Depth from Videos in the Wild: Unsupervised Monocular Depth Learning from Unknown Cameras

Ariel Gordon[1 2], Hanhan Li[2], Rico Jonschkowski[1 2], Anelia Angelova[1 2]
[1]Robotics at Google, [2]Google AI
{gariel, uniqueness, rjon, anelia}@google.com

Abstract

We present a novel method for simultaneously learning depth, egomotion, object motion, and camera intrinsics from monocular videos, using only consistency across neighboring video frames as a supervision signal. Similarly to prior work, our method learns by applying differentiable warping to frames and comparing the result to adjacent ones, but it provides several improvements: We address occlusions geometrically and differentiably, directly using the depth maps as predicted during training. We introduce randomized layer normalization, a novel regularizer, and we account for object motion relative to the scene. To the best of our knowledge, our work is the first to learn the camera intrinsic parameters, including lens distortion, from video in an unsupervised manner, thereby allowing us to extract accurate depth and motion from arbitrary videos of unknown origin at scale. We evaluate our results on the Cityscapes, KITTI, and EuRoC MAV datasets, establishing new state of the art on depth prediction and odometry, and demonstrate qualitatively that depth prediction can be learned from a collection of YouTube videos. The code is publicly available[1].

1. Introduction

Estimating 3D structure and camera motion from video is a key problem in computer vision. Traditional approaches to this problem rely on identifying the same points in the scene in multiple consecutive frames, then solving for a 3D structure and camera motion that is maximally consistent across those frames [23]. But such correspondences between frames can only be established for a subset of all pixels, which leaves the problem of estimating depth underdetermined. As commonly done with inverse problems, the gaps are filled based on assumptions of continuity, planarity, etc.

[1] github.com/google-research/google-research/tree/master/depth_from_video_in_the_wild

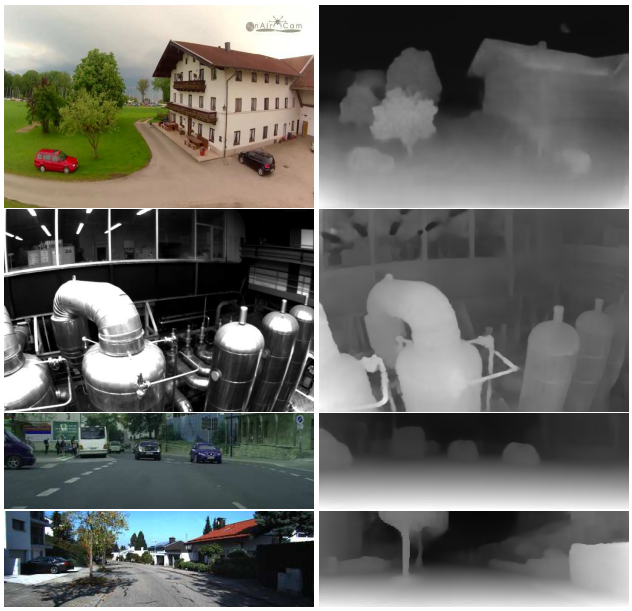

Figure 1. Qualitative results of our approach for learning depth from videos of unknown sources, which is enabled by simultaneously learning the camera extrinsic *and intrinsic* parameters. Since our method does not require knowing the camera parameters, it can be applied to any set of videos. All depth maps (visualized on the right, as disparity) were learned from raw videos without using any camera intrinsics groundtruth. From top to bottom: frames from YouTube8M [1], from EuRoC MAV dataset [5], from Cityscapes [7] and from KITTI [11].

Rather than specifying these assumptions manually, deep learning is able to obtain them from data. Wherever information is insufficient to resolve ambiguities, deep networks can produce depth maps and flow fields by generalizing from prior examples they have seen. Unsupervised approaches allow learning from raw videos alone, using similar consistency losses as traditional methods but optimizing them during training. At inference, the trained networks are able to predict depth from a single image and egomotion from pairs or longer sequences of images.

As research in this direction gained traction [47, 10, 12, 33, 24, 34], it became clear that object motion is a major obstacle because it violates the assumption of a static scene. Several directions have been proposed to address the issue [44, 40], including leveraging semantic understanding of the scene through instance segmentation [6]. Occlusions have been another limiting factor, and lastly, in all prior work in this direction, the intrinsic parameters of the camera had to be provided.

This work addresses the problems above and, as a result, reduces supervision and improves the quality of depth and motion prediction from unlabeled videos. First, we show that a deep network can be trained to predict the *intrinsic* parameters of the camera, including lens distortion, in an *unsupervised* manner from the video itself (see Fig. 1). Second, we are the first in this context to address occlusions directly, in a geometric way, from the predicted depth as it is. Lastly, we substantially reduce the amount of semantic understanding needed to address moving elements in the scene: Instead of segmenting every instance of a moving object and tracking it across frames [6], we need a single mask that covers pixels that *could* belong to a moving object. This mask can be as rough as a union of rectangular bounding boxes. Obtaining such a rough mask is a much simpler problem than instance segmentation and can be solved more reliably with existing models.

In addition to these qualitative advances, we conduct an extensive quantitative evaluation of our method and find that it establishes a new state of the art on multiple widely used benchmark datasets. Pooling datasets together, a capability which is greatly advanced by our method, proves to enhance quality. Finally, we demonstrate for the first time that depth and camera intrinsics prediction can be learned on YouTube videos, which were captured with multiple different cameras, each with unknown and generally different intrinsics.

2. Related work

Estimating scene depth is an important task for robot navigation and manipulation. Historically much research has been devoted to it, including a large bodies of research on stereo, multi-view geometry, and active sensing [29, 21, 9]. Recently, learning-based approaches for dense depth prediction have gained attention [9, 22, 19, 45]. In these, scene depth is predicted from input RGB images and the depth estimation function is learned using supervision provided by a sensor, such as a LiDAR. Similar approaches are used for other dense predictions such as surface normals [8, 38].

Unsupervised depth learning. Unsupervised learning of depth, where the only supervision is obtained from the monocular video itself and no depth sensors are needed, has also been popularized recently [47, 10, 12, 33, 24, 34, 44]. Garg et al. [10] introduced joint learning of depth and egomotion. Zhou et al. [47] demonstrated a fully differentiable approach where depth and egomotion are predicted jointly by deep neural networks. Techniques were developed for the monocular [33, 42, 24, 41, 44, 35, 6] and binocular [12, 33, 40, 46, 35, 46] settings. In the latter, it was shown that monocular depth quality at inference is improved when stereo inputs are used during training. Other methods learn directly the stereo disparity [17, 18, 43]. Other novel techniques include the use of motion [41, 35, 44, 6, 40].

Learning from images or videos from unknown cameras. This is an active research field, focusing on single or multi-view images [2, 30, 20]. It is especially hard for internet photos due to the diversity of input sources and lack of the camera parameters, as shown by Li et al. [20]. Our work makes a step in addressing this challenge by learning camera intrinsics for videos in the wild.

Occlusion aware learning. Multiple approaches disconnected from geometry have been proposed for handling occlusions in the context of optical flow [36, 15, 25]. Differentiable mesh rendering [26, 16] adopts a geometric approach to occlusions. In the context of learning to predict depth and egomotion, occlusions were addressed via a learned explainability mask [47], by penalizing the minimum reprojection loss between the previous frame or the next frame into the middle one, and through optical flow [40]. In the latter context, we are the first to address occlusions in a direct geometric approach via a differentiable loss.

Learning of intrinsics. Learning to predict the camera intrinsics has mostly been limited to strongly supervised approaches. The sources of groundtruth vary: Workman et al. [37] use focal lengths estimated employing classical 1D structure from motion. Yan et al. [39] obtain the focal length based on EXIF. Bogdan et al. [4] synthesize images from panoramas using virtual cameras with known intrinsics, including distortion. To our knowledge, our approach is the only one that learns the camera intrinsics in an unsupervised manner directly from video, jointly with depth, egomotion and object motion.

3. Preliminaries

Similarly to prior work [47, 12, 44, 32], the backbone of our method is the equation that ties together two adjacent video frames using a depth map (z) and the camera matrix K. Eq. 1 describes the shift in a pixel position p due to a rotation matrix R and a translation vector t:

$$z'p' = KRK^{-1}zp + Kt \tag{1}$$

p' and z' are the new homogeneous coordinates of the pixel and the new depth.

