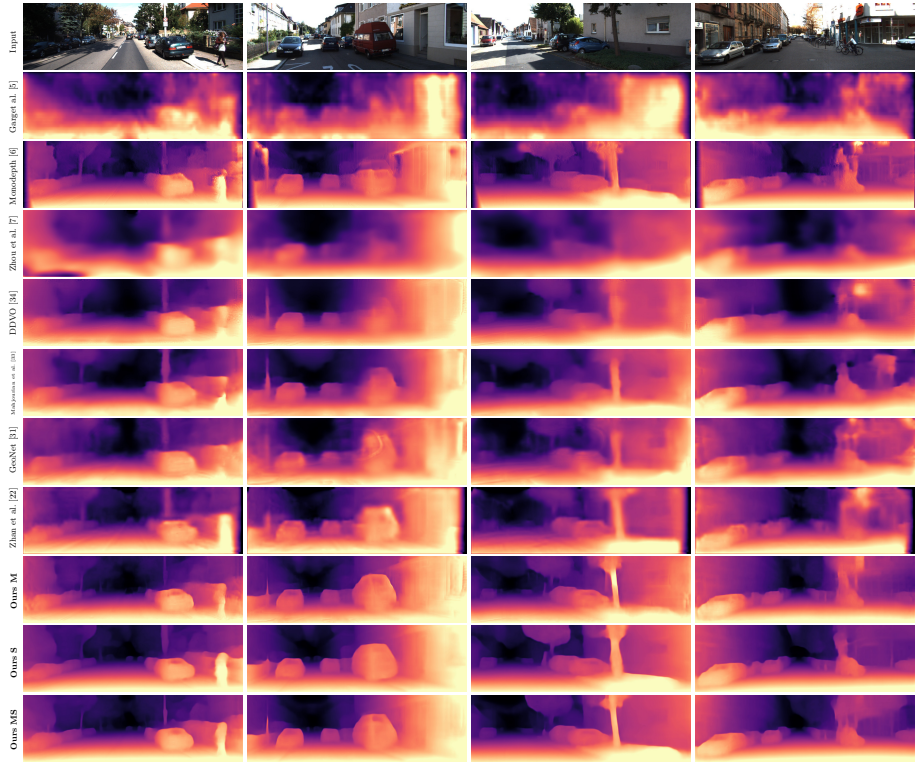

Fig. 3. Qualitative results on the KITTI Eigen split. We can see that our approaches in the last three rows produce the sharpest depth maps, which is reflected by the quantitative results in Table 1.

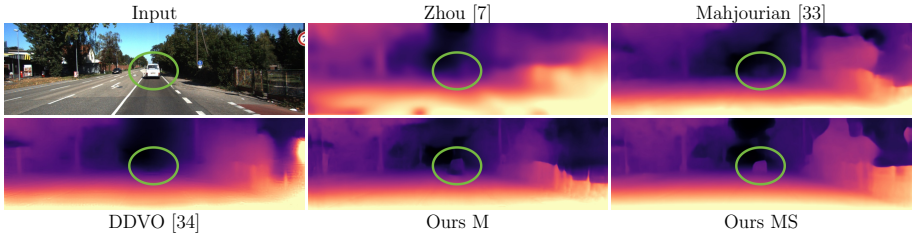

Fig. 4. Monocular failures. Monocular methods can fail at predicting depth for objects that were typically observed to be in motion during training e.g. moving cars.

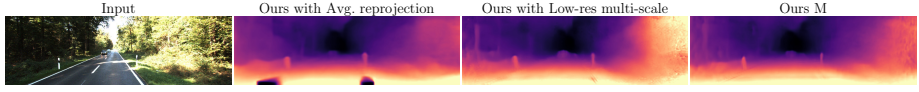

Fig. 5. Comparison with 'Low-res multi-scale' and 'Avg. reprojection'. We see that our model results in less artifacts in the final depth image.

4.2 Additional Results

Here we present quantitative results on the Make3D dataset [2] using our model trained on KITTI monocular data. In Table 3 we outperform all methods that do not use depth supervision. However, caution should be taken with Make3D, as the ground truth depth and input images are not well aligned in the original dataset, causing potential evaluation issues. Qualitative results can be seen in Figs. 6 and 7 and our supplementary material.

