2022 26th International Conference on Pattern Recognition (ICPR)
August 21-25, 2022, Montréal, Québec, Canada

# PMPNet: Pixel Movement Prediction Network for Monocular Depth Estimation in Dynamic Scenes

Kebin Peng, John Quarles, Kevin Desai
{kebin.peng, john.quarles, kevin.desai}@utsa.edu
Department of Computer Science
The University of Texas at San Antonio
One UTSA Circle, San Antonio, TX 78249

2022 26th International Conference on Pattern Recognition (ICPR) | 978-1-6654-9062-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICPR56361.2022.9956096

***Abstract*—In this paper, we propose a novel method for monocular depth estimation in dynamic scenes. We first explore the arbitrariness of object's movement trajectory in dynamic scenes theoretically. To overcome the arbitrariness, we use assume that points move along a straight line over short distances and then summarize it as a triangular constraint loss in two dimensional Euclidean space. This triangular loss function is used as part of our proposed pixel movement prediction network, PMPNet, to estimate a dense depth map from a single input image. To overcome the depth inconsistency problem around the edges, we propose a deformable support window module that learns features from different shapes of objects, making depth value more accurate around edge area. The proposed model is trained and tested on two outdoor datasets - KITTI and Make3D, as well as an indoor dataset - NYU Depth V2. The quantitative and qualitative results reported on these datasets demonstrate the success of our proposed model when compared against other approaches. Ablation study results on the KITTI dataset also validate the effectiveness of the proposed pixel movement prediction module as well as the deformable support window module.**

## I. Introduction

Monocular depth estimation for dynamic scenes, i.e., estimating depth value for a moving point seen from a monocular moving camera, is a challenging task. In static scenes, like Middleburry in the 2014 dataset [1], only camera moves and the objects are static. However in dynamic scenes, such as in autonomous driving, not only will the camera move but also the objects. The moving objects such as cars and trucks violate the static world assumption [2]. Hence, the object motion relative to the camera should contain the movement of the object as well as the movement of the camera [3]. For this reason, previous works have used two neural networks, one for estimating depth the other one for estimating camera motion from monocular video [4] or synchronized stereo pairs [5], [6]. Alternatively, [3] used a neural network to recognize possible moving objects then estimate depth with other neural networks.

Although existing monocular depth estimation methods [7], [8], [9], [10] perform well in autonomous driving datasets such as KITTI [11], most of them do not consider dynamic scenes, which is common in real-world autonomous driving The methods that work on such dynamic scenes [3], [5], [6], [12], [13] do not consider the arbitrariness of object's movement trajectory. In other words, a pixel could move to any place in the next time instance and these methods do not add any constraints on the pixel/object movement trajectory. Such arbitrariness could result in a huge search space, which makes it hard for neural networks to learn useful information.

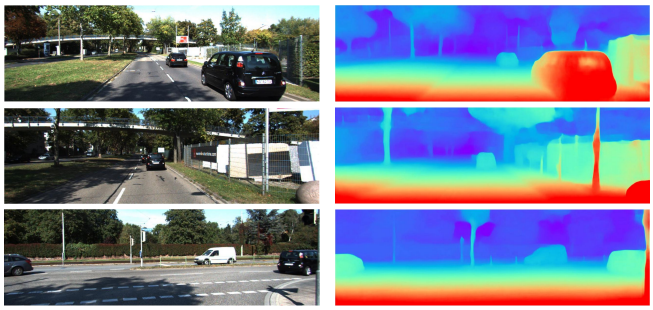

Fig. 1: **Monocular Depth Estimation Results:** Two examples from the KITTI dataset [11]. Our model estimates accurate depth maps with non-blur object edges.

In this paper, we try to reduce such arbitrariness by constraining the trajectory of the pixel. For this goal, we use the assumption stated in [14], i.e., the pixel moving along a straight-line and the camera center of projection traces a straight line during camera motion. Such an assumption is reasonable in an autonomous driving dataset because in most cases the cars and the camera on cars will move along a straight-line (left/right turn can be viewed as a combination of many small straight-lines). Considering this assumption, we propose the *Pixel Movement Prediction Module* to estimate two two-dimensional vectors $v_1$, $v_2$ that could span a null space. As concluded by [14], in the null space, the straight-line $L$ could be represented by the linear combination of $v_1$, $v_2$. We summarize this conclusion into a *Pixel Movement Triangular Constrain Loss Function* to train our model. Using the *Pixel Movement Prediction Module*, we synchronize a "right image" feature map according to the input image's feature map (the input image is viewed as the "left image" of the stereo pairs). One thing needs to be noticed is that we synchronize the feature map of "right image" instead of the "right image" itself. This is because the feature map can be used directly to construct the cost volume (see Section III - Formula 1).

