Regional Deformer Networks for Unsupervised Depth Estimation from Unconstrained Monocular Videos

Haofei Xu¹, Jianmin Zheng², Jianfei Cai² and Juyong Zhang¹
¹University of Science and Technology of China
²Nanyang Technological University
xhf@mail.ustc.edu.cn, {asjmzheng, asjfcai}@ntu.edu.sg, juyong@ustc.edu.cn

Abstract
While learning based depth estimation from images/videos has achieved substantial progress, there still exist intrinsic limitations. Supervised methods are limited by small amount of ground truth or labeled data and unsupervised methods for monocular videos are mostly based on the static scene assumption, not performing well on real world scenarios with the presence of dynamic objects. In this paper, we propose a new learning based method consisting of DepthNet, PoseNet and Region Deformer Networks (RDN) to estimate depth from unconstrained monocular videos without ground truth supervision. The core contribution lies in RDN for proper handling of rigid and non-rigid motions of various objects such as rigidly moving cars and deformable humans. In particular, a deformation based motion representation is proposed to model individual object motion on 2D images. This representation enables our method to be applicable to diverse unconstrained monocular videos. Our method can not only achieve the state-of-the-art results on standard benchmarks KITTI and Cityscapes, but also show promising results on a crowded pedestrian tracking dataset, which demonstrates the effectiveness of the deformation based motion representation.

1 Introduction
Depth sensing plays an important role in 3D scene understanding. For instance, it is crucial for robots to be aware of how far the surrounding objects are away from themselves, which help robots keep clear of obstacles and adjust future behaviour. Recently, learning based single-image depth estimation has attracted a lot of attention due to the rapid progress of Convolutional Neural Networks (CNN). Supervised methods [Eigen et al., 2014; Li et al., 2015; Liu et al., 2016] aim at learning a mapping from color image to per-pixel depth by neural networks. However, these methods require a large quantity of color-depth pairs and collecting such dataset is challenging, especially in outdoor scenarios. Unsupervised learning gets rid of the dependence on ground truth depth and shows a promising direction. The key idea of the unsupervised learning is to use the warping based image reconstruction loss between adjacent frames to guide the learning process. Several methods have been proposed to use stereo images to estimate depth [Garg et al., 2016; Godard et al., 2017]. Although no ground truth depth is required, the stereo images are still not as common as monocular videos and they need to be carefully synchronized.

In this paper, we consider unsupervised depth estimation from monocular videos. In practice, videos are ubiquitous and unlimited. Many previous works in this line are based on the static scene assumption [Mahjourian et al., 2018; Wang et al., 2018; Zhou et al., 2017], where only camera motion is considered, leading to inaccurate results for moving objects. Several works tried to explicitly model object motion, either with optical flow [Yin and Shi, 2018] or SE(3) transforms [Casser et al., 2019] by assuming rigid motion of objects. However, Yin and Shi [2018] do not report obvious improvement even with the residual flow learning scheme and Casser et al. [2019] only model the motion of rigid objects like cars in driving scenes. For real world scenarios,
deformable objects are often present in various forms such as pedestrians and animals.

We observe that the coupling of camera motion and individual object motion often causes ambiguities and may confuse the learning process in dynamic scenes. Therefore we propose to disentangle camera motion and individual object motion between adjacent frames by introducing an additional transformation \( f_\theta : \mathbb{R}^2 \rightarrow \mathbb{R}^2 \) for every independently moving object to deform itself to adjacent frame, as illustrated in Fig. 1. The transformation is modeled as a bicubic deformation and the transformation parameters \( \theta \) are learned by a CNN without any predefined correspondences, which is fully guided by image appearance dissimilarity (see Sec. 3.1 for detail). To realize the individual transformation, the existing instance segmentation method, Mask R-CNN [He et al., 2017], is used to segment objects out in each frame, which is only needed at training time and helps the motion representation to learn a better depth network.

