How Far Can I Go ? : A Self-Supervised Approach for Deterministic Video Depth Forecasting

Sauradip Nag *
University of Surrey, UK
s.nag@surrey.ac.uk

Nisarg A. Shah *
IIT Jodhpur
shah.2@iitj.ac.in

Anran Qi *
University of Surrey, UK
a.qi@surrey.ac.uk

Raghavendra Ramachandra
NTNU, Norway
raghavendra.ramachandra@ntnu.no

Abstract

In this paper we present a novel self-supervised method to anticipate the depth estimate for a future, unobserved real-world urban scene. This work is the first to explore self-supervised learning for estimation of monocular depth of future unobserved frames of a video. Existing works rely on a large number of annotated samples to generate the probabilistic prediction of depth for unseen frames. However, this makes it unrealistic due to its requirement for large amount of annotated depth samples of video. In addition, the probabilistic nature of the case, where one past can have multiple future outcomes often leads to incorrect depth estimates. Unlike previous methods, we model the depth estimation of the unobserved frame as a view-synthesis problem, which treats the depth estimate of the unseen video frame as an auxiliary task while synthesizing back the views using learned pose. This approach is not only cost effective - we do not use any ground truth depth for training (hence practical) but also deterministic (a sequence of past frames map to an immediate future). To address this task we first develop a novel depth forecasting network *DeFNet* which estimates depth of unobserved future by forecasting latent features. Second, we develop a channel-attention based pose estimation network that estimates the pose of the unobserved frame. Using this learned pose, estimated depth map is reconstructed back into the image domain, thus forming a self-supervised solution. Our proposed approach shows significant improvements of $\sim 5\%/8\%$ in Abs Rel metric compared to state-of-the-art alternatives on both short and mid-term forecasting setting, benchmarked on KITTI and Cityscapes.

1 Introduction

Monocular video depth estimation, which requires to estimate the depth of all object instances appearing in given frames of a video that is captured using a single camera, has drawn more and more attention in recent past. Most of the existing approaches are developed for after-the-fact estimation, where the depth of images/frames which are to be estimated are accessible to the system (refer to Fig 1(a)). However, it is often required in many practical cases that the system can predict future depth

*All authors contributed equally

Machine Learning for Autonomous Driving Workshop at the 35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

estimation before the corresponding images/frames are observed (refer to Fig 1(b)). Future depth estimation (depth forecasting) is thus much more important than after-the-fact depth estimation in real-world applications like media interpretation, autonomous driving, etc. For example, as illustrated in fig 1(e) due to the sudden right turn of the vehicle, a collison can be avoided if the depth of the car and free space around it is known to the driving system beforehand to perform an escape maneuver.

Video Depth forecasting is a challenging problem mostly due to uncertainty in appearance caused by object movement, occlusion and viewpoint. Qi et al. [36] was the first to predict depth estimates for future time instants by reasoning over optical-flow (2D/3D), but assume access to additional information - depth images and semantic maps for past frames. More specifically, [36] designed a complicated two-stage (prediction then refinement) architecture to jointly predict future motion, RGB frames, depth maps, and semantic maps. Very recently, Hu et al. [17] adopted a conditional variational approach to model the inherent stochasticity of future prediction. The architecture learns rich representations that incorporate both local and global spatio-temporal context which are then used to forecast segmentation , depth and flow. While these approaches have attempted to reason the future using depth forecasting, both of them have some shortcomings : (a) both the approaches [36, 17] require labeled data like semantic, depth maps, motion estimates of the past/all frames during training/pre-training stage hence not generalizable to any video; (b) although [36] handles rigid scene assumptions for a very short-term (next frame), both [36, 17] fails to handle the scene consistency beyond that - resulting in artefacts of moving object.

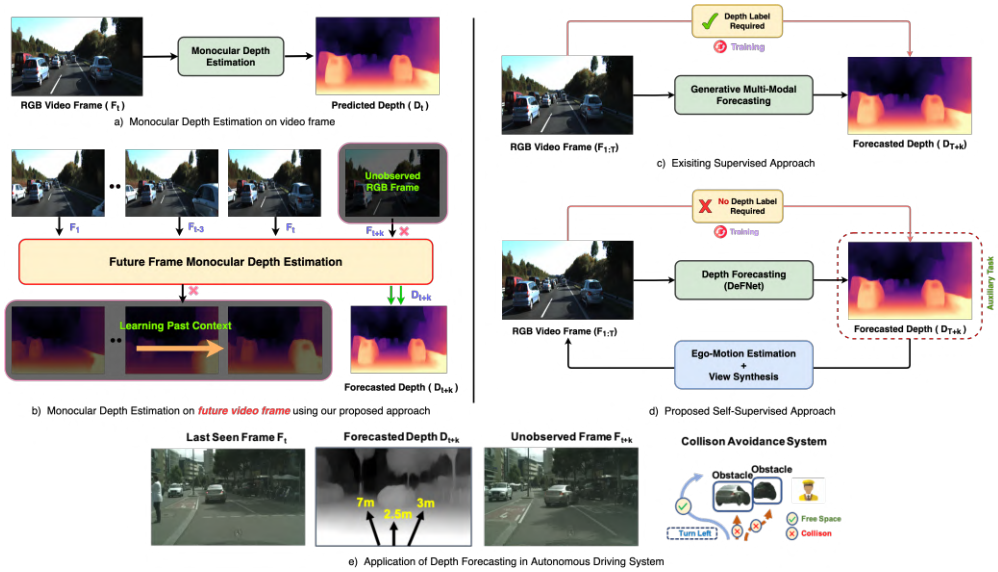

Figure 1: **Overview of the problem.** Illustration of the standard problem of depth forecasting with existing baselines [17] and its practical application in autonomous driving solution.

