FisheyeDistanceNet: Self-Supervised Scale-Aware Distance Estimation using Monocular Fisheye Camera for Autonomous Driving

Varun Ravi Kumar1, Sandesh Athni Hiremath1, Stefan Milz1, Christian Witt1, Clément Pinnard2, Senthil Yogamani3 and Patrick Mäder4
1Valeo DAR Kronach, Germany 2ENSTA ParisTech Palaiseau, France 3Valeo Vision Systems, Ireland 4Technische Universität Ilmenau, Germany

Abstract—Fisheye cameras are commonly used in applications like autonomous driving and surveillance to provide a large field of view (> 180°). However, they come at the cost of strong non-linear distortion which require more complex algorithms. In this paper, we explore Euclidean distance estimation on fisheye cameras for automotive scenes. Obtaining accurate and dense depth supervision is difficult in practice, but self-supervised learning approaches show promising results and could potentially overcome the problem. We present a novel self-supervised scale-aware framework for learning Euclidean distance and ego-motion from raw monocular fisheye videos without applying rectification. While it is possible to perform piece-wise linear approximation of fisheye projection surface and apply standard rectilinear models, it has its own set of issues like re-sampling distortion and discontinuities in transition regions. To encourage further research in this area, we will release this dataset as part of our WoodScape project [1]. We further evaluated the proposed algorithm on the KITTI dataset and obtained state-of-the-art results comparable to other self-supervised monocular methods. Qualitative results on an unseen fisheye video demonstrate impressive performance.

I. INTRODUCTION

Recently, there has been a significant rise in the usage of fisheye cameras in various computer vision tasks, automotive applications [2], depth estimation [2], [3], [4], surveillance [5], [6] and robotics [7] due to their large Field of View (FOV). Depth estimation is an important task in autonomous driving as it is used to avoid obstacles and plan trajectories. While depth estimation has been substantially studied for narrow FOV cameras, it has barely been explored for fisheye cameras.

Previous learning-based approaches [8], [9], [10], [11] have solely focused on traditional 2D content captured with cameras following a typical pinhole projection model based on rectified image sequences. With the surge of efficient and cheap wide angle fisheye cameras and their larger field-of-view (FOV) in contrast to pinhole models, there has been significant interest in the computer vision community to perform depth estimation from omnidirectional content similar to traditional 2D content via omnidirectional stereo [12], [13], [14], [15] and structure-from-motion (SfM) [16] approaches.

Depth estimation models may be learned in a supervised fashion on LiDAR distance measurements, such as KITTI [17]. In previous work, we followed this approach and demonstrated the possibility to estimate high-quality distance maps using LiDAR ground truth on fisheye images [2]. However, setting up the entire rig for such recordings is expensive and time consuming, and therefore limits the amount of data on which a model can be trained. To overcome this problem, we propose FisheyeDistanceNet, the first end-to-end self-supervised monocular scale-aware training framework. FisheyeDistanceNet uses convolutional neural networks (CNN) and operates on raw fisheye image sequences to regress a Euclidean distance map. Applying FisheyeDistanceNet, we provide a baseline for single frame Euclidean distance estimation based on a structure-from-motion (SfM) setting and raw fisheye image sequences. We summarize our contributions as follows:

- A self-supervised training strategy that aims at inferring a distance map from a sequence of images on distorted and unrectified raw fisheye images.
- A solution to the scale factor uncertainty with the bolster from ego-motion velocity allows outputting metric distance maps. This facilitates the map’s practical use for self-driving cars equipped with fisheye or pinhole cameras.
- A novel combination of super resolution networks and deformable convolution layers [18] to output high resolution distance maps with sharp boundaries from a low resolution input. Inspired by the super resolution of images approach [19] this approach allows us to accurately resolve distances replacing the deconvolution [20] and naive nearest neighbor or bilinear upsampling.

![Image](https://youtu.be/Sgq1WzoQmXg)
problem as there could exist a large number of possible incorrect distances per pixel, which can also recreate the novel view, given the relative pose between $I_t \rightarrow t'$.

Using view-synthesis as the supervising technique we can train the network using the viewpoint of $I_{t-1}$ and $I_{t+1}$ to estimate the appearance of a target image $I_t$ on raw fisheye images. A naive approach would be correcting raw fisheye images to one of the projections shown in Fig. 4 and would essentially render the problem equivalent to Zhou et al.'s work [9]. In contrast, at the core of our approach there is a simple yet efficient technique for obtaining scale-aware distance maps.

This section starts with discussing the geometry of the problem and how it is used to obtain differentiable losses. We describe the scale-aware FisheyeDistanceNet and its effects on the output distance estimates. Additionally, we provide an in-depth discussion of the various losses.

A. Modeling of Fisheye Geometry

1) Projection from world coordinates to image coordinates: The projection function $X_w \rightarrow \Pi(X_w) = p$ of a 3D point $X_w = (X, Y, Z)^T$ to a pixel $p = (u, v)^T$ in the image coordinates is obtained via a 4th order polynomial in the following way:

$$
\begin{align*}
\begin{bmatrix}
    x_c \\
y_c \\
z_c \\
1
\end{bmatrix} &=
\begin{bmatrix}
    R & t \\
0 & 1
\end{bmatrix}
\begin{bmatrix}
    X \\
Y \\
Z \\
1
\end{bmatrix} \\
\phi &= \arctan2(y_c, x_c) \\
\theta &= \frac{\pi}{2} - \arctan2(Z, r_w) \\
\rho(\theta) &= k_1 \cdot \theta + k_2 \cdot \theta^2 + k_3 \cdot \theta^3 + k_4 \cdot \theta^4 \\
p &= \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \rho(\theta) \cdot \cos \varphi \cdot a_x + c_x \\ \rho(\theta) \cdot \sin \varphi \cdot a_y + c_y \end{bmatrix}
\end{align*}
$$

where $r_w = \sqrt{X^2 + Y^2}$, $\theta$ is the angle of incidence, $(a_x, a_y)$ is the aspect ratio and $(c_x, c_y)$ is the principal point.