The paper has made the following contributions:

- We present a learning based approach to estimate depth from unconstrained monocular videos. The approach consists of DepthNet, PoseNet and Region Deformer Networks (RDN), which does not need ground truth supervision.

- We propose a deformation based motion representation to model non-rigid motion of individual objects between adjacent frames on 2D images. This representation is general and enables our method to be applicable to unconstrained monocular videos.

- We conduct extensive experiments on three datasets across diverse scenes. Our method can not only achieve state-of-the-art performance on standard benchmarks KITTI and Cityscapes, but also show promising results on a crowded pedestrian tracking dataset, which validates the effectiveness of the proposed deformation based motion representation.

2 Related Works

This section briefly reviews some learning based depth estimation work that is most related to ours.

**Supervised Depth Estimation.** Many supervised methods have been developed to estimate depth [Eigen et al., 2014; Li et al., 2015; Liu et al., 2016; Xie et al., 2016]. These methods use CNN to learn a mapping from RGB images to depth maps. However, they need a training dataset with ground truth depth of real world scenes which is hard to acquire, especially in outdoor scenarios, and hence limits its applicability. Several works try to resolve this limitation by using synthetic data [Mayer et al., 2018; Zheng et al., 2018] or images from Internet [Chen et al., 2016; Li and Snavely, 2018], but special care must be taken to generate high quality training data, which can be very time-consuming.

**Unsupervised Depth Estimation.** Unsupervised approaches use image reconstruction loss between adjacent frames to provide self-supervision. Garg et al. [2016] propose to use calibrated stereo pairs as supervision to train a single view depth CNN. Godard et al. [2017] further improve the performance by imposing left-right consistency constraints. Zhou et al. [2017] propose to learn depth and ego-motion from monocular videos under the static scene assumption, with an additional learned explainability mask to ignore the motion of objects. Yin and Shi [2018] propose to learn a residual flow to handle the motion of objects. Zou et al. [2018] jointly learn depth and optical flow from monocular videos with a cross-task consistency loss in the rigid scene, but the depth of moving objects does not benefit from the learned optical flow since depth and flow consistency is enforced only between static regions.

Our work focuses on depth estimation from monocular videos as videos are more easily available than rectified stereo pairs. The work most similar to ours is [Casser et al., 2019]. However, there are two important differences: (i) Casser et al. [2019] model object motion in 3D with SE(3) transformation, which is good for rigidly moving objects, like cars in driving scenes. We use a deformation based representation to model object motion in 2D image plane, which is more general to be applicable to diverse real world scenarios. (ii) To handle the common issue that cars moving in front of camera at roughly the same speed are often projected into infinite depth in monocular setting, Casser et al. [2019] propose to impose object size constraints depending on the height of object segmentation mask, which is not suitable for deformable objects as the actual scale can be varied over time. Also, the constraints in [Casser et al., 2019] are learned by network which can be tricky to find the good hyper-parameters. Instead we choose to use a simple yet efficient prior inspired from [Ranftl et al., 2016], which is more general for diverse scenes and has no parameters to learn.

**Learning Geometric Transformation.** Spatial Transformer Networks (STN) [Jaderberg et al., 2015] build the first learnable module in the network architecture to handle geometry variation of input data, which is realized by learning a global parametric transformation. Deformable ConvNets [Dai et al., 2017] further extend STN by learning offsets to regular grid sampling locations in the standard convolution. STN and Deformable ConvNets are both aiming at designing network architectures with geometry invariant for supervised tasks like classification and segmentation. Our deformation based motion representation aims at learning a transformation for each of individual objects to model object motion between adjacent frames. WarpNet [Kanazawa et al., 2016] shares a similar spirit to match images by learning a transformation, but training WarpNet needs the supervision of artificial correspondences. Our approach is fully unsupervised and in the context of depth estimation from videos.