Table 3. Make3D results.

Method	Type	Abs Rel	Sq Rel	RMSE	$\log_{10}$
Train set mean	D	0.893	13.98	12.27	0.307
Karsch [10]	D	0.428	5.079	8.389	0.149
Liu [53]	D	0.475	6.562	10.05	0.165
Laina [11]	D	**0.204**	**1.840**	**5.683**	**0.084**
Monodepth [6]	S	0.544	10.94	11.760	0.193
Zhou [7]	M	0.383	5.321	10.470	0.478
DDVO [34]	M	0.387	4.720	8.090	0.204
Ours	M	**0.361**	**4.170**	**7.821**	**0.175**

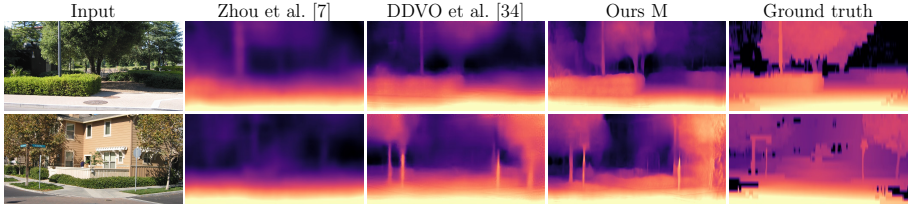

Fig. 6. Qualitative results on the Make3D dataset. All methods were trained on KITTI using monocular supervision.

Fig. 7. Qualitative results on the Cityscapes and Wander datasets. VGG feature loss was used for Wander as it produced more complete, yet blurrier depth maps.

4.3 Discussion

Across the different self-supervised settings we studied, we observed that the following design decisions are important for realizing better quality depth estimation models: (1) Independent object motion is very challenging for monocular methods. This can be substantially mitigated by excluding frames with significant motion from training, if possible, or by including them through the use of stereo pairs, where available. However it is clear that an explicit handling of object motion is required to truly exploit general monocular sequences. (2) Higher resolution leads to increased performance. This is observed at both the final depth resolution, and during multi-scale image reconstruction. (3) When predicting camera pose, it is beneficial to use shared weights between the depth and pose networks. This results in more stable training, fewer parameters to learn, and better performance. (4) Compared to the averaging baseline, a minimum reprojection loss is a simple and elegant solution to deal with occluded pixels. (5) Pretraining encoders on general image recognition tasks enables faster convergence, and results in better accuracy for depth estimation.

5 Conclusions

We presented a versatile model for self-supervised monocular depth estimation. We showed that with some careful, but non-obvious design choices, our model can outperform the existing state-of-the-art depth estimation algorithms, whether they leverage self-supervision with monocular training data, stereo training data, or both. We observed that models that are trained with stereo images still outperform those that use only monocular videos. Existing datasets like KITTI slightly conceal motion-induced limitations, due to the relatively small number of moving objects at training time. We expect scene-flow based performance differences to become more apparent as the community moves to more complex and general-world training imagery.

Acknowledgements We would like to thank Chaoyang Wang, Luo Yue, Reza Mahjourian, Zhichao Yin and Huangying Zhan for sharing their KITTI results with us. We would also like to thank Peter Hedman for sharing his stereo tricks.