In most monocular depth estimation methods, there exists depth inconsistency around edge area. Such inconsistencies exist because standard CNNs consist of fixed geometric

structures in their architecture, which restrict the learning to standard geometric transformations. [15] further proposed deformable convolutions that enable free form transformations of the sampling grid and thereby achieving higher accuracy. In our work, we adopt deformable convolution layers which especially improves the depth estimation at the object edges, where the pixel movement may involve compound transformations. These deformable convolution layers help better estimate depth near the edges. Figure 1 shows the depth estimation results of our proposed PMPNet on three different examples from the KITTI dataset [11].

### A. Contributions

Following are the major contributions of our work:

- We analyze the arbitrariness of object's / pixel's trajectory for dynamic scenes theoretically and use a straight-line assumption to constrain the trajectory.
- We propose a novel deep neural network architecture called PMPNet. It consists of a *Pixel Movement Prediction Module* which provides two pixel movement predictions and a third straight-line prediction. The relation between pixel movements and straight-line is summarized into a novel *Triangular Constraint Loss Function*.
- We propose a *Deformable Convolution Support Window*, which better addresses the problem of blurred edges during monocular depth estimation.

## II. Related Work

### A. Supervised Monocular Depth Estimation

[7] view Monocular Depth Estimation as a regression problem and proposed a transformer based architecture. It divides the depth range into bins whose center value is estimated adaptively per image. [9] proposed using the vision transformers as a backbone for monocular depth estimation. [16] proposed a framework where networks act as competitors and collaborators to reach specific goals by benefiting from each other. [17] proposed the deep ordinal regression network, using spacing-increasing discretization strategy and the ordinary regression loss. [18] learned the 3D scene geometry from monocular videos to simultaneously estimate depth, pixel movement, and camera movement.

### B. Unsupervised / Self-Supervised Monocular Depth Estimation

As ground truth is expensive to acquire in monocular depth estimation, many researchers focus on unsupervised approaches. [19] proposed feature-metric loss to replace photometric loss and the authors used three different neural networks to estimate depth, image feature, and pose. [20] proposed an adaptive approach that can make use of sequence information. They also proposed a consistency loss that can make the network ignore unreliable cost volume. [21] proposed a framework for learning depth map, camera ego-motion, and object motions by representing objects in 3D, modeling its motion as SE3 transforms, and incorporating the SE3 transforms into the training process. [22] proposed a view decoder to capture the 3D scene structure as a supervised signal and use that to jointly train a single-view depth estimation network and a pose estimation network.

### C. Monocular Depth Estimation in Dynamic Scenes

Although all the methods in II-A and II-B perform well, they do not consider real-world dynamic scenes. Because dynamic scenes break static world assumption, many methods focus on how to recover depth value when both the camera and the scene are in motion. [23] proposed a two stage method for the same. The first stage performs motion segmentation and the second stage jointly reasons about the scales of different components and their location relative to the camera. This method is motivated by optical flow estimation and hence will fail if optical flow estimation fails. [12] proposed a framework for monocular depth estimation in dynamic scenes by using super-pixel relations between neighboring frames in a monocular video. To solve the scale ambiguity problem in super-pixel relations, the authors assumed that motion and spatial relations between neighboring super-pixels correspond one-to-one. [3] presented a method for training the estimation of depth, ego-motion, and a dense 3D translation field of objects in dynamic scenes. The authors assumed that the dynamic scene is sparse, i.e., most of the scene is static, and they tend to be piece-wise constant for rigid moving objects. However, as the authors pointed out, they did not consider the dense dynamic scenes. Our proposed approach does not make such assumptions on sparseness of the dynamic scene.

## III. PMPNet - Pixel Movement Prediction Network

Our proposed neural network architecture PMPNet, as shown in Figure 2, consists of two main modules - individual pixel movement prediction and deformable support window. Given an input RGB image $I$, we first extract the feature pyramid $\{F_1, F_2\}$ at $1/6^{th}$ and $1/12^{th}$ resolutions respectively. Next, we predict the initial pixel movement $PM_1$ by passing in the feature $F_2$ and the convolved feature $F_1$ to a warp layer. This initial pixel movement $PM_1$ is concatenated in the channel direction with the feature $F_2$. Next, a novel deformable support window approach is used to predict a second pixel movement $PM_2$ that better incorporates the compound transformations, especially at the edges. We then perform multiple convolutions as part of the final pixel movement prediction $PM_3$. This $PM_3$ is used to better constrain the first two pixel movements using a triangular loss function. The final pixel movement $PM_3$ is convolved and passed together with the input image feature $F_2$ to construct the cost volume. This cost volume is computed similar to [24], using the following equation:

$$C(d, x, y) = \frac{1}{N} \langle PM_3(x, y), F_2(x, y - d) \rangle \tag{1}$$

During our 3-part pixel movement prediction, we up-sample the feature $F_2$ before concatenating with the feature $F_1$. Because of this, the computed cost volume will inherently contain the features of the original left image at two different scales.