2) Projection from image coordinates to world coordinates: The unprojection function $p \rightarrow \Pi^{-1}(p) = X_w$ of an image pixel $p = (u, v)^T$ to the world point $X_w = (X, Y, Z(D))^T$ is obtained via the following steps. Letting $(x_c, y_c)^T = (u - c_x, v - c_y)^T$ T, we obtain the angle of incidence $\theta$ by numerically calculating the 4th order polynomial roots of $p = \sqrt{x_c^2 + y_c^2} u^2 + \rho^2$ using the distortion coefficients $k_1, k_2, k_3, k_4$ (see Eq. 4). For training efficiency, we pre-calculate the roots and store it as a lookup table for all the pixel coordinates. Now, $\theta$ is used to get

$$
\begin{align*}
r_w &= \sin(\theta) \cdot D \\
Z(D) &= \cos(\theta) \cdot D
\end{align*}
$$

where $D$ is the distance estimate generated from the network and represents the Euclidean distance $\|X_w\| = \sqrt{X^2 + Y^2 + Z^2}$ of a 3D point $X_w$. The angle of incidence $\varphi$ and the $X, Y$ components can be obtained as follows:

$$
\begin{align*}
\varphi &= \arctan2(y_c, x_c), \quad X = r_w \cdot \cos(\varphi), \quad Y = r_w \cdot \sin(\varphi).
\end{align*}
$$

- We depict the importance of using backward sequences for training and construct a loss for this sequence with a combination of filtering static pixels and an ego mask. The incorporated bundle-adjustment framework [21] jointly optimizes distances and camera poses within a sequence by increasing the baseline and providing additional consistency constraints.

II. SELF-SUPERVISED SCALE-AWARE FISHEYE-DISTANCE-NET

Zhou et al.'s [9] self-supervised monocular structure-from-motion (SfM) framework aims at learning:

1) a monocular depth model $g_d : I \rightarrow D$ predicting a scale-ambiguous depth $D = g_d(I(p))$ per pixel $p$ in the target image $I_t$; and
2) an ego-motion predictor $g_e : (I_t, I_{t'}) \rightarrow I_{t \rightarrow t'}$ predicting a set of six degrees of freedom rigid transformations for $T_{t \rightarrow t'} \in SE(3)$, between the target image $I_t$ and the set of reference images $I_{t'}$. Typically, $t' \in \{t+1, t-1\}$, i.e. the frames $I_{t-1}$ and $I_{t+1}$ are used as reference images, although using a larger window is possible.

A limitation of this approach is that both depth and pose were estimated up to an unknown scale factor in the monocular SfM pipeline.

The distance which acts as an intermediary variable is obtained from the network by constraining the model to perform image synthesis. Distance estimation is an ill-posed
B. Photometric Loss

Let us consider the image reconstruction error from a pair of images \( I_t \) and \( I_{t'} \), distance estimate \( D_{t'} \) at time \( t' \), \( D_t \) at time \( t \), and the relative pose for \( I_{t-1} \), with respect to the target image \( I_t \)'s pose, as \( T_{t-1} \). Using the distance estimate \( D_t \) of the network a point cloud \( P_t \) is obtained via:

\[
P_t = D_t \circ \Pi^{-1}(p_t)
\]

where, \( \Pi^{-1} \) represent the nonprojection from image to world coordinates as explained in Section II-A.2 \( p_t \) the pixel set of image \( I_t \). Pose estimate \( T_{t-1} \) from the pose network is used to get an estimate \( P_{t'} = T_{t-1}^{-1} P_t \) for the point cloud for the image \( I_{t'} \). \( P_{t'} \) is then projected onto the fisheye camera at frame \( t' \) using the projection model \( \Pi \) described in Section II-A.2. Combining transformation and projection with Eq. 7 establishes a mapping from image coordinates at time \( t \) to image coordinates at time \( t' \). This mapping allows reconstructing target frame \( \hat{I}_t \) by warping the source frame \( I_{t'} \) based on \( D_t, T_{t-1} \).

\[
\hat{p}_{t'=t} = \Pi \circ T_{t-1} \circ \left( \Pi^{-1}(p_t) \right)
\]

\[
\hat{I}_{t'}^{uv} = \hat{I}_t^{uv} = \langle \hat{p}_{t'=t} \rangle
\]

Since \( \hat{I}_t \) is with continuous coordinates, we apply the differentiable spatial transformer network introduced by [22] to compute \( \hat{I}_t \) by performing bilinear interpolation from the four pixels corresponding to \( I_t \), whose coordinates overlap with \((\hat{u}, \hat{v})\), \( \hat{I}_t \) is the sampling operator.

Following [10], [23] the image reconstruction error between the target image \( I_t \) and the reconstructed target image \( \hat{I}_t \) is estimated using the L1 pixel-wise loss term combined with Structural Similarity (SSIM) [24], as our photometric loss given by Equation (11) below.

\[
\mathcal{L}_p(I_t, \hat{I}_t) = \alpha \frac{1 - \text{SSIM}(I_t, \hat{I}_t)}{2} + (1 - \alpha) \| I_t - \hat{I}_t \|_1
\]

\[
\mathcal{L}_p = \min_{t'} \langle I_t, \hat{I}_t \rangle \circ M_{t,t'}
\]

where \( \alpha = 0.85 \) and \( M_{t,t'} \) is the binary mask as discussed in Section II-D.