References

1. Jiang, H., Learned-Miller, E., Larsson, G., Maire, M., Shakhnarovich, G.: Self-supervised depth learning for urban scene understanding. arXiv preprint arXiv:1712.04850 (2017)
2. Saxena, A., Sun, M., Ng, A.Y.: Make3d: Learning 3d scene structure from a single still image. PAMI (2009)
3. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: NIPS. (2014)
4. Xie, J., Girshick, R., Farhadi, A.: Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In: ECCV. (2016)
5. Garg, R., Kumar BG, V., Reid, I.: Unsupervised CNN for single view depth estimation: Geometry to the rescue. In: ECCV. (2016)
6. Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: CVPR. (2017)
7. Zhou, T., Brown, M., Snavely, N., Lowe, D.: Unsupervised learning of depth and ego-motion from video. In: CVPR. (2017)
8. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In: CVPR. (2012)
9. Hoiem, D., Efros, A.A., Hebert, M.: Automatic photo pop-up. TOG (2005)
10. Karsch, K., Liu, C., Kang, S.B.: Depth transfer: Depth extraction from video using non-parametric sampling. PAMI (2014)
11. Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., Navab, N.: Deeper depth prediction with fully convolutional residual networks. In: 3DV. (2016)
12. Žbontar, J., LeCun, Y.: Stereo matching by training a convolutional neural network to compare image patches. JMLR (2016)
13. Mayer, N., Ilg, E., Häusser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In: CVPR. (2016)
14. Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., Brox, T.: Demon: Depth and motion network for learning monocular stereo. In: CVPR. (2017)
15. Kendall, A., Martirosyan, H., Dasgupta, S., Henry, P., Kennedy, R., Bachrach, A., Bry, A.: End-to-end learning of geometry and context for deep stereo regression. ICCV (2017)
16. Fischer, P., Dosovitskiy, A., Ilg, E., Häusser, P., Hazırbaş, C., Golkov, V., van der Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: ICCV. (2015)
17. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: Flownet 2.0: Evolution of optical flow estimation with deep networks. In: CVPR. (2017)
18. Wang, Y., Yang, Y., Yang, Z., Zhao, L., Xu, W.: Occlusion aware unsupervised learning of optical flow. arXiv preprint arXiv:1711.05890 (2017)
19. Wu, Y., Ying, S., Zheng, L.: Size-to-depth: A new perspective for single image depth estimation. arXiv preprint arXiv:1801.04461 (2018)
20. Zoran, D., Isola, P., Krishnan, D., Freeman, W.T.: Learning ordinal relationships for mid-level vision. In: ICCV. (2015)
21. Chen, W., Fu, Z., Yang, D., Deng, J.: Single-image depth perception in the wild. In: NIPS. (2016)
22. Zhan, H., Garg, R., Weerasekera, C.S., Li, K., Agarwal, H., Reid, I.: Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In: CVPR. (2018)

23. Nath Kundu, J., Krishna Uppala, P., Pahuja, A., Babu, R.V.: Adadepth: Unsupervised content congruent adaptation for depth estimation. arXiv preprint arXiv:1803.01599 (2018)
24. Mayer, N., Ilg, E., Fischer, P., Hazirbas, C., Cremers, D., Dosovitskiy, A., Brox, T.: What makes good synthetic training data for learning disparity and optical flow estimation? IJCV (2018)
25. Flynn, J., Neulander, I., Philbin, J., Snavely, N.: Deepstereo: Learning to predict new views from the world's imagery. In: CVPR. (2016)
26. Kuznietsov, Y., Stückler, J., Leibe, B.: Semi-supervised deep learning for monocular depth map prediction. In: CVPR. (2017)
27. Luo, Y., Ren, J., Lin, M., Pang, J., Sun, W., Li, H., Lin, L.: Single view stereo matching. In: CVPR. (2018)
28. Li, R., Wang, S., Long, Z., Gu, D.: Undeepvo: Monocular visual odometry through unsupervised deep learning. arXiv preprint arXiv:1709.06841 (2017)
29. Byravan, A., Fox, D.: Se3-nets: Learning rigid body motion using deep neural networks. In: ICRA. (2017)
30. Vijayanarasimhan, S., Ricco, S., Schmid, C., Sukthankar, R., Fragkiadaki, K.: Sfm-net: Learning of structure and motion from video. arXiv preprint arXiv:1704.07804 (2017)
31. Yin, Z., Shi, J.: Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In: CVPR. (2018)
32. Yang, Z., Wang, P., Xu, W., Zhao, L., Nevatia, R.: Unsupervised learning of geometry with edge-aware depth-normal consistency. arXiv preprint arXiv:1711.03665 (2017)
33. Mahjourian, R., Wicke, M., Angelova, A.: Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. CVPR (2018)
34. Wang, C., Buenaposada, J.M., Zhu, R., Lucey, S.: Learning depth from monocular videos using direct methods. CVPR (2018)
35. Furukawa, Y., Hernández, C.: Multi-view stereo: A tutorial. Foundations and Trends in Computer Graphics and Vision (2015)
36. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial transformer networks. In: NIPS. (2015)
37. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. (2015)
38. Zhao, H., Gallo, O., Frosio, I., Kautz, J.: Loss functions for image restoration with neural networks. Transactions on Computational Imaging (2017)
39. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. TIP (2004)
40. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. IJCV (2002)
41. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. (2016)
42. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. IJCV (2015)
43. Clevert, D.A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289 (2015)
44. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. (2017)

45. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
46. Ren, Z., Yan, J., Ni, B., Liu, B., Yang, X., Zha, H.: Unsupervised deep learning for optical flow estimation. In: AAAI. (2017)
47. Sun, D., Yang, X., Liu, M.Y., Kautz, J.: Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. arXiv preprint arXiv:1709.02371 (2017)
48. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR. (2015)
49. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)
50. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: CVPR. (2016)
51. Eigen, D., Fergus, R.: Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In: ICCV. (2015)
52. Liu, F., Shen, C., Lin, G., Reid, I.: Learning depth from single monocular images using deep convolutional neural fields. PAMI (2015)
53. Liu, M., Salzmann, M., He, X.: Discrete-continuous depth estimation from a single image. In: CVPR. (2014)

Digging Into Self-Supervised Monocular Depth Estimation - Supplementary Material

1 Effect of moving cars

Cars moving at the same speed as the camera, or close to it, can appear seemingly static for many consecutive frames - masquerading as if they were extremely far away objects. This often causes a monocular video based network to assign a very large depth to such cars, to reach a low reprojection error. These incorrect predictions can be seen at test time, as shown in Fig. 1. This problem is amplified by pretraining on Cityscapes data [5], as this dataset contains many more sequences with such ambiguous situations, where the camera is following similar-speed cars. As we can see in Fig. 1, our monocular model, pretrained on Cityscapes, makes more dramatic mistakes in certain specific situations: cars just ahead are interpreted as "punched" out depths.

While we can mitigate the problem by excluding such nearly-matched-speed cars when training on KITTI, we found a better overall solution: we found that training with both monocular and stereo supervision addresses the issue directly.

2 Network architecture

In Table 1 we describe the parameters of each layer used in our depth decoder and pose network. The depth decoder makes use of ELU non-linearities [6].

3 Single scale test time evaluation

Our approach, like all self-supervised baselines, has no guarantee of producing results with a metric scale. Nonetheless, we anticipate that there could be value in estimating depth-outputs that are, without special measures, consistent with each other over the length of a video clip.

To evaluate the stability of our depth estimation, we modified the evaluation protocol from [1] to scale (or align) the predicted depths with a single scalar per method, instead of a *different* scalar per test depth map. In [1], the authors independently scale each predicted depth map by the ratio of the median of the ground truth and predicted depth map - for each individual test image. This is in contrast to stereo based training where the scale is known and as a result no additional scaling is required during the evaluation e.g. [7, 8]. This per-image depth scaling hides unstable scale estimation in both depth and pose estimation and presents a best case scenario for the monocular training case i.e.

Table 1. Our network architecture Here **k** is the kernel size, **s** the stride, **chns** the number of output channels for each layer, **res** is the downscaling factor for each layer relative to the input image, and **input** corresponds to the input of each layer where ↑ is a 2× nearest-neighbor upsampling of the layer.

Depth Decoder					
layer	**k**	**s**	**chns**	**res**	**input**
upconv5	3	1	256	32	econv5
iconv5	3	1	256	16	↑upconv5, econv4
upconv4	3	1	128	16	iconv5
iconv4	3	1	128	8	↑upconv4, econv3
disp4	3	1	1	1	iconv4
upconv3	3	1	64	8	iconv4
iconv3	3	1	64	4	↑upconv3, econv2
disp3	3	1	1	1	iconv3
upconv2	3	1	32	4	iconv3
iconv2	3	1	32	2	↑upconv2, econv1
disp2	3	1	1	1	iconv2
upconv1	3	1	16	2	iconv2
iconv1	3	1	16	1	↑upconv1
disp1	3	1	1	1	iconv1