In recent years, self-supervised methods have attracted increasing interests in monocular depth estimation [14, 13, 31, 52] tasks thus making it a suitable alternative to supervised approaches [6, 10, 28]. In the absence of ground truth, one can still recover scene depth and ego-motion from monocular video sequences using novel view synthesis [13, 11] (by jointly optimizing the depth and pose network). However, all of these self-supervised approaches are designed for after-the-fact depth estimation (fig 1(a)). Taking motivation from this, we propose a new self-supervised depth forecasting setting which is achieved by designing a novel Depth Forecasting Network and a new pipeline for Pose Estimation conditioned on the unseen future. The Depth Forecasting Network (DeFNet) consists of a novel ConvGRU based feature forecasting block and a convLSTM based flow forecasting block. The forecasting blocks take aggregated features till time step $\{t\}$ and forecast a feature at time step $\{t+k\}$ which is then post-processed to form forecasted depth. Additionally, we also designed a new channel-attention based Pose Network for the forecasting task that enforces the network to leverage the pose w.r.t the unseen target frame at time $\{t+k\}$. As a result of these changes, we modify the existing self-supervised after-the-fact setting to propose a new *self-supervised forecasting setting*. Thus, we a) solve depth forecasting as an auxiliary task thereby eliminated the need of expensive

depth labels hence making our approach generalizable to any form of video scene and b) handle the rigid scene-assumption for both short and mid-term forecast (by varying k in $\{t+k\}$).

Contributions In summary, our contributions are three-folds : **(1)** We proposed a novel multi-scale feature-level forecasting network (DeFNet) for depth forecasting **(2)** We also designed a new pose estimation network using Channel attention (PCAB) and adapted it for the forecasting task. **(3)** We formulated a new self-supervised setting for depth forecasting task that eliminates the need of depth annotation making it generalizable and scalable. Extensive experiments show that the proposed method yields new state-of-the-art performance on two benchmark datasets (KITTI and Cityscapes).

2 Related Works

Self-Supervised Depth Estimation: A more promising substitute for supervised depth models [4, 25, 37, 46, 2, 51, 28, 6] is the self-supervised approach. A less constrained form of self-supervision is to use monocular videos, where consecutive temporal frames provide the training signal. In one of the first monocular self-supervised approaches, [52] trains a depth estimation network along with a separate pose network. Inspired by [3], [45] proposes a more sophisticated motion model using multiple motion masks. [50] also decomposes motion into rigid and non-rigid components, using depth and optical flow to explain object motion. In the context of optical flow estimation, [22] shows that it helps to explicitly model occlusion. Additional data such as pre-computed instance segmentation masks were used by [4] to help deal with moving objects. In another work, [13] shows that the inclusion of a local structure based appearance loss [47] significantly improves depth estimation performance compared to simple pairwise pixel differences [48, 11, 52]. Self-supervision was achieved by [14] by jointly training a depth and pose network using novel view synthesis. Nevertheless, all of these existing self-supervised approaches are formulated for after-the-fact setting depth estimation. In this paper, we for the first time develop a self-supervised monocular depth solution for our new forecasting setting.

Video Forecasting: These approaches are useful in generating a future frame by learning past frame correspondence. Ranzato et al. [38] introduces the first baseline in this domain by presenting a generative approach for the future frame prediction. Few works followed this [33, 32, 24], which improves the learning of past context using Long Short Term Memory (LSTM). Different from predicting frames, Jin et al. [23] directly predicts the semantic label of the future frame i.e semantic forecasting by observing the past frames. Several works [34, 30] thereby followed that fused optical flow to model semantic forecast for longer durations. Along similar motivation to our work, semantic forecasting was addressed by forecasting intermediate latent representations of past observed frames using CNN [29, 39] and ConvLSTMs [42]. Depth forecasting was first addressed by [36] as a part of future frame synthesis that can only predict the depth of next time instant. Recently, Hu et al. [17] addressed this task using generative modelling of a probabilistic future. However, both of these approaches require expensive depth annotation during training or pre-training stage to generate the depth forecast. Our approach is thus the first work that attempts to solve the depth forecasting task in a completely self-supervised fashion without any label requirement.

3 Proposed Methodology

We introduce the novel Depth Forecasting Network and describe overall design in Sec 3.1. In Sec 3.2, we describe our new pipeline for Pose Estimation and discuss about adapting it for the new forecasting setting, achieving self-supervision using novel-view synthesis. Finally, we discuss the learning objective as well as the inference process in Sec 3.3. The overall architecture is illustrated in Fig 2.

