Supplementary material

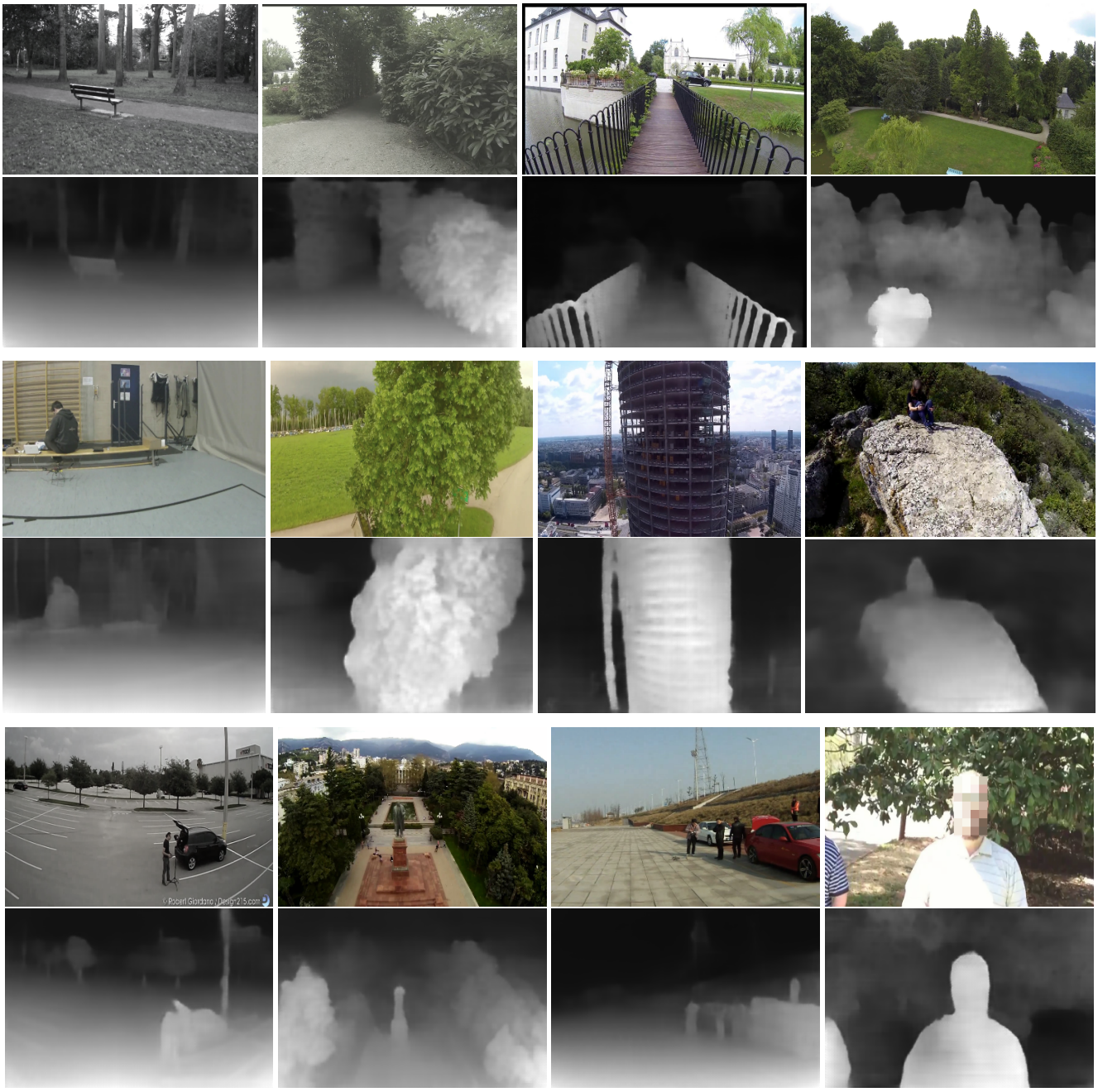

Figure SM1. Images and learned disparity maps from the set collected from YouTube8M.

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/

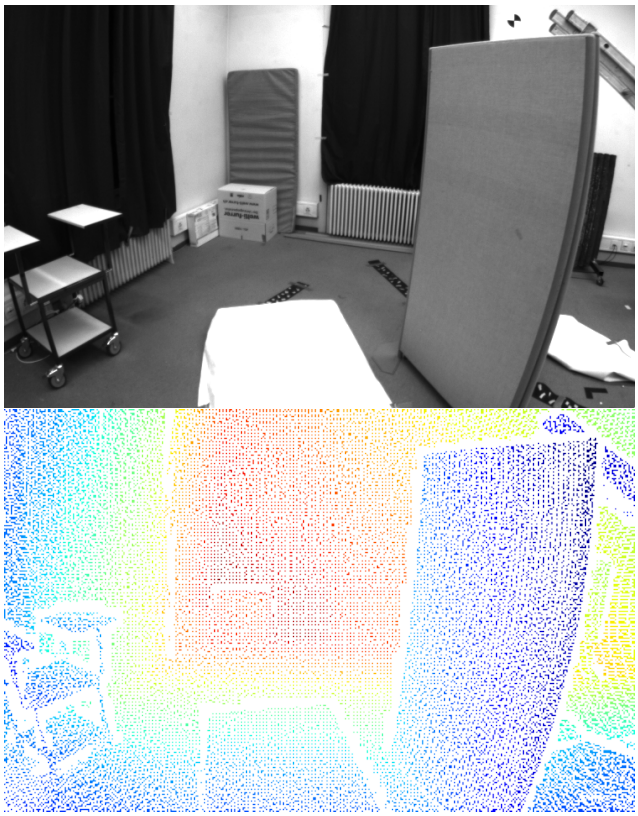

Figure SM2. Illustration of a depth map (below) generated from the EuRoC point cloud of Vicon Room 2, by projecting onto the view of the RGB camera (above).

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.

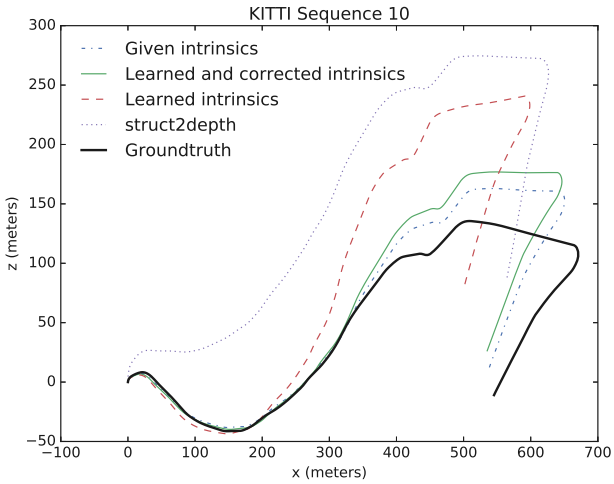

Figure SM3. Predicted location on the KITTI odometry sequence 10 (the counterpart of Fig. 10 from the main paper), by a model trained with given intrinsics, a model that learned the intrinsics, and the latter model with inference time correction applied. The groundtruth and the struct2depth [6] results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