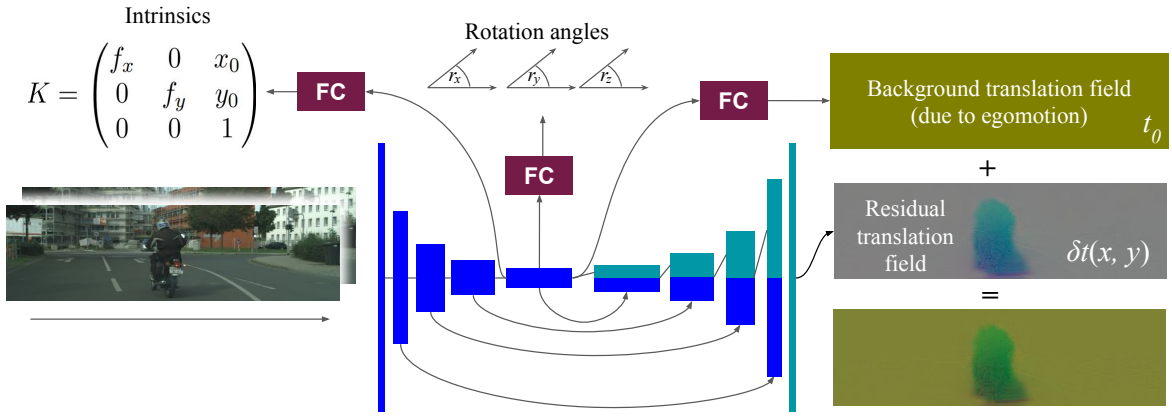

Figure 2. A schematic of our motion-prediction network. The network receives two images as input. A stack of convolutions creates a bottleneck, form which fully-connected (FC) network heads predict the intrinsics, rotation angles and translation vector components of the background (due to egomotion). A series of decoder layers predict a residual translation field, which predicts motion of objects with respect to the scene (color coding is explained in Fig. 4. A mask (not shown in the figure, see Eq. 2) multiplies by zero the residual translation field at all pixels that don't belong to possibly-moving objects. The residual translation field is then added to the background translation to obtain a total translation field. The images are taken from Cityscapes. A separate network (now shown) predicts depth from a single image.

Using z, R, and t as predicted by deep networks, Eq. 1 is used to warp one video frame onto the other. The result is then compared to the actual other frame, and the differences constitute the main component of the training loss. The premise is that through penalizing the differences, the networks will learn to correctly predict z, R, and t.

4. Method

In this work we propose simultaneous learning of depth, egomotion, object motion, and camera intrinsics from monocular videos. A motion-prediction network predicts camera motion, motion of every pixel with respect to the background, and the camera intrinsics: focal lengths, offsets and distortion. A second network predicts depth maps. By imposing consistency across neighboring frames as a loss, the networks simultaneously learn to predict depth maps, motion fields and the camera intrinsics. To apply this loss only in unoccluded pixels, we estimate occlusions geometrically based on estimated depth maps. We regularize motion fields based on masks that indicate which pixels might belong to moving objects, obtained from a pretrained segmentation or object detection network.

4.1. Networks and losses

Networks Depth is predicted form *a single image* by a UNet [28] encoder-decoder network with a ResNet 18 base and a softplus activation ($z(\ell) = \log(1 + \mathrm{e}^{\ell})$) to convert the logits (ℓ) to depth (z).

A second network (shown in Fig. 2) predicts camera motion, a dense residual translation representing motion of objects relative to the scene, as well as the camera intrinsics, from *two consecutive images*. Further details about the network are given in the Supplementary Material (SM).

Losses Based on the estimated depth map, camera intrinsics, rotation, and the translation field, we warp the first frame to match the second one and compare those using two losses: 1) a structural similarity (SSIM) loss and 2) the sum of $L1$ distances for the color channels, following Casser et al. [6]. Additionally, we impose a cycle consistency loss on the motion field by estimating both forward and backward motion, which we obtain by applying the networks on the frames in normal and in reversed order. Since those motion estimates at corresponding pixels should be opposite, we define an L2 loss on the relative deviation from opposite rotation and translation. Additionally, we apply spatial L1 smoothness losses for the depth and motion fields, a temporal L1 smoothness loss for depth, and an L2 weight regularization term. More details are given in the SM.

4.2. Occlusion-aware consistency

When the camera and / or objects move, areas in the scene that were visible in one frame may become occluded in another, and vice versa. Photometric consistency cannot be enforced in pixels that correspond to these areas. Given a depth map and a motion field in one frame, one could actually detect where occlusion is about to occur, and exclude the occluded areas from the consistency loss. Detecting occluded pixels requires some sort of reasoning about the connectivity of the surface represented by the depth map, and z-buffering. Keeping the mechanism differentiable and efficient enough for a training loop, may pose a challenge.

We therefore take a different approach, as illustrated in Fig. 3. For each pixel (i, j) in the source frame, the pre-

dicted depth z_{ij} and the camera intrinsic matrix are used to obtain the respective point in space, (x_{ij}, y_{ij}, z_{ij}). The point is moved in space according to the predicted motion field. In particular, the depth changes to z'. The new spatial location is reprojected back onto the camera frame, and falls at some generally-different location (i', j') on the target frame. i' and j' are generally non-integer. Therefore obtaining the depth on the target frame at (i', j'), $z^t_{i',j'}$, requires interpolation.

Occlusions happen at (i', j') where z' becomes multivalued. At such points, color and depth consistency should be applied only to the visible branch of z', that is, the branch where z' is smaller. If the source and target frames are nearly consistent, the visible branch will be close to target depth at (i', j'), $z^t_{i',j'}$. The way we propose to pick the visible branch is to include in the losses only points (i', j') where $z'_{i',j'} \leq z^t_{i',j'}$. In other words, only if a transformed pixel on the source frame lands in front of the depth map in the target frame, do we include that pixel in the loss. This scheme is not symmetrical with respect to interchanging the source and target frames, which is why we always apply it in a symmetrized way: We transform the source onto the tagret, calculate the losses, and then switch the roles of source and target. Fig. 3 illustrates the method.

The losses described in Sec. 4.1 are invoked in an "occlusion-aware" manner, as described in this section, except for SSIM. For the latter, we handle occlusions by replacing all averaging operations by a weighted averaging, where the weight of a pixel is a decreasing function of the depth error in that pixel. The exact expression is given in the SM.

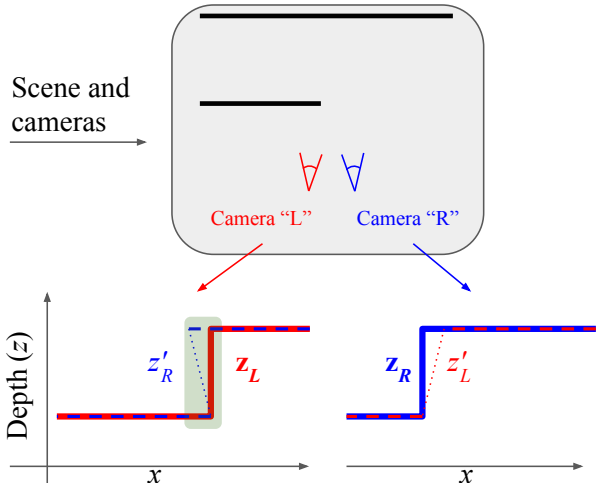

Figure 3. An illustration of our proposed method for handling occlusions. At the top we show a two-dimensional "scene", consisting of two straight surfaces, one partially occluding the other. Two cameras, left ("L") and right ("R"), are observing the scene. Our method is *monocular*, so these represent two locations of the same camera that moved, and "left" and "right" are used for convenience. At the bottom, the depth map observed by each camera is illustrated as a solid line on the respective side ($\mathbf{z}_L$ and $\mathbf{z}_R$). A dashed line shows the depth map obtained from warping one view onto the other (z'_R and z'_L). The warped depth map can become a multivalued function, which indicates occlusions (see the green-shaded rectangle). To handle that, we apply the photometric and geometric losses only at pixels where $z'_R \leq \mathbf{z}_L$ and $z'_L \leq \mathbf{z}_R$. When the depth maps and motion estimation are correct, the loss in this scheme would indeed evaluate to zero.

4.3. Regularization

Semantic regularization of the translation field Eq. 1 can propagate frame inconsistency losses to z, R and t at every pixel. However without further regularization, they remain greatly underdetermined. While continuity of z, R and t is a powerful regularizer, we found that further regularization helps significantly. In particular, we impose constancy of R throughout the image, and allow t to deviate from a constant value only at pixels that are designated as *possibly mobile*. Unlike in prior work [6], instance segmentation and tracking are not required, as all we need is a single "possibly mobile" mask $m(x, y)$. We write

$$t(x, y) = t_0 + m(x, y)\delta t(x, y), \tag{2}$$

where t_0 and $\delta t(x, y)$ are the background motion (due to camera motion) and residual motion (due to object motion).

We show in ablation experiments that $m(x, y)$ can be as rough as a union of bounding boxes (see Fig. 4). In addition, an $L1$ smoothing operator is applied to $t(x, y)$.

Randomized layer normalization In our experiments, we observed the following anomalous behavior in relation to Batch Normalization (BN):

- Evaluation metrics were consistently better when running inference at the "training mode" of BN. That is, instead of long-term averaged means and variances, the ones obtained from the image itself during inference were used[2], rendering batch normalization more similar to layer normalization [3].
- As we increased the batch size at training, the eval accuracy was consistently worse and worse, no matter how we scaled the learning rate.