Problem Formulation: Depth forecasting is defined as the task when given a video $\mathbf{V}$ with *t* frames $\{I_1, I_2, ..., I_t\} \in V$ and the corresponding optical flow $\mathbf{M} \in \{O_1, O_2, ..., O_t\}$, we anticipate the depth $\mathbf{D}_{t+k}$ at a fixed number of timesteps *k* from the last observed frame at time t. This anticipated depth corresponds to the depth of an unobserved future frame I_{t+k}. In this paper, we will discuss models for both short-term $(k = 5)$ and mid-term $(k = 10)$ prediction. For training, we sample 4 frames from V and M with frame interval 3 per batch : $V_{1:t} = \{I_{t-9}, I_{t-6}, I_{t-3}, I_t\}$; $M_{1:t} = \{O_{t-9}, O_{t-6}, O_{t-3}, O_t\}$ and feed it to our Depth Forecasting Network (DeFNet). Additionally, we pass $\{I_t, I_{t+k}, I_{t+2k}\}$ as input to the Pose Estimation network.

3.1 Depth Forecasting Network

Depth forecasting network primarily consists of (a) Feature encoder - to encode pyramidal features , (b) Forecasting module - to forecast the features, (c) Decoder module - to decode forecasted depth from the features and (d) Fusion block - which refines the depth by fusing flow forecast.

Feature Encoder: Our approach begins with a encoder each for RGB and Flow input that extracts the multi-scale pyramidal features. In general any encoder backbone can be used, but due to performance gain we chose a pre-trained ResNext [27] as RGB Encoder backbone and a pretrained LiteFlowNet2.0 [19] as Flow Encoder backbone. Formally, given a video $V_{1:t}$ and optical flow $M_{1:t}$ having frames of size $W \times H$, the encoder extracts $\{F_{rgb}^{t-9}, F_{rgb}^{t-6}, .., F_{rgb}^{t}\}$ for rgb and $\{F_{flow}^{t-9}, F_{flow}^{t-6}, .., F_{flow}^{t}\}$ for flow image where F^t indicates the pyramid features $\{P_t^i\}_{i=1}^{L}$ (L scales in total) extracted from t-th frame. P_t^l is the feature of the l-th pyramid level at t-th frame having dimension $[\frac{W}{2^l}, \frac{H}{2^l}]$. The resultant rgb and flow features are then aggregated along the temporal dimension to obtain $\mathbf{F}_{rgb}^{1:t}$ and $\mathbf{F}_{flow}^{1:t}$ as follows :

$$\mathbf{F}_{rgb}^{1:t} = F_{rgb}^{t-9} \oplus F_{rgb}^{t-6} \oplus .. \oplus F_{rgb}^{t}, \qquad \mathbf{F}_{flow}^{1:t} = F_{flow}^{t-9} \oplus F_{flow}^{t-6} \oplus .. \oplus F_{flow}^{t} \tag{1}$$

where $\oplus$ indicates channel-wise concatenation of features. The resultant multi-scale feature maps ($\mathbf{F}_{rgb}^{1:t}$ and $\mathbf{F}_{flow}^{1:t}$) are simultaneously fed into the feature forecasting module to predict the features for time-step t+k at multiple-scales.

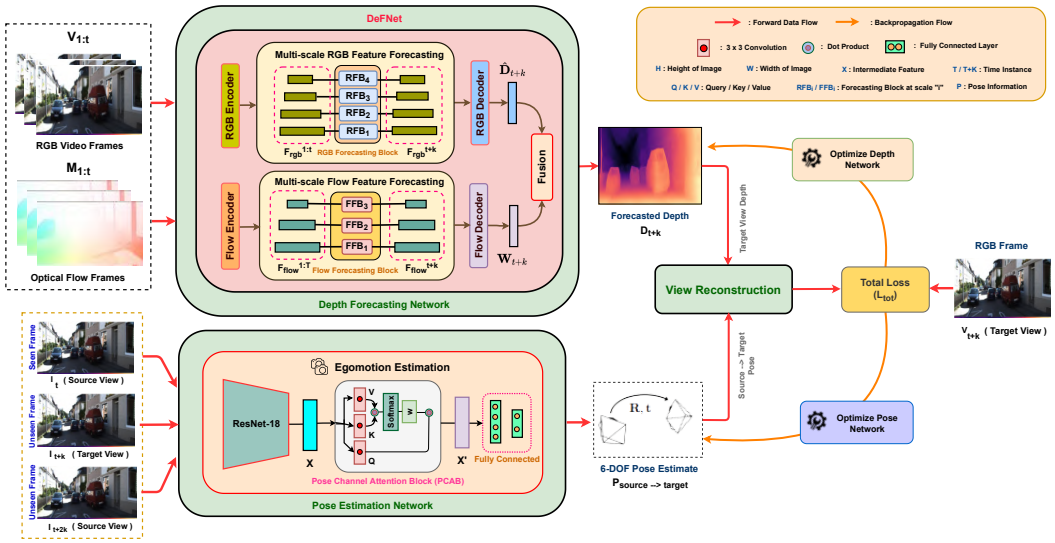

Figure 2: **Overview of our proposed self-supervised Depth Forecasting approach**. Our novel Depth Forecasting network (DeFNet) takes in rgb and flow frames and forecasts depth D_{t+k}. The Pose Estimation Network takes in source and target input frames to produce pose estimate P conditioned on target frame unobserved by the depth network. The View reconstruction module reconstructs back the target view using depth D_{t+k} and pose P. The reconstruction cost obtained from the target view and reconstructed target view acts as the supervision signal to train both the depth and pose networks.

