Feature-metric Loss for Self-supervised Learning of Depth and Egomotion

Chang Shu[1⋆], Kun Yu[2], Zhixiang Duan[2], and Kuiyuan Yang[2]

[1] Meituan Dianping Group
[2] DeepMotion
shuchang02@meituan.com
{kunyu,zhixiangduan,kuiyuanyang}@deepmotion.ai

Abstract. Photometric loss is widely used for self-supervised depth and egomotion estimation. However, the loss landscapes induced by photometric differences are often problematic for optimization, caused by plateau landscapes for pixels in textureless regions or multiple local minima for less discriminative pixels. In this work, feature-metric loss is proposed and defined on feature representation, where the feature representation is also learned in a self-supervised manner and regularized by both first-order and second-order derivatives to constrain the loss landscapes to form proper convergence basins. Comprehensive experiments and detailed analysis via visualization demonstrate the effectiveness of the proposed feature-metric loss. In particular, our method improves state-of-the-art methods on KITTI from 0.885 to 0.925 measured by δ_1 for depth estimation, and significantly outperforms previous method for visual odometry.

1 Introduction

Estimating depth and egomotion from monocular camera is a fundamental and valuable task in computer vision, which has wide applications in augmented reality [35], robotics navigation [8] and autonomous driving [31]. Though monocular camera is cheap and lightweight, the task is hard for conventional SfM/SLAM algorithms [12,34,42] and continues challenging deep learning based approaches [4,24,1,2,56].

Deep learning for depth and egomotion estimation can be broadly categorized into supervised and self-supervised learning. For depth estimation, supervised learning takes images paired with depth maps as input [11,13,23], where depth maps are sparsely collected from expensive LiDAR sensors [14] or densely rendered from simulation engines [29], while supervision from LiDAR limits the generalization to new cameras and supervision from rendering limits the generalization to real scenes. For egomotion estimation, supervised signals come from trajectories computed by classical methods with high precision sensors like IMU and GPS, which are also costly and cannot guarantee absolute accuracy. Self-supervised learning unifies these two tasks into one framework, and

⋆ This work is done when Chang Shu is an intern at DeepMotion.

only uses monocular videos as inputs, and supervision is from view synthesis [56,52,27,16,15]. The setup is simpler, and easy to generalize among cameras.

However, self-supervised approaches are still inferior to supervised ones by large margins when compared on standard benchmarks. The main problem lies in the weak supervision added as photometric loss, which is defined as the photometric difference between a pixel warped from source view by estimated depth and pose and the pixel captured in the target view. Nevertheless, small photometric loss does not necessarily guarantee accurate depth and pose, especially for pixels in textureless regions. The problem can be partially solved by adding smoothness loss on depth map, which encourages first-order smoothness [4,16,15] or second-order smoothness [50,51,49,26], and forces depth propagation from discriminative regions to textureless regions. However, such propagation is with limited range and tends to cause over-smooth results around boundaries.

Considering the basic limitation is from representation, feature-metric loss is proposed to use learned feature representation for each pixel, which is explicitly constrained to be discriminative even in textureless regions. For learning feature representation, a single view reconstruction pathway is added as an auto-encoder network. To ensure loss landscapes defined on the learned feature representation having desired shapes, two additional regularizing losses are added to the auto-encoder loss, *i.e.*, discriminative loss and convergent loss. The discriminative loss encourages feature differences across pixels modeled by first-order gradients, while the convergent loss ensures a wide convergence basin by penalizing feature gradients' variances across pixels.

In total, our network architecture contains three sub-networks, *i.e.*, DepthNet and PoseNet for cross-view reconstruction, and FeatureNet for single-view reconstruction, where features generated by FeatureNet are used to define feature-metric loss for DepthNet and PoseNet.

In experiment, feature-metric loss outperforms widely used first-order and second-order smoothness losses, and improves state-of-the-art depth estimation from 0.885 to 0.925 measured by δ_1 on KITTI dataset. In addition, our method generates better egomotion estimation and results in more accurate visual odometry.

In general, our contributions are summarized as three-fold:

- Feature-metric loss is proposed for self-supervised depth and egomotion estimation.
- FeatureNet is proposed for feature representation learning for depth and egomotion estimation.
- State-of-the-art performances on depth and egomotion estimation are achieved on KITTI dataset.

2 Related Work

In this section, we review related works of self-supervised learning for two tasks, *i.e.*, monocular depth and egomotion estimation, as well as visual representation learning.

Monocular depth and egomotion estimation: SfMLearner is a pioneering work [56] for this task, where geometry estimation from DepthNet and PoseNet is supervised by photometric loss. To tackle moving objects that break the assumption of static scenes, optical flow is estimated to compensate these moving pixels [52,49,26,57], segmentation masks provided by pre-trained segmentation models are also to handle potential moving objects separately [4,30,17].

More geometric priors are also used to strengthen the self-supervised learning. Depth-normal consistency loss is proposed as as extra constraint [50,51]. 3D consistency between point clouds backprojected from adjacent views is considered in [27,5,2]. In addition, binocular videos are used for training to solve both scale ambiguity and scene dynamics [24,15,49,26], where only inference can be carried on monocular video.

In contrast to all above methods where focuses are on the geometry parts of the task, deep feature reconstruction [53] proposed to use deep features from pre-trained models to define reconstruction loss. Our method shares the same spirit, but takes a step further to explicitly learn deep features from the geometry problem under the same self-supervised learning framework.

Visual representation learning: It is of great interest of self-supervised visual representation learning for downstream tasks. Without explicitly provided labels, the losses are defined by manipulating the data itself in different ways, which could be reconstructing input data [28,45,10,32], predicting spatial transformations [9,36,37,38], coloring grayscale input images [7,21,22,54] etc. Our work belongs to reconstruct the input through an auto-encoder network. Different from previous works mainly aiming for learning better features for recognition tasks, our method is designed to learn better features for the geometry task.