3 Proposed Method

This section presents our generic framework for unsupervised depth estimation from unconstrained monocular videos. The input is a sequence of video frames \( \{I_t\}_{t=1}^N \), where \( H, W, C \) represent frame \( I_t \)'s height, width and number of channels, respectively. Our goal is to estimate the corresponding depth maps \( \{D_t\}_{t=1}^N \), where \( H \times W \). For this purpose, we build a DepthNet to learn the mapping from color
3.1 Basic Formulation

With \( D_t, I_{t+1}, T_{t+1 \rightarrow t} \), we synthesize image

\[
\hat{I}_{t+1 \rightarrow t} = W(D_t, I_{t+1}, T_{t+1 \rightarrow t})
\]

corresponding to frame \( I_t \), where \( W \) is a warping function. We first construct \( W \) based on the static scene assumption, and then improve it for dynamic scenes by adding individual object motions in Sec. 3.2.

For each pixel coordinates \( p = (x, y) \) in frame \( I_t \), we can obtain its projected coordinates \( \hat{p} = (\hat{x}, \hat{y}) \) in frame \( I_{t+1} \) with the estimated depth \( D_t \) and camera transformation \( T_{t+1 \rightarrow t} \)

\[
\hat{p} \sim KT_{t+1 \rightarrow t}D_t(p)K^{-1}h(p),
\]

where \( h(p) = (x, y, 1) \) denotes the homogeneous coordinates of \( p \) and \( K \) denotes the camera intrinsic matrix. We use the bilinear mechanism used in [Jaderberg et al., 2015] to sample frame \( I_{t+1} \) to create image \( \hat{I}_{t+1 \rightarrow t} \).

The appearance dissimilarity between reconstructed image \( \hat{I}_{t+1 \rightarrow t} \) and \( I_t \) is defined as

\[
\ell = \rho(I_t, \hat{I}_{t+1 \rightarrow t}),
\]

where function \( \rho \) is a combination of \( L_1 \) photometric error and Structure Similarity (SSIM) [Wang et al., 2004]:

\[
\rho(I, \hat{I}) = \frac{1 - SSIM(I, \hat{I})}{2} + (1 - \alpha)||I - \hat{I}||_1.
\]

To handle occlusion/disocclusion between adjacent frames, per-pixel minimum of the dissimilarities with previous frame and next frame is used as proposed in [Godard et al., 2018]:

\[
L_{ap} = \min(\rho(I_t, \hat{I}_{t+1 \rightarrow t}), \rho(I_t, \hat{I}_{t-1 \rightarrow t})),
\]

where \( \hat{I}_{t \rightarrow t} \) is reconstructed image from \( I_{t-1} \).

To regularize the depth, we further impose an image-aware depth smoothness constraint as commonly used in previous works [Godard et al., 2017; Yin and Shi, 2018]:

\[
L_s = \sum_{x,y} ||\partial_x D_t||e^{-\|\partial_x I_t\|} + ||\partial_y D_t||e^{-\|\partial_y I_t\|}.
\]

The total loss function is a combination of \( L_{ap} \) and \( L_s \) applied on four different image scales, ranging from the model’s input resolution, to an image that is 1/8 in height and width:

\[
L = \sum_{k=0}^{3} L_{ap}^{(k)} + \lambda \cdot \frac{1}{2k} L_s^{(k)},
\]

where \( \lambda \) is a hyper-parameter. We use Eq. 7 as our baseline model, which is similar to that in [Casser et al., 2019].

3.2 Region Deformer Networks

For dynamic scenes, individual objects may have their own motions besides camera motion. We propose to explicitly model individual object motion on 2D images. Specifically, our goal is to learn the displacement vector \( \Delta p = (\Delta x, \Delta y) \) of every pixel belonging to moving objects. Then the corresponding pixel of \( p \) can be found by \( p' = \hat{p} + \Delta p \). This process is accomplished by learning a function \( f : \mathbb{R}^2 \rightarrow \mathbb{R}^2 \) for each object to map \( (x, y) \) to the displacement \( (\Delta x, \Delta y) \). The
basic requirement for $f$ is that it should be flexible enough to model non-rigid object motion. We choose the bicubic function, which is expressed as

$$\begin{aligned}
\Delta x &= \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j, \\
\Delta y &= \sum_{i=0}^{3} \sum_{j=0}^{3} b_{ij} x^i y^j.
\end{aligned}$$

(8)

The bicubic function is widely used in various applications. It has low computational cost, and meanwhile it contains 32 coefficients, providing sufficient degrees of freedom for modeling freeform deformation (or motion).