Feature Forecasting Module: Our feature forecasting module receives processed input rgb and flow features ($\mathbf{F}_{rgb}^{1:t}/\mathbf{F}_{flow}^{1:t}$) and directly regresses the future features $\mathbf{F}_{rgb}^{t+k}, \mathbf{F}_{flow}^{t+k}$ respectively. More specifically, the forecasting module learns a mapping ϕ between the pyramid features extracted from past video frames and future frames. This mapping distinguishes our forecasting setting from the standard after-the-fact setting by predicting features for unobserved frames. The features predicted for the unobserved frame I_{t+k} is formulated as :

$$\mathbf{F}_s^{t+k} = \phi_s(\mathbf{F}_s^{1:t}) \quad where \quad s \in \{rgb, flow\} \tag{2}$$

The rgb forecasting function ϕ_{rgb} consists of a feature forecasting block (termed as RFB) at each pyramidal level as illustrated in Fig 3. Each RFB block comprises of a series of convGRUs (cGRU) [1] to model spatio-temporal relation (intra-level) among the features of the same pyramid level. In addition, our method also introduces connections (inter-level) between different cGRUs (in different RFBs) in order to capture the spatio-temporal context across different levels. The inter-connected

cGRUs form our model (i.e., the mapping ϕ_{rgb} in formula (2)) for predicting future rgb features. More details on intra-level and inter-level cGRU is provided in appendix. Similar to rgb forecasting, the flow forecasting function ϕ_{flow} consists of a flow feature forecasting block (termed as FFB) at each pyramidal level. Each FFB block consists of a single convLSTM [44] and a 1-D convolution. For simplicity and ease of training, we do not consider inter or intra-level connections among different FFBs at each level. The extracted flow features $\mathbf{F}_{flow}^{1:t}$ are passed as input to the FFB with a 3×3 kernel and 1×1 padding. It then produces the flow forecasted feature $\mathbf{F}_{flow}^{t+k}$ using a single 1×1 convolution layer after the ConvLSTM to reduce the channels for the flow field.

Feature Decoder: The input of our feature decoder is the predictive features ($\mathbf{F}_{rgb}^{t+k}/\mathbf{F}_{flow}^{t+k}$) forecasted by our feature-forecasting module. As illustrated in fig 3, the decoders progressively predicts depth and flow features at each pyramidal level in a coarse-to-fine manner. The rgb decoder first predicts the low resolution depth map $\hat{\mathbf{D}}_{t+k}^{L}$ at the top-level with size $\frac{W}{2^L} \times \frac{H}{2^L}$ as the initial scene layout using a convolution operation. As illustrated in fig 3(b), the pyramidal feature $\mathbf{P}_{t+k}^{i}$ and the depth map $\hat{\mathbf{D}}_{t+k}^{i+1}$ predicted at the upper level are integrated together to produce a refined depth map $\hat{\mathbf{D}}_{t+k}^{i}$ in the current scale. The output depth representation $\hat{\mathbf{D}}_{t+k}$ thereby contains the refined depth from multiple scales that carry both high-level scene information and low-level detail information. Similarly, for the flow decoder, we follow the cascaded flow refinement design of [19] at each pyramid level. Given the forecasted features $\mathbf{F}_{flow}^{t+k}$, it obtains refined flow $\mathbf{W}_{t+k}$ using a 2-stage refinement strategy. First, the pixel by-pixel matching of high-level feature vectors yields a coarse flow estimate. This is achieved by upsampling the previous pyramid level flow $\mathbf{W}_{t+k}^{i+1}$ and warping it on to the current level flow feature P_{t+k}^{i}. Second, a subsequent refinement on the coarse flow further improves it to a refined flow. A correlation between the warped coarse flow and the current level feature P_{t+k}^{i} results in a refined $\mathbf{W}_{t+k}^{i}$. The final decoded depth $\hat{\mathbf{D}}_{t+k}$ and flow $\mathbf{W}_{t+k}$ is obtained from last (bottom) pyramidal layer output.

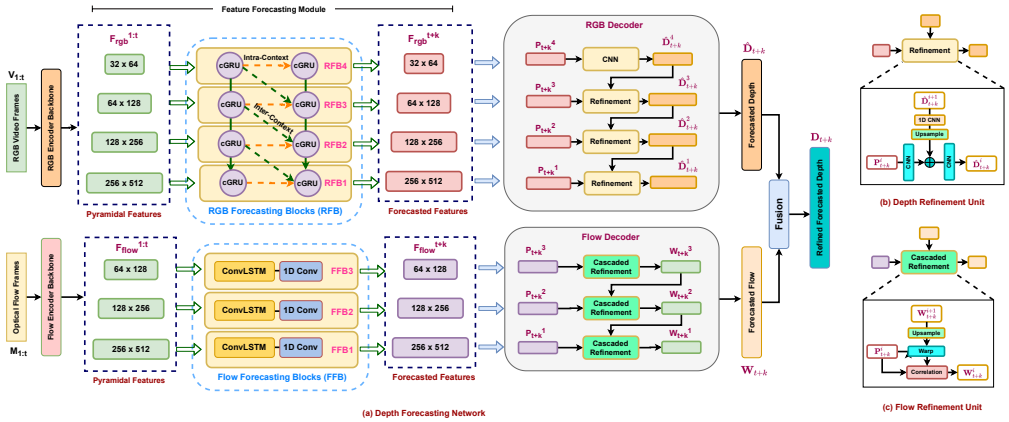

Figure 3: **Illustration of our novel Depth Forecasting Network (DeFNet)**It consists of a encoder , a forecasting and decoder module. The encoded pyramidal features are forecasted using the feature forecasting blocks and then decoded in a coarse-to-fine manner. The refined forecasted depth is obtained by fusing both rgb and flow forecast. The refinement units of depth and flow decoders are illustrated in (b) and (c) respectively.