3 Method

In this section, we firstly introduce geometry models with required notations, then define two reconstruction losses, one for depth and ego-motion learning, the other for feature representation learning. Finally, we present our overall pipeline and implementation details about loss settings and network architectures.

3.1 Geometry models

Camera model and depth. The camera operator $\pi : \mathbb{R}^3 \to \mathbb{R}^2$ projects a 3D point $P = (X, Y, Z)$ to a 2D pixel $p = (u, v)$ by:

$$\pi(P) = (f_x \frac{X}{Z} + c_x, f_y \frac{Y}{Z} + c_y) \tag{1}$$

where (f_x, f_y, c_x, c_y) are the camera intrinsic parameters. Similarly, a pixel p is projected to a 3D point P given its depth $D(p)$, i.e., backprojection $\pi^{-1} : \mathbb{R}^2 \times \mathbb{R} \to \mathbb{R}^3$:

$$\pi^{-1}\big(p, D(p)\big) = D(p)\Big(\frac{x - c_x}{f_x}, \frac{y - c_y}{f_y}, 1\Big)^\top \tag{2}$$

Ego-motion. Ego-motion is modeled by transformation $G \in \mathbb{SE}(3)$, together with π and π^{-1}, we can define a projective warping function $\omega : \mathbb{R}^2 \times \mathbb{R} \times \mathbb{SE}(3) \rightarrow \mathbb{R}^2$, which maps a pixel p in one frame to the other frame transformed by G:

$$\widehat{p} = \omega\big(p, D(p), G\big) = \pi\Big(G \cdot \pi^{-1}\big(p, D(p)\big)\Big) \tag{3}$$

3.2 Cross-view reconstruction

With the above geometry models, target frame I_t can be reconstructed from source frame I_s via,

$$\widehat{I}_{s\rightarrow t}(p) = I_s(\widehat{p}) \tag{4}$$

where $\widehat{p}$ is defined in Eq. 3 and depends on both depth and ego-motion. $I_t(p)$ and $I_s(\widehat{p})$ should be similar given a set of assumptions, including both depth and ego-motion are correct; the corresponding 3D point is static with Lambertian reflectance and not occluded in both views. Then, a multi-view reconstruction loss can be defined for learning depth and motion, i.e.,

$$\mathcal{L}_{s\rightarrow t} = \sum_p \ell\big(I_s(\widehat{p}), I_t(p)\big), \tag{5}$$

where $\ell(,)$ is the per-pixel loss which measures the photometric difference, i.e, photometric loss.

Though the loss works, it is fundamentally problematic since correct depth and pose is sufficient but not necessary for small photometric error, e.g., pixels in a textureless with the same photometric values can have small photometric losses even the depth and pose are wrongly estimated. The problem can be formally analysed from the optimization perspective by deriving the gradients with respect to both depth $D(p)$ and egomotion G,

$$\frac{\partial \mathcal{L}_{s\rightarrow t}}{\partial D(p)} = \frac{\partial \ell\big(I_s(\widehat{p}), I_t(p)\big)}{\partial I_s(\widehat{p})} \cdot \frac{\partial I_s(\widehat{p})}{\partial \widehat{p}} \cdot \frac{\partial \widehat{p}}{\partial D(p)}, \tag{6}$$

$$\frac{\partial \mathcal{L}_{s\rightarrow t}}{\partial G} = \sum_p \frac{\partial \ell\big(I_s(\widehat{p}), I_t(p)\big)}{\partial I_s(\widehat{p})} \cdot \frac{\partial I_s(\widehat{p})}{\partial \widehat{p}} \cdot \frac{\partial \widehat{p}}{\partial G}, \tag{7}$$

where both gradients depend on the image gradient $\frac{\partial I_s(\widehat{p})}{\partial \widehat{p}}$. For textureless region, the image gradients are close to zero which further causes zero gradients for Eq. 6 and contributes zero to Eq. 7 for egomotion estimation. In addition, locally non-smooth gradient directions are also challenging convergence due to inconsistent update directions towards minima.

Therefore, we propose to learn feature representation $\phi_s(p)$ with better gradient $\frac{\partial \phi_s(\widehat{p})}{\partial \widehat{p}}$ to overcome the above problems, and generalizes photometric loss to feature-metric loss accordingly,

$$\mathcal{L}_{s\rightarrow t} = \sum_p \ell\big(\phi_s(\widehat{p}), \phi_t(p)\big). \tag{8}$$

3.3 Single-view reconstruction

The feature representation $\phi(p)$ is also learned in self-supervised manner with single-view reconstruction through an auto-encoder network. The auto-encoder network contains an encoder for deep feature extractions from an image and an decoder to reconstruct the input image based on the deep features. The deep features are learned to encode large patterns in an image where redundancies and noises are removed. To ensure the learned representation with good properties for optimizing Eq. 8, we add two extra regularizers $\mathcal{L}_{dis}$ and $\mathcal{L}_{cvt}$ to the image reconstruction loss $\mathcal{L}_{rec}$, i.e.,

$$\mathcal{L}_s = \mathcal{L}_{rec} + \alpha\mathcal{L}_{dis} + \beta\mathcal{L}_{cvt} \tag{9}$$

where α and β are set to 1e-3 via cross validation. These three loss terms are described in detail below.

For simplicity, we denote first-order derivative and second-order derivative with respect to image coordinates by ∇^1 and ∇^2, which equals $\partial_x + \partial_y$ and $\partial_{xx} + 2\partial_{xy} + \partial_{yy}$ respectively.

Image reconstruction loss Image reconstruction loss $\mathcal{L}_{rec}$ is the standard loss function for an auto-encoder network, which requires the encoded features can be used to reconstruct its input, i.e.,

$$\mathcal{L}_{rec} = \sum_p |I(p) - I_{rec}(p)|_1 \tag{10}$$

where $I(p)$ is the input image, and $I_{rec}(p)$ is the image reconstructed from the auto-encoder network.