Following [8] instead of averaging the photometric error over all source images, we adopt per-pixel minimum. This significantly sharpens the occlusion boundaries and reduces the artifacts resulting in higher accuracy.

The self-supervised framework assumes a static scene, no occlusion and appearance \( i.e. \) brightness constancy. A large photometric cost is incurred, potentially worsening the performance, if there exist dynamic objects and occluded regions. These areas are treated as outliers similar to [21] and clip the photometric loss values to a \( q \)-th percentile \( i.e. \) \( q = 95 \). Zero gradient is obtained for errors larger than 95%.

This improves the optimization process and provides a way to strengthen the photometric error.

C. Solving Scale Factor Ambiguity at Training Time

For a pinhole projection model, \( \text{Depth} \propto 1/\text{disparity} \). Henceforth, the network’s sigmoid output \( \sigma \) can be converted to depth with \( D = 1/(a \sigma + b) \), where \( a \) and \( b \) are chosen to constrain \( D \) between 0.1 and 100 units [8]. For a spherical image, we can only obtain angular disparities [25] by rectification on spherical images. To perform distance estimation on raw fisheye images, we would require distance metric values to warp the source image \( I_{t'} \) onto the target frame \( I_t \). Due to the limitations of the monocular SFM objective, both the monocular depth \( g_d \) and ego-motion predictors \( g_x \) predict scale-ambiguous \( \text{values} \) which would make it impossible to estimate distance maps on fisheye images.

To achieve scale-aware distance values, we normalize the pose network’s estimate \( T_{t-1} \) and scale it with \( \Delta x \), the displacement magnitude relative to target frame \( I_t \) which is calculated using vehicle’s instantaneous velocity estimates \( v_{t'} \) at time \( t' \) and \( v_t \) at time \( t \). We also apply this technique on KITTI [17] to obtain metric depth maps.

\[
\overline{T}_{t-1} = T_{t-1} / \| T_{t-1} \| \cdot \Delta x
\]

D. Masking Static Pixels and Ego Mask

Following [8], we incorporate a masking approach to filter out static pixels which do not change its appearance from one frame to the other in the training sequence. The approach would filter out objects which move at the same speed as the ego-car, and also ignore the static frame when the ego-car stops moving. Similar to other approaches [8], [9], [26], [27] the per-pixel mask \( \sigma \) is applied to the loss by weighting the pixels selectively. Instead of being learned from the object motion [28], the mask is computed in the forward pass of the network, yielding a binary mask output where \( \sigma \in \{0, 1\} \).

Wherever the photometric error of the warped image \( I_{t'}^{t-1} \) is lower than that of the original unwarped source frame \( I_{t'} \), \( \sigma \) is set to ignore the loss of such pixel, \( i.e. \)

\[
\sigma = \left[ \min_{t'} \langle p_{t'}(I_t, I_{t'}^{t-1}) < \min_{t'} \langle p_{t'}(I_t, I_{t'}) \rangle \right],
\]

where \( \left[ \right] \) is the Iverson bracket. Additionally, we add a binary ego mask \( M_t \) proposed in [29] that ignores computing the photometric loss on the pixels that do not have a valid mapping \( i.e. \) some pixels of the source image \( I_{t'} \) may not be projected onto the target image \( I_t \) given the estimated target distance \( D_t \).

E. Backward Sequence

In the forward sequence, we synthesize the target frame \( I_t \) with the source frames \( I_{t-1} \) and \( I_{t+1} \) \( i.e. \) as per above discussion \( t' \in \{t + 1, t - 1\} \). Analogously, backward sequence is carried out by using \( I_{t-1} \) and \( I_{t+1} \) as target frames and \( I_t \) as source frame. We include warps from \( I_{t-1} \) and \( I_{t+1} \) thereby inducing more constraints to avoid overfitting and resolves unknown distances in the borders at the test time, as also observed in previous works [8], [9], [30]. We construct the cost for the additional backward sequence in a similar manner to the forward. This comes with the cost of high computation effort and longer training time as we perform two forward and backward warps which yields superior results on the Fisheye and KITTI dataset than...
the previous approaches [8], [9] which train only with one forward sequence and one backward sequence.

F. Edge-Aware Smoothness Loss

In order to regularize distance and avoid divergent values in occluded or texture-less low-image gradient areas, we add a geometric smoothing loss. We adopt the edge-aware term similar to [10], [29], [31]. The regularization term is imposed on the inverse distance map. Unlike previous works, the loss is not decayed for each pyramid level by a factor of 2 due to down-sampling, as we use a super resolution network (cp. Section III-A)

\[ \mathcal{L}_s(\hat{D}_t) = |\partial_x \hat{D}_t| e^{-|\hat{D}_t I_{t}|} + |\partial_y \hat{D}_t| e^{-|\hat{D}_t I_{t}|} \]  

(12)

To discourage shrinking of estimated distance [11], mean-normalized inverse distance of \( \hat{D}_t \) is considered, i.e. \( \hat{D}_t = \hat{D}_t^{-1}/\hat{D}_t \), where \( \hat{D}_t \) denotes the mean of \( D_t^{-1} := 1/D_t \).

G. Cross-Sequence Consistency Loss

The SfM setting uses an N-frame training snippet \( S = \{I_1, I_2, \cdots, I_N\} \) from a video as inputs. The FisheyeDistanceNet can estimate the distance of each image in the training sequence. Another constraint can be enforced among the frames in \( S \), since the distance of a 3D point estimated from different frames should be consistent.