Shared-encoder pose network					
layer	**k**	**s**	**chns**	**res**	**input**
pconv0	1	1	256	32	econv5
pconv1	3	2	256	64	pconv0_{t-1}, pconv0_{t}, pconv0_{t+1}
pconv2	3	2	256	128	pconv1
pconv3	1	1	12	128	pconv3

if a method outputs wildly varying scales for each sequence, then this evaluation protocol will significantly hide the issue. We thus modified the original protocol to instead use a single scale for all predicted depth maps of each method. For each method, we compute this single scale by averaging all the individual ratios of the depth medians on the *test* set. While this is still not ideal as it makes use of the ground truth depth, we believe it to be a more fair and representative of the performance of each method. We also calculated the standard deviation σ_{scale} of the individual scales, where lower values indicate more consistent output depth map scales. As can be seen in Table 2, our method still outperforms all previously published self-supervised monocular methods, and is more stable.

Table 2. Comparison to existing monocular supervised methods on KITTI 2015 [9] using the Eigen split using a *single* scale for each method. † indicates newer results from the respective online implementations. Here we compare our monocular trained model, where LR is our model trained to predict at 128×416 resolution, otherwise we use 192×640.

Method	σ_{scale}	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Zhou [1]†	0.201	0.278	2.636	7.428	0.335	0.576	0.836	0.930
Mahjourian [3]	0.184	0.234	1.874	6.616	0.297	0.642	0.871	0.948
GeoNet [4]	0.167	0.216	1.778	6.389	0.277	0.681	0.890	0.957
DDVO [2]	0.104	0.167	1.408	5.770	0.243	0.778	0.923	0.970
Ours LR	0.100	0.149	1.259	5.525	0.221	0.810	0.943	0.976
Ours	**0.089**	**0.139**	**1.165**	**5.253**	**0.214**	**0.832**	**0.946**	**0.977**

4 Effect of pre-training

While pre-training provides an improvement in both convergence speed and final accuracy, our method isn't relying on it. As we can see in Table 3 our method still outperforms *all but one* [2] published state of the art methods.

Table 3. Even without pre-training, our model still outperforms or matches recent SoA methods at depth estimation (best two shown). The best result is bolded, second is underlined.

Method	Train	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
UnDeepVO [10]	MS	0.183	1.730	6.57	0.268	-	-	-
Zhan Temporal [22]	MS	0.144	1.391	5.869	0.241	0.803	0.928	0.969
Zhan FullNYU [22]	D*MS	0.135	1.132	5.585	0.229	0.820	0.933	0.971
Ours scratch **LR**	MS	0.137	1.155	5.660	0.234	0.810	0.930	0.970
Ours scratch	MS	0.135	1.161	5.470	0.229	0.821	0.934	0.971
Ours	MS	**0.114**	**0.991**	**5.029**	**0.203**	**0.864**	**0.951**	**0.978**
Garg [5]†	S	0.152	1.226	5.849	0.246	0.784	0.921	0.967
Monodepth R50 [6]	S	0.133	1.142	5.533	0.230	0.830	0.936	0.970
Ours scratch **LR**	S	0.140	1.300	5.649	0.235	0.818	0.930	0.968
Ours scratch	S	0.130	1.214	5.468	0.226	0.836	0.935	0.970
Ours	S	**0.111**	**1.012**	**5.127**	**0.209**	**0.861**	**0.947**	**0.975**
Zhou [1]†	M	0.183	1.595	6.709	0.270	0.734	0.902	0.959
Yang [11]	M	0.182	1.481	6.501	0.267	0.725	0.906	0.963
Mahjourian [3]	M	0.163	1.240	6.220	0.250	0.762	0.916	0.968
GeoNet [31]	M	0.155	1.296	5.857	0.233	0.793	0.931	0.973
DDVO [34]	M	0.151	1.257	5.583	0.228	0.810	0.936	0.974
Ours scratch **LR**	M	0.154	1.218	5.699	0.231	0.798	0.932	0.973
Ours scratch	M	0.154	1.270	5.610	0.229	0.803	0.933	0.972
Ours	M	**0.129**	**1.112**	**5.180**	**0.205**	**0.851**	**0.952**	**0.978**