Discriminative loss $\mathcal{L}_{dis}$ is defined to ensure the learned features have gradients $\frac{\partial\phi(\widehat{p})}{\partial\widehat{p}}$ by explicitly encouraging large gradient, i.e.,

$$\mathcal{L}_{dis} = -\sum_p |\nabla^1\phi(p)|_1 \tag{11}$$

Furthermore, image gradients are used to emphasize low-texture regions,

$$\mathcal{L}_{dis} = -\sum_p e^{-|\nabla^1 I(p)|_1}|\nabla^1\phi(p)|_1 \tag{12}$$

where low-texture regions receive large weights.

Convergent loss $\mathcal{L}_{cvt}$ is defined to encourage smoothness of feature gradients, which ensures consistent gradients during optimization and large convergence radii accordingly. The loss is defined to penalize the second-order gradients, i.e.,

$$\mathcal{L}_{cvt} = \sum_p |\nabla^2\phi(p)|_1 \tag{13}$$

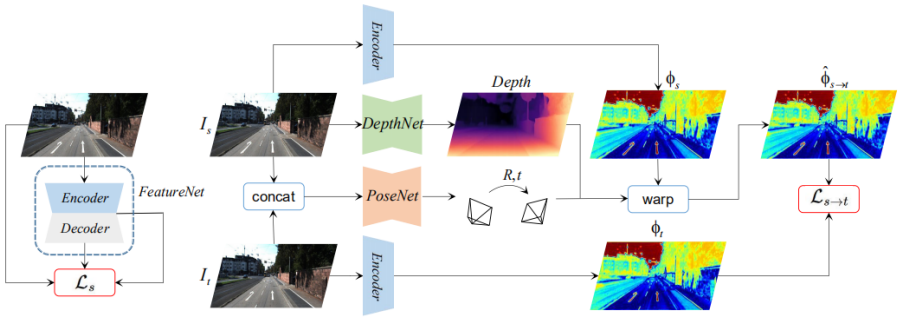

Fig. 1: An illustration of the overall framework, which contains DepthNet, PoseNet and FeatureNet for depth map prediction, egomotion prediction and feature learning respectively. FeatureNet uses $\mathcal{L}_s$ to learn require visual representation, the encoder from FeatureNet is used to extract features for cross-view reconstruction loss $\mathcal{L}_{s\to t}$.

3.4 Overall pipeline

Single-view reconstruction and cross-view reconstruction are unified to form the final framework as illustrated in Fig. 1. DepthNet is a monodepth estimator which takes the target frame as input and outputs a depth map. PoseNet is an egomotion estimator, which takes two frames from both source and target view and outputs the relative pose between them. DepthNet and PoseNet provide the geometry information to establish point-to-point correspondences for cross-view reconstruction. FeatureNet is for feature representation learning, which follows the auto-encoder architecture and supervised by single-view reconstruction loss. Features from FeatureNet are used to define the cross-view reconstruction loss.

Therefore, the total loss for the whole architecture contains two parts, where $\mathcal{L}_s$ constrains the quality of learned features through single-view reconstruction, whilst $\mathcal{L}_{s\to t}$ penalizes the discrepancy from cross-view reconstruction, i.e.,

$$\mathcal{L}_{total} = \mathcal{L}_s + \mathcal{L}_{s\to t} \tag{14}$$

Toward better performance, the proposed feature-metric loss is combined with used photometric loss, i.e.,

$$\begin{aligned} \mathcal{L}_{s\to t} = &\sum_p \mathcal{L}_{fm}\big(\phi_s(\widehat{p}), \phi_t(p)\big) \\ &+ \sum_p \mathcal{L}_{ph}(I_s(\widehat{p}), I_t(p)) \end{aligned} \tag{15}$$

where $\mathcal{L}_{fm}$ and $\mathcal{L}_{ph}$ are the feature-metric loss and photometric loss respectively. Specifically, feature-metric loss is defined by

$$\mathcal{L}_{fm} = |\phi_s(\widehat{p}) - \phi_t(p)|_1, \tag{16}$$

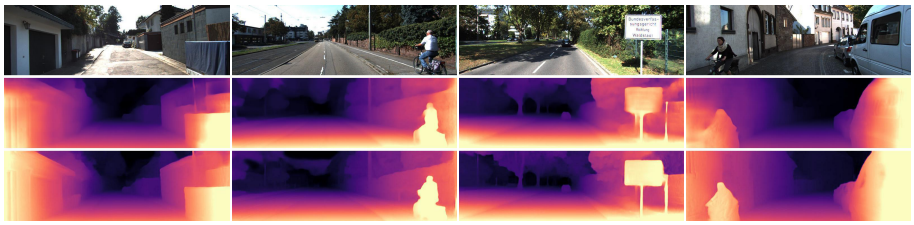

Fig. 2: Qualitative comparison between Monodepth2 [15] (second row) and our method (last row). It can be seen that we achieve better performance on the low-texture regions like walls and billboards, and finer details are present like silhouette of humans and poles.

and photometric loss is defined following [16] using a combination of L_1 and SSIM losses, i.e.,

$$\begin{aligned}\mathcal{L}_{ph} = 0.15 \sum_{p} |I_s(\widehat{p}) - I_t(p)|_1 + \\ 0.85 \frac{1 - \text{SSIM}(I_s(\widehat{p}), I_t(p))}{2}\end{aligned} \tag{17}$$

Furthermore, we resolve the occlusion problem following the practices in [15,46,6,53], where two source views are used to define the cross-view reconstruction loss,

$$\mathcal{L}'_{s\to t} = \sum_{p} \min_{s\in V} \mathcal{L}_{s\to t}\big(\phi_s(\widehat{p}), \phi_t(p)\big) \tag{18}$$

Where V is a set composed of source frames. When trained on the monocular videos, V contains the previous and posterior source frames of current target frame; when trained on the calibrated binocular videos, an extra frame of opposite stereo pair is added.