To learn the transformation parameters $\{a_{ij}\}_{i,j=0}^{3}$ and $\{b_{ij}\}_{i,j=0}^{3}$, we design the Region Deformer Networks (RDN). The workflow of the RDN is illustrated in Fig. 1. Given two adjacent frames $I_t$ and $I_{t+1}$, an instance segmentation model [He et al., 2017] is used to segment objects within each frame. Let $M_i^t$ and $M_i^{t+1}$ denote the $i$-th binary object segmentation masks in frame $I_t$ and $I_{t+1}$, respectively. We first compute the reconstructed image $\hat{I}_{t+1\rightarrow t}$ and mask $\hat{M}_{t+1\rightarrow t}$ by camera motion using Eq. 1, which eliminates camera motion between adjacent frames. The input of RDN is the concatenation of objects $O_i^t = I_t \odot M_i^t$ and $\hat{O}_i^{t+1\rightarrow t} = \hat{I}_{t+1\rightarrow t} \odot \hat{M}_{t+1\rightarrow t}$ in $I_t$ and $I_{t+1\rightarrow t}$, respectively, where $\odot$ denotes element-wise multiplication, making only the $i$-th object visible for every independent RDN. When multiple objects exist in a single frame, multiple RDNs are used in parallel to model every independent object motion. The outputs of RDN are the parameters of the transformation. By applying the transformation to its corresponding object, we obtain the displacement of the pixels belonging to that object.

Now we are ready to refine the function $\mathcal{W}$ by defining a new warping function $\mathcal{W}^\star$. If a pixel $p$ belongs to static background, we compute its correspondence by Eq. 2. If $p$ belongs to moving objects, its correspondence is found by $p^\star = \hat{p} + \Delta p$, where $\hat{p}$ is from camera motion using Eq. 2 and $\Delta p$ is from object motion obtained by RDN. In general, for every pixel $p$ in frame $I_t$, we can get its correspondence by

$$p^\star = \hat{p} + M_t \cdot \Delta p,$$

(9)

where $M_t = \sum_{i=1}^{S} M_i^t$ is the binary instance segmentation mask for $I_t$: $M_i^t(p)$ is 1 if $p$ belongs to moving objects; otherwise 0, and $S$ is the total number of masks in $I_t$. By modeling individual object motion with RDN, we can get more accurately reconstructed image

$$I_{t+1\rightarrow t}^\star = \mathcal{W}^\star(D_t, I_{t+1}, T_{t+1\rightarrow t}, f_1, f_2, \cdots, f_S),$$

(10)

where $\{f_i\}_{i=1}^{S}$ are the individual transformations learned by the RDN. Similarly, we can generate image $I_{t\rightarrow t-1}^\star$.

### 3.3 Object Depth Prior

Depth estimation from monocular videos has an issue that objects moving with camera at the roughly same speed are often projected to infinite depth, as this shows very little appearance change, resulting in low reprojection error [Godard et al., 2018]. Casser et al. [2019] propose to impose object size constraints by additionally learning the actual scales of objects. These constraints are internally based on the assumption that object scales are fixed. However, in real world scenarios, deformable objects are often present, like pedestrians and animals, which are not applicable for these constrains. Furthermore, as the actual scales are also learned during the training process, it can be tricky to find the good hyper-parameters.

