An Analysis of Feature-metric Loss on Self-supervised Monocular Depth Estimation

Richard Lin
Stanford University
yifengl@stanford.edu

Michael Lin
Stanford University
mlin4@stanford.edu

Abstract

Phomometric loss is widely used for monocular depth estimation, but the loss is hard to optimize due to plateau landscapes for pixels in textureless regions and multiple local minimas for less discriminative pixels. FeatureNet, the new state-of-the-art model, introduced feature-metric loss to address this issue. In this work, we apply feature-metric loss to the previous state-of-the-art model, Monodepth2, and explore the efficacy of feature-metric loss. We evaluate our results on the KITTI dataset and the indoor NYU depth dataset. We find the addition of feature-metric loss generally improves the model's performance but not consistently, and we also see that model performance doesn't always correlate with the quality of the feature representations.

Our implementations can be found in the following repositories:
https://github.com/richardlyf/nyuv2-python-toolbox
https://github.com/richardlyf/monodepth2
https://github.com/richardlyf/FeatDepth

1. Introduction

Recently, there has been significant research carried out in regards to self-supervised monocular depth estimation. One such paper, "Feature-metric Loss for Self-supervised Learning of Depth and Egomotion" [4], recently achieved state-of-the-art, utilizing feature-metric loss on learned feature representation in addition to photometric loss to create more accurate depth and pose estimations.

The addition of feature-metric loss addresses the problem where photometric loss is unable to optimize over textureless regions and non-discriminative pixels. And since the feature representations are pretrained in a self-supervised manner, feature-metric loss can easily be included in other monocular depth estimation models and improve their performances at very little cost.

In this work, we explore the efficacy of feature-metric loss, and we do so by employing feature-metric loss in the Monodepth2 [2] model, which uses photometric loss. We evaluate our results on the KITTI dataset as in the original papers, and we also evaluate the model's performance on the NYU depth dataset, where textureless regions are prevelant in indoor scenes.

2. Related Work

In this section, we discuss the implementation and contribution of both Monodepth2 and FeatureNet.

2.1. Monodepth2

Monodepth2 [2] was previously considered the state-of-the-art model for outdoor monocular depth estimation. The model is trained using self-supervision, where a source image is used to directly predict depth or disparity, and this depth information is used to project the image from the source frame into the target frame. The reprojected image is then compared to the ground truth target image, and the objective is to minimize the photometric reprojection error[5].

The model supports training using consecutive monocular video frames (M), calibrated stereo image pairs (S), or both (MS). In the case of training with only monocular data, the source image is the video frame at time t, and the target image is the next frame at $t + 1$. The model uses a pose estimation network to estimate the transforms between the two frames, and the estimated pose is used to reproject the source image. For stereo training, the source and target frames are the left right images, and the cameras are calibrated so the source image can be reprojected directly using the intrinsic and extrinsic parameters.

To address issues with out-of-view pixels and occlusions, [2] takes the minimum of the reprojection loss, instead of using the average over all pixels. The model

also auto-masks out the stationary pixels to account for the common assumption in monocular depth estimation that the camera is moving and the scene is static. Multi-scale estimation is also introduced to mitigate the issue where 'holes' appear in the depth estimates where the region has low texture. The architecture of the model is show in 1.

2.2. FeatureNet

FeatureNet[4], recently accepted by ECCV 2020, achieved new state-of-the-art results on the KITTI dataset for outdoor monocular depth estimation. Similar to [2], FeatureNet also accepts training using monocular videos, stereo image pairs, or both. The main contribution of this paper is the introduction of feature-metric loss, which addresses the issue of photometric loss failing on low-texture regions. Though photometric loss is effective in most cases, it is problematic because low-texture regions with similar photometric values may result in small photometric losses even when the depths and poses are wrongly estimated.

Feature-metric loss deals with this problem by computing loss from the reprojection of learned feature representations. The feature representations capture image features and focus on low image gradient regions, and the loss itself is computed in a similar way to photometric loss, where the images features from the source images are reprojected and directly compared to the features extracted from the target image.

The feature representations are learned in a self-supervised manner with singleview reconstruction through a resnet-based auto-encoder. In addition to the image reconstruction loss, discriminative loss and convergent loss are added to force the model to focus on useful features. The discriminative loss is defined to ensure the learned feature representations have large gradients and put extra emphasis on low-texture regions. The convergent loss is defined to encourage smoothness of feature gradients, which ensures consistent gradients during optimization.