3.5 Implementation details

For FeatureNet, ResNet-50 [18] with fully-connected layer removed is used as the encoder, where deepest feature map goes through 5 downsampling stages and reduces to 1/32 resolution of input image, the decoder contains five 3×3 convolutional layers and each followed by a bilinear upsampling layer. Multi-scale feature maps from convolutional layers of the decoder are used to generate multi-scale reconstructed images, where feature map of each scale further goes through a 3×3 convolution with sigmoid function for image reconstruction. The largest feature map with 64 channels from encoder is regularized by $\mathcal{L}_{dis}$ and $\mathcal{L}_{cvt}$ and will be used for feature-metric loss.

DepthNet also adopts an encoder-decoder structure, where ResNet-50 without fully-connected layer is used as encoder and multi-scale feature maps are outputted. The decoder for depth is implemented in a cascaded refinement manner, which decodes depth maps in a top-down pathway. Specifically, multiple-scale features from encoder are used to predict maps of corresponding sizes via a 3×3

convolution followed by sigmoid, and these maps are refined in a coarse-to-fine manner towards the final depth map. Both FeatureNet and DepthNet take image size of 320×1024 as inputs.

The PoseNet is a pose estimator with a structure of ResNet-18 [18], which is modified to receive a concatenated image pair and predicts a relative pose therein. Here axis angle is chosen to represent the 3D rotation. The input resolution is 192×640. Comparing with both FeatureNet and DepthNet, PoseNet uses lower image resolution and more light-weight backbone, which observes this has no obvious influence to pose accuracy, but significantly save both memory and computation.

We adopt the setting in [15] for data preprocessing. Our models are implemented on PyTorch [39] with distributed computing, and trained for 40 epochs using Adam [20] optimizer, with a batch size of 2, on the 8 GTX 1080Ti GPUs. The learning rate is gradually warmed up to $1e^{-4}$ in 3 steps, where each step increases learning rate by $1e^{-4}/3$ in 500 iterations. After warmping, learning rate $1e^{-4}$ is used for the first 20 epochs and halved twices at 20th and 30th epoch. As for online refinement technique we used during testing, we follow the practice proposed by [5,4]. We keep the model training while performing inference. The batch size is set to 1. Each batch consists of the test image and its two adjacent frames. Online refinement is performed for 20 iterations on one test sample with the same setting introduced before. No data augmentation is used in the inference phase.

4 Experiments

In this section we show extensive experiments for evaluating the performance of our approach. We make a fair comparison on KITTI 2015 dataset [14] with prior art on both single view depth and visual odometry estimation tasks. And detailed ablation studies of our approach are done to show the effectiveness of the **feature-metric loss**.

KITTI 2015 dataset contains videos in 200 street scenes captured by RGB cameras, with sparse depth ground truths captured by Velodyne laser scanner. We follow [56] to remove static frames as pre-processing step. We use the Eigen split of [11] to divide KITTI raw data, and resulting in 39,810 monocular triplets for training, 4,424 for validation and 697 for testing.

For depth evaluation, we test our depth model on divided 697 KITTI testing data. For odometry evaluation, we test our system to the official KITTI odometry split which containing 11 driving sequences with ground truth odometry obtained through the IMU and GPS readings. Following previous works [53,2,56], we train our model on the sequence 00-08 and use the sequence 09-10 for testing.

4.1 Depth evaluation.

Performance metrics. Standard metrics are used for depth evaluation, as shown in Tab. 1. During evaluation, depth is capped to 80m. For the methods

Abs Rel : $\frac{1}{\lvert D\rvert}\sum_{d\in D}\lvert d^*-d\rvert/d^*$	**RMSE** : $\sqrt{\frac{1}{\lvert D\rvert}\sum_{d\in D}\lvert\lvert d^*-d\rvert\rvert^2}$
Sq Rel : $\frac{1}{\lvert D\rvert}\sum_{d\in D}\lvert\lvert d^*-d\rvert\rvert^2/d^*$	**RMSE log** : $\sqrt{\frac{1}{\lvert D\rvert}\sum_{d\in D}\lvert\lvert logd^*-logd\rvert\rvert^2}$
$\delta_{\mathbf{t}}$: $\frac{1}{\lvert D\rvert}\lvert\{d\in D\lvert\, max(\frac{d^*}{d},\frac{d}{d^*})\ <1.25^t\}\rvert\times 100\%$	

Table 1: Performance metrics for depth evaluation. d and d^* respectively denotes predicted and ground truth depth, D presents a set of all the predicted depth values of an image, $|.|$ returns the number of the elements in the input set.

trained on monocular videos, the depth is defined up to scale factor [56], which is computed by

$$scale = median(D_{gt})/median(D_{pred}) \tag{19}$$

For evaluation, those predicted depth maps are multiplied by computed *scale* to match the median with the ground truth, this step is called **median scaling**.

Comparison with state-of-the-art. Tab. 2 shows performances of current state-of-the-art approaches for monocular depth estimation. They are trained on different kinds of data — monocular videos (M), stereo pairs (S), binocular videos (MS) and labelled single images (Sup), while all of them are tested with single image as input.

We achieve the best performance compared to all self-supervised methods, no matter which training data is used. Our method achieves more significant improvement in the performance metric Sq Rel. According to Tab. 1, this metric penalizes more on large errors in short range, where more textureless regions exist due near objects are large in images and our method handles well. The closest results in self-supervised methods are from DepthHint [47], which uses the same input size but adds an extra post processing step. It utilizes a traditional stereo matching method — SGM [19] to provide extra supervisory signals for training, since SGM is less likely to be trapped by local minimums. However, in its settings, the object function of SGM is still photometric loss, the drawbacks of photometric loss are still inevitable. In contrast, proposed feature-metric loss will largely avoid the interference of local minimums.

Moreover, compared with state-of-the-art **supervised** methods [13,23], which achieve top performances on the KITTI depth prediction competition, our model with online refinement technique even exceeds in many metrics. Our advantage over supervised methods is that the gap between the distributions of training and testing data does exist, we can make full use of online refinement technique. What is more, as shown in Sec. 4.3, the introduction of feature-metric loss can obtain more performance gain from online refinement technique.