Let us assume \( \hat{D}_t \) and \( \hat{D}_t \) are the estimates of the images \( I_t \) and \( I_t \) respectively. For each pixel \( p_t \in I_t \), we can use Eq. [8] to obtain \( \hat{p}_t \). Since it is real valued, we apply the differentiable spatial transformer network introduced by [22] and estimate the distance value of \( \hat{p}_t \) by performing bilinear interpolation from the four pixels in \( \hat{D}_t \) whose coordinates overlap with \( (\hat{u}, \hat{v}) \). Let us denote the distance map obtained through this as \( \hat{D}_t \). Next, We can transform the point cloud in frame \( t \) to frame \( t' \) by first obtaining \( \hat{P}_t \) using Eq. [7] We transform the point cloud \( P_t \) using the pose network’s estimate using \( \hat{P}_t = T_{t 

H. Final Training Loss

The overall self-supervised structure-from-motion (SfM) from motion objective consists of a photometric loss term \( \mathcal{L}_p \) imposed between the reconstructed target image \( \hat{I}_t \) and the target image \( I_t \), and a distance regularization term \( \mathcal{L}_s \) ensuring edge-aware smoothing in the distance estimates. Finally, \( \mathcal{L}_d \), a cross-sequence distance consistency derived from the chain of frames in the training sequence \( S \). To prevent the training objective getting stuck in the local minima due to the gradient locality of the bilinear sampler [22], we adopt 4 scales to train the network as followed in [9], [10]. The final objective function is averaged over per-pixel, scale and image batch.

\[ \mathcal{L}(I_t, \hat{I}_t) = \sum_{n=1}^{4} \frac{\mathcal{L}_n}{2^{n-1}}, \]  

(15)

\[ \mathcal{L}_n = n \mathcal{L}_p + n \mathcal{L}_p + \gamma (n \mathcal{L}_d + \mathcal{L}_d) + \beta_n \mathcal{L}_s \]

III. NETWORK DETAILS

A. Deformable Super-Resolution Distance and PoseNet

The distance estimation network is mainly based on the U-net architecture [37], an encoder-decoder network with skip connections. After testing different variants of ResNet family, such as ResNet50 with 25M parameters, we chose a ResNet18 [38] as the encoder. The key aspect here is replacing normal convolutions with deformable convolutions since regular CNNs are inherently limited in modeling large, unknown geometric distortions due to their fixed structures, such as fixed filter kernels, fixed receptive field sizes, and fixed pooling kernels [39], [18].

In previous works [8], [9], [10], [11], [30], the decoded features were upsampled via a nearest neighbor interpolation or with learnable transposed convolutions. The main drawback of this process is that it may lead to large errors at object boundaries in the upscaled distance map as the interpolation simply combines distance values of background and foreground. For effective and detailed preservation of the decoded features, we leverage the concept of sub-pixel convolutions [19] to our super resolution network. We use pixel shuffle convolutions and replace the convolutional feature upsampling, performed via a nearest neighbor interpolation or with learnable transposed convolutions. We can see in supplementary Section 5 the resulting distance maps are super-resolved and has sharp boundaries and exposes more details of the scene.

The backbone of our pose estimation network is based on [8] and predicts rotation using Euler angle parameterization. The output is a set of six DOF transformations between \( I_{t-1} \) and \( I_t \). We replace normal convolutions with deformable convolutions for the encoder-decoder setting.

B. Implementation Details

We use Pytorch [40] and employed Adam [41] optimizer to minimize the training objective function (15) with \( \beta_1 = 0.9, \beta_2 = 0.999 \). We train the model for 25 epochs, with a
batch size of 20 on 24GB Titan RTX with initial learning rate of $10^{-4}$ for the first 20 epochs, then drop to $10^{-5}$ for the last 5 epochs. The sigmoided output $\sigma$ from the distance decoder is converted to distance with $D = a \cdot \sigma + b$. For the pinhole model, $D = 1/(a \cdot \sigma + b)$, where $a$ and $b$ are chosen to constrain $D$ between 0.1 and 100 units. The original input resolution of the fisheye image is $1280 \times 800$ pixels, we crop it to $1024 \times 512$ to remove the vehicle’s bumper, shadow and other artifacts of the vehicle. Finally the cropped image is downscaled to $512 \times 256$ before feeding to the network. For pinhole model on KITTI, we use $640 \times 192$ pixels as the network input. We experimented with batch normalization [42] and group normalization [43] layers in the encoder-decoder setting. We found that group normalization with $G = 32$ significantly improved the results than the former [44]. The smoothness weight term $\beta$ and cross-sequence distance consistency weight term $\gamma$ are set to 0.001.

We applied deformable convolutions to the $3 \times 3$ conv layers in stages conv3, conv4, and conv5 in ResNet18 and ResNet50, with 12 layers of deformable convolution in the encoder part compared to 3 layers in [39], all in the conv5 stage for ResNet50. We replaced the subsequent layers of the decoder with deformable convolutions for the distance and pose network. For the pinhole model, on KITTI Eigen split in Section IV-A.2 we use normal convolutions instead of deformable convolutions.

Finally, to alleviate checkerboard artifacts from the output distance maps using sub-pixel convolution [19], we initialized the last convolutional layer in a specific way before the pixel shuffle operation as described in [45].

### IV. Experiments

#### A. Datasets

1) Fisheye WoodScape: The dataset contains 40,000 raw images obtained with a fisheye camera and a sparse Velodyne HDL-64E rotating 3D laser scanner as ground truth for the test set. The training set contains 39,038 images collected by driving around various parts of Bavaria, Germany. The validation and the test split contain 1,214 and 697 images respectively. The dataset distribution is similar to the KITTI Eigen split used in [8], [9] for the pinhole model. The training set comprises three scene categories: city, residential and sub-urban. While training, these categories are randomly shuffled and fed to the network. We filter static scenes based on the speed of the vehicle with a threshold of 2 km/hr to remove image frames that only observe minimal camera ego-motion, since distance cannot be learned under these circumstances. Comparable to previous experiments on pinhole SfM [9], [8], we set the length of the training sequence to 3.