These two anomalies led to the conclusion that BN is acting like Layer Normalization (LN) [3], and for each item in the batch, the others act as a source of noise. Replacing BN by LN with multiplicative Gaussian noise led to improved evaluation metrics and allowed increasing the batch size (accompanied with a linear increase of the learning rate [14]) without loss and even with a slight improvement in the evaluation metrics.

[2]Even at batch size 1, there would still be means and variances over the spatial dimensions.

Figure 4. Use of a "possibly mobile" mask to regularize the translation field. An object detection network identifies all instances of objects that are capable of motion, such as pedestrians, cyclists and cars. The union of the bounding boxes comprises the "possibly mobile" mask, within which the translation field is allowed to vary. The top picture, from Cityscapes, illustrates the mask, and the bottom one is the translation field predicted by the network (x, y, z coded as RGB). The **golden background** corresponds to motion in the negative z direction, as the entire scene is moving towards the camera. The **greenish silhouette** is the cyclist moving slightly to the left and slightly towards the camera. Note that the network carves the silhouette out of a rough mask.

4.4. Learning the intrinsics

Through Eq. 1, photometric consistency losses across neighboring frames provide a supervision signal for K, but only when there is a nonzero camera rotation between them $R \neq \mathbb{1}$. Indeed, when $R = \mathbb{1}$ Eq. 1 is reduced to $z'p' = zp + Kt$, which means that the loss depends on K only through the product Kt. Kt, however, can be perfectly correct even if K and t are incorrect. In fact, for any (non-singular) $\tilde{K}$, there exists a $\tilde{t} = \tilde{K}^{-1}Kt$, such that $\tilde{K}\tilde{t} = Kt$. Especially when K and t are predicted by two heads stemming from the same network, the latter can "escape" the supervision signal as long as it predicts a K and a t whose *product* is correct.

Fortunately, rotations can provide a supervision signal for K. Eq. 3 (derived in the SM) ties the tolerance with which the focal lengths can be determined (δf_x and δf_y, denominated in pixels) to the amount of camera rotation that occurs between the two:

$$\delta f_x < \frac{2f_x^2}{wsr_y}; \quad \delta f_y < \frac{2f_y^2}{hsr_x}. \tag{3}$$

r_y and r_x are the rotation angles along the respective axes in radians, w and h are the image width and height respectively, and $s = \max(h, w)$.

5. Experiments

In this section, we evaluate our method on depth prediction, odometry estimation, and the recovery of camera intrinsics across a range of diverse datasets.

5.1. Datasets

KITTI The KITTI dataset is collected in urban environments and is the main benchmark for depth and egomotion estimation. It is accompanied with a LIDAR sensor, which is used for evaluation only. We use standard splits into training, validation, and test sets, commonly referred to as the Eigen split. 39835 training examples are used from KITTI.

Cityscapes The Cityscapes dataset is a more recent urban driving dataset, which we use for both training and evaluation. It is a more challenging dataset with many dynamic scenes. With a few exceptions [27, 6] it has not been used for depth estimation evaluation. It has 38675 training examples. We use depth from the disparity data for evaluation on a standard evaluation set of 1250 samples [27, 6].

EuRoC Micro Aerial Vehicle Dataset The EuRoC Micro Aerial Vehicle (MAV) Dataset [5] is a very challenging dataset collected by an aerial vehicle indoors. While the data contains a comprehensive suite of sensor measurements, including stereo pairs, IMU, accurate Leica laser tracker ground truth, Vicon scene 3d scans, and camera calibration, we only used the monocular videos for training. Since the camera has significant lens distortion, this is an opportunity to test our method for learning lens distortions.

YouTube8M videos To demonstrate that depth can be learned on videos in the wild from unknown cameras, we collected videos from the YouTube8M dataset [1]. From the 3079 videos in YouTube8M that have the label "quadcopter", human raters selected videos that contain significant amount of footage from a quadcopter. Naturally, the videos were taken with different unknown cameras, with varying fields of view and varying degrees of lens distortion. The YouTube8M IDs are listed in the SM.

5.2. Depth

Since monocular methods can only estimate depth up to a global scale factor, we follow the field's common practice [47] of normalizing out the scale factor based on the medians of predicted and groundtruth depths (code).

KITTI Table 1 summarizes the evaluation results on the KITTI Eigen partition of a model trained on KITTI. The metrics are the ones defined in Zhou et al. [47]. Only the best methods and the first three metrics are displayed in Table 1, the rest are given in the SM. As seen in the table, we improve on the state-of-the-art results. More importantly, we observed that learned intrinsics, rather than given ones, consistently help performance.

Cityscapes Table 2 summarizes the evaluation metrics of models trained and tested on Cityscapes. We follow the established protocol by previous work, using the disparity for

Method	M	Abs Rel	Sq Rel	RMSE
Zhou [47]		0.208	1.768	6.856
Yang [42]		0.182	1.481	6.501
Mahjourian [24]		0.163	1.240	6.220
LEGO [41]	✓	0.162	1.352	6.276
GeoNet [44]	✓	0.155	1.296	5.857
DDVO [35]		0.151	1.257	5.583
Godard [13]		0.133	1.158	5.370
Struct2Depth [6]	✓	0.141	1.026	5.291
Yang [40]		0.137	1.326	6.232
Yang [40]	✓	0.131	1.254	6.117
Ours:				
Given intrinsics	✓	0.129	0.982	**5.23**
Learned intrinsics	✓	**0.128**	**0.959**	**5.23**

Table 1. Evaluation of depth estimation of our method, with given and learned camera intrinsics, for models trained and evaluated on KITTI, compared to other monocular methods. The depth cutoff is always 80m. The "M" column is checked for all models where object motion is taken into account.

evaluation [6, 27]. Since this is a very challenging benchmark with many dynamic objects, very few approaches have evaluated on it. As seen in Table 2, our approach outperforms previous ones and benefits from learned intrinsics.

Method	M	Abs Rel	Sq Rel	RMSE
Pilzer [27]		0.440	5.713	5.443
Struct2Depth [6]	✓	0.145	1.736	7.279
Ours:				
Given intrinsics	✓	0.129	1.35	**6.96**
Learned intrsinsics	✓	**0.127**	**1.33**	**6.96**

Table 2. Evaluation of depth estimation of models trained on Cityscapes on the cityscapes test set, with a depth cutoff of 80m, and comparison to prior art.

Cityscapes + KITTI Being able to learn depth without the need for intrinsics opens up the opportunity for pooling videos from any data source. Figure 5 shows the results of pooling Cityscapes and KITTI datasets and evaluating on either one. In this experiment the intrinsincs are assumed unknown and are learned. Training jointly on both datasets improves the depth metrics even beyond the best results obtained on either dataset separately. This is a key result which demonstrates the impact of our method to leverage data sources of potentially unlimited size.

Cityscapes + KITTI: ablation experiments Table 3 summarizes the results of ablation experiments we ran in order to study the impact of each of the techniques described in this paper on the end results. In order to reduce the number of combinations of results, in all experiments the training set was Cityscapes and KITTI mixed together. Each model was evaluated on both Cityscapes and KITTI separately.

Using a union of bounding boxes as a "possibly-mobile" mask (as depicted in Fig. 4) is found to be as good as using segmentation masks, which makes our technique more broadly applicable. Object motion estimation is shown to play a crucial role, especially on Cityscapes, which is characterized by more complex scenes with more pedestrians and cars. Randomized LN is shown to be superior to standard BN, and lastly – occlusion-awareness improves the quality of depth estimation, especially on Cityscapes, which has richer scenes with more occlusions. Figure 6 visualizes the type of artifacts that occlusion-aware losses reduce.

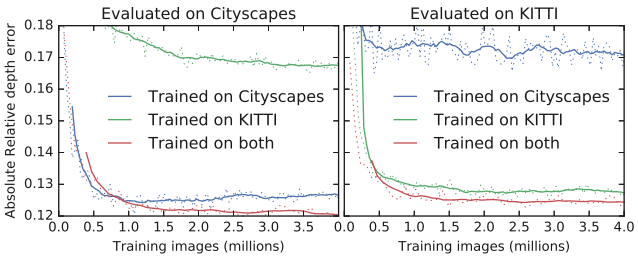

Figure 5. Depth prediction on Kitti and Cityscapes when training on each dataset and on both. Joining the two datasets improves the results on both. Learning intrinsics allows to similarly pool many datasets, even if they are from unknown cameras.

Experiment	Abs Rel depth CS	KITTI
Our algorithm	0.121	**0.124**
Boxes instead of segmentation masks	**0.120**	0.125
w/o occlusion-aware loss	0.127	0.126
w/o object motion	0.172	0.130
w/o randomized layer normalization	0.124	0.127

Table 3. Ablation experiments on depth estimation. In all experiments the training set was Cityscapes (CS) and KITTI combined, and we tested the model on Cityscapes (CS) and KITTI (Eigen partition) separately. Each row represents an experiment where one change was made compared to the main method, as described in the "Experiment" row. Smaller numbers are better.