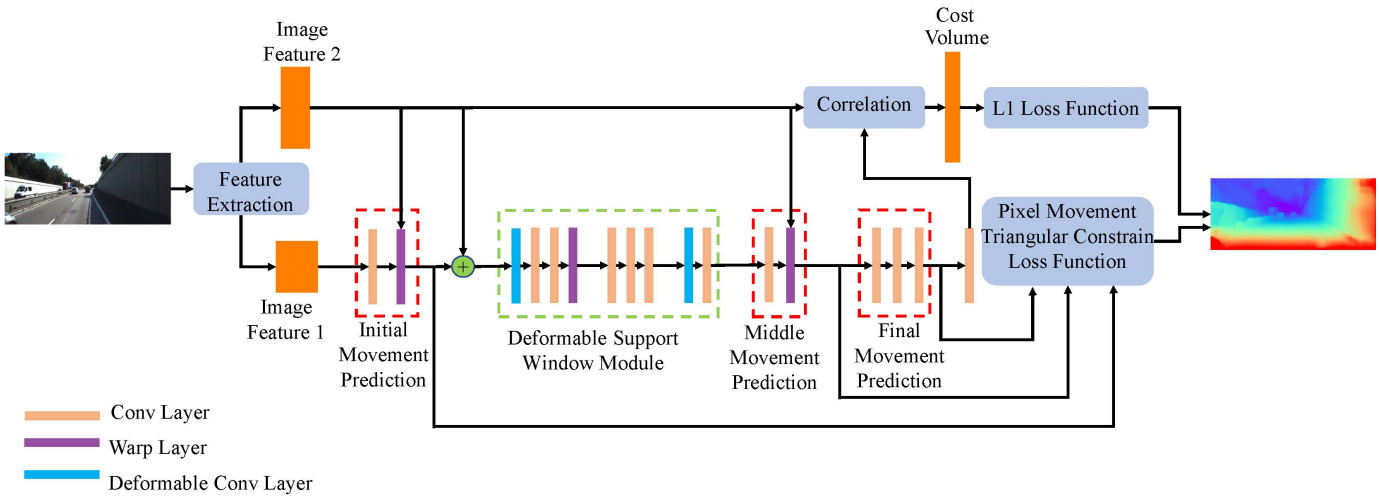


Fig. 2: **Overall Network Architecture:** The model first extracts two features $F_1$ and $F_2$ at $1/6^{th}$ and $1/12^{th}$ scale respectively. These features are passed through the deformable support window module (green dotted box) and the 3-part pixel movement prediction module (red dotted boxes) to finally obtain the cost volume for depth prediction. The overall loss function consists of the traditional depth L1 loss and a Pixel Movement Triangular Constraint loss.

Hence, we do not need to perform a multi-scale aggregation step as done in [24], thereby saving us some training time. The cost volume is then used to estimate the final depth as proposed within the deformable support window module.

### A. *Pixel Movement Prediction Module*

Previous works [16], [18] used the input image and additional information such as monocular video to predict the pixel movements. These approaches assume that all objects and pixels in the scene are static. Rather, we lift that assumption and predict different possible movements for each pixel. However, without that assumption, it is hard to estimate depth value unless some constraints are added on the object/pixel trajectory. Similar to [14], we assume that the pixel moves along a straight-line and the camera center of projection traces a straight line during camera motion. With these two assumptions, there exist two two-dimensional vectors, named $v_1$, $v_2$ which span a null space and a straight line $L$ could be represented in that null space by $L \cong \lambda_1 v_1 + \lambda_2 v_2$. As different CNN layers represent different scales, two parameters $\lambda_1$ and $\lambda_2$ are needed for the prediction of $v_1$ from $F_1$, $v_2$ from $F_2$, and $L$ from $v_1$ and $v_2$.

Figure 2 shows all three pixel movement prediction parts. Based on where the pixel movements are incorporated in our model, we call them as initial or $v_1$, middle or $v_2$, and final or $L$. The initial and middle pixel movement represent two possible move directions of a pixel. Such two possible move directions have good generalization because they could point to any direction.

Also we propose a Pixel Movement Triangular Constraint (PMTC) Loss according to $L \cong \lambda_1 v_1 + \lambda_2 v_2$ as defined by the following equation:

$$L_{PMTC} = \mathcal{L}(v_1 + v_2, L) \tag{2}$$

The final loss function for our monocular depth estimation model is obtained by combining our proposed Pixel Movement Triangular Constraint Loss $L_{PMTC}$ with the standard L1 depth loss used in [24].

As shown in Figure 2, our initial and middle pixel movement prediction parts consist of two layers - convolution and warping. The prediction of the initial pixel movement $v_1$ is performed on the features extracted from the input image. The middle pixel movement $v_2$ is predicted on the output of the deformable support window module. $PM_2$ is passed into the final pixel movement prediction part, which consists of three traditional convolution layers. This final pixel movement $L$ is convolved again and passed to the correlation step alongside the input image feature $F_2$ to compute the cost volume for the depth L1 loss function.