We propose to use a simple yet efficient prior inspired from [Ranftl et al., 2016]: objects are supported by their surrounding environment, which is often true in most real world scenes. This prior can be used by requiring the depths of moving objects to be smaller or equal to their horizontal neighbors. However, noticing that overlapping objects may exist in the real world, which might violate this depth prior, we thus introduce a soft constraint, which is formulated as

$$L_{\text{prior}} = \max(d_{\text{obj}} - d_{\text{neigh}} - \delta, 0),$$

(11)

where $d_{\text{obj}}$ is the mean depth of an individual object, $d_{\text{neigh}}$ is the mean depth of its horizontal neighbors in a small range, and $\delta$ is a small positive number to handle exceptions violating our depth prior. The main idea here is to use the depth prior to prevent the degenerated cases of infinite depth. Note that if this prior is satisfied, which happens most of the time, Eq. 11 actually has no use (i.e., the loss becomes 0).

By incorporating the RDN and object depth prior, our final loss function can be expressed as

$$L^\star = \sum_{k=0}^{3} L_{ap}^{(k)} + \lambda \cdot \frac{1}{2} L_{s}^{(k)} + \mu \cdot L_{\text{prior}},$$

(12)

where $\mu$ is a hyper-parameter. The only difference between $L_{ap}^{(k)}$ and $L_{s}^{(k)}$ is that we replace $I^*_{t+1\rightarrow t}, I_{t\rightarrow t-1}$ in Eq. 1 with $I^*_{t+1\rightarrow t}, I^*_{t\rightarrow t-1}$. We treat Eq. 12 as our motion model.

### 4 Experiments

#### 4.1 Datasets

We conduct experiments on several datasets across diverse scenes, including not only standard benchmarks KITTI and Cityscapes, but also a publicly available pedestrian tracking dataset. The details of each dataset are given below.

**KITTI.** The KITTI dataset [Geiger et al., 2012] is a popular benchmark for scene understanding in outdoor driving scenario. Only the monocular video sequences are used for training and no ground truth depth is needed. Following the approach of [Zhou et al., 2017] to pre-process the dataset, we randomly split the processed images as training and validation sets (about 40K images for training and 4K images for validation). The performance is evaluated on Eigen split [Eigen et al., 2014] using the standard evaluation protocol.

**Cityscapes.** The Cityscapes [Cordts et al., 2016] is an outdoor driving dataset similar to KITTI, but with more moving objects. We do the same data pre-processing as [Casser et al., 2019], resulting in 38,675 images for training. We use this dataset for training and the evaluation is done on Eigen split.

**Pedestrian Tracking Dataset.** To validate that our motion representation is general enough to model deformable objects, we collect videos from a publicly available pedestrian
Figure 3: Qualitative comparisons on Eigen test split. GT denotes ground truth depth, which is interpolated for visualization purpose, and the upper regions are cropped as they are not available. Compared with other algorithms, our method can better capture scene structures and moving objects like cars and riding people.

Table 1: Unsupervised monocular depth estimation results on Eigen test split of KITTI raw dataset. We use K and C to denote models trained on KITTI and Cityscapes dataset, respectively. Abs Rel, Sq Rel, RMSE and RMSE log are error metrics (lower is better). $\delta < 1.25$, $\delta < 1.25^2$ and $\delta < 1.25^3$ are accuracy metrics (higher is better). The best performance in each group is highlighted in bold. For ablation study, we add two baselines: Ours (Rigid) and Ours (w/o prior), where the former is the one by replacing bicubic transform in our full model with rigid motion representation, and the latter is Ours (motion) without the depth prior.