EuRoC MAV Dataset We further use the EuRoC MAV Dataset to evaluate depth. We selected also a very challenging out-of-sample evaluation protocol in which we trained on the "Machine room" sequences and tested on the "Vicon Room 2 01", which has 3D groundtruth. Table 4 reports the results. The SM details how depth ground truth was generated from the provided Vicon 3D scans.

Abs R	Sq R	RMS	lgRMS	a_1	a_2	a_3
0.332	0.389	0.971	0.396	0.420	0.743	0.913

Table 4. Evaluation of depth estimation for EuRoC dataset, no prior results are available for this dataset. $a_i = \delta < 1.25^i$.

5.3. YouTube Videos

To demonstrate that depth can be learned on collections of videos in the wild for which the camera parameters are

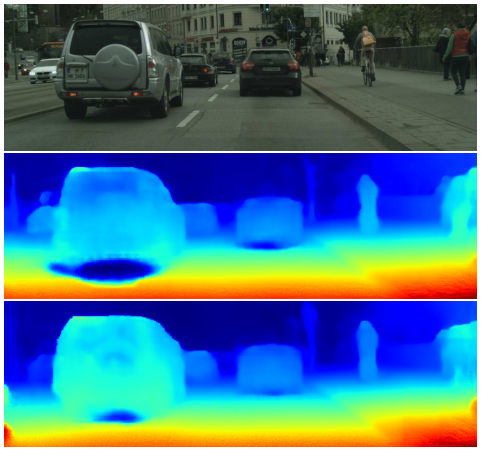

Figure 6. Illustration of the effect of occlusion aware losses. The center and bottom images are inferred disparity maps obtained from the image at the top. In the center image, the model was trained without occlusion-aware losses. At areas that become disoccluded, under cars, occlusion aware loss is shown to reduce artifacts. The top image belongs to the Cityscapes test set and is one of images whose depth prediction metrics were hurt the most upon removing occlusion aware losses.

not known, and differ across videos, we trained our model on the YouTube8M videos described in Sec. 5.1. Figure 7 visualizes the results. We note that this dataset is very challenging as it features objects at large ranges of depth.

Figure 7. Frames from from YouTube and learned disparity maps. The camera intrinsics are learned.

5.4. Camera intrinsics evaluation

In evaluating our method for learning camera intrinsics, two separate questions can be asked. First, how good is the supervision signal that cross-frame consistency provides for the camera intrinsics. Second is how good a deep network is in learning them and generalizing to test data.

Quality of the supervision signal. To evaluate the supervision signal for the intrinsics, we represented each intrinsic parameter as a separate learned variable. This is suitable for the EuRoC dataset, since it was captured with the same camera throughout. Each of the 11 subsets of EuRoC were trained in a separate experiment, until the intrinsics converged, yielding 11 independent results. The resulting sample mean and standard deviation for each intrinsic parameter are summarized in Table 5. All parameters agree with groundtruth within a few pixels. Since the groundtruth values were not accompanied by tolerances, it is hard to tell whether or not the differences are within tolerance. The learned disparity maps are shown in Fig. 8.

Quantity	Learned	GT
Horizontal focal length (f_x)	253.7 ± 1.1	250.2
Vertical focal length (f_y)	265.4 ± 1.3	261.3
Horizontal center (x_0)	189.0 ± 0.9	187.2
Vertical center (y_0)	132.2 ± 1.1	132.8
Quadratic radial distortion	-0.267 ± 0.003	-0.283
Quartic radial distortion	0.064 ± 0.002	0.074

Table 5. Camera intrinsics learned on the EuRoC datasets. Learning of depth, egomotion and intrinsics was done separately on each of the 11 datasets, using monocular images ("cam0") only. Constancy of the intrinsics throughout each dataset separately was imposed, and statistics (mean and standard deviation) for each intrinsic parameter were collected across the results. The groundtruth (GT) was adjusted to an image size of (256×384).

Figure 8. Frames from EuRoC [5] and the corresponding learned disparity maps. The camera intrinsics are learned.

Learning and generalization Prior work [37, 39, 4] has shown that deep networks can learn and generalize camera intrinsics in a strongly supervised setting. In our setting, the camera intrinsics and motion are predicted by the same network and are thus correlated. In other words, the losses imposed on the motion / intrinsics network only impose correctness of the intrinsics within the limits of Eq. 3.

We evaluated our model's predictions of the intrinsics on the KITTI odometry series. The model was trained on the union of the Cityscapes and KITTI training sets, which significantly differ in their typical focal lengths. Figure 9 shows the scatter plot of the predicted f_x as function of the predicted r_y. The predictions fall within the limits imposed by Eq. 3. While Table 5 indicates a high-quality supervision signal for the intrinsics, Fig. 9 shows that when the intrinsics and motion are predicted by the same network, the latter learns them "in aggregate", only to the extent needed for the depth-prediciton network to learn depth.

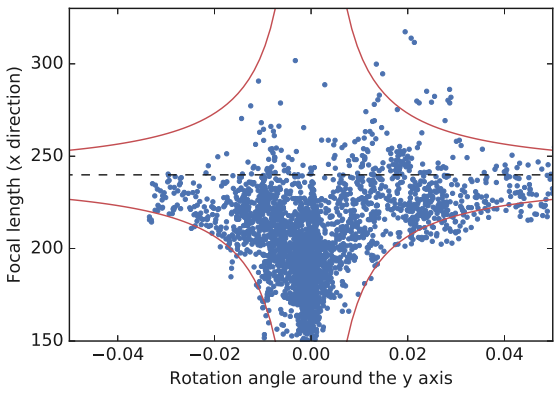

Figure 9. Predicted f_x as function of the predicted r_y for all images in the KITTI odometry sequences 09 and 10. The dashed line is groundtruth; red curves show tolerance limits imposed by Eq. 3.

5.5. Odometry

We evaluated our egomotion prediction on the KITTI sequences 09 and 10. The common 5-point Absolute Trajectory Error (ATE) metric [47, 6, 44, 13] measures local agreement between the estimated trajectories and the respective groundtruth. However assessing the usefulness for a method for localization requires evaluating its accuracy in predicting location. A common metric for localization is average relative translational drift t_{rel} [31, 46] – the distance between the predicted location and the groundtruth location divided by distance traveled and averaged over the trajectory. Table 6 summarizes both metrics, demonstrating the improvements our method achieves on both.

When evaluating for odometry, the most naive way is to calculate the inference of egomotion for every pair of adjacent frame. That leads to the red "learned intrinsics" curve in Fig. 10. However it is also possible to make an inference-time correction if we know the intrinsics of the camera at inference time. In that case, one can leverage the fact that for small errors in the rotation angles and focal lengths, $r_x f_y$ and $r_y f_x$ are approximately constant (Eq. SM8). Therefore if the network predicted r'_y and f'_x for a given pair of images, and we know the true focal length f_x, we can correct our estimate of r_y to $r'_y f'_x / f_x$. This is the procedure invoked in generating the "Learned and corrected intrinsics" curve in Fig. 10, and the respective entry in Table 6. The trajectories and metrics obtained with learned itrinsics with inference time correction and with given intrinsics are similar. Both notably improve prior art, which is especially prominent for localization, as the t_{rel} metric indicates.

6. Conclusions

This work addresses important challenges for unsupervised learning of depth and visual odometry through geometric handling of occlusions, a simple way of accounting for object motion, and a novel form of regularization. Most importantly, it takes a major step towards leveraging the vast amounts of existing unlabeled videos for learning depth estimation: Through unsupervised learning of the camera intrinsic parameters, including lens distortion, it enables, for the first time, learning depth from raw videos captured by unknown cameras.

Metric	Seq. 09 ATE	Seq. 09 t_{rel}	Seq. 10 ATE	Seq. 10 t_{rel}
Zhou [47]	0.021	17.84%	0.020	37.91%
GeoNet [44]	0.012	/	0.012	/
Zhan [46]	/	11.92%	/	12.45%
Mahjourian [24]	0.013	/	0.012	/
Struct2depth [6]	0.011	10.2%	0.011	28.9%
Ours, with intrinsics:				
Learned	0.012	7.5%	0.010	13.2%
Learned & corrected	0.010	**2.7%**	**0.007**	6.8%
Given	**0.009**	3.1%	0.008	**5.4%**

Table 6. Absolute Trajectory Error (ATE) [47] and average relative translational drift (t_{rel}) [31] on the 09 and 10 KITTI odometry sequences. Our method with both learned and given intrinsics is compared to prior work.

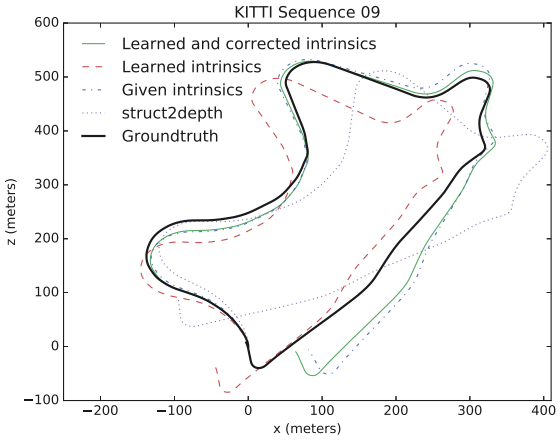

Figure 10. Predicted location on the KITTI odometry sequence 09, generated by models trained on KITTI, with given intrinsics and with learned intrinsics (with and without inference time correction), compared to groundtruth and to struct2depth results.