### B. *Deformable Support Window Module*

Majority of the monocular depth estimation method suffers from the depth inconsistency problem of blurring the edge regions. To overcome this, we use a deformable support window module, as shown in Figure 2. This idea is motivated by the property of deformable convolutions that it enables free form deformation of the sampling grid, which would lead to better learning of compound transformations [15]. The use of a deformable support window is inspired and built on top of [25] who proposed a slanted support window to compute disparity more accurately for a slanted surface such as a corridor, and [26] who used a CNNs layer to learn the slanted support window. However, our support window module is different from [26] in that our method uses the deformable convolution layer to learn depth, which performs better near edge areas. The use of these deformable convolution layers provides better depth estimates near the edges, thereby addressing the problem of having blurry-edges.

### C. Depth Regression

To estimate the depth image, we propose a new depth generation approach. Compared with [7], [24], our formula combines the offsets from the deformable convolution layer, which learn the compound geometric transformations near the edges. We first apply softmax [36] to build an initial depth $\widetilde{d}(p_{i,j})$ for each pixel $p_{i,j}$. This depth value can be given by the following equation:

$$\widetilde{d}(p_{i,j}) = \sum_{d=0}^{N-1} d(p_{i,j}) \times \sigma(C(d(p_{i,j}))) \quad (3)$$

where $N$ is the number of channels for the cost volume layer. $\sigma$ is the softmax function and $C(d(p_{i,j}))$ is the matching cost for $p_{i,j}$ with depth $d$.

Next, we use this initial depth $\widetilde{d}(p_{i,j})$ in our new depth equation as follows:

$$d_{i,j} = (1-\alpha)\widetilde{d}(p_{i,j}) + \alpha \sum_{k=1}^{8} d(p_{m,n} + \Delta_k(i,j)) \quad (4)$$

where $\Delta_k(i,j)$ is the k-th offset of $p_{i,j}$ from the deformable convolution layer used in the proposed Deformable Support Window Module. As we select the support window of size $3 \times 3$, hence there can only be 8 offsets. $d(p_{m,n} + \Delta_k(m,n))$ represents the depth of one of the 8 neighboring pixels of $p_{i,j}$ plus an offset $\Delta$. The final depth values for all pixels are regressed by using a traditional L1 loss and the proposed Pixel Movement Triangular Constraint Loss $L_{PMTC}$, represented by Equation 2.

## IV. Experiments & Results

In this section, we evaluate our proposed model by conducting experimental analysis on two outdoor datasets: KITTI [11] and Make3D [37] as well as an indoor dataset: NYU Depth V2 [38]. For each dataset we report two types of results, quantitative and qualitative, in comparison with other state-of-the-art monocular depth estimation methods.

For *quantitative* analysis, the evaluation metrics used are as in the previous works [27], [31]. For the KITTI dataset, we use seven metrics - Abs Rel, Sq Rel, RMSE, RMSE $log$, $\delta < 1.25$, $\delta < 1.25^2$, and $\delta < 1.25^3$. For the Make3D dataset and NYU Depth V2, we use - Abs Rel, Sq Rel, RMSE, and RMSE $log$.

For *qualitative* analysis, we compare the visual results of our approach for a sample image against the other methods.

### A. Implementation Details

Our model is built on the PyTorch framework [39]. We train our model on the KITTI dataset using a single NVIDIA RTX 3080 GPU for 60 epochs with a batch size of 4. Adam optimizer [40] is used with the settings $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We use an adaptive learning rate, starting at $0.001$ and decreasing it by half every 10 epochs after the 20th epoch.

*1) KITTI Dataset Details [11]:* From the KITTI dataset [11], we use all the 61 scenes from the "City", "Residential", "Road", and "Campus" categories in the raw data as our training/testing sets. We use the split by Eigen et al. [27], where we test our model on 697 images from the 29 scenes and train on 10K images picked uniformly from the remaining 32 scenes. For efficient computation, all input images are resized to $336 \times 960$. We cap the depth to $80m$ during evaluation.