After training the auto-encoder, the encoder weights are frozen and then used to train the depth estimation model. The feature encoder doesn't update its weights during the depth estimation training.

3. Approach

In this work, we first setup our baselines by replicating the results in the papers using their publicly released code, training on the KITTI dataset. We specifically look at the models trained using monocular videos and stereo image pairs. After doing so, we adapt and retrain these models on the indoor NYU dataset to see how these models perform in an indoor setting. We only evaluate the models on monocular video training data for the NYU dataset because stereo image pairs are not provided. Also, since it's harder to estimate camera poses with many textureless regions in indoor datasets, we expect Monodepth2 to perform worse on the NYU dataset.

We then add feature-metric loss to the Monodepth2 model and repeat the experiments on both KITTI and NYU datasets. For experiments performed on the KITTI dataset, the models use the pretrained weights for the feature encoder provided by [4]. For the NYU experiments, we trained our own feature encoder using the NYU dataset. We expect all results, where feature-metric loss is included, to outperform their baseline counterparts.

3.1. Dataset

We look at two different datasets in this project: the Outdoor KITTI dataset [1] and the NYU Depth Dataset V2 [3], both mentioned above.

KITTI is a dataset that includes almost 8000 training images that come from autonomous driving vehicles and several videos over a span of a few days. These are represented as color mono and stereo sequences stored in the .png file format.

The NYU Depth Dataset V2 is also a set of video sequences, but from an indoor dataset rather than from autonomous driving vehicles. Its labeled dataset contains 1500 densely labeled pairs of aligned RGB and depth images, while the raw dataset contains approximately 400k unlabeled frames.

For KITTI, we are using the eigen split described in the Monodepth2 paper. For NYU, we are using a randomly sampled subset of the raw video frames. We train on about 40k frames and evaluate the model on the labeled dataset with ground-truth depth information.

Since the NYU dataset is processed in MATLAB, we wrote our own data processing code in Python3, and here's the link to the repo:
https://github.com/richardlyf/nyuv2-python-toolbox

3.2. Evaluation

Qualitatively, we compare the depth maps generated by all three models to see which one performs better. Specifically, we look at examples with textureless regions and compare if the feature metric improves the depth estimation.

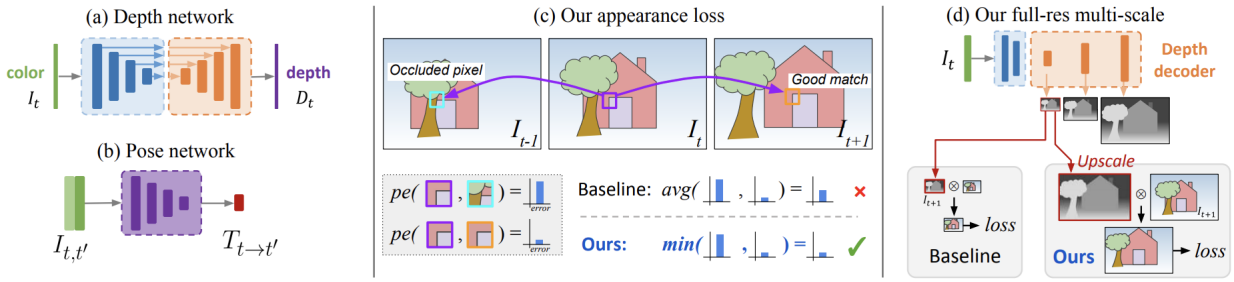

Figure 1. Monodepth2 model, as described in [2]

Quantitatively, we use the standard metrics for depth estimation as defined in the two papers:

$$\textbf{Abs Rel}: \frac{1}{|D|}\sum_{d\in D}|d^* - d|/d^*$$

$$\textbf{Sq Rel}: \frac{1}{|D|}\sum_{d\in D}||d^* - d||^2/d^*$$

$$\textbf{RMSE}: \sqrt{\frac{1}{|D|}\sum_{d\in D}||d^* - d||^2}$$

$$\textbf{Sq Rel}: \sqrt{\frac{1}{|D|}\sum_{d\in D}||logd^* - logd||^2}$$

$$\delta_t: \frac{1}{|D|}|\{d\in D|max(\frac{d^*}{d},\frac{d}{d^*}) < 1.25^t\}| \times 100\%$$

$$t\in\{1,2,3\}$$

d and d^* respectively denotes predicted and ground truth depth, D presents a set of all the predicted depth values of an image.