Fusion: Recall, that the forecasted depth estimate $\hat{\mathbf{D}}_{t+k}$ already carries structural geometries and the forecasted flow information $\mathbf{W}_{t+k}$ carries the object motions for the unobserved scene. However, the occluded objects (not observed in the past frames) tend to lose its structure in the forecasted depth $\hat{\mathbf{D}}_{t+k}$ and thus can suffer from incomplete depth estimates and self-occlusions. Since flow information can still predict the motion of such occluded pixels , fusing both forecasted depth and flow helps our network to make meaningful assumptions of the future scene by considering observed motion of depth feature activations. The final forecasted depth $\mathbf{D}_{t+k}$ is obtained by using a single learnable convolution [16] layer Θ as follows :

$$\mathbf{D}_{t+k} = \sigma(\Theta(\left\| \hat{\mathbf{D}}_{t+k} \right\|, \|\mathbf{W}_{t+k}\|)) \tag{3}$$

where σ is sigmoid operation. It is to be noted that both $\hat{\mathbf{D}}_{t+k}, \mathbf{W}_{t+k}$ are normalized before passing into Θ. For non-translational motion, Θ can compensate for some geometric distortions by learning some depth for the occluded pixels thus producing a refined forecast.

3.2 View Reconstruction using Future Conditioned Ego-Motion

The key supervision signal for our network comes from the task of *novel view synthesis*: given one input view of a scene, synthesize a new image of the scene seen from a different camera pose. We can synthesize a target view given a per-pixel depth in that image, plus the pose and visibility in a nearby view. As we will show next, this view-synthesis approach is re-designed for our forecasting setting to achieve self-supervision.

Ego-Motion: The ego-motion network shown in the bottom half of fig. 2 estimates relative pose $P \in \mathbb{SE}(3)$ introduced by motion fields across frames. Specifically, it learns a function Φ which is a CNN for predicting the camera motion of input frames. For instance, given a sequence of images (I_{t-1}, I_t, I_{t+1}) with target frame I_t and temporally neighboring reference frames I_{t-1}, I_{t+1}, Φ predicts a vector $\mathbf{P}_{t-1\to t}$ and $\mathbf{P}_{t+1\to t}$ comprising of translation (t_x, t_y, t_z) and rotation (ϱ, θ, ψ) parameters of the camera between the frames. Mathematically, it is denoted as

$$\mathbf{P}_{so\to ta} = \Phi(I_{so}, I_{ta}) \quad where \quad I_{so} \in \{I_t, I_{t+1}\}, I_{ta} \in \{I_t\} \tag{4}$$

where I_{so} denotes source images and I_{ta} denotes target images. This formulation is commonly used in after-the-fact setting for pose estimation where pose is calculated for the frames seen by the depth network. In order to adapt this for our forecasting setting we used an interesting trick : instead of passing a sequence of consecutive images (I_{t-1}, I_t, I_{t+1}) , we input (I_t, I_{t+k}, I_{t+2k}) (i.e $I_{so} \in \{I_t, I_{t+2k}\}$ and $I_{ta} \in \{I_{t+k}\}$ in formula 4). It is interesting to note that the frames I_{t+k}, I_{t+2k} are unobserved video frames for the depth network. We argue that by passing these unobserved frames, the Pose-Network Φ predicts $\mathbf{P}_{t\to t+k}$ and $\mathbf{P}_{t+2k\to t+k}$ which enforces the network to learn the camera translation (t_x, t_y, t_z) and rotation (ϱ, θ, ψ) parameters conditioned on the target frame I_{t+k}. This leverages a pose information that captures both seen pose context ($\mathbf{P}_{t\to t+k}$) and also unseen pose context ($\mathbf{P}_{t+2k\to t+k}$) while making an estimate.

Network Design: The pose-network Φ consists of a ResNet-18 [15] encoder , a channel attention block (referred as PCAB) followed by 2 fully connected layers to predict $6 - DOF$ output. Given a sequence of images $\{I_t, I_{t+k}, I_{t+2k}\}$ the encoder takes inputs in pairs and outputs a feature $X \in \mathbb{R}^{7\times7\times512}$. Since we are not passing consecutive frames ($k \geqslant 1$ in I_{t+k}, I_{t+2k}) through the network it is prone to uncertainties introduced by moving objects, occlusions. Such error prone regions between the target and source image feature is highlighed using our channel attention PCAB. The PCAB primarily consists of 3 convolution layers having kernel size 3 to obtain Q/K/V which is used to obtain the channel weight vector ω. The weighted pose feature $X^{'}$ is obtained by multiplying weight vector ω with the input feature X. More details on PCAB is provided in appendix. The weighted feature $X^{'}$ is then passed into the fully connected layers to obtain 6-DOF pose $\mathbf{P}$.