7. Acknowledgements

We Thank Roland Siegwart for the permission to use EuRoC dataset, Sergey Ioffe for enlightening discussions about batch normalization, Jason Su and Kwea Koi for spotting an error in the derivation in an earlier version of the manuscript, Kurt Konolige for consultation and critical reading of the manuscript, and Chad Richards for copyediting the paper.

Supplementary material

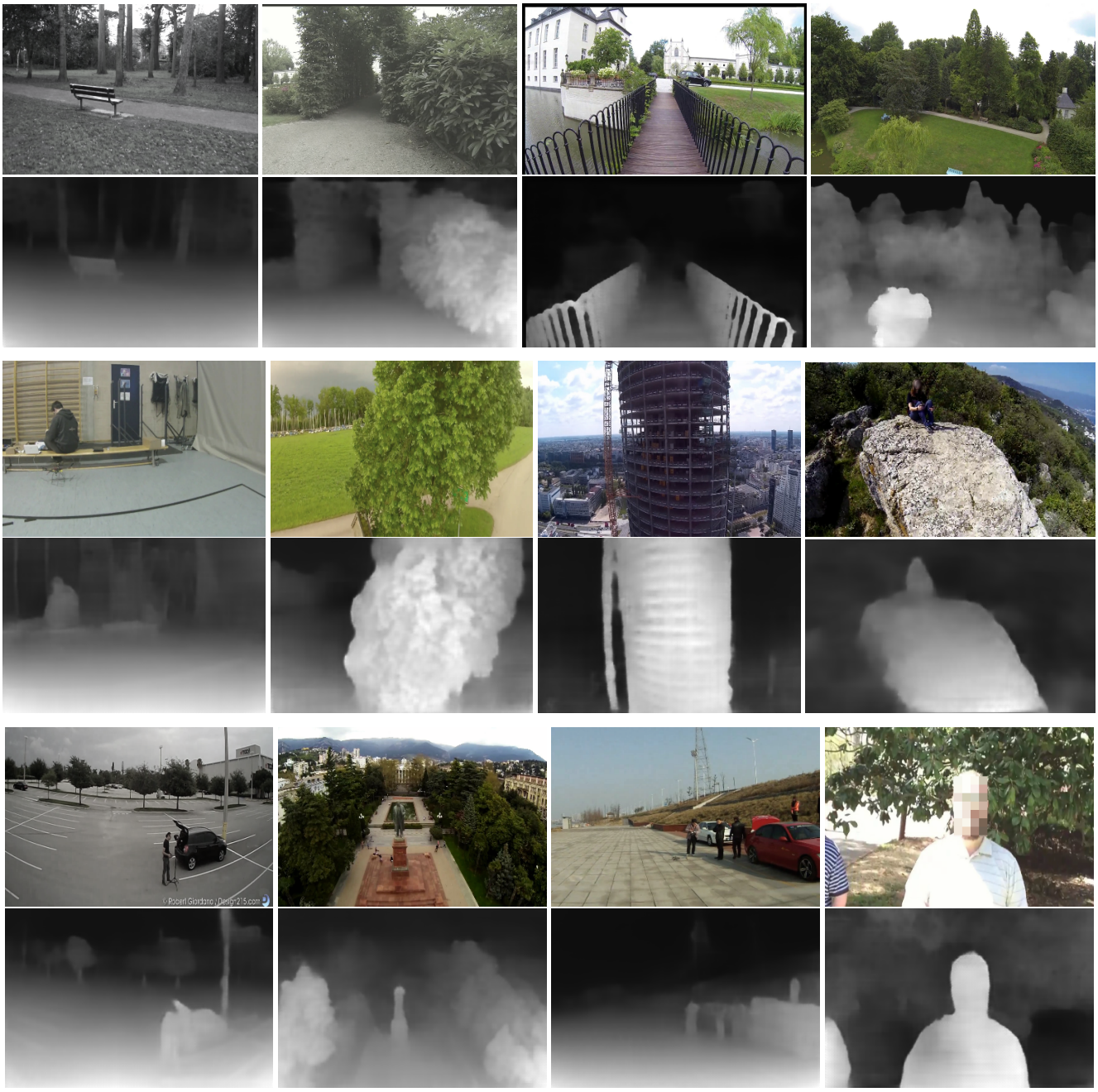

Figure SM1. Images and learned disparity maps from the set collected from YouTube8M.

A. Supplementary material

A.1. Accuracy of camera intrinsics – derivation

In this section we derive Eq. 3, which estimates the accuracy of the supervision signal that rotations provide for learning the camera intrinsics. Let R and t be the rotation that occurs between two frames, and K be the intrinsic matrix

$$K = \begin{pmatrix} f_x & 0 & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{SM1}$$

For every pixel location p in the first frame, Eq. 1 provides the shifted location p' due to R and t.

The photometric loss terms provide a supervision signal for p'. Therefore, they do not discriminate between combinations of R, t and K, as long as they produce the correct p'. Let $\tilde{R}$, $\tilde{t}$ and $\tilde{K}$ be a set of possibly-incorrect predictions for R, t and K. If we are able to satisfy

$$KRK^{-1}zp + Kt = \tilde{K}\tilde{R}\tilde{K}^{-1}zp + \tilde{K}\tilde{t}, \tag{SM2}$$

$\tilde{R}$, $\tilde{t}$ and $\tilde{K}$ are as good a solution as R, t and K.

As argued in Sec. 4.4, $\tilde{t}$ can always be chosen such that the Kt and $\tilde{K}\tilde{t}$ in Eq. SM2 cancel each other. Translations are thus be henceforth omitted. z cancels out as well, which intuitively makes sense, since for a pinhole camera, in the absence of translation, the amount of shift in pixel space due to a rotation depends on the rotation only, not on the distance of the object containing the pixel from the camera.

p in homogeneous coordinates can be written as $(p_x, p_y, 1)$, where p_x and p_y are the coordinates of a pixel in pixel space. After the rotation, p_x will me displaced to

$$p'_x = \frac{(KRK^{-1}p)_1}{(KRK^{-1}p)_3} \quad \tilde{p}'_x = \frac{(\tilde{K}\tilde{R}\tilde{K}^{-1}p)_1}{(\tilde{K}\tilde{R}\tilde{K}^{-1}p)_3}, \tag{SM3}$$

depending on whether we use K and R or $\tilde{K}$ and $\tilde{R}$. The $_1$ and $_3$ subscripts indicate the respective component of the three dimensional vector obtained by multiplying the 3x3 matrix KRK^{-1} or $\tilde{K}\tilde{R}\tilde{K}^{-1}$ by the three-dimensional vector p. p'_y and $\tilde{p}'_y$ have analogous expressions, with the subscript 1 replaced by 2.

In what follows, we only consider small rotations, since imposing photometric consistency across frames is typically only possible when the rotation is small enough to allow significant overlaps between the fields of view before and after the rotation, and because small-angle approximations facilitate deriving simple analytic equations like Eq. 3.

We thus write R as

$$R = \mathbb{1} + r, \quad \text{where} \quad r = \begin{pmatrix} 0 & r_z & -r_y \\ -r_z & 0 & r_x \\ r_y & -r_x & 0 \end{pmatrix}, \tag{SM4}$$

and similarly for $\tilde{R}$. Expanding Eq. SM3 in Taylor series with respect to r, we obtain

$$p'_x = p_x + (KrK^{-1}p)_1 - p_x(KrK^{-1}p)_3, \tag{SM5}$$

and similarly for $\tilde{p}'_x$.

Substituting Eq. SM1 and Eq. SM4 to Eq. SM5 gives a

$$\begin{aligned} p'_x &= -f_x r_y - r_y \frac{(p_x - x_0)^2}{f_x} + \\ &\quad r_x \frac{(p_x - x_0)(p_y - y_0)}{f_y} + r_z \frac{(p_y - y_0) f_x}{f_y} \end{aligned} \tag{SM6}$$

$$\begin{aligned} p'_y &= f_y r_x + r_x \frac{(p_y - y_0)^2}{f_y} + \\ &- r_y \frac{(p_x - x_0)(p_y - y_0)}{f_x} - r_z \frac{(p_x - x_0) f_y}{f_x} \end{aligned} \tag{SM7}$$

and similarly for $\tilde{p}'_x$, $\tilde{p}'_y$.

We assume that errors in estimating K and R that result in a difference that is much less than a single pixel do not affect the photometric loss, and thus cannot be eliminated by it. We therefore need to derive an expression for the range where $\tilde{K}$ and $\tilde{R}$ can be such that $|\tilde{p}'_x - p'_x| \ll 1$ and $|\tilde{p}'_y - p'_y| \ll 1$.