3.3. Implementation details

The models are trained on Google Cloud VM with 1 NVIDIA Tesla P100 GPU. We used PyTorch 0.4, consistent with the original Monodepth2 framework. We have tried to switch the model to PyTorch 1.1+ but that resulted in the model diverging rapidly. Due to limitations in time and compute resources (GCP constantly reporting not having enough GPUs), we were only able to train our models up to 5 epochs, which took roughly a day. The full model requires 20 epochs of training. When the pretrained feature encoder is not used, we train our own feature encoder for 5 epochs, which took half a day. We used ResNet50 as the backbone for the feature encoder, and ResNet18 as the backbone for the encoder of the depth estimation model. The input images are scaled down to 640×192 for the KITTI dataset, and 480×192 for the NYU dataset.

4. Experiments

As we discussed in the previous sections, we are able to build and train models in several different ways. But to put it more concretely, there are three main variables that we want to take a deeper dive in: 1) photometric vs. photometric and feature-metric loss, 2) KITTI (outdoor) vs. NYU Depth (indoor) datasets, and 3) mono vs. stereo inputs.

First, for photometric vs. photometric and feature-metric loss, we replicate the Monodepth2 paper using the KITTI dataset and use that as our baseline. This is the first row in Table 1, below. Then, we add feature-metric loss to it (even rows in Table 1). In this experiment, we directly compare how effective the feature-metric loss is.

Secondly, for KITTI vs. NYU Depth Dataset, we want to see how the models perform on an indoor dataset. We repeat all experiments on both datasets because we want to know if the models can be generalized to other domains, and if feature-metric loss can still be effective.

Finally, for Mono vs. Stereo, we want to see if feature-metric loss is affected by the mode of training. Just like our investigation for KITTI vs. NYU, this looks at how different inputs can affect model performance. This is especially relevant, since FeatureNet does not report their results on pure stereo training.

4.1. Quantitative Results

Our results are displayed in Table 1. Every single model is only trained on 5 epochs, due to time and resource constraints. This is important to note, as these results are very much a rough estimates and don't represent how the models would actually perform if they are run through to completion.

Through a basic scan of the table, we can see that for KITTI, there was a split in whether photometric with

Quantitative Results									
Method	dataset	train	Lower the better				Higher the better		
			abs_rel	sq_rel	rmse	rmse_log	a1	a2	a3
Photometric	KITTI	M	**0.125**	**0.938**	**4.849**	**0.197**	**0.859**	**0.956**	**0.980**
Photometric & feature-metric*	KITTI	M	0.126	0.947	4.867	0.200	0.858	0.955	**0.980**
Photometric	KITTI	S	0.115	0.902	4.982	0.206	0.856	**0.949**	**0.977**
Photometric & feature-metric*	KITTI	S	**0.113**	**0.838**	**4.880**	**0.205**	**0.858**	**0.949**	**0.977**
Photometric	NYU	M	0.345	0.512	1.139	0.528	0.469	0.740	0.859
Photometric & feature-metric†	NYU	M	**0.321**	**0.435**	**1.038**	**0.463**	**0.502**	**0.768**	**0.886**

Table 1. Comparison of performances are reported on models trained on 5 epochs from scratch. Values in bold are better. M: trained on monocular videos. S: trained on stereo pairs. ∗: Using the pretrained autoencoder weights provded by [4]. †: Using the autoencoder trained for 5 epochs from scratch.