View Reconstruction: Given a target and source image pair (I_{ta}, I_{so}) with known transformation ($\mathbf{P}_{so\to ta}$) between the images and the forecasted depth map ($\mathbf{D}_{ta}$), the target image can be reconstructed by sampling pixels from the source image through image inverse warping. Let j_{ta} denotes the 2D homogeneous coordinate of an pixel in frame I_{ta} and K denotes the intrinsic camera matrix. We can compute the corresponding point of j_{ta} (denoted as j_{so}) in frame I_{so} using the following equation:

$$j_{so} \sim K P_{so\to ta} D_{ta}(j_{ta}) K^{-1} j_{ta} \tag{5}$$

The reconstructed target image $I_{so\to ta}$ is obtained by populating the value of $I_{ta}(j_{so})$ using differentiable inverse warping [21]. This reconstruction is feasible for monocular videos which is mainly based on the assumptions that the scene is static without moving object, the vision difference is caused by the camera pose change and no new object appears into the view between the target view and the source views. But this is hard to satisfy for all training sequences collected in real world. Inspired by [14, 6], we use a auto-mask (denoted by A) that filters out such pixels which do not change appearance from one frame to the next in the sequence. It is described as follows :

$$A_{so\to ta} = \min_{so} L_{pe}(I_{ta}, I_{so\to ta}) < \min_{so} L_{pe}(I_{ta}, I_{so}) \tag{6}$$

where L_{pe} denotes photometric reprojection loss (described in Section 3.3). This mask $\mathbf{A}$ forces the pose-network to ignore objects which move at the same velocity as the camera, and even to ignore whole frames in monocular videos when the camera stops moving.

3.3 Learning Objective and Inference :

We train our network (both depth and pose) using the reconstruction error between the reconstructed target $I_{so\rightarrow ta}$ and original target I_{ta}. The loss components are defined as follows :
(a) **Masked Photometric(L_{mpe})** : Photometric loss is a standard L1 loss calculated between the reconstructed view $I_{so\rightarrow ta}$ and target view I_{ta} denoted as L_{pe}. We formulate the mask photometric loss L_{mpe} by performing the dot product with mask L_{pe} as follows:

$$L_{mpe} = \sum_{s}(A_s.L_{pe}), s \in (so \rightarrow ta, ta \rightarrow so) \tag{7}$$

(b) **Dissimilarity (L_{ds}):** It calculates the dissimilarity between source and target views by also being differentiable. It is denoted by

$$L_{ds} = 0.5 * (1 - SSIM(I_{so\rightarrow ta}, I_{ta})) \tag{8}$$

(c) **Smoothness (L_{sm}):** It enforces DeFNet to produce sharp edge distribution while producing smoother depth. It is denoted by L_{sm} and has similar formulation as in [11].
(d) **Pose Consistency (L_{pc}):**It a simple L1 loss that ensures the 'past' and 'future' inter-frame translations are consistent with each other. It is represented as

$$L_{pc} = \|P_{t,t+k} - P_{t+2k,t+k}\|_1 \tag{9}$$

. We combine all the losses to form our final training objective as follows : $L_{tot} = \alpha.L_{mpe} + (1-\alpha).L_{ds} + \lambda.L_{sm} + \gamma.L_{pc}$ where α, λ, γ are hyperparameters. Following standard practise in self-supervised after-the-fact setting [14], both the DeFNet and Pose networks are jointly optimized using this loss. More details on loss are given in the appendix.

During Inference, we discard the pose estimation network similar to after-the-fact setting [14] and only use the depth forecasting network DeFNet to estimate the forecasted depth of testing videos.

4 Experiments

Dataset: We use the **KITTI** [12] and **Cityscapes** [7] datasets to evaluate our method for outdoor scenes. For KITTI, we adopt the training protocols used in Eigen et al. [9], and specifically, we use the KITTI Eigen splits that contain 22600 train, 888 validation, and 697 test stereo image pairs. For Cityscapes, it is a more challenging dataset with many dynamic scenes. With a few exceptions [35, 5, 17] it has not been used for depth estimation evaluation. It has 38675 training examples. We use depth from the disparity data for evaluation on a standard evaluation set of 1250 samples [35, 5]. We evaluate on both these datasets following the same evaluation strategy used in semantic forecasting [30, 29].

Implementation Details: We implement our framework in PyTorch and conduct experiments on a 11 GB Nvidia RTX 2080-Ti GPU. The ResNext [49] backbone is pre-trained on ImageNet [8] using batch normalisation. The input dimensions of the video frames are set as 512×1024 H'= 512 and W'= 1024. Optical Flow for the video frames are computed using the flow network in FlowNet2.0 [20]. Our DeFNet and Ego-Motion Network is jointly trained for 30 epochs with learning rate of 0.0001 using Adam Optimizer [26]. The entire network takes 53 mins to converge. The learning objective hyperparameters $\alpha/\lambda/\gamma$ are set to 0.4/0.5/0.6. For fair comparison, we follow existing work [17], where we consider k = 5 and k = 10 frames in the future (corresponding to respectively 0.29s and 0.59s). The mean inference speed of our DeFNet on image of size 512×1024 is 13 FPS on 2080GTX GPU.

Evaluation Metric: We quantify our approach following standard monocular depth estimation setting in short and mid term setting. Specifically, we report the square relative error (Sq Rel), absolute relative error (Abs Rel), root mean square error (RMSE), log scale invariant RMSE (RMSE-Log) and the amount of inliers (δ) for both KITTI and Cityscapes dataset. We refer the reader to [9] or appendix for a complete description of these metrics.