2) Rectilinear KITTI Eigen Split: We use the KITTI dataset and data split according to Eigen et al. [46] for the experiments with pinhole image data. We filter static frames as proposed by Zhou et al. [9]. The resulting training set contains 39,810 images and the validation split comprises 4,424 images. We use the standard test set of 697 images. The length of the training sequence is set to 3.
Fig. 3 Qualitative results on the Fisheye WoodScape dataset. Our FisheyeDistanceNet produces sharp distance maps on distorted fisheye images.

| Method | FS | BS | SR | CSDCL | DCN | Abs Rel | Sq Rel | RMSE | RMSE log | $\delta < 1.25$ | $\delta < 1.25^2$ | $\delta < 1.25^3$ |
|--------|----|----|----|-------|-----|---------|-------|------|----------|----------------|----------------|----------------|
| Ours   | ✓  | ✓  | ✓  | ✓     | ✓   | 0.152   | 0.768 | 2.723| 0.210    | 0.812          | 0.954          | 0.974          |
| Ours   | ✓  | ✓  | ✓  | ✓     | ✓   | 0.172   | 0.829 | 2.925| 0.243    | 0.802          | 0.952          | 0.970          |
| Ours   | ✓  | ✓  | ✓  | ✓     | ✓   | 0.181   | 0.913 | 3.180| 0.250    | 0.823          | 0.938          | 0.963          |
| Ours   | ✓  | ✓  | ✓  | ✓     | ✓   | 0.190   | 0.997 | 3.266| 0.258    | 0.796          | 0.930          | 0.963          |
| Ours   | ✓  | ✓  | ✓  | ✓     | ✓   | 0.201   | 1.282 | 3.589| 0.276    | 0.590          | 0.898          | 0.949          |

TABLE II Ablation study of our algorithm on different variants on our FisheyeDistanceNet using the Fisheye WoodScape dataset [1]. Distances are capped at 40m. BS, SR, CSDCL and DCN represent backward sequence, super-resolution network with PixelShuffle or sub-pixel convolution initialized to convolution NN resize (ICNR) [45], cross-sequence distance consistency loss and deformable convolutions respectively. The input resolution is $512 \times 256$ pixels.

B. Evaluation

We evaluate FisheyeDistanceNet’s depth and distance estimation results using the metrics proposed by Eigen et al. [36] to facilitate comparison. Quantitative results are shown in Table I; additionally we report qualitative results on the KITTI Eigen split in the supplementary material Fig. [7] and for the Fisheye WoodScape dataset in Fig. [5]. Further comparison on the results of previous methods for pinhole model can be found in the supplementary Section [VI].

Since the projection operators are different, previous SfM approaches will not be feasible without adaption of the network and projection model. As there are no existing methods available for comparison out of the box, we performed detailed ablation study of various design aspects. It is important to note that due to the geometry of the fisheye, it would not be a fair comparison to evaluate the distance estimates up to 80 m. Our fisheye automotive cameras also undergo high data compression and our dataset contains images of inferior quality when compared with KITTI. Our fisheye cameras can perform well up to a range of 40 m. Therefore, we also report results on a 30 m and a 40 m range (cp. Table [II]).

C. Fisheye Ablation Study

We conduct an ablation study to evaluate the importance of different components. We ablate the following components and report their impact on the distance evaluation metrics in Table [II] (i) Without Backward Sequence: The network is only trained for the forward sequence which consists of two warps as explained in Section [II-E]; (ii) Without Backward Sequence and Super Resolution using sub-pixel convolution: Removal of sub-pixel convolution has a huge impact compared to KITTI Eigen split ablation study which can be seen in the supplementary Section [VI]. This is mainly attributed to the fisheye model, as far-away objects are tiny and can not be resolved accurately with naive nearest neighbor or transposed convolution [20]; (iii) Removing the cross sequence distance loss mainly reduces the baseline; (iv) Without Backward Sequence and Super Resolution using sub-pixel convolution and deformable convolution: If we remove all the major components, especially deformable convolution layers [18], our model will fail miserably as the distortion introduced by the fisheye model will not be learned correctly by the normal convolutional layers.

V. Conclusion

We propose a novel self-supervised training strategy to obtain metric distance maps on unrectified fisheye images. Through extensive experiments, we show that our FisheyeDistanceNet establishes a new state of the art in self-supervised monocular distance and depth estimation on Fisheye WoodScape and KITTI dataset respectively. We obtain promising results demonstrating the potential of using CNN based approach for deployment in commercial automotive systems, in particular for replacing current classical depth estimation approaches. However, it is essential to have a large scale dataset covering various scenarios and corner cases.
A. Aitken, C. Ledig, L. Theis, J. Caballero, Z. Wang, and W. Shi, “Checkerboard artifact free sub-pixel convolution: A note on sub-pixel convolution, resize convolution and convolution resize,” arXiv preprint arXiv:1707.02937, 2017.

D. Eigen and R. Fergus, “Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture,” in Proceedings of the IEEE international conference on computer vision, 2015, pp. 2650–2658.

M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 3213–3223.

T. J. Herbert, “Area projections of fisheye photographic lenses,” Agricultural and Forest Meteorology, vol. 39, no. 2-3, pp. 215–223, 1987.