Equations SM6 and SM7 provide the foundation for a full error analysis on K and R, but a full analysis falls beyond the scope of the present paper. Instead, we limit our discussion to the error analysis of the focal length. Suppose that the networks mispredicted f_x and f_y, yielding $\tilde{f}_x$ and $\tilde{f}_y$ instead. Since the same network predicts R, we assume it chose an $\tilde{R}$ that tries to undo some of the effects of the mispredicted f-s. For simplicity, we choose R such that at least at the pixels on the optical axis ($p_x = x_0, p_y = y_0$), p'_x and p'_y remain unchanged. From Eqs. SM6 and SM7 one can see that this requires

$$\tilde{r}_y = r_y f_x / \tilde{f}_x, \quad \tilde{r}_x = r_x f_y / \tilde{f}_y. \tag{SM8}$$

We can now write the *tilde* version Eqs. SM6 and SM7, using rprime to eliminate $\tilde{r}_x$ and $\tilde{r}_y$ form the equations. The result is:

$$\begin{aligned} \tilde{p}'_x &= -f_x r_y - r_y f_x \frac{(p_x - x_0)^2}{\tilde{f}_x^2} + \\ &\quad r_x f_y \frac{(p_x - x_0)(p_y - y_0)}{\tilde{f}_y^2} + r_z \frac{(p_y - y_0)\tilde{f}_x}{\tilde{f}_y} \end{aligned} \tag{SM9}$$

$$\begin{aligned} \tilde{p}'_y &= f_y r_x + r_x f_y \frac{(p_y - y_0)^2}{\tilde{f}_y^2} + \\ &- r_y f_x \frac{(p_x - x_0)(p_y - y_0)}{\tilde{f}_x^2} - r_z \frac{(p_x - x_0)\tilde{f}_y}{\tilde{f}_x} \end{aligned} \tag{SM10}$$

Putting $\tilde{p}'_x = p'_x + \delta p'_x$, $\tilde{f}_x = f_x + \delta f_x$, and similarly for their y counterparts, and subtracting Eqs. SM6 and SM7 from Eqs. SM9 and SM10, we obtain

$$\begin{aligned}\delta p'_x &= 2r_y \delta f_x \frac{(p_x - x_0)^2}{f_x^2} \\ &- 2r_x \delta f_y \frac{(p_x - x_0)(p_y - y_0)}{f_y^2} \\ &+ r_z \frac{(p_y - y_0) f_x}{f_y} \left(\frac{\delta f_x}{f_x} - \frac{\delta f_y}{f_y} \right)\end{aligned} \tag{SM11}$$

$$\begin{aligned}\delta p'_y &= -2r_x \delta f_y \frac{(p_y - y_0)^2}{f_y^2} \\ &+ 2r_y \delta f_x \frac{(p_y - y_0)(p_x - x_0)}{f_x^2} \\ &- r_z \frac{(p_x - x_0) f_y}{f_x} \left(\frac{\delta f_y}{f_y} - \frac{\delta f_x}{f_x} \right),\end{aligned} \tag{SM12}$$

where terms of higher than first order in δf_x and δf_y have been dropped.

Eqs. SM11 and SM12, along with the requirement that $|\delta p'_x| \ll 1$ and $|\delta p'_y| \ll 1$, can be used to estimate the bounds of δf_x and δf_y given an arbitrary small rotation r. In order to gain some insight into Eqs. SM11 and SM12, in what follows we derive explicit expressions for the bounds of δf_x and δf_y for the cases where two out of the three components of r are zero, that is, rotations around the x, y and z axes.

Rotations around the z axis If only r_z is nonzero, Eqs. SM11 and SM12, along with $|\delta p'_x| \ll 1$ and $|\delta p'_y| \ll 1$ reduce to

$$\begin{aligned}\left| \frac{\delta f_y}{f_y} - \frac{\delta f_x}{f_x} \right| &\ll \frac{f_x}{r_z f_y |p_x - x_0|} \\ \left| \frac{\delta f_y}{f_y} - \frac{\delta f_x}{f_x} \right| &\ll \frac{f_y}{r_z f_x |p_y - y_0|}\end{aligned} \tag{SM13}$$

If f_y and f_x are of similar magnitudes, and $x0 \approx w/2$, $y_0 \approx h/2$, Eq. SM13 reduces to

$$\begin{aligned}\left| \frac{\delta f_y}{f_y} - \frac{\delta f_x}{f_x} \right| &\ll \frac{2}{w r_z} \\ \left| \frac{\delta f_y}{f_y} - \frac{\delta f_x}{f_x} \right| &\ll \frac{2}{h r_z}\end{aligned} \tag{SM14}$$

We learn from Eqs. SM13 and SM14 that rotations along z:

- Constrain the ratio between f_x and f_y
- Do not otherwise provide a supervision signal for the magnitudes of the f-s
- The strength of the supervision signal is inversely proportional to the magnitude of the rotation and the height / width of the image in pixels.

Rotations around the y axis If only r_y is nonzero, Eqs. SM11 and SM12, along with $|\delta p'_x| \ll 1$ and $|\delta p'_y| \ll 1$ reduce to

$$\begin{aligned}|\delta f_x| &\ll \frac{f_x^2}{2r_y (p_x - x_0)^2} \\ |\delta f_x| &\ll \frac{f_x^2}{2r_y |p_y - y_0| |p_x - x_0|}\end{aligned} \tag{SM15}$$

Just like before, the pixels that are farthest away from the center provide the tightest bound on $|\delta f_x|$. If $x0 \approx w/2$, $y_0 \approx h/2$, Eq. SM15 leads to

$$|\delta f_x| \ll \min\left(\frac{2f_x^2}{r_y w^2}, \frac{2f_x^2}{r_y w h} \right)$$

We learn from Eqs. SM15 and SM16 that rotations along y:

- Provide a supervision signal for f_x
- The magnitude of the supervision signal is inversely proportional to the magnitude of the rotation and height / width of the image in pixels *squared*.

Since rotations around the x axis lead to an expression identical to Eq. SM16 with x and y swapped and w and h swapped, this concludes the derivation of Eq. 3.

A.2. The motion- and intrinsics-prediction network

A schematic of the network is shown in Fig. 2 in the main paper. A stack of convolutions with stride 2 (the"encoder"), with average pooling in the last one, forms a bottleneck of 1024 channels with a 1x1 spatial resolution. From the bottleneck, the following heads stem:

- A fully-connected layer with 3 outputs each predict the global rotation angles (r_0) and the global translation vector (t_0). The latter two represent the movement of the entire scene with respect to the camera, due to camera motion.
- Each of the intrinsic parameters is predicted by a 1x1 convolution. Softplus activations keep the focal lengths positive and the distortion curve monotonically-increasing.
- A stack of decoder layers predicts a *dense residual translation vector field* $\delta t(x, y)$, with 3 output channels, representing the 3D movement of each pixel with respect to the scene. Each decoder layer receives as input the outputs of the previous decoder layer and the outputs of the corresponding encoder layer following the UNet architecture.

For the results shown in Table 5, instead of predicting the intrinsic parameters from the network, each of the parameters listed in 5 was assigned a separate trainable variable. This is a way to incorporate the constraint that the intrinsics should be the same for all training example, since entire EuRoC dataset was captured with the same camera. The distortion variables were initialized to zero, x_0 and f_x were initialized to $w/2$, and y_0 and f_y are initialized to $h/2$.

Method	M	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Zhou [47]		0.208	1.768	6.856	0.283	0.678	0.885	0.957
Yang [42]		0.182	1.481	6.501	0.267	0.725	0.906	0.963
Mahjourian [24]		0.163	1.240	6.220	0.250	0.762	0.916	0.968
LEGO [41]	✓	0.162	1.352	6.276	0.252	0.783	0.921	0.969
GeoNet [44]	✓	0.155	1.296	5.857	0.233	0.793	0.931	0.973
DDVO [35]		0.151	1.257	5.583	0.228	0.810	0.936	0.974
Godard [13]		0.133	1.158	5.370	**0.208**	0.841	**0.949**	0.978
Struct2Depth [6]	✓	0.141	1.026	5.291	0.2153	0.8160	0.9452	**0.9791**
Yang [40]		0.137	1.326	6.232	0.224	0.806	0.927	0.973
Yang [40]	✓	0.131	1.254	6.117	0.220	0.826	0.931	0.973
Ours:								
Given intrinsics	✓	0.129	0.982	5.23	0.213	0.840	0.945	0.976
Learned intrinsics	✓	**0.128**	**0.959**	**5.23**	0.212	**0.845**	0.947	0.976

Table SM1. Evaluation of depth estimation of our method, with given and learned camera intrinsics, for models trained and evaluated on KITTI, compared to other monocular methods. The depth cutoff is always 80m. The "M" column is checked for all models where object motion is taken into account. This extends Table 1 in the main paper.