5 Odometry

In Table 4 we evaluate our pose estimation network following Zhou et al. [1] evaluation protocol. We trained our monocular model with 3 frames on sequences 0-8 from the KITTI odometry split and tested on 9-10. The absolute trajectory error (ATE) is averaged over all overlapping 5-frame snippets in the test sequences. Here, unlike [1] and others, that use specific architectures for the odometry task, we use the *same* architecture for this task as our depth estimation network from Table 1, and simply train it again from scratch on these new sequences. In order to compare our 3-frame model, we only use one relative transformation $T_{t\to t-1}$ and combine the frame-to-frame estimates to form local trajectories. For completeness we repeat the same process with Zhou's [1] predicted poses, which we indicate with a $*$ in the table. As we can see in Table 4, our frame-to-frame poses are better than both [1] and the previous state-of-the-art for monocular depth estimation [2]. They however fall short of the independent 5-frame estimations from previous self-supervised methods i.e. [3, 4].

Table 4. Odometry results on the KITTI [9] odometry dataset. Results show the average absolute trajectory error, and standard deviation, in meters. †indicates newer results from the respective online implementations.

	Sequence 09	**Sequence 10**
ORB-Slam [12]	0.014±0.008	0.012±0.011
Zhou [1]	0.021±0.017	0.020±0.015
Zhou [1]†	0.016±0.009	0.013±0.009
Mahjourian [3]	0.013±0.010	0.012±0.011
GeoNet [4]	0.012±0.007	0.012±0.009
DDVO [2]	0.045±0.108	0.033±0.074
Zhou* [1]	0.050±0.039	0.034±0.028
Ours LR	0.023±0.013	0.018±0.014

Table 5. Comparison of the revised evaluation code on KITTI 2015 [9] using the Eigen split. D refers to methods that use KITTI depth supervision at training time, D* use auxiliary depth supervision, S use stereo, and M use monocular supervision. LR is our model trained to predict at 128×416 resolution, otherwise we use 192×640.

Method	Train	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Ours LR	MS	0.122	1.164	5.244	0.212	0.850	0.947	0.975
Ours	MS	0.114	0.991	5.029	0.203	0.864	0.951	0.978
Ours LR	S	0.118	1.044	5.264	0.216	0.849	0.944	0.974
Ours	S	0.111	1.012	5.127	0.209	0.861	0.947	0.975
Ours LR	M	0.137	1.153	5.353	0.212	0.836	0.947	0.978
Ours	M	0.133	1.111	5.182	0.209	0.845	0.950	0.977

6 Revised evaluation code

For the main paper we used the evaluation code from [8] which is also used by most subsequent work. We found that it uses an incorrect flag which made the depths to be computed with respect to the LIDAR instead of the cameras. Because most competing methods use this evaluation code, we evaluated without any changes. However for completeness we present our evaluation with this revised evaluation code in Table 5. In the first version of this paper on arXiv, we incorrectly stated that the main paper evaluation was computed using the revised evaluation code.

7 Convergence

In Fig. 2 we see that our model with the shared pose encoder converges faster, and to a better accuracy, compared to using a separate pose network. This test accuracy is not monitored during training as all our networks are trained for 15 epochs.

8 Additional qualitative comparisons

We include additional qualitative results from the KITTI test set in Fig. 3. We also include a **supplementary video** that shows qualitative results on KITTI, Cityscapes, and a dataset collected from Youtube that features an individual walking with a hand-held camera in a non-European environment. This last video is quite different from the car mounted cameras of KITTI and Cityscapes as it only features a monocular hand-held camera that we use for training i.e. there is no stereo data.