J. P. Barreto, “Unifying image plane liftings for central catadioptric and dioptric cameras,” Imaging Beyond the Pinhole Camera, pp. 21—38, 2006.

B. Khomutenko, G. Garcia, and P. Martinet, “An enhanced unified camera model,” IEEE Robotics and Automation Letters, vol. 1, no. 1, pp. 137–144, 2016.

C. Hughes, P. Denny, E. Jones, and M. Glavin, “Accuracy of fish-eye lenses models,” Applied Optics, vol. 49, no. 17, pp. 3338–3347, 2010.

Z. Kukelova, J. Heller, M. Bujnak, and T. Pajdla, “Radial distortion homography,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 639–647.

Y.-C. Su and K. Grauman, “Kernel transformer networks for compact spherical convolution,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 9442–9451.

B. Coors, A. Paul Condurache, and A. Geiger, “SphereNet: Learning spherical representations for detection and classification in omnidirectional images,” in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 518–533.

C. Plagemann, C. Stachniss, J. Hess, F. Endres, and N. Franklin, “A nonparametric learning approach to range sensing from omnidirectional vision,” Robotics and Autonomous Systems, vol. 58, no. 6, pp. 762–772, 2010.

M. Rudr, A. Dosovitskiy, and T. Brox, “Artistic style transfer for videos and spherical images,” International Journal of Computer Vision, vol. 126, no. 11, pp. 1199–1219, 2018.

R. Monroy, S. Lutz, T. Chalasani, and A. Smolic, “Salnet360: Saliency maps for omni-directional images with cnn,” Signal Processing: Image Communication, vol. 69, pp. 26–34, 2018.

P. Frossard and R. Khasanova, “Graph-based classification of omnidirectional images,” in 2017 IEEE International Conference on Computer Vision Workshops (ICCVW). IEEE, 2017, pp. 860–869.

Y.-C. Su and K. Grauman, “Learning spherical convolution for fast features from 360 imagery,” in Advances in Neural Information Processing Systems, 2017, pp. 529–539.

Y. Jeon and J. Kim, “Active convolution: Learning the shape of convolution for image classification,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 4201–4209.

K. Tateno, N. Navab, and F. Tombari, “Distortion-aware convolutional filters for dense prediction in panoramic images,” in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 707–722.

L. Deng, M. Yang, H. Li, T. Li, B. Hu, and C. Wang, “Restricted deformable convolution based road scene semantic segmentation using surround view cameras,” arXiv preprint arXiv:1801.00708, 2018.

R. Garg, V. K. BG, G. Carneiro, and I. Reid, “Unsupervised cnn for single view depth estimation: Geometry to the rescue,” in European Conference on Computer Vision. Springer, 2016, pp. 740–756.

M. Klodt and A. Vedaldi, “Supervising the new with the old: learning sfm from sfm,” in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 698–713.

N. Yang, R. Wang, J. Stuckler, and D. Cremers, “Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry,” in Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 817–833.

J. Li, Y. Chen, L. Cai, I. Davidson, and S. Ji, “Dense transformer networks,” arXiv preprint arXiv:1705.08881, 2017.

S. Pillai, R. Ambruš, and A. Gaidon, “Superdepth: Self-supervised, super-resolved monocular depth estimation,” in 2019 International Conference on Robotics and Automation (ICRA). IEEE, 2019, pp. 9250–9256.
SUPPLEMENTARY MATERIAL

I. OVERVIEW OF FISHEYE CAMERA PROJECTIONS

Fisheye cameras exhibit significantly more complex projection geometry and the images display severe distortion. Typical camera datasets like KITTI [17] and CityScapes [47] consist of relatively narrow FOV camera data where a simple pinhole projection model is commonly employed. In case of fisheye camera images, it is imperative that the appropriate camera model is well understood either to handle distortion in the algorithm or to warp the image prior to processing. This section is intended to highlight to the reader that the fisheye camera model requires specific attention. We provide a brief overview and references for further details, and discuss the merits of operating on the raw fisheye versus undistortion of the image.

Fisheye distortion is modeled by a radial mapping function \( r(\theta) \), where \( r(\theta) \) is the distance on the image from the centre of distortion, and is a function of the angle \( \theta \) of the incident ray against the optical axis of the camera system. The centre of distortion is the intersection of the optical axis with the image plane, and is the origin of the radial mapping function \( r(\theta) \). Stereographic projection [48] is the simplest model which uses a mapping from a sphere to a plane. More recent projection models are Unified Camera Model (UCM) [49], [7] and eUCM (Enhanced UCM) [50]. More detailed analysis of accuracy of various projection models is discussed in [51]. These models are not a perfect fit for fisheye cameras as they encode a specific geometry (e.g. spherical projection), and errors arising in the model are compensated by using an added distortion correction component.

We use model parameters for a more generic fisheye intrinsic calibration that is independent of any specific projection model and does not require the added step of distortion correction. Our model is based on a fourth order polynomial mapping incident angle to image radius in pixels \( r(\theta) = a_1\theta + a_2\theta^2 + a_3\theta^3 + a_4\theta^4 \). In our experience, higher orders provide no additional accuracy.

The inverse of \( r(\theta) \) for the 4th order polynomial requires root solving, which can be expensive (albeit, this can be overcome by the use of a look-up table for the inverse as we are certain that the distortion stays the same throughout the whole dataset. UCM and eUCM, and geometric models in general, are analytically reversible.