Fig. 2 shows the qualitative results. Compared with state-of-the-art method MonoDepth2 [15], we achieve better performance on low-texture regions and finer details, e.g., walls, billboards, silhouette of humans and poles.

However, MonoDepth2 is built on the photometric loss, which is easily trapped by local minimums especially on low-texture regions like walls and billboards. In contrast, the introduction of feature-metric loss leads the network into jumping

Method	Train	The lower the better				The higher the better		
		Abs Rel	Sq Rel	RMSE	RMSE log	δ_1	δ_2	δ_3
SfMLearner [56]	M	0.208	1.768	6.958	0.283	0.678	0.885	0.957
DNC [51]	M	0.182	1.481	6.501	0.267	0.725	0.906	0.963
Vid2Depth [27]	M	0.163	1.240	6.220	0.250	0.762	0.916	0.968
LEGO [50]	M	0.162	1.352	6.276	0.252	0.783	0.921	0.969
GeoNet [52]	M	0.155	1.296	5.857	0.233	0.793	0.931	0.973
DF-Net [57]	M	0.150	1.124	5.507	0.223	0.806	0.933	0.973
DDVO [46]	M	0.151	1.257	5.583	0.228	0.810	0.936	0.974
EPC++ [26]	M	0.141	1.029	5.350	0.216	0.816	0.941	0.976
Struct2Depth [4]	M	0.141	1.036	5.291	0.215	0.816	0.945	0.979
SIGNet [30]	M	0.133	0.905	5.181	0.208	0.825	0.947	0.981
CC [43]	M	0.140	1.070	5.326	0.217	0.826	0.941	0.975
LearnK [17]	M	0.128	0.959	5.230	0.212	0.845	0.947	0.976
DualNet [55]	M	0.121	0.837	4.945	0.197	0.853	0.955	0.982
SuperDepth [40]	M	0.116	1.055	-	0.209	0.853	0.948	0.977
Monodepth2 [15]	M	0.115	0.882	4.701	0.190	0.879	0.961	0.982
Ours	M	**0.104**	**0.729**	**4.481**	**0.179**	**0.893**	**0.965**	**0.984**
Struct2Depth [4]	M*	0.109	0.825	4.750	0.187	0.874	0.958	**0.983**
GLNet [5]	M*	0.099	0.796	4.743	0.186	0.884	0.955	0.979
Ours	M*	**0.088**	**0.712**	**4.137**	**0.169**	**0.915**	**0.965**	0.982
Dorn [13]	Sup	0.099	0.593	3.714	0.161	0.897	0.966	0.986
BTS [23]	Sup	0.091	0.555	4.033	0.174	0.904	0.967	0.984
MonoDepth [16]	S	0.133	1.142	5.533	0.230	0.830	0.936	0.970
MonoDispNet [48]	S	0.126	0.832	**4.172**	0.217	0.840	0.941	0.973
MonoResMatch [44]	S	0.111	0.867	4.714	0.199	0.864	0.954	0.979
MonoDepth2 [15]	S	0.107	0.849	4.764	0.201	0.874	0.953	0.977
RefineDistill [41]	S	**0.098**	0.831	4.656	0.202	0.882	0.948	0.973
UnDeepVO [24]	MS	0.183	1.730	6.570	0.268	-	-	-
DFR [53]	MS	0.135	1.132	5.585	0.229	0.820	0.933	0.971
EPC++ [26]	MS	0.128	0.935	5.011	0.209	0.831	0.945	0.979
MonoDepth2 [15]	MS	0.106	0.818	4.750	0.196	0.874	0.957	0.979
DepthHint [47]	MS†	0.100	0.728	4.469	0.185	0.885	0.962	**0.982**
Ours	MS	0.099	**0.697**	4.427	**0.184**	**0.889**	**0.963**	**0.982**
Ours	MS*	0.079	0.666	3.922	0.163	0.925	0.970	0.984

Table 2: Comparison of performances are reported on the KITTI dataset. Best results are in bold, second best are underlined. M: trained on monocular videos. S: trained on stereo pairs. MS: trained on calibrated binocular videos. Sup: trained on labelled single images. $*$: using the online refinement technique [4], which advocated keeping the model training while performing inference. †: using post processing steps.

out of local minimums, since our features are designed to form a desirable loss for easier optimization.

4.2 Odometry evaluation

Performance metric. Average translational root mean square error drift (t_{err}) and average rotational root mean square error drift (r_{err}) on length of 100 m - 800

Method	Seq. 09		Seq. 10	
	t_{err}	r_{err}	t_{err}	r_{err}
ORB-SLAM [33]	15.30	0.26	3.68	0.48
SfMLearner [56]	17.84	6.78	37.91	17.78
DFR [53]	11.93	3.91	12.45	3.46
MonoDepth2 [15]	10.85	2.86	11.60	5.72
NeuralBundler [25]	**8.10**	2.81	12.90	**3.17**
SC-SfMlearner [2]	8.24	2.19	10.70	4.58
Ours	8.75	**2.11**	**10.67**	4.91

Table 3: Comparison of performances are reported on the KITTI odometry dataset [14]. Best results are in bold.

m are adopted for evaluation. For the methods who suffer from scale ambiguity, one global scale that best align the whole sequence is used.

Comparison with state-of-the-art. As shown in Tab. 3, we report the performance of ORB-SLAM[33] as a reference and compare with recent deep methods. our method gets top performances in two metrics and comparable performance in the rest metrics compared to other deep learning methods. When compared to traditional SLAM method [33], our translation performance is comparable, while in the rotation estimation we still fall short like other deep learning methods. We believe that it is because the bundle adjustment of the traditional SLAM method can optimize subtler rotation errors along a long sequence which can't be observed in a small sequence used by current deep learning based methods. Moreover current reconstruction process may be not sensible to variation of rotation [3].

4.3 Ablation study

To get a better understanding of the contribution of proposed losses—feature-metric loss, discriminative loss and convergent loss—to the overall performance, we perform an ablation study in Tab. 4.