Method	M	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Pilzer [27]		0.440	5.71	5.44	0.398	0.730	0.887	0.944
Struct2Depth [6]	✓	0.145	1.74	7.28	0.205	0.813	0.942	0.978
Ours:								
Given intrinsics	✓	0.129	1.35	6.96	0.198	0.827	0.945	0.980
Learned intrsinsics	✓	**0.127**	**1.33**	**6.96**	**0.195**	**0.830**	**0.947**	**0.981**

Table SM2. Evaluation of depth estimation of models trained on Cityscapes on the cityscapes test set using the procedure and code in Ref. [6], with a depth cutoff of 80m, and comparison to prior art. This table extends Table 2 from the main paper.

Trained on	Evaluated on	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Cityscapes	Cityscapes	0.127	1.33	6.96	0.195	0.830	0.947	0.981
Cityscapes	KITTI	0.172	1.37	6.21	0.250	0.754	0.921	0.967
KITTI	Cityscapes	0.167	2.31	9.99	0.272	0.747	0.894	0.957
KITTI	KITTI	0.128	0.959	5.23	0.212	0.845	0.947	0.976
Cityscapes + KITTI	Cityscapes	**0.121**	**1.31**	**6.92**	**0.189**	**0.846**	**0.953**	**0.983**
Cityscapes + KITTI	KITTI	**0.124**	**0.930**	**5.12**	**0.206**	**0.851**	**0.950**	**0.978**

Table SM3. Evaluation of depth estimation of models trained on Cityscapes and KITTI together, on the Cityscapes and KITTI test sets separately. The depth cutoff is of 80m. This table extends Figure 5 in the main paper.

The code is at github.com/google-research/google-research/tree/master/depth_from_video_in_the_wild.

A.3. Full tables of metrics for depth estimation

The numbers in Table 1 and 2, as well as in Fig. 5, are given for only part of the metrics commonly published for depth estimation. In this section we give the rest of the metrics, for completeness. Tables SM1, SM2 and SM3 provide the full set of numbers for the former ones, respectively.

A.4. Further details about the losses

Structural Similarity (SSIM) As explained in Sec. 4.2 and Fig. 3, when warping one frame onto the other, the depth map can become mutivalued, which indicates newly-occluded areas. In a multivalued depth map, we need to pick the branch that is closer to the camera when demanding consistency.

Calculating SSIM involves calculating the mean, variance and covariance of image patches (the formula is given at en.wikipedia.org/wiki/Structural_similarity). For example, the mean of $3x3$ image patch would be

$$\mu = \frac{1}{9} \sum_{i=-1,j=-1}^{i=1,j=1} I_{ij}, \qquad \text{(SM16)}$$

where I_{ij} is the pixel value of one of the channels of the image.

We replace Eq. by a weighted average:

$$\mu = \frac{\sum_{i=-1,j=-1}^{i=1,j=1} w_{ij} I_{ij}}{\sum_{i=-1,j=-1}^{i=1,j=1} w_{ij}}, , \qquad \text{(SM17)}$$

where w_{ij} is a positive weight function. The weight function we used was

$$w_{ij} = \frac{1}{1 + (z_{ij} - z'_{ij})^2 / \langle (z - z')^2 \rangle}, \qquad \text{(SM18)}$$

where z_{ij} is the predicted depth at the pixel at i, j and z'_{ij} is the transformed depth from the other frame, interpolated to i, j. $\langle\cdot\rangle$ denotes an average over the entire image.

In words, Eq. SM17 downweighs the contribution of pixels where the depth reprojection error is greater than the root mean square depth reprojection error calculated over the entire image. The rationale is that if the depth reprojection error is large, the point more likely belongs to the occluded branch of a multivalued depth map.

The same weighing is applied for the other statistics (variances and covariances) calculated in the SSIM formula.

Other losses The RGBD consistency losses, SSIM loss and motion cycle consistency losses are implemented in the `consistency_lossses.py` file in our repository. The smoothing losses the weights of all the losses are in `model.py`.

A.5. Generating depth groundtruth for the EuRoC dataset

In the EuRoC dataset, the Vicon Room 2 series has point-clouds that were obtained from merging depth sensor readings. In addition, there is groundtruth for the position and orientation of the camera at given timestamps, as well as the intrinsics. For every frame, we reprojected the point clouds onto the camera using the intrinsics and extrinsics. To address occlusions, each point was given some finite angular width. If two 3D points were projected onto close enough locations on the image plane, and their depth ratio was greater than a certain threshold, only the one closer to the camera was kept. Finally, the rendered depth maps were made more uniform by introducing a uniform grid in projective space and keeping at most one point in a each cell. An example of the result is shown in Fig. SM2.

A.6. YouTube8M IDs of used for training

The YouTube8M IDs are listed below:

```
1ofm 2Ffk 2Gc7 2hdG 4Kdy 4gbW 70eK 77cq
7We1 8Eff 8W2O 8bfg 9q4L A8cd AHdn Ai8q
B8fJ BfeT C23C C4be CP6A EOdA Gu4d IdeB
Ixfs Kndm L1fF M28T M92S NSbx NSfl NT57
Q33E Qu62 U4eP UCeG VRdE W0ch WU6A WWdu
WY2M XUeS YLcc YkfI ZacY aW8r bRbL d79L
d9bU eEei ePaw iOdz iXev j42G j97W k7fi
kxe2 lIbd lWeZ mw3B nLd8 olfE qQ8k qS6J
sFb2 si9H uofG yPeZ zger
```

The YouTube8M website[3] provides the instructions for mapping them you YouTube IDs. Two consecitive frames were sampled off of each video every second.

[3] research.google.com/youtube8m/

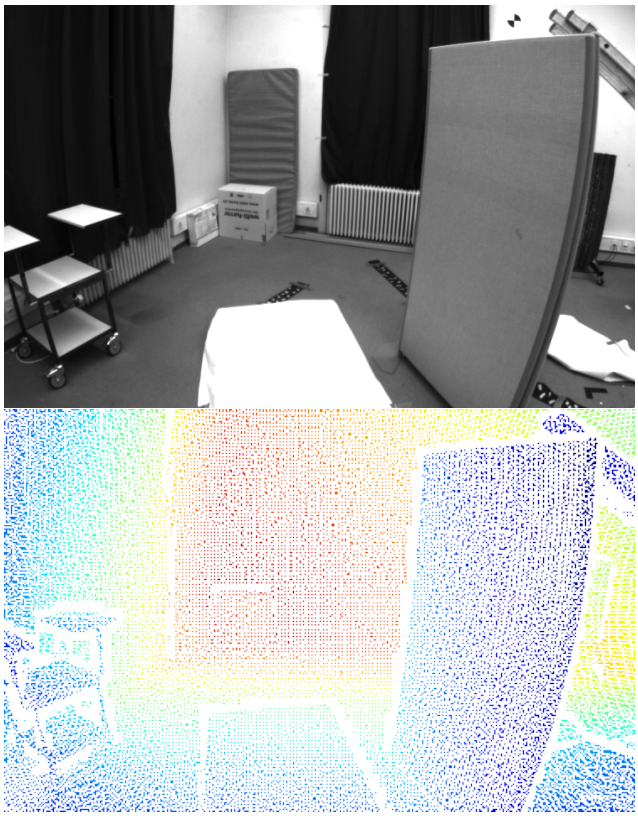

Figure SM2. Illustration of a depth map (below) generated from the EuRoC point cloud of Vicon Room 2, by projecting onto the view of the RGB camera (above).

A.7. Intrinsics tranformation on the EuRoC dataset

The intrinsics of cam0 in the EuRoC set are (752, 480) for the width and height, 458.654, 457.296 for the focal lengths in the x and y direction respectively, and 367.215, 248.375 for x_0 and y_0 respectively. The radial distortion coefficients are -0.28340811 and 0.07395907, and the higher-order coefficients are small. In our experiments, we first center-cropped the images to (704, 448). This does not change the focal lengths nor the distortion coefficients, and changes x_0 and y_0 to 343.215, 232.375 respectively. Next, we resized the images to (384, 256), which multiplies all x-related parameters by 384/704, and all y-related parameters by 256/448. The results are in the last column of Table 5.

A.8. Odometry

The KITTI Sequence 10 is shown in Figure SM3. Tables SM4 and SM5 extend Table 6 with more metrics.

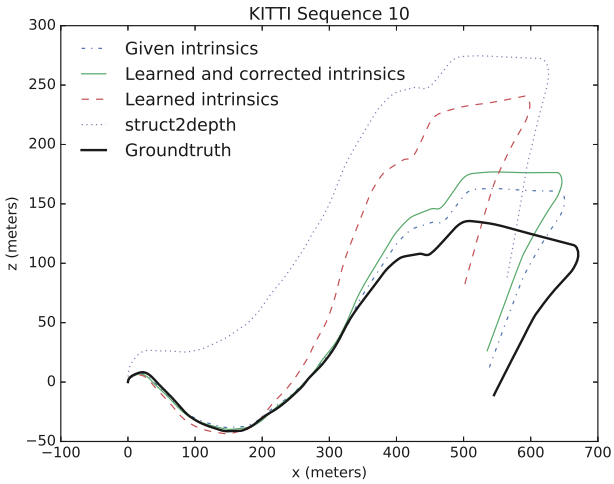

Figure SM3. Predicted location on the KITTI odometry sequence 10 (the counterpart of Fig. 10 from the main paper), by a model trained with given intrinsics, a model that learned the intrinsics, and the latter model with inference time correction applied. The groundtruth and the struct2depth [6] results are displayed as well.