References

1. Zhou, T., Brown, M., Snavely, N., Lowe, D.: Unsupervised learning of depth and ego-motion from video. In: CVPR. (2017)
2. Wang, C., Buenaposada, J.M., Zhu, R., Lucey, S.: Learning depth from monocular videos using direct methods. CVPR (2018)
3. Mahjourian, R., Wicke, M., Angelova, A.: Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. CVPR (2018)
4. Yin, Z., Shi, J.: Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In: CVPR. (2018)
5. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: CVPR. (2016)
6. Clevert, D.A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elus). arXiv preprint arXiv:1511.07289 (2015)
7. Garg, R., Kumar BG, V., Reid, I.: Unsupervised CNN for single view depth estimation: Geometry to the rescue. In: ECCV. (2016)
8. Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: CVPR. (2017)
9. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In: CVPR. (2012)
10. Li, R., Wang, S., Long, Z., Gu, D.: Undeepvo: Monocular visual odometry through unsupervised deep learning. arXiv preprint arXiv:1709.06841 (2017)
11. Yang, Z., Wang, P., Xu, W., Zhao, L., Nevatia, R.: Unsupervised learning of geometry with edge-aware depth-normal consistency. arXiv preprint arXiv:1711.03665 (2017)
12. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: Orb-slam: a versatile and accurate monocular slam system. Transactions on Robotics (2015)
13. Zhan, H., Garg, R., Weerasekera, C.S., Li, K., Agarwal, H., Reid, I.: Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In: CVPR. (2018)