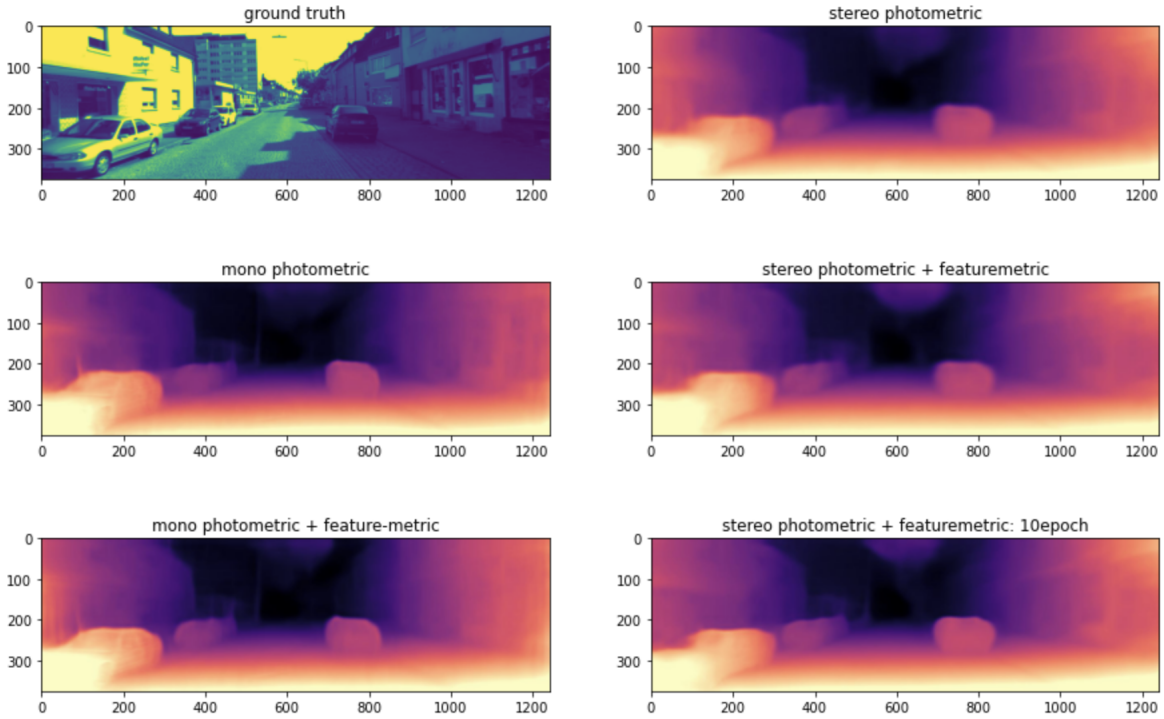

Figure 2. KITTI Dataset Example 1

feature-metric performed better than purely photometric.

For KITTI with monocular input, the regular baseline Monodepth2 actually outperformed our version of Monodepth2 with feature-metric loss. On the other hand, for stereo, our feature-metric loss model performed better than the baseline Monodepth2. Considering the mono baseline only did better than the feature-metric version by a slight margin, we think the feature-metric version might still be able to do better if it's trained to completion.

With regards to our NYU Dataset results, it was nice to see that our predictions for performance were correct. Across the board, the model with feature-metric loss outperformed the baseline, showing that the feature representations do help with low texture indoor scenes.

4.2. Qualitative Results

We have several figures (Figures 2-5) that illustrate how well our models did. Figures 2-3 show how well the KITTI models performed and Figures 4-5 (below) demonstrate how well the NYU Dataset performed.

If we first take a look at the KITTI dataset results (Figures 2 and 3), we can see that, overall, they all performed pretty similarly, and they seemed pretty accurate compared to the ground truth.

However, if we were to examine the generated depth maps a little more closely, we can see that, overall, mono