| Method | Abs Rel | Sq Rel | RMSE | RMSE $log$ | $\delta < 1.25$ | $\delta < 1.25^2$ | $\delta < 1.25^3$ |
|---|---|---|---|---|---|---|---|
| | Lower is better | | | | Higher is better | | |
| Eigen[27] | 0.203 | 1.548 | 6.307 | 0.282 | 0.702 | 0.890 | 0.890 |
| Yin∗ [28] | 0.155 | 1.296 | 5.857 | 0.233 | 0.793 | 0.931 | 0.973 |
| Garg∗ [5] | 0.152 | 1.226 | 5.849 | 0.246 | 0.784 | 0.921 | 0.967 |
| DDVO∗ [29] | 0.151 | 1.257 | 5.583 | 0.228 | 0.810 | 0.936 | 0.974 |
| EPC++∗ [30] | 0.141 | 1.029 | 5.350 | 0.216 | 0.816 | 0.941 | 0.976 |
| mono2∗ [31] | 0.115 | 0.903 | 4.863 | 0.193 | 0.877 | 0.959 | 0.981 |
| DORN [17] | 0.072 | 0.307 | 2.727 | 0.120 | 0.932 | 0.984 | 0.994 |
| FeatureDepth∗[19] | 0.079 | 0.666 | 3.922 | 0.163 | 0.925 | 0.970 | 0.984 |
| SC-GAN[32] | 0.063 | 0.178 | 2.129 | 0.097 | 0.961 | 0.993 | 0.998 |
| DPT-Hybrid[9] | 0.062 | 0.198 | 2.573 | 0.092 | 0.959 | 0.995 | 0.999 |
| AdaBins[7] | 0.058 | 0.190 | 2.360 | 0.088 | 0.964 | 0.995 | 0.999 |
| SingleNet∗[33] | 0.094 | 0.681 | 4.392 | 0.185 | 0.892 | 0.965 | 0.981 |
| You[34] | 0.058 | 0.194 | 2.415 | 0.092 | 0.960 | 0.995 | 0.999 |
| ManyDepth[20] | 0.087 | 0.685 | 4.142 | 0.167 | 0.920 | 0.968 | 0.983 |
| EdgeConv[35] | 0.056 | 0.169 | 1.925 | 0.087 | 0.964 | 0.994 | 0.999 |
| Ours | **0.049** | **0.157** | **1.831** | **0.071** | **0.994** | **0.996** | **0.999** |

TABLE I: **Quantitative Results - KITTI [11]:** Comparison of our model on the KITTI test split by Eigen [27]. Best results for each metric are in **bold**; second best are underlined. All results presented here are without post-processing. The ∗ denotes that the approach is either self-supervised or unsupervised, and the remaining are supervised approaches.

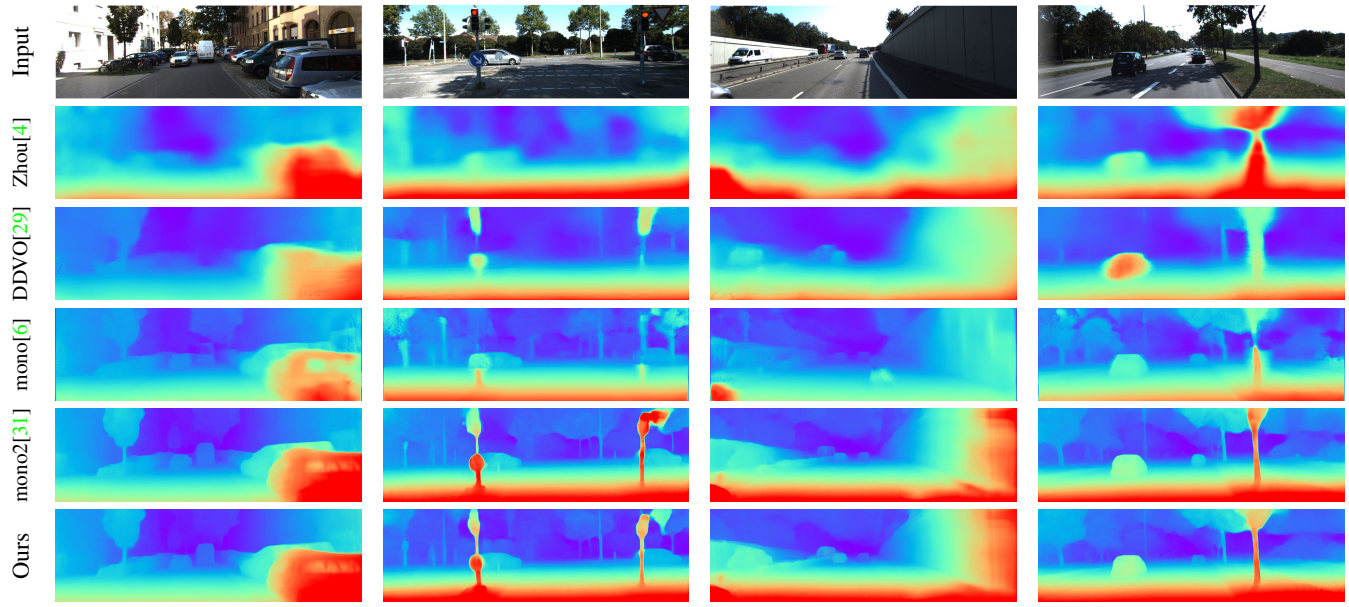


Fig. 3: **Qualitative Results - KITTI [11]:** Depth estimated on four different test images (first row) using our approach (last row) in comparison with four other methods (remaining rows).