| Method          | Dataset | Abs Rel | Sq Rel | RMSE  | RMSE log | $\delta < 1.25$ | $\delta < 1.25^2$ | $\delta < 1.25^3$ |
|-----------------|---------|---------|--------|-------|----------|----------------|------------------|------------------|
| Zhou et al.     | K       | 0.208   | 1.768  | 6.856 | 0.283    | 0.678          | 0.885            | 0.957            |
| Mahjourian et al.| K       | 0.163   | 1.240  | 6.220 | 0.250    | 0.762          | 0.916            | 0.968            |
| Yin and Shi     | K       | 0.155   | 1.296  | 5.857 | 0.233    | 0.793          | 0.931            | 0.973            |
| Wang et al.     | K       | 0.151   | 1.257  | 5.583 | 0.228    | 0.810          | 0.936            | 0.974            |
| Zou et al.      | K       | 0.150   | 1.124  | 5.507 | 0.223    | 0.806          | 0.933            | 0.973            |
| Casser et al.   | K       | 0.1412  | 1.0258 | 5.2905| 0.2153   | 0.8160         | 0.9452           | 0.9791           |
| Ours (Baseline) | K       | 0.1436  | 1.1732 | 5.4468| 0.2188   | 0.8202         | 0.9428           | 0.9762           |
| Ours (Motion)   | K       | 0.1377  | 1.0155 | 5.3523| 0.2165   | 0.8226         | 0.9428           | 0.9762           |
| Ours (Rigid)    | K       | 0.1391  | 1.0384 | 5.3518| 0.2174   | 0.8231         | 0.9425           | 0.9759           |
| Ours (w/o prior)| K       | 0.1419  | 1.1635 | 5.4739| 0.2206   | 0.8217         | 0.9416           | 0.9751           |
| Casser et al.   | C       | 0.1876  | 1.3541 | 6.3166| 0.2641   | 0.7135         | 0.9046           | 0.9667           |
| Ours (Baseline) | C       | 0.1929  | 1.6131 | 6.4098| 0.2680   | 0.7189         | 0.9071           | 0.9641           |
| Ours (Motion)   | C       | 0.1816  | 1.3160 | 6.1484| 0.2573   | 0.7263         | 0.9124           | 0.9677           |
| Ours (Rigid)    | C       | 0.1820  | 1.3500 | 6.1657| 0.2575   | 0.7313         | 0.9119           | 0.9672           |
| Ours (w/o prior)| C       | 0.1973  | 1.9500 | 6.6228| 0.2695   | 0.7200         | 0.9074           | 0.9630           |

4.2 Implementation Details

Our method is implemented in TensorFlow. The input images are resized to $128 \times 416$ when training on KITTI and Cityscapes datasets, and to $240 \times 320$ when training on the pedestrian tracking dataset. The loss weights $\mu, \alpha$ are set to 0.5, 0.15, respectively. The smoothness weight $\lambda$ is set to 0.04 when training on KITTI and the pedestrian tracking dataset [Ess et al., 2009], which was recorded on a crowded pedestrian zone. This dataset is very challenging as large human deformations are frequently observed, where 9,369 images are used for training.
dataset, and to 0.008 when training on Cityscapes dataset. The \( \delta \) in Eq. 11 is set to 0.01 when training on KITTI and the pedestrian tracking dataset, and to 0.5 when training on Cityscapes dataset due to the different data distributions of the datasets. The batch size is chosen to be 4 and Adam [Kingma and Ba, 2014] is used to optimize the network with \( \beta_1 = 0.9, \beta_2 = 0.99 \). When training the baseline model, the learning rate is set to \( 2 \times 10^{-4} \). Our motion model is trained with learning rate of \( 2 \times 10^{-5} \) and the network weights are initialized from our trained baseline model. We use the same DepthNet and PoseNet architectures as [Casser et al., 2019]. Our RDN uses the same architecture as PoseNet except for the last output layer. We will make our code and trained models publicly available to the community.

**Evaluation Metrics:** We evaluate our method with standard metrics reported in [Eigen et al., 2014], which include 4 error metrics (Abs Rel, Sq Rel, RMSE, RMSE log) and 3 accuracy metrics (\( \delta < 1.25, \delta < 1.25^2, \delta < 1.25^3 \)). The definition of these metrics is referred to [Eigen et al., 2014].