The losses for cross-view reconstruction. In Tab. 4a, different components of $\mathcal{L}_{s\to t}$ have been tried. The smoothness losses which are widely used are used as baselines:

$$\mathcal{L}^i_{ds} = \sum_p e^{-|\nabla^i I(p)|_1} |\nabla^i \widehat{D}(p)|_1 \tag{20}$$

where $\widehat{D}(p) = D(p)/\bar{D}$, this operation is the mean normalization technique advocated by [46]. i denotes the order of the derivatives. These smoothness losses are used as baselines to verify the effectiveness of the feature-metric loss.

Compared with smoothness losses, feature-metric loss leads to much better effect. We can see that a biggest performance boost is gained by introducing the feature-metric loss. As we discussed before, the propagation range of smoothness losses is limited, in contrast, the feature-metric loss enable a long-range

Method	OR	The lower the better				The higher the better		
		Abs Rel	Sq Rel	RMSE	RMSE log	δ_1	δ_2	δ_3
$\mathcal{L}_{ph} + \mathcal{L}_{ds}^1$	×	0.105	0.748	4.835	0.191	0.878	0.956	0.979
$\mathcal{L}_{ph} + \mathcal{L}_{ds}^2$	×	0.103	0.740	4.754	0.187	0.881	0.959	0.981
$\mathcal{L}_{ph} + \mathcal{L}_{ds}^1 + \mathcal{L}_{ds}^2$	×	0.103	0.735	4.554	0.187	0.883	0.961	0.981
$\mathcal{L}_{ph} + \mathcal{L}_{ds}^1 + \mathcal{L}_{ds}^2$	✓	0.088	0.712	4.237	0.175	0.905	0.965	0.982
$\mathcal{L}_{ph} + \mathcal{L}_{fm}$	×	0.099	0.697	4.427	0.184	0.889	0.963	0.982
$\mathcal{L}_{ph} + \mathcal{L}_{fm}$	✓	**0.079**	**0.666**	**3.922**	**0.163**	**0.925**	**0.970**	**0.984**

(a) **Different loss combinations in $\mathcal{L}_{s\to t}$** (Eq. 8), the term 'OR' denotes whether the online refinement [4] is used.

Loss	The lower the better				The higher the better			Seq. 09		Seq. 10	
	Abs Rel	Sq Rel	RMSE	RMSE log	δ_1	δ_2	δ_3	t_{err}	r_{err}	t_{err}	r_{err}
$\mathcal{L}_{rec}$	0.105	0.739	4.585	0.191	0.883	0.961	**0.982**	4.30	1.18	8.50	4.06
$\mathcal{L}_{rec} + \mathcal{L}_{dis}$	0.103	0.723	4.535	0.187	0.884	0.961	**0.982**	4.10	1.07	8.03	3.94
$\mathcal{L}_{rec} + \mathcal{L}_{cvt}$	0.100	0.721	4.474	0.187	0.885	0.962	**0.982**	3.29	1.16	5.91	3.48
$\mathcal{L}_{rec} + \mathcal{L}_{dis} + \mathcal{L}_{cvt}$	**0.099**	**0.697**	**4.427**	**0.184**	**0.889**	**0.963**	**0.982**	**3.07**	**0.89**	**3.83**	**1.78**

(b) **Different loss combinations in $\mathcal{L}_s$** (Eq. 9).

Table 4: The ablation study of different loss settings of our work.

propagation, since it has a large convergence radius. We also observe that when feature-metric loss can benefit more from the performance gain provided by online refinement than other loss combination. Higher performance gain is attributed to better supervised signal provided by feature-metric loss during online refinement phase, where incorrect depth values can be appropriately penalized with larger losses based on more discriminative features.

The losses for single-view reconstruction. Tab.4b shows that the model without any of our contributions performs the worst. When combined together, all our components lead to a significant improvement.

And as shown in right part of Tab. 4b, although small deviations are less obvious in some metrics of the depth evaluation, small errors will be magnified via accumulation and propagation during trajectory prediction, big differences are shown in the odometry evaluation. Note that different from previous odometry evaluation, we directly applied the model trained on the kitti raw data to sequence 09-10 to get t_{err} and r_{err}.

Merely using $\mathcal{L}_{rec}$ gets similar performance as merely using photometric loss (the third row in Tab. 4a), since it plays a similar role as the photometric loss at textureless regions. Results get better when equipped with $\mathcal{L}_{dis}$, since discrimination at low-texture regions is improved. Best performance is achieved when added $\mathcal{L}_{cvt}$, which means discrimination is not enough, a correct optimization direction is also important.

Visualization analysis. In order to see whether learned visual representations have promised properties, we visualize it in Fig. 3. The feature maps learned with different loss combinations: $\mathcal{L}_{rec}$, $\mathcal{L}_{rec} + \mathcal{L}_{dis}$ and $\mathcal{L}_{rec} + \mathcal{L}_{dis} + \mathcal{L}_{cvt}$ are sequentially shown from the second to the fourth row. Although we require our feature to be discriminative, this effect is not sufficient to be shown in a