| Method | Abs Rel | Sq Rel | RMSE | RMSE *log* |
|---|---|---|---|---|
| | Lower is better | | | |
| mono* [6] | 0.443 | 7.112 | 8.860 | 0.142 |
| DDVO∗ [29] | 0.387 | 4.720 | 8.090 | 0.204 |
| mono2∗ [31] | 0.322 | 3.589 | 7.417 | 0.163 |
| SC-GAN[32] | 0.206 | 3.550 | 5.478 | 0.801 |
| DORN [17] | 0.197 | 3.461 | 5.671 | 0.746 |
| DPT-Hybrid[9] | 0.182 | 3.021 | 5.773 | 0.766 |
| AdaBins[7] | 0.158 | 2.874 | 5.355 | 0.702 |
| SingleNet∗[33] | 0.146 | 2.287 | 5.536 | 0.728 |
| You[34] | 0.120 | 2.258 | 6.471 | 0.248 |
| ManyDepth∗[20] | 0.119 | 1.857 | 7.633 | 0.195 |
| EdgeConv[35] | 0.125 | 1.987 | 5.105 | 0.077 |
| Ours | **0.101** | **1.736** | **4.337** | **0.062** |

TABLE II: **Quantitative Results - Make3D [37]:**Best results for each metric are in **bold**; second best are underlined.

| Method | Abs Rel | Sq Rel | RMSE | RMSE *log* |
|---|---|---|---|---|
| | Lower is better | | | |
| mono* [6] | 0.476 | 7.155 | 9.368 | 0.142 |
| DDVO∗ [29] | 0.405 | 5.271 | 8.815 | 0.413 |
| mono2∗ [31] | 0.352 | 3.905 | 7.141 | 0.196 |
| SC-GAN[32] | 0.401 | 1.875 | 5.367 | 0.251 |
| DORN [17] | 0.374 | 3.792 | 8.238 | 0.201 |
| DPT-Hybrid[9] | 0.110 | 0.335 | 0.357 | 0.045 |
| AdaBins[7] | 0.103 | 0.312 | 0.364 | 0.044 |
| SingleNet∗[33] | 0.138 | 0.382 | 0.475 | 0.058 |
| You[34] | 0.110 | 0.301 | 0.392 | 0.047 |
| ManyDepth[20] | 0.117 | 1.182 | 6.031 | 0.179 |
| EdgeConv[35] | 0.107 | 0.298 | 0.373 | 0.046 |
| Ours | **0.075** | **0.105** | **0.194** | **0.028** |

TABLE III: **Quantitative Results - NYU Depth V2 [38]:**Best results for each metric are in **bold**; second best are underlined.

*2) Make3D Dataset Details [37]:* We use our model pre-trained on the KITTI dataset and fine-tune the parameters before testing it on all 534 images in the Make3D dataset. As the size of the ground truth $55 \times 305$ and the size of the input image $1704 \times 2272$ are not aligned, we center crop the input image with a ratio of $1 \times 2$ and then resize it to $336 \times 960$. The ground truth image is also center cropped with the same ratio.

*3) NYU Depth V2 [38]:* . This is a dataset that provides images and depth maps for different indoor scenes captured at a pixel resolution of $640 \times 480$. The dataset contains 120K training samples and 654 testing samples. We train our network on a 10K subset. The upper bound of depth maps is 10 meters. The resolution of our output is $320 \times 240$. Then we upsample it to $640 \times 480$ to match the ground truth resolution during both training and testing.

### *B. Results on KITTI dataset [11]*

Table I shows the quantitative results for our experimental analysis on the KITTI test split by Eigen [27]. As evident from the results, our model outperforms other methods in all evaluation metrics.

Figure 3 shows the qualitative results for estimating depth on the images from the KITTI test set. As visible from the depth maps, the results obtained using our approach have depth values which can clearly delineate the objects and do not have edge-blurring effect as in the other approaches.