4.1 Comparison with the state-of-the-art methods

Competitors: For comparative comparison, we consider a multi-scale instance segmentation model [42], a unsupervised depth estimation model [14] and the latest depth forecasting approaches [17, 36]. Because [14] is not designed for forecasting task, we add a basic F2F[29] before the decoder of

KITTI								
Method	Supervision	Higher is better			Lower is better			
		$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	Abs Rel	Sq Rel	RMSE	RMSE-Log
Oracle	-	0.888	0.970	0.984	0.097	0.734	4.442	0.187
Copy Last	-	0.816	0.941	0.976	0.141	1.029	5.350	0.216
Qi et al. [36]	Supervised	-	-	-	0.108	0.806	4.630	0.193
Sun et al. [42]	Supervised	0.852	0.947	0.977	0.112	0.875	4.958	0.207
Goddard et al. [14]	Unsupervised	0.877	0.959	0.981	0.115	0.903	4.863	0.193
Ours	Unsupervised	**0.878**	0.953	**0.983**	**0.107**	0.849	**4.614**	**0.182**
Ours w/o Automask	Unsupervised	0.803	**0.960**	0.986	0.113	**0.741**	4.621	0.189
Ours w/o PCAB	Unsupervised	0.864	0.954	0.979	0.111	0.867	4.714	0.199
Ours w/o PCAB + Automask	Unsupervised	0.844	0.941	0.978	0.119	1.201	5.888	0.208
Ours w/o Flow Forecast	Unsupervised	0.859	0.947	0.980	0.113	0.894	4.804	0.208
Cityscapes								
Method	Supervision	Higher is better			Lower is better			
		$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	Abs Rel	Sq Rel	RMSE	RMSE-Log
Oracle	-	0.836	0.943	0.974	0.127	1.031	5.266	0.221
Copy Last	-	0.765	0.893	0.940	0.257	4.238	7.273	0.448
Qi et al. [36]	Supervised	0.678	0.885	0.957	0.208	1.768	6.865	0.283
Hu et al. [17]	Supervised	0.725	0.906	0.963	0.182	1.481	6.501	0.267
Sun et al. [42]	Supervised	0.801	0.913	0.950	0.227	3.80	6.91	0.414
Goddard et al. [14]	Unsupervised	**0.836**	0.930	0.958	0.193	1.438	5.887	0.234
Ours	Unsupervised	0.793	**0.931**	**0.973**	**0.174**	**1.296**	**5.857**	**0.233**
Ours w/o Automask	Unsupervised	0.680	0.898	0.967	0.201	1.584	6.471	0.273
Ours w/o PCAB	Unsupervised	0.784	0.916	0.961	0.198	1.438	6.216	0.270
Ours w/o PCAB + Automask	Unsupervised	0.776	0.903	0.949	0.234	3.776	7.104	0.416
Ours w/o Flow Forecast	Unsupervised	0.730	0.919	0.958	0.189	1.533	6.315	0.279

Table 1: **Quantitative Evaluation** of our DeFNet model with existing approaches and baselines on KITTI val (eigen split) and Cityscapes dataset for short-term forecasting.

the depth network. Since KITTI does not have vehicle control data, we cannot train [17] on KITTI, instead we compare another similar approach [36] for both short and mid-term forecast. Following [29, 17], we report the accuracy of oracle as a upper bound and we use a trivial copy baseline as a lower bound.

Results: The short-term forecast results are compared in Tab 1. Refer to appendix for mid-term forecast results. It is evident that our method outperforms unsupervised approaches alongside performing competitively with existing supervised approaches for both the forecasts. This suggests the superiority of our forecasting model design over existing approaches. As illustrated in fig 4., in short-term forecast on KITTI our method gains by $\sim 1-5\%$ in Abs Rel over a supervised methods [36, 17] which is a significant contribution while beating strongly customised unsupervised baseline by $\sim 5-8\%$. For mid-term forecasts, [17, 14] performs competitively as they use convLSTMs however being second-best to ours in most of the metrics. Our method performs better in mid-term forecast probably due to the PCAB block and multi-scale features. Interestingly, the variant of Ours without PCAB and Automask (last row of Tab 1) performs worse than adapted baseline of [14]. This indicates that vanilla PoseCNN along with cGRU cannot boost the performance alone. In cityscapes, we similarly observe that our method still gains over other approaches by $\sim 8\%$ while [42] performs the worst among all. This is because of the challenging scenes in the dataset and inability to handle occlusion. On the other hand, [17] performs better than [36, 14] suggesting that probabilistic modeling of harder scenes is a solution. Our variant without the flow forecast shows a sharp decrease ($\sim 8\%/11\%$) in performance for short/mid-term forecasts suggesting that 2D motion is useful in refining the depth in occluded scenes particularly in mid-term forecast. Please refer appendix for more qualitative and quantitative evaluation.

4.2 Ablation Studies

Effect of Channel Attention and Automasking in Ego-Motion In order to validate the effectivity of the proposed PCAB block and automasking, we compare the test performance of our network trained without i) PCAB ii) Automasking iii) both PCAB and Automasking. From Tab 1, we can see that the variant (i) without PCAB drops in performance by $4/9\%$ in Abs Rel for short/mid forecast. This indicates that attention is necessary when there is a lack of annotated data. For variant (ii) without Automasking, we observe that the performance drop is slightly higher ($\sim 6/11\%$) in short/mid forecast. This is expected since inability to handle rigid motion is an existing problem even in self-supervised monocular depth setting. The illustration in Fig 4 clearly indicates the need of automasking for this task. For variant(iii), we observe the highest drop of $\sim 12/19\%$, which suggests that using a simple PoseCNN is ineffective for the task of depth forecasting similar to our previous finding.