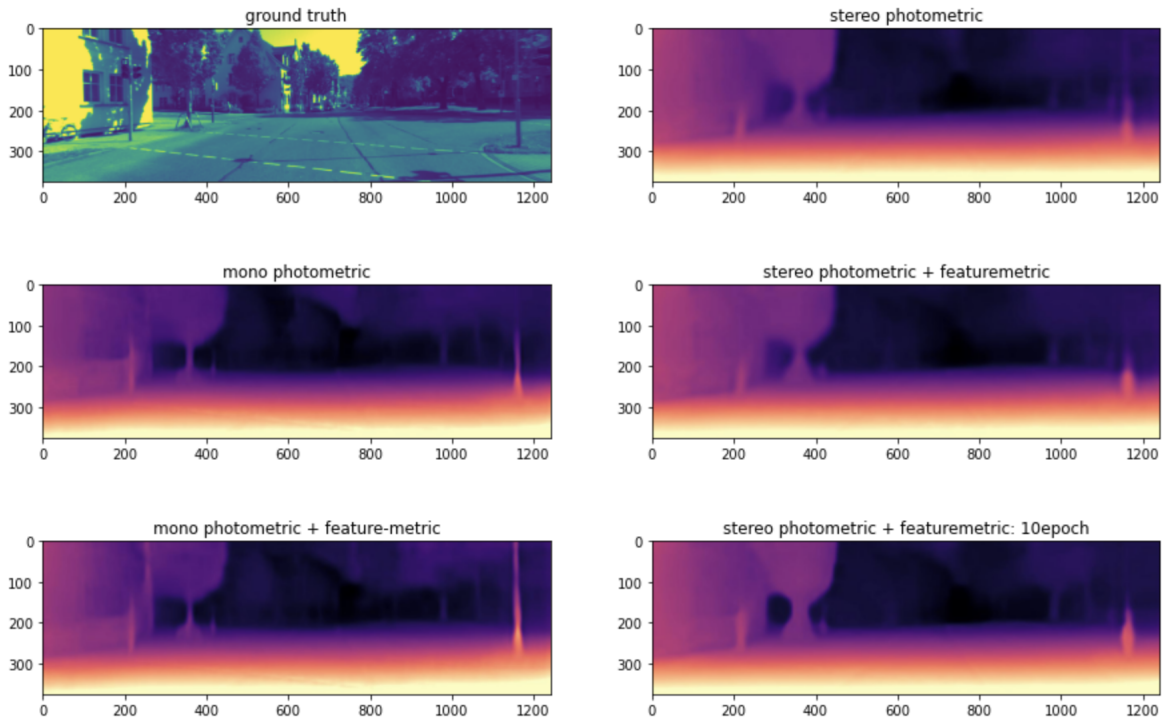

Figure 3. KITTI Dataset Example 2

performed better than stereo, for both photometric and photometric with feature-metric. We are not entirely surprised by the result, since in the Monodethp2 paper, the qualitative results for the stereo model are slightly worse than the mono model's, so perhaps it make senses why the performance would reflect that.

Additionally, you can see that photometric and feature-metric performs better than just photometric (looking at just mono), especially if you take a look at the details. In Figure 2, the car behind the closest car on the left is much more defined in the photometric + feature-metric model, and in Figure 3, you can see that the tree on the left is more crisp in the lowest left corner compared to its photometric counterpart.

As stated before, all of our models were trained with only 5 epochs, but we wanted to see how well our model performs with longer training time, hence the 10-epoch image on the bottom right for both KITTI datasets. Surprisingly, more epochs made the training worse, which is something to investigate in the future.

If we take a look at Figures 4-5, which represent the qualitative results for the NYU dataset, we can see that, on average, all models performed much more poorly than that of the KITTI dataset. There are rarely any defined lines, and it's hard to make out the scene at all.

One of the reasons for this, perhaps, is because there are many textureless regions in these scenes compared to the KITTI outdoor dataset, where objects and structures are easier to distinguish. Although the feature representations are trained to focus on low image gradient regions, low texture patches can still make the pose estimation network incredibly hard to train, and correct pose estimation is crucial when monocular video is the training input.

While figures 4 and 5 are less than ideal, we can still see that the photometric + feature-metric pictures appear better than the photometric model. there is slightly more detail in the pictures, especially with the metal shelves in Figure 4.

Overall, while the quantitative results don't unanimously justify the effectiveness of feature-metric loss, the quantitative results show that, in every situation, the feature-metric model is an improvement on the photometric model.

4.3. Ablation Study

Regarding ablation study, we try to take our experimentation one step further to see which factors made a difference. One of the main factors that we mentioned earlier was the number of epochs we used to train. Here we first look at the direct effect of training time on the depth estimation model itself, and then we look at the effect of training time on the feature encoder.