### 4.3 Results

**The KITTI and Cityscapes Datasets.** We report our depth estimation results on standard Eigen test split [Eigen et al., 2014] of KITTI raw dataset in Tab. 1. All methods are evaluated on KITTI dataset, no matter whether they are trained on KITTI or Cityscapes dataset. We achieve comparable results with [Casser et al., 2019] when training on KITTI dataset, which is specially designed for outdoor driving scenes. When training on more dynamic dataset Cityscapes, our performance is consistently better than that of [Casser et al., 2019]. More evidences can be seen from the qualitative comparisons in Fig. 3. On the other hand, comparing our motion model with the baseline, the results are consistently better no matter training on KITTI or Cityscapes dataset (see Tab. 1 and the visual results in Fig. 4).

**Ablation Study.** To further evaluate our proposed bicubic motion representation, we create a new baseline named ‘Ours (rigid)’, where we replace the bicubic function in Eq. 8 with rigid motion representation and keep all the other settings the
The scale ambiguity issue [Wang (Cars)]. We analyze the possible reason is that the coupling better than the rigid representation, even on the rigid objects in Tab. 1. Our bicubic motion representation is consistently which is quite different from KITTI and Cityscapes datasets.

To evaluate the contribution from the object depth prior, we add another baseline called ‘Ours (w/o prior)’, which disables the prior in our full motion model. As shown in Tab. 1, compared with Ours (Motion), the performance of Ours (w/o prior) degrades a lot, which verifies that our object prior is simple yet efficient to handle the infinite depth issue as explained in Sec. 3.3.

Evaluation on Moving Objects. In order to further analyze the effect of the proposed algorithm on different types of objects, we classify the objects in Eigen test split and show the results of different algorithms on different object categories. Tab. 2 shows the quantitative results on cars (Cars), people (People) and all possibly moving objects (All, include other objects like motorcycle besides cars and people), respectively. Notable improvements can be observed when comparing our full motion model (Bicubic) with baseline (Baseline) no matter training on KITTI (K) or Cityscapes (C) dataset. On the other hand, we replace the bicubic function with rigid motion representation and keep all the other settings the same to fully compare the rigid motion representation with our proposed bicubic motion representation. We denote this model as Rigid, which is the same as Ours(Rigid) in Tab. 1. Our bicubic motion representation is consistently better than the rigid representation, even on the rigid objects (Cars). We analyze the possible reason is that the coupling of rigid representation and depth (see Eq. 2) may suffer from the scale ambiguity issue [Wang et al., 2018], which makes learning harder in the unsupervised setting. As a comparison, our bicubic motion representation directly models object motion on 2D image plane, leading to better performance for various objects.

The Pedestrian Tracking Dataset. To illustrate the generality of our proposed bicubic motion representation, we conduct experiments on a crowded pedestrian tracking dataset, which is quite different from KITTI and Cityscapes datasets and particularly challenging due to the presence of many deformable pedestrians. Fig. 5 visualizes the depth prediction results of samples from this dataset. Clear improvements can be seen from our motion model. The promising results show the generality of our proposed bicubic motion representation and indicate that our framework is applicable to unconstrained monocular videos.

5 Conclusion

We have presented a learning based approach to estimate depth from unconstrained monocular videos. The approach consists of DepthNet, PoseNet and RDN, for which a deformation based bicubic motion representation is proposed to model object motions in diverse scenes. The experimental results on several datasets show the promising performance of the proposed approach and validate the effectiveness of the deformation based motion representation as well. In future, we would like to incorporate more domain knowledge such as non-rigid structure from motion into the learning process in order to improve the depth estimation in dynamic scenes.

Acknowledgements. We would like to thank Dr. Haiyong Jiang for the helpful discussion during this work. The authors are supported by the National Key R&D Program of China (No. 2016YFC0800501), the National Natural Science Foundation of China (No. 61672481), and the Youth Innovation Promotion Association CAS (No. 2018495).

References

[Casser et al., 2019] V. Casser, S. Pirk, R. Mahjourian, and A. Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. AAAI, 2019.

[Chen et al., 2016] W. Chen, Z. Fu, D. Yang, and J. Deng. Single-image depth perception in the wild. In NIPS, pages 730–738, 2016.