A. Image Undistortion vs. Model Adaptation

Standard computer vision models do not generalize easily to fisheye cameras because of large non-linear distortion. For example, translation invariance is lost for a standard convolutional neural net (CNN). The naive way to develop algorithms for fisheye cameras is to perform rectilinear correction so that standard models can be applied. The simplest undistortion is to re-warp pixels to a rectilinear image as shown in Figure 4 (a). But there are two major issues. Firstly, the FOV is greater than 180\(^\circ\), hence there are rays incident from behind the camera and it is not possible to establish a complete mapping to a rectilinear viewport. This leads to a loss of FOV, this is seen via the missing yellow pillars in the corrected image. Secondly, there is an issue of resampling distortion, which is more pronounced near the periphery of the image where a smaller region gets mapped to a larger region.

The missing FOV can be resolved by multiple linear viewports as shown in Figure 4 (b). However there are issues in the transition region from one plane to another. This can be viewed as a piecewise linear approximation of the fisheye lens manifold. Figure 4 (c) demonstrates a quasi-linear correction using a cylindrical viewport, where it is linear in vertical direction and straight vertical objects like pedestrians are preserved. However, there is a quadratic distortion along the horizontal axis. In many scenarios, it provides a reasonable trade-off but it still has limitations. In case of learning algorithms, a parametric transform can be optimized for optimal performance of the target application accuracy.

Because of fundamental limitations of undistortion, an alternate approach of adapting the algorithm incorporating fisheye projection model discussed in Section II-A could be an optimal solution. In case of classical geometric algorithms, an analytical version of non-linear projection can be incorporated. For example, Kukelova et al. [52] extend homography estimation by incorporating radial distortion model. In case of deep learning algorithms, a possible solution could be to train the CNN model to learn the distortion. However, the translation invariance assumption of CNN fundamentally breaks down due to spatially variant distortion and thus it is not efficient to let the network learn it implicitly. This had led to several adaptations of CNN to handle spherical images such as [53] and [54]. However, spherical models do not provide an accurate fit for fisheye lenses and it is an open problem.

II. RELATED WORK

We aim to learn the task of dense distance estimation on raw fisheye images without any rectification. However, we
Fig. 5 Qualitative results on the Fisheye WoodScape [1]. In the 4th row of the table, we can see that our model adapts to the extreme distortion induced by the fisheye camera and produces sharp distance maps. In the 6th row, we can clearly see the sharp curbs on the street. Finally, in the last few rows, our model adapts to most of the complex scenes and produces very sharp scale-aware distance maps.
are not aware of any previous work on specifically this task and therefore review learning based approaches for spherical content. Furthermore, we discuss recent monocular dense depth estimation methods for pinhole models.

A. Learning for spherical images

Plagemann et al. [55] proposed one of the earliest approaches using machine learning for estimating distances purely from spherical input by predicting over a range value per image rather than predicting the distance per pixel to drive robotic navigation. However, modern ConvNets have dramatically changed the landscape of distance estimation. This enables us to apply spherical images to CNN framework by either directly on a projected (equirectangular) image, or the spherical information is projected to the faces of a cube and inferring the estimations on them, and finally merged by upproojecting them to the spherical domain. Ruder [56] used cubemap projection, where each face of the cube was re-styled separately and re-mapping was performed to equirectangular domain for an artistic style transfer work. The saliency predictions on the cube’s faces is refined by SalNet360 [57] using spherical coordinates and finally merged back to 360°.

A graph based deep learning approach was followed by Frossard et al. [58] due to the difficulty to model the distortion directly in CNNs as well as achieve in-variance to the viewpoint’s rotation. The distortion was modelled into the graph’s structure and was applied directly to a classification task. In our previous work [2], we trained distance estimation models in a supervised manner using sparse LiDAR data as ground truth. A novel approach by Su et al. [59] used an existing trained network on 2D images and learned the weights of the convolution for equirectangular projected spherical images. A consistency check was in place for the conversion from the 2D to the 360° domain by enforcing constrains between the estimations of the 2D projected views and those in the 360° image. For learning the distortion model of fisheye images many recent works [60], [61], [39] on convolutions learn their shape in addition to their weights. They have only been applied to fisheye lenses up to now by Deng et al. [62] for scene segmentation. These approaches are applied in the spectral domain on single variable regression and classification problems where as we formulate our network design for the spatial image domain.

B. Self-Supervised Monocular Depth and Pose estimation

As supervised techniques for depth estimation advanced rapidly, the availability of target depth labels became challenging, especially for outdoor applications. To this end, [63], [10] provided an alternative strategy involving training a monocular depth network with stereo cameras, without requiring ground-truth depth labels. By leveraging Spatial Transformer Networks [22], Godard et al [10] use stereo imagery to geometrically transform the right image plus a predicted depth of the left image into a synthesized left image. The loss between the resulting synthesized and original left images is then defined in a fully-differentiable manner, using a Structural Similarity [24] term and additional depth regularization terms, thus allowing the depth network to be self-supervised in an end-to-end fashion.

Zhou et al. [9] was the pioneer in self-supervised training using monocular approach, where depth and pose was simultaneously learned from unlabeled monocular video sequences. Following Zhou’s SfM-learner approach several methods [30], [29], [28], [31], [64], [11], [65] have evolved by integrating additional loss terms and constraints. All the stated methods can predict depth and pose up to an unknown scale factor on rectified pinhole images only and depend upon the ground-truth LiDAR data to scale their depth predictions appropriately for evaluation purposes [9].

In contrast to the approaches discussed above, we show that using view synthesis as the main supervisory signal along with deformable convolutions [39] and bolster from the car’s instantaneous velocity during training, we are able to learn a scale-aware distance and pose model on fisheye images, eliminating the impractical usage of LiDAR ground-truth distance measurements at inference-time.