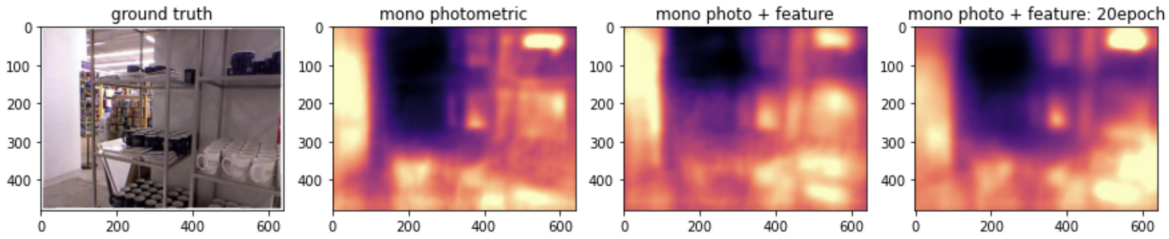

Figure 4. NYU Dataset Example 1

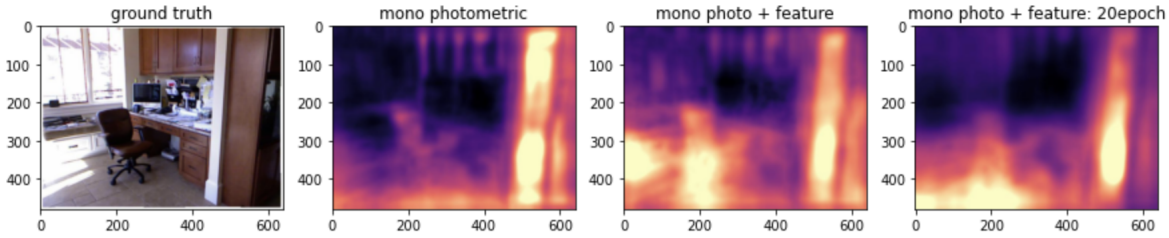

Figure 5. NYU Dataset Example 2

In Table 2, we show the stereo model trained with feature-metric loss on the KITTI dataset for 5 and 10 epochs respectively, and we see that the 5-epoch model actually performed better than the 10-epoch model. This is consistent with Figure 2 and 3, as mentioned in the qualitative results section. We're not really sure why this is the case, but it would be interesting to see if this trend continues to 15 or even to 20 epochs. It is possible that this is only a slight dip during training.

Stereo + feature-metric on KITTI				
Method	abs_rel	sq_rel	rmse	rmse_log
5 epochs	**0.113**	**0.838**	**4.880**	**0.205**
10 epochs	0.114	0.954	4.969	0.207
Method	a1	a2	a3	
5 epochs	**0.858**	**0.949**	**0.977**	
10 epochs	**0.858**	**0.949**	0.976	

Table 2. Comparing the same model trained for 5 and 10 epochs using the pretrained feature-metric encoder.

In Table 3, we show two mono models trained with feature-metric loss on the NYU dataset for 5 epochs. One model has a feature encoder that was trained for 5 epochs, and another with a feature encoder that was trained for 20 epochs.

Surprisingly, the 20-epoch version also performed more poorly than the 5-epoch version, which, if studied more, could point to a discreditation of the feature-metric approach. Of course, without further investigation, we cannot draw any conclusions based on our current results.

feature-metric on NYU				
Method	abs_rel	sq_rel	rmse	rmse_log
5 epochs	**0.321**	**0.435**	**1.038**	**0.463**
20 epochs	0.339	0.488	1.116	0.533
Method	a1	a2	a3	
5 epochs	**0.502**	**0.768**	**0.886**	
20 epochs	0.468	0.749	0.870	

Table 3. Comparing the same model trained for 5 epochs, using feature-metric encoder that was trained for 5 and 20 epochs.

4.4. Conclusion

Overall, this research has shown that adding feature-metric loss to photometric less does, indeed, increase the performance of self-supervised monocular depth estimation. It works best on the NYU dataset where textureless regions are much more prevalent; however, it isn't consistent, as we can see in our ablation studies.

A big portion of this uncertainty is due to the short amount of training time we had available, and in the future, the next steps are to figure out why a feature encoder trained for more epochs performs worse, and what part of the feature-metric loss is actually impacting the model in a positive way.

References

[1] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2

[2] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow. Digging into self-supervised monocular depth prediction. Oc-

tober 2019. 1, 2, 3

[3] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 2

[4] C. Shu, K. Yu, Z. Duan, and K. Yang. Feature-metric loss for self-supervised learning of depth and egomotion. In *ECCV*, 2020. 1, 2, 4

[5] H. R. S. Zhou Wang, Alan Conrad Bovik and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. In *TIP*, 2004. 1