[Cordts et al., 2016] 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 CVPR, pages 3213–3223, 2016.

[Dai et al., 2017] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. CoRR, abs/1703.06211, 1(2):3, 2017.

[Eigen et al., 2014] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In NIPS, pages 2366–2374, 2014.

[Ess et al., 2009] A. Ess, B. Leibe, K. Schindler, and Luc Van Gool. Robust multiperson tracking from a mobile platform. IEEE Transactions on Pattern Analysis and Machine Intelligence, 31(10):1831–1846, 2009.

[Garg et al., 2016] R. Garg, V. K. BG, G. Carneiro, and I. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In ECCV, pages 740–756. Springer, 2016.

[Geiger et al., 2012] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In CVPR, pages 3354–3361. IEEE, 2012.
[Godard et al., 2017] C. Godard, O. Aodha, and G. Brostow. Unsupervised monocular depth estimation with left-right consistency. In CVPR, volume 2, page 7, 2017.

[Godard et al., 2018] C. Godard, Oisin Mac Aodha, and G. Brostow. Digging into self-supervised monocular depth estimation. arXiv preprint arXiv:1806.01260, 2018.

[He et al., 2017] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In ICCV, pages 2980–2988. IEEE, 2017.

[Jaderberg et al., 2015] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In NIPS, pages 2017–2025, 2015.

[Kanazawa et al., 2016] A. Kanazawa, D. Q. Jacobs, and M. Chandraker. Warpnet: Weakly supervised matching for single-view reconstruction. In CVPR, pages 3253–3261, 2016.

[Kingma and Ba, 2014] Diederik P Kingma and J. Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[Li and Snavely, 2018] Z. Li and N. Snavely. Megadepth: Learning single-view depth prediction from internet photos. In CVPR, pages 2041–2050, 2018.

[Li et al., 2015] B. Li, C. Shen, Y. Dai, Anton Van Den Hengel, and M. He. Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In CVPR, pages 1119–1127, 2015.

[Liu et al., 2016] F. Liu, C. Shen, G. Lin, and I. Reid. Learning depth from single monocular images using deep convolutional neural fields. IEEE Trans. Pattern Anal. Mach. Intell., 38(10):2024–2039, 2016.

[Mahjourian et al., 2018] R. Mahjourian, M. Wicke, and A. Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In CVPR, pages 5667–5675, 2018.

[Mayer et al., 2018] N. Mayer, E. Ilg, P. Fischer, C. Hazirbas, D. Cremers, A. Dosovitskiy, and T. Brox. What makes good synthetic training data for learning disparity and optical flow estimation? International Journal of Computer Vision, pages 1–19, 2018.

[Ranftl et al., 2016] R. Ranftl, V. Vineet, Q. Chen, and V. Koltun. Dense monocular depth estimation in complex dynamic scenes. In CVPR, pages 4058–4066, 2016.

[Wang et al., 2004] Z. Wang, A. C. Bovik, Hamid R Sheikh, and E. Simoncelli. Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing, 13(4):600–612, 2004.

[Wang et al., 2018] C. Wang, J. M. Buenaposada, R. Zhu, and S. Lucey. Learning depth from monocular videos using direct methods. In CVPR, pages 2022–2030, 2018.

[Xie et al., 2016] J. Xie, R. Girshick, and A. Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In ECCV, pages 842–857. Springer, 2016.

[Yin and Shi, 2018] Z. Yin and J. Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In CVPR, volume 2, 2018.

[Zheng et al., 2018] C. Zheng, T.-J. Cham, and J. Cai. T2net: Synthetic-to-realistic translation for solving single-image depth estimation tasks. In ECCV, pages 767–783, 2018.

[Zhou et al., 2017] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and egomotion from video. In CVPR, volume 2, page 7, 2017.

[Zou et al., 2018] Y. Zou, Z. Luo, and J.-B. Huang. Df-net: Unsupervised joint learning of depth and flow using cross-task consistency. In ECCV, pages 38–55. Springer, 2018.