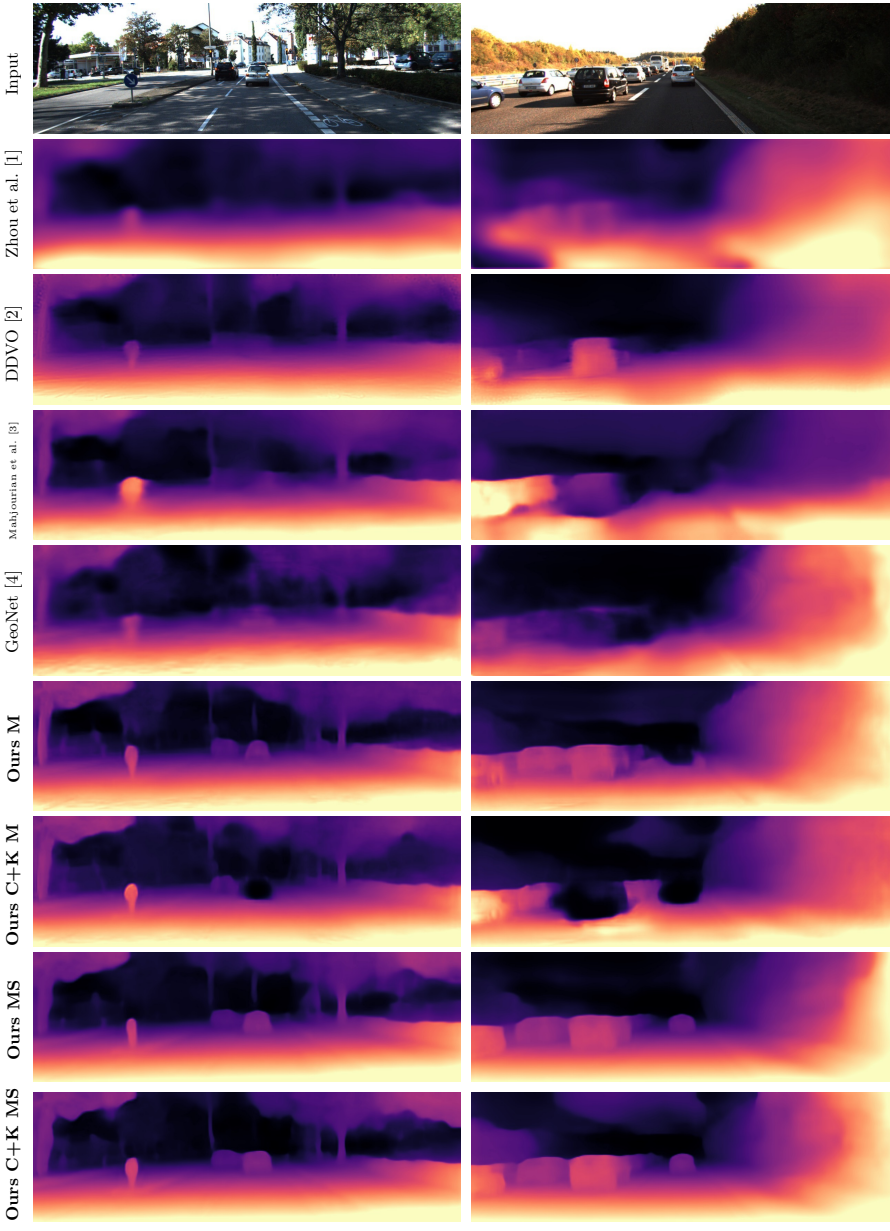

Fig. 1. Moving cars and Cityscapes pretraining. All monocular methods tend to put cars which are ahead of the camera at a very large depth value. Pretraining on Cityscapes [5], which has more moving cars in the training data compared to KITTI, only makes things worse especially for our method. Training with the addition of stereo data significantly reduces the impact of this problem.

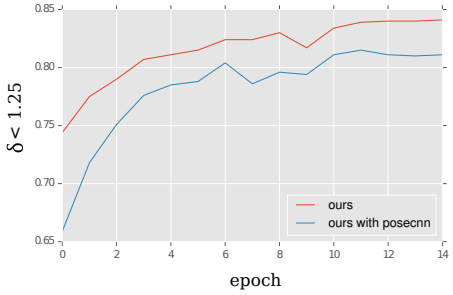

Fig. 2. Convergence comparison. Here we compare the depth prediction accuracy of our model with our shared pose network (ours) and our model with the separate pose network (ours with posecnn) using the $\delta < 1.25$ metric on the KITTI test set, where higher values are better.

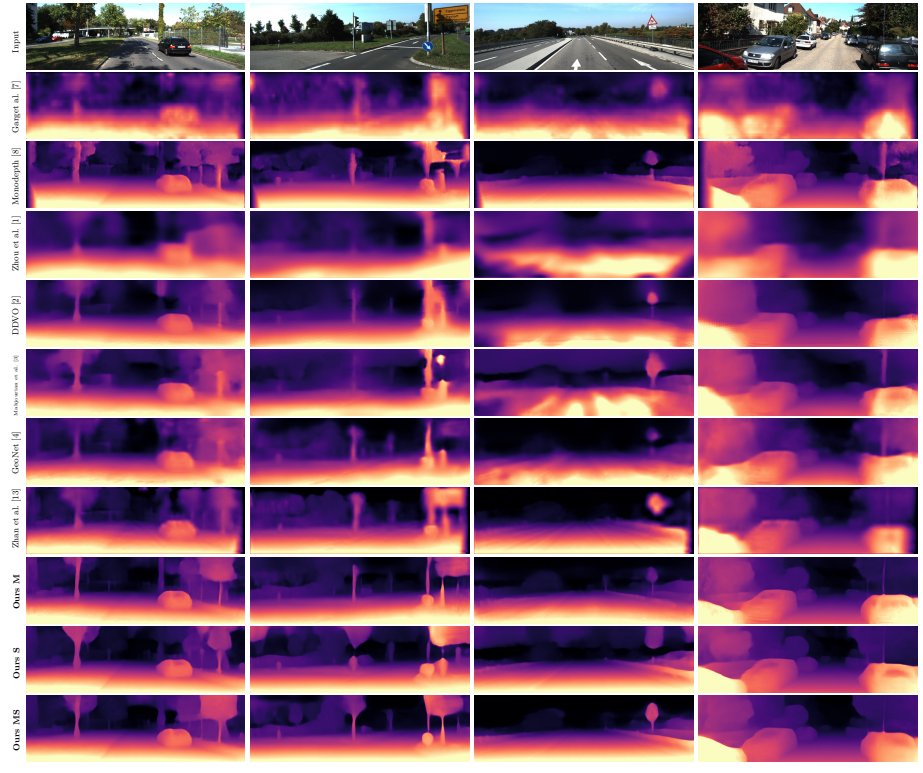

Fig. 3. Qualitative results on test images from the KITTI Eigen split. We can see that our approaches in the last three rows produce the sharpest depth maps.