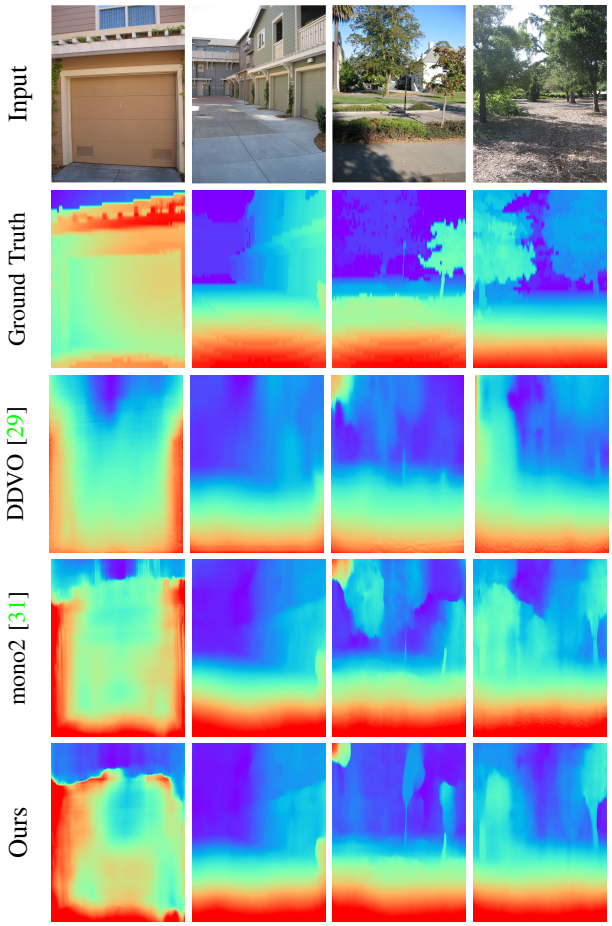


Fig. 4: **Qualitative Results - Make3D [37]**

### C. Results on Make3D dataset [37]

Quantitative results obtained on applying our trained model on the images from the Make3D dataset are reported using Abs Rel, Sq Rel, RMSE, and RMSE *log* as the metrics. As shown in Table II, our model performs better than all other approaches.

Figure 4 shows the qualitative depth estimation results on four images from the Make3D dataset. Visual comparison demonstrates that our results are closer to the ground truth as compared to those of the other approaches.

### D. Results on NYU Depth V2 [38]

As shown in Table III, our model performs better than all other approaches. It gives best values for the Abs Rel metric, Sq Rel metric, and second to the best values for the remaining two metrics RMSE, and RMSE *log*. The results on NYU Depth V2 shows our model not only can work in an outdoor environment but also in an indoor scenario.

### E. Ablation Study on KITTI dataset

We conduct an ablation study on the KITTI dataset in order to better understand the importance of the two proposed modules - deformable support window and pixel movement prediction. Table IV shows the results of the different variants of our model on the KITTI dataset using the Eigen split. As seen in the results, the baseline model with just our neural network architecture but none of the two proposed modules, performs the worst. Whereas the model with all our proposed contributions performs the best with significant improvements. The variant with the pixel movement prediction module works the second best. This proposed 3-part pixel movement prediction along with the pixel movement triangle constraint loss is the primary contributor towards successful monocular depth estimation. The deformable support window module also contributes to improving the quantitative results. However, this module's importance is realized more from a qualitative standpoint of obtained non-blurred edges.

## V. Conclusion

In this paper, we propose a novel deep neural network called PMPNet for monocular depth estimation in dynamic scenes. The model performs individual pixel movement prediction so as to lift the static scene assumption while simultaneously ensuring geometric consistency during depth estimation. We assume that objects move along a straight line and use this to predict two possible movements for each pixel and one straight line to constrain them. A novel pixel movement triangle constraint loss is proposed for restricting the relationship between the pixel movement predictions and the straight line, thereby overcoming the inconsistency between the pixel movement and the depth map. The proposed model also consists of a support window module consisting of deformable convolutions which overcomes the incorrect depth estimation problem around object edges. Experiments were conducted on two outdoor datasets - KITTI and Make3D, and one indoor dataset - NYU Depth V2. Results of these, alongside the ablation study results on the KITTI dataset, demonstrate the effectiveness of our proposed PMPNet for monocular depth estimation in dynamic scenes.

## References

[1] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nešić, X. Wang, and P. Westling, "High-resolution stereo datasets with subpixel-accurate ground truth," in *German conference on pattern recognition*. Springer, 2014, pp. 31–42. 1

| Deformable Support Window | Pixel Movement Prediction | Abs Rel | Sq Rel | RMSE | RMSE *log* | $\delta < 1.25$ | $\delta < 1.25^2$ | $\delta < 1.25^3$ |
|---|---|---|---|---|---|---|---|---|
| | | Lower is better | | | | Higher is better | | |
| | | 0.179 | 0.254 | 3.578 | 0.131 | 0.527 | 0.801 | 0.753 |
| ✓ | | 0.150 | 0.187 | 3.221 | 0.125 | 0.624 | 0.828 | 0.826 |
| | ✓ | <u>0.064</u> | <u>0.178</u> | <u>2.356</u> | <u>0.104</u> | <u>0.878</u> | <u>0.861</u> | <u>0.865</u> |
| ✓ | ✓ | **0.049** | **0.157** | **1.831** | **0.071** | **0.994** | **0.996** | **0.999** |

TABLE IV: **KITTI [11] Ablation Study:** Best results for each metric are in **bold**; second best are <u>underlined</u>.