Method	Seq. 09	Seq. 10
Zhou [47]	0.021 ± 0.017	0.020 ± 0.015
Mahjourian [24]	0.013 ± 0.010	0.012 ± 0.011
GeoNet [44]	0.012 ± 0.007	0.012 ± 0.009
Godard [13]	0.023 ± 0.013	0.018 ± 0.014
Struct2depth [6]	0.011 ± 0.006	0.011 ± 0.010
Ours, with intrinsics:		
Given	$\mathbf{0.009} \pm 0.015$	0.008 ± 0.011
Learned	0.012 ± 0.016	0.010 ± 0.010
Learned & corrected	0.010 ± 0.016	$\mathbf{0.007 \pm 0.009}$

Table SM4. 5-point Absolute Trajectory Error, (ATE) calculated following the procedure outlined in [47]. The three variants of our method are a model trained with given intrinsics, a model trained with learned intrinsics, and the latter model with test-time correction of the intrinsics. The trajectories are shown in (Fig. 10 and Fig. SM3). This table extends Table 6 in the main paper.

	Seq. 09		Seq. 10	
Method	t_{rel}	r_{rel}	t_{rel}	r_{rel}
Zhou [47] a la [46]	17.8	6.78	37.9	17.8
Zhou [47] a la [31]	21.63	3.57	20.5	10.9
Zhan [46]	11.9	3.60	12.6	3.43
Struct2depth [6]	10.2	2.64	29.0	4.28
Ours, with intrinsics:				
Given	3.18	0.586	**5.38**	**1.03**
Learned	7.47	0.960	13.2	3.09
Learned & corrected	**2.70**	**0.462**	6.87	1.36

Table SM5. Average relative translation error (t_{rel}, in percents) and average relative rotation error (r_{rel}, degrees per 100 meters) calculated on the KITTI odometry sequences 09 and 10. The results for the method in Zhou et al. [47] were taken from two different evaluations [31, 46]. The number for Struct2depth [6] were evaluated using their published code and models. As in prior work, [31, 46] the metrics are calculated starting after the first 100 meters. This table extends Table 6 in the main paper.

References

[1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *CoRR*, abs/1609.08675, 2016. 1, 5

[2] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building Rome in a day. *ICCV*, 2009. 2

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 4

[4] Oleksandr Bogdan, Viktor Eckstein, Francois Rameau, and Jean-Charles Bazin. Deepcalib: A deep learning approach for automatic intrinsic calibration of wide field-of-view cameras. In *Proceedings of the 15th ACM SIGGRAPH European Conference on Visual Media Production*, CVMP '18, pages 6:1–6:10, New York, NY, USA, 2018. ACM. 2, 7

[5] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W Achtelik, and Roland Siegwart. The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 2016. 1, 5, 7

[6] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Unsupervised learning of depth and ego-motion: A structured approach. In *AAAI-19*, 2019. 2, 3, 4, 5, 6, 8, 14, 16

[7] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 1

[8] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. *ICCV*, 2015. 2

[9] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *NIPS*, 2014. 2

[10] Ravi Garg, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. *ECCV*, 2016. 2

[11] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 1

[12] Clément Godard, Oisin Mac Aodha, and Gabriel J. Brostow. Unsupervised monocular depth estimation with left-right consistency. *CVPR*, 2017. 2

[13] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel Brostow. Digging into self-supervised monocular depth estimation. *arxiv.org/pdf/1806.01260*, 2018. 6, 8, 14, 16

[14] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017. 4

[15] Joel Janai, Fatma Guney, Anurag Ranjan, Michael Black, and Andreas Geiger. Unsupervised learning of multi-frame optical flow with occlusions. *ECCV*, 2018. 2

[16] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[17] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry. End-to-end learning of geometry and context for deep stereo regression. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 2

[18] Sameh Khamis, Sean Fanello, Christoph Rhemann, Adarsh Kowdle, Julien Valentin, and Shahram Izadi. Stereonet: Guided hierarchical refinement for real-time edge-aware depth prediction. In *The European Conference on Computer Vision (ECCV)*, September 2018. 2

[19] Iro Laina, Christian Rupprecht, Vasileios Belagiannis, Federico Tombari, and Nassir Navab. Deeper depth prediction with fully convolutional residual networks. *arXiv:1606.00373*, 2016. 2

[20] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. *CVPR*, 2018. 2

[21] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. *CVPR*, 2015. 2

[22] F. Liu, C. Shen, G. Lin, and I. Reid. Learning depth from single monocular images using deep convolutional neural fields. *PAMI*, 2015. 2

[23] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004. 1

[24] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5667–5675, 2018. 2, 6, 8, 14, 16

[25] Michal Neoral, Jan Sochman, and Jir Matas. Continual occlusions and optical flow estimation. *ECCV*, 2018. 2

[26] Thu H Nguyen-Phuoc, Chuan Li, Stephen Balaban, and Yongliang Yang. Rendernet: A deep convolutional network for differentiable rendering from 3d shapes. In *Advances in Neural Information Processing Systems*, pages 7902–7912, 2018. 2

[27] Andrea Pilzer, Dan Xu, Mihai Marian Puscas, Elisa Ricci, and Nicu Sebe. Unsupervised adversarial depth estimation using cycled generative networks. *3DV*, 2018. 5, 6, 14

[28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing. 3

[29] Ashutosh Saxena, Jamie Schulte, and Andrew Y. Ng. Depth estimation using monocular and stereo cues. In *Proceedings*

of the 20th International Joint Conference on Artifical Intelligence, IJCAI'07, pages 2197–2203, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc. 2

[30] Johannes L. Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. *CVPR*, 2016. 2

[31] Jared Shamwell, Sarah Leung, and William Nothwang. Vision-aided absolute trajectory estimation using an unsupervised deep network with online error correction. 10 2018. 8, 16

[32] Shubham Tulsiani, Tinghui Zhou, Alexei A. Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. *CVPR*, 2017. 2

[33] Benjamin Ummenhofer, Huizhong Zhou, Jonas Uhrig, Nikolaus Mayer, Eddy Ilg, Alexey Dosovitskiy, and Thomas Brox. Demon: Depth and motion network for learning monocular stereo. *CVPR*, 2017. 2

[34] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv:1704.07804*, 2017. 2

[35] Chaoyang Wang, Jose Miguel Buenaposada, Rui Zhu, and Simon Lucey. Learning depth from monocular videos using direct methods. *CVPR*, 2018. 2, 6, 14

[36] Yang Wang, Yi Yang, Zhenheng Yang, Liang Zhao, Peng Wang, and Wei Xu. Occlusion aware unsupervised learning of optical flow. *CVPR*, 2018. 2

[37] Scott Workman, Connor Greenwell, Menghua Zhai, Ryan Baltenberger, and Nathan Jacobs. DeepFocal: A Method for Direct Focal Length Estimation. In *International Conference on Image Processing*, 2015. 2, 7

[38] Abhinav Gupta Xiaolong Wang, David F. Fouhey. Designing deep networks for surface normal estimation. *CVPR*, 2015. 2

[39] Han Yan, Yu Zhang, Shunli Zhang, Sicong Zhao, and Li Zhang. Focal length estimation guided with object distribution on focalens dataset. *Journal of Electronic Imaging*, 26(3):1 – 14 – 14, 2017. 2, 7

[40] Zhenheng Yang, Peng Wang, Yang Wang, Wei Xu, and Ram Nevatia. Every pixel counts: Unsupervised geometry learning with holistic 3d motion understanding. *arxiv.org/pdf/1806.10556*, 2018. 2, 6, 14

[41] Zhenheng Yang, Peng Wang, Yang Wang, Wei Xu, and Ram Nevatia. Lego: Learning edge with geometry all at once by watching videos. *CVPR*, 2018. 2, 6, 14

[42] Zhenheng Yang, Peng Wang, Wei Xu, Liang Zhao, and Ramakant Nevatia. Unsupervised learning of geometry with edge-aware depth-normal consistency. *arXiv:1711.03665*, 2017. 2, 6, 14

[43] Yao Yao, Zixin Luo, Shiwei Li, and Tian Fangand Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. *ECCV*, 2018. 2

[44] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. *CVPR*, 2018. 2, 6, 8, 14, 16

[45] Zifeng Wu Yuanzhouhan Cao and Chunhua Shen. Estimating depth from monocular images as classification using deep fully convolutional residual networks. *CoRR:1605.02305*, 2016. 2

[46] Huangying Zhan, Ravi Garg, Chamara Saroj Weerasekera, Kejie Li, Harsh Agarwal, and Ian Reid. Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. *CVPR*, 2018. 2, 8, 16

[47] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. *CVPR*, 2017. 2, 5, 6, 8, 14, 16