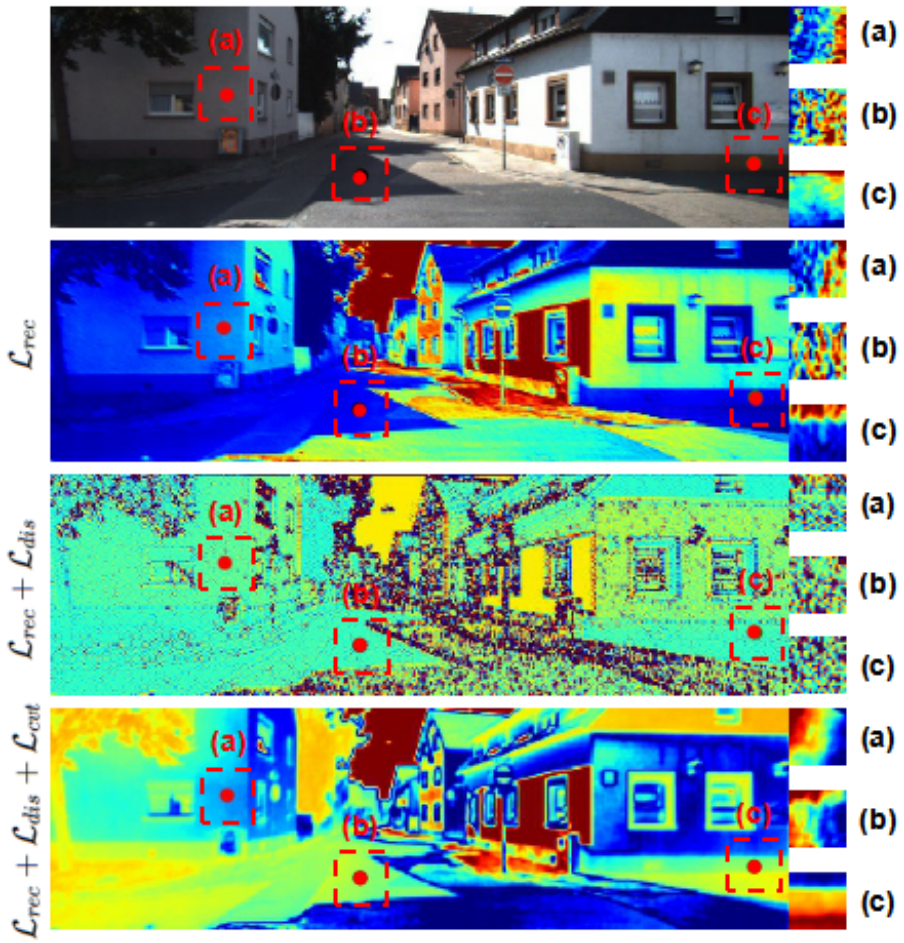

Fig. 3: A visualization of a learned visual representation, which is achieved by selecting one principle channel through PCA decomposition, then showing the feature map as a heat map, hotter color indicates a higher feature value. First row shows a typical image which is full of textureless regions like walls and shadows. The visualization of corresponding feature maps is shown in second to fourth rows. The feature maps are respectively learned with different loss combinations, which sequentially correspond with the settings in the first three rows in Tab. 4b. In order to get a better understanding, we crop three typical textureless regions as shown in (a-c), cropped feature maps are visualized according to the dynamic range after cropping.

large view, since the gap between the features of different sorts are much larger than that of spatially adjacent features. Therefore, we cropped three typical textureless regions, and visualize them again according to the dynamic range after cropping.

We can see that merely using $\mathcal{L}_{rec}$ get small variations at textureless regions. The close-ups of original images are similar to feature maps only trained with $\mathcal{L}_{rec}$, which verifies the proposed losses in improving feature representations. The feature map learned with $\mathcal{L}_{rec} + \mathcal{L}_{dis}$ is not smooth and disordered, since $\mathcal{L}_{dis}$ overemphasizes the discrepancy between adjacent features, the network degenerates to form a landscape of a zigzag shape. This phenomenon can be approved by the results in the second row of Tab. 4b, which is only slightly higher than merely using $\mathcal{L}_{rec}$.

A desired landscape for feature maps is a smooth slope, in this way, feature-metric loss will be able to form a basin-like landscape. The feature map learned with all the proposed losses approximates this ideal landscape, from zoom-in views we can see a clear and smooth transition along a certain direction. On this landscape, gradient descent approaches can move smoothly toward optimal solutions.

5 Conclusion

In this work, feature-metric loss is proposed for self-supervised learning of depth and egomotion, where feature representation is additionally learned with two extra regularizers to ensure convergence towards correct depth and pose. The whole framework is end-to-end trainable in self-supervised setting, and achieves state-of-the-art depth estimation which is even comparable to supervised learning methods. Furthermore, visual odometry based on estimated egomotion also significantly outperforms previous state-of-the-art methods.

Acknowledgements This research is supported by Beijing Science and Technology Project (No. Z181100008918018).

References

1. Andraghetti, L., Myriokefalitakis, P., Dovesi, P.L., Luque, B., Poggi, M., Pieropan, A., Mattoccia, S.: Enhancing self-supervised monocular depth estimation with traditional visual odometry. arXiv:1908.03127 (2019)
2. Bian, J.W., Li, Z., Wang, N., Zhan, H., Shen, C., Cheng, M.M., Reid, I.: Unsupervised scale-consistent depth and ego-motion learning from monocular video. NeurIPS (2019)
3. Bian, J.W., Zhan, H., Wang, N., Chin, T.J., Shen, C., Reid, I.: Unsupervised depth learning in challenging indoor video: Weak rectification to rescue. arXiv:2006.02708 (2020)
4. Casser, V., Pirk, S., Mahjourian, R., Angelova, A.: Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In: AAAI (2019)
5. Chen, Y., Schmid, C., Sminchisescu, C.: Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. In: ICCV (2019)
6. Cheng, X., Zhong, Y., Dai, Y., Ji, P., Li, H.: Noise-aware unsupervised deep lidar-stereo fusion. In: CVPR (2019)
7. Deshpande, A., Rock, J., Forsyth, D.: Learning large-scale automatic image colorization. In: ICCV (2015)
8. DeSouza, G.N., Kak, A.C.: Vision for mobile robot navigation: A survey. TPAMI (2002)
9. Doersch, C., Gupta, A., Efros, A.A.: Unsupervised visual representation learning by context prediction. In: ICCV (2015)
10. Donahue, J., Krähenbühl, P., Darrell, T.: Adversarial feature learning. arXiv preprint arXiv:1605.09782 (2016)
11. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: NeurIPS (2014)
12. Engel, J., Koltun, V., Cremers, D.: Direct sparse odometry. TPAMI (2017)
13. Fu, H., Gong, M., Wang, C., Batmanghelich, K., Tao, D.: Deep ordinal regression network for monocular depth estimation. In: CVPR (2018)
14. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: CVPR (2012)
15. Godard, C., Mac Aodha, O., Brostow, G.: Digging into self-supervised monocular depth estimation. In: ICCV (2019)
16. Godard, C., Mac Aodha, O., Brostow, G.J.: Unsupervised monocular depth estimation with left-right consistency. In: CVPR (2017)
17. Gordon, A., Li, H., Jonschkowski, R., Angelova, A.: Depth from videos in the wild: Unsupervised monocular depth learning from unknown cameras. In: ICCV (2019)
18. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
19. Hirschmuller, H.: Stereo processing by semiglobal matching and mutual information. TPAMI (2007)
20. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv:1412.6980 (2014)
21. Larsson, G., Maire, M., Shakhnarovich, G.: Learning representations for automatic colorization. In: ECCV (2016)
22. Larsson, G., Maire, M., Shakhnarovich, G.: Colorization as a proxy task for visual understanding. In: CVPR (2017)