III. LIMITATION OF NORMAL 2D CONVOLUTION

The deeper layers in a convolutional neural network (CNN) with weak spatial information encode high-level scene information including object- or category-level information. The middle layers features are likely to describe object parts and retain spatial information. The lower layer features encode information like corners, edges, shapes, etc. Thios signifies the fact that the learning of spatial structures takes place in the middle and lower layers of a CNN. The spatial structures are vulnerable to major variations if the deformable convolution is applied to the lower or middle layers of a CNN. It would be difficult to retain the spatial correspondence between input images and output distance maps. This is the spatial correspondence problem indicated in [66], which is critical in pixel-wise distance estimation. Hence, deformable convolution is applied to the last few convolution layers as proposed by [39].

To alleviate this problem, Zhu et al. [18] proposed a better, more Deformable ConvNet with enhanced modeling power which can effectively model geometric transformations. We incorporate the enhanced modulated deformable convolutions to our FisheyeDistanceNet and PoseNet.

IV. DEPICTING THE IMPORTANCE OF ADDITIONAL WARPS

In the forward sequence, we synthesize the target frame \(I_t\) with the source frames \(I_{t-1}\) and \(I_{t+1}\) \((i.e.\ as\ per\ the\ discussion\ in\ Section\ [II-B]\ in\ \{t+1, t-1\})\). The reconstructed image \(I_{t-1\rightarrow t}\) will result in a zoom-in operation where the border distance values are useful and will get meaningful gradients to train with, but center values will have a lot of noise due to the low displacement. On the other hand since \(I_{t+1\rightarrow t}\) will result in a zoom-out effect, the border distance values are insignificant and should be filtered out from photo-metric loss because the pixel that has to be retrieved does not exist. Center distance though will have a warp that sample values from border of the frame, \(i.e.\ with\)
large displacement and the gradient will be less noisy than with $I_{t-1\rightarrow t}$. Additional warps, will induce more constraints to avoid overfitting and resolves unknown distances in the borders at the test time.

V. FURTHER QUANTITATIVE COMPARISONS

The quantitative results shown in the Table I show that our scale-aware self-supervised approach outperforms all the state-of-the-art monocular approaches. More specifically, we outperform recent methods that explicitly compute motion masks (cp. EPC++ [27], Ranjan [33], Godard [8]). Due to a lack of odometry data on the Cityscapes, we could not leverage the dataset to benchmark our scale-aware framework. In contrast to PackNet-SfM [34], which presumably uses a superior architecture compared to our super-resolution ResNet18, with capability of estimating scale-aware depths with their velocity supervision loss, we could achieve higher accuracy with subtle improvements to the standard ResNet18 and the training framework.

We could significantly achieve higher accuracy after 25 epochs compared to Pillai et al. [67] 1024×384, PackNet-SfM’s [34] 1280×384 approach trained for 200 epochs. All previous monocular methods mentioned in the Table I evaluate their depth predictions by scaling to the median ground-truth data from LiDAR as introduced by [9]. We do not perform any per-image median scaling on the test set, our FisheyeDistanceNet outputs scale-aware depth estimates for pinhole models. Initially, an unscaled model is trained for several epochs. As the training progresses, scale-awareness is then induced into the framework.

VI. KITTI EIGEN SPLIT ABLATION STUDY

We conduct an ablation study to evaluate the importance of different components. We ablate the following components and report their impact on the distance evaluation metrics in Table III: (i) Without Super Resolution using sub-pixel convolution and Backward Sequence: Removal of sub-pixel convolution has a minor impact on the pinhole model compared to dropping these layers in fisheye [1]. Although, when compared to state-of-the-art method [8] we could significantly achieve higher accuracy and resolve depths at longer distances accurately; (ii) Without Backward Sequence: The network is only trained for the forward sequence which consists of two warps as explained in Section II-E; (iii) Removing the cross sequence distance loss mainly reduces the baseline.

![Fig. 6 Failure Cases on the Fisheye WoodScape [1] dataset. The photometric loss fails to learn good distances for reflective regions which can be seen in the 1st figure. In the 2nd and 3rd figures shown above, the model fails to accurately delineate objects where boundaries are ambiguous.](image-url)

| Method | FS | BS | SR | CSDCL | Abs Rel | Sq Rel | RMSE | RMSE log | $\delta < 1.25$ | $\delta < 1.25^2$ | $\delta < 1.25^3$ |
|--------|----|----|----|-------|---------|--------|------|----------|----------------|----------------|----------------|
| Ours   | ✓  | ✓  | ✓  | ✓     | 0.117   | 0.867  | 4.739| 0.190    | 0.869          | 0.960          | 0.982         |
| Ours   | ✓  | ✓  |     | ✓     | 0.133   | 0.884  | 4.951| 0.212    | 0.845          | 0.952          | 0.980         |
| Ours   | ✓  |     | ✓  |      | 0.134   | 0.929  | 5.063| 0.215    | 0.835          | 0.947          | 0.979         |
| Ours   |     | ✓  | ✓  |      | 0.144   | 1.029  | 5.146| 0.258    | 0.817          | 0.945          | 0.974         |

TABLE III Ablation study of our algorithm on the KITTI Eigen split dataset [17]. Distances are capped at 80m. BS, SR (ICNR), CSDCL and DCN represent backward sequence, super-resolution network with PixelShuffle or sub-pixel convolution initialized to convolution NN resize (ICNR) [45] and cross-sequence distance consistency loss, respectively. The input resolution is 640×192 pixels.
Fig. 7 Qualitative results on the KITTI Eigen split. The qualitative table is taken from [8] for a fair comparison, as our results produce similar sharp depth maps. When compared to Monodepth2 M [8], our model resolves the low textured areas such as sky i.e. infinite depth and provides sharper transition in the boundaries of objects. Our model (M) in the last row yields superior quantitative results which are reflected in Table I.