[2] M. Klingner, J.-A. Termöhlen, J. Mikolajczyk, and T. Fingscheidt, "Self-Supervised Monocular Depth Estimation: Solving the Dynamic Object Problem by Semantic Guidance," in *European Conference on Computer Vision (ECCV)*, 2020. 1

[3] H. Li, A. Gordon, H. Zhao, V. Casser, and A. Angelova, "Unsupervised monocular depth learning in dynamic scenes," *arXiv preprint arXiv:2010.16404*, 2020. 1, 2

[4] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1851–1858. 1, 5

[5] 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. 1, 4

[6] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 270–279. 1, 5

[7] S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins: Depth estimation using adaptive bins," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4009–4018. 1, 2, 4, 5

[8] M. Song, S. Lim, and W. Kim, "Monocular depth estimation using laplacian pyramid-based depth residuals," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021. 1

[9] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 179–12 188. 1, 2, 4, 5

[10] L. Wang, J. Zhang, O. Wang, Z. Lin, and H. Lu, "Sdc-depth: Semantic divide-and-conquer network for monocular depth estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 541–550. 1

[11] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 3354–3361. 1, 2, 4, 5, 6

[12] Y. Di, H. Morimitsu, S. Gao, and X. Ji, "Monocular piecewise depth estimation in dynamic scenes by exploiting superpixel relations," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4363–4372. 1, 2

[13] J. S. Yoon, K. Kim, O. Gallo, H. S. Park, and J. Kautz, "Novel view synthesis of dynamic scenes with globally coherent depths from a monocular camera," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5336–5345. 1

[14] S. Avidan and A. Shashua, "Trajectory triangulation: 3d reconstruction of moving points from a monocular image sequence," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 4, pp. 348–357, 2000. 1, 3

[15] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 764–773. 2, 3

[16] A. Ranjan, V. Jampani, L. Balles, K. Kim, D. Sun, J. Wulff, and M. J. Black, "Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 240–12 249. 2, 3

[17] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2002–2011. 2, 4, 5

[18] Z. Yin and J. Shi, "Geonet: Unsupervised learning of dense depth, optical flow and camera pose," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1983–1992. 2, 3

[19] C. Shu, K. Yu, Z. Duan, and K. Yang, "Feature-metric loss for self-supervised learning of depth and egomotion," in *European Conference on Computer Vision*. Springer, 2020, pp. 572–588. 2, 4

[20] J. Watson, O. Mac Aodha, V. Prisacariu, G. Brostow, and M. Firman, "The temporal opportunist: Self-supervised multi-frame monocular depth," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1164–1174. 2, 4, 5

[21] V. Casser, S. Pirk, R. Mahjourian, and A. Angelova, "Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 8001–8008. 2

[22] B. Bozorgtabar, M. S. Rad, D. Mahapatra, and J.-P. Thiran, "Syndemo: Synergistic deep feature alignment for joint learning of depth and ego-motion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 4210–4219. 2

[23] R. Ranftl, V. Vineet, Q. Chen, and V. Koltun, "Dense monocular depth estimation in complex dynamic scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4058–4066. 2

[24] H. Xu and J. Zhang, "Aanet: Adaptive aggregation network for efficient stereo matching," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1959–1968. 2, 3, 4

[25] M. Bleyer, C. Rhemann, and C. Rother, "Patchmatch stereo-stereo matching with slanted support windows." in *Bmvc*, vol. 11, 2011, pp. 1–11. 3

[26] V. Tankovich, C. Häne, S. Fanello, Y. Zhang, S. Izadi, and S. Bouaziz, "Hitnet: Hierarchical iterative tile refinement network for real-time stereo matching," *arXiv preprint arXiv:2007.12140*, 2020. 3

[27] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *arXiv preprint arXiv:1406.2283*, 2014. 4, 5

[28] W. Yin, Y. Liu, C. Shen, and Y. Yan, "Enforcing geometric constraints of virtual normal for depth prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5684–5693. 4

[29] C. Wang, J. M. Buenaposada, R. Zhu, and S. Lucey, "Learning depth from monocular videos using direct methods," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2022–2030. 4, 5, 6

[30] C. Luo, Z. Yang, P. Wang, Y. Wang, W. Xu, R. Nevatia, and A. Yuille, "Every pixel counts++: Joint learning of geometry and motion with 3d holistic understanding," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 10, pp. 2624–2641, 2019. 4

[31] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3828–3838. 4, 5, 6

[32] Z. Wu, X. Wu, X. Zhang, S. Wang, and L. Ju, "Spatial correspondence with generative adversarial network: Learning depth from monocular videos," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7494–7504. 4, 5

[33] Z. Chen, X. Ye, W. Yang, Z. Xu, X. Tan, Z. Zou, E. Ding, X. Zhang, and L. Huang, "Revealing the reciprocal relations between self-supervised stereo and monocular depth estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 529–15 538. 4, 5

[34] Z. You, Y.-H. Tsai, W.-C. Chiu, and G. Li, "Towards interpretable deep networks for monocular depth estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 879–12 888. 4, 5

[35] M. Lee, S. Hwang, C. Park, and S. Lee, "Edgeconv with attention module for monocular depth estimation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 2858–2867. 4, 5

[36] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry, "End-to-end learning of geometry and context for deep stereo regression," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 66–75. 4

[37] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: Learning 3d scene structure from a single still image," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 5, pp. 824–840, 2008. 4, 5, 6

[38] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *European conference on computer vision*. Springer, 2012, pp. 746–760. 4, 5, 6

[39] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *arXiv preprint arXiv:1912.01703*, 2019. 4

[40] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 4