23. Lee, J.H., Han, M.K., Ko, D.W., Suh, I.H.: From big to small: Multi-scale local planar guidance for monocular depth estimation. arXiv:1907.10326 (2019)
24. Li, R., Wang, S., Long, Z., Gu, D.: Undeepvo: Monocular visual odometry through unsupervised deep learning. In: ICRA (2018)
25. Li, Y., Ushiku, Y., Harada, T.: Pose graph optimization for unsupervised monocular visual odometry. arXiv:1903.06315 (2019)
26. Luo, C., Yang, Z., Wang, P., Wang, Y., Xu, W., Nevatia, R., Yuille, A.: Every pixel counts++: Joint learning of geometry and motion with 3d holistic understanding. arXiv:1810.06125 (2018)
27. Mahjourian, R., Wicke, M., Angelova, A.: Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In: CVPR (2018)
28. Masci, J., Meier, U., Cireşan, D., Schmidhuber, J.: Stacked convolutional auto-encoders for hierarchical feature extraction. In: ICANN (2011)
29. Mayer, N., Ilg, E., Hausser, 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: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4040–4048 (2016)
30. Meng, Y., Lu, Y., Raj, A., Sunarjo, S., Guo, R., Javidi, T., Bansal, G., Bharadia, D.: Signet: Semantic instance aided unsupervised 3d geometry perception. In: CVPR (2019)
31. Menze, M., Geiger, A.: Object scene flow for autonomous vehicles. In: CVPR (2015)
32. Mescheder, L., Nowozin, S., Geiger, A.: Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks. In: ICML (2017)
33. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: Orb-slam: a versatile and accurate monocular slam system. TR (2017)
34. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: Orb-slam: a versatile and accurate monocular slam system. IEEE transactions on robotics **31**(5), 1147–1163 (2015)
35. Newcombe, R.A., Lovegrove, S.J., Davison, A.J.: Dtam: Dense tracking and mapping in real-time. In: ICCV (2011)
36. Noroozi, M., Favaro, P.: Unsupervised learning of visual representations by solving jigsaw puzzles. In: ECCV (2016)
37. Noroozi, M., Pirsiavash, H., Favaro, P.: Representation learning by learning to count. In: ICCV (2017)
38. Noroozi, M., Vinjimoor, A., Favaro, P., Pirsiavash, H.: Boosting self-supervised learning via knowledge transfer. In: CVPR (2018)
39. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: NeurIPS-W (2017)
40. Pillai, S., Ambrus, R., Gaidon, A.: Superdepth: Self-supervised, super-resolved monocular depth estimation. In: ICRA (2019)
41. Pilzer, A., Lathuilière, S., Sebe, N., Ricci, E.: Refine and distill: Exploiting cycle-inconsistency and knowledge distillation for unsupervised monocular depth estimation. In: CVPR (2019)
42. Pire, T., Fischer, T., Castro, G., De Cristóforis, P., Civera, J., Berlles, J.J.: S-ptam: Stereo parallel tracking and mapping. Robotics and Autonomous Systems **93**, 27–42 (2017)
43. Ranjan, A., Jampani, V., Kim, K., Sun, D., Wulff, J., Black, M.J.: Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In: CVPR (2019)
44. Tosi, F., Aleotti, F., Poggi, M., Mattoccia, S.: Learning monocular depth estimation infusing traditional stereo knowledge. In: CVPR (2019)

45. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: ICML (2008)
46. Wang, C., Buenaposada, J.M., Zhu, R., Lucey, S.: Learning depth from monocular videos using direct methods. In: CVPR (2018)
47. Watson, J., Firman, M., Brostow, G.J., Turmukhambetov, D.: Self-supervised monocular depth hints. In: ICCV (2019)
48. Wong, A., Hong, B.W., Soatto, S.: Bilateral cyclic constraint and adaptive regularization for unsupervised monocular depth prediction. In: CVPR (2019)
49. Yang, Z., Wang, P., Wang, Y., Xu, W., Nevatia, R.: Every pixel counts: Unsupervised geometry learning with holistic 3d motion understanding. In: ECCV-w (2018)
50. Yang, Z., Wang, P., Wang, Y., Xu, W., Nevatia, R.: Lego: Learning edge with geometry all at once by watching videos. In: CVPR (2018)
51. Yang, Z., Wang, P., Xu, W., Zhao, L., Nevatia, R.: Unsupervised learning of geometry with edge-aware depth-normal consistency. In: AAAI (2018)
52. Yin, Z., Shi, J.: GeoNet: Unsupervised learning of dense depth, optical flow and camera pose. In: CVPR (2018)
53. 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)
54. Zhang, R., Isola, P., Efros, A.A.: Colorful image colorization. In: ECCV (2016)
55. Zhou, J., Wang, Y., Qin, K., Zeng, W.: Unsupervised high-resolution depth learning from videos with dual networks. In: ICCV (2019)
56. Zhou, T., Brown, M., Snavely, N., Lowe, D.G.: Unsupervised learning of depth and ego-motion from video. In: CVPR (2017)
57. Zou, Y., Luo, Z., Huang, J.B.: Df-net: Unsupervised joint learning of depth and flow using cross-task consistency. In: ECCV (2018)