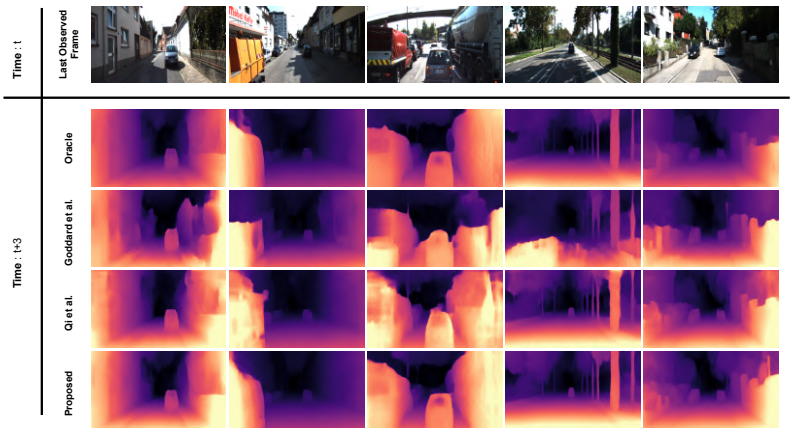

Figure 4: **Qualitative Illustration on KITTI:** Illustrates the short-term forecast on KITTI

Does Ego-Motion Forecast help ? In our approach we have forecasted rgb and flow to obtain a forecasted depth. The next natural question that can be asked is : *Can we also forecast ego-motion to improve the forecasted depth ?*. In order to validate this setup, we adapted the ego-motion prediction module from [36] in our forecasting setting denoted as $Ours^{\dagger}$. We obtained forecasted flow $\mathbf{W}_{t+k}$, point cloud depth from our forecasted $\mathbf{D}_{t+k}$, and the pose estimate $[R|T]_{t-1,t}$ to predict $[R|T]_{t,t+k}$. From Tab 2, we can observe that the adapted setting ($Ours^{\dagger}$) performs competitively with our original setting for short term but surprisingly drops by $\sim 8\%$ in Abs Rel metric for mid-term forecast. A possible reason for this is that, the performance of ego-motion forecast is dependent on the quality of depth and flow maps. Since we are using forecasted depth and forecasted flow maps (which are generated in a unsupervised setting) to estimate the ego-motion forecast the performance drop in mid-term forecast is self-explanatory. This justifies our choice to not incorporate ego-motion forecast and instead condition the camera parameters on unobserved rgb frames.

Methodology	Short	Mid
	Abs Rel	Abs Rel
Ours	**0.107**	**0.125**
Ours†	0.108	0.134

Table 2: **Effect of Ego-Motion forecast** on KITTI dataset. We evaluate the effectiveness of Our approach vs our variant ($Ours^{\dagger}$) with ego-motion forecast on short and mid level forecast

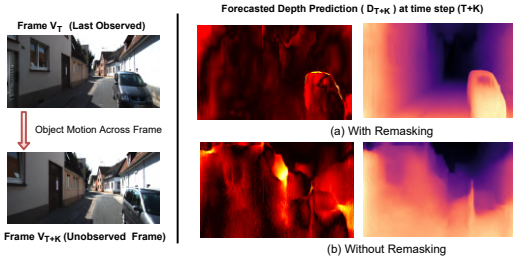

Figure 4. **Illustration of the effect of automask** on depth forecasting

5 Limitations of our approach

This problem although can work on wide array of unconstrained videos, it is not designed to handle jittery/unstable videos. As a result, our method will struggle to estimate depth for the videos captured using hand-held devices. Additionally, our approach is based on how the scene is illuminated – which makes our design vulnerable to scene illumination. Hence, creating a design that is invariant to mode of capture and illumination is thus a part of our future work.

6 Conclusion

We present a novel spatio-temporal forecasting framework DeFNet for the depth forecasting problem. Our method is designed to mitigate the practical barriers to predicting future frame depth such as annotation cost, generalization ability etc. We for the very first time solve the depth forecasting problem in a self-supervised fashion by formulating the depth estimation task as a novel-view synthesis problem. This allows our approach to solve the depth forecasting as an auxiliary task and thus making our approach annotation free. Experiments on two popular datasets verify the superiority of our approach. Moreover, being able to learn depth without the need for intrinsics/labels opens up the opportunity for pooling videos from any data sources together for a wider application.

References

[1] Nicolas Ballas, Li Yao, Chris Pal, and Aaron Courville. Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*, 2015.

[2] Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J Davison. Codeslam—learning a compact, optimisable representation for dense visual slam. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2560–2568, 2018.

[3] Arunkumar Byravan and Dieter Fox. Se3-nets: Learning rigid body motion using deep neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 173–180. IEEE, 2017.

[4] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8001–8008, 2019.

[5] Vincent Casser, Soeren Pirk, Reza Mahjourian, and Anelia Angelova. Unsupervised learning of depth and ego-motion: A structured approach. In *Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)*, volume 2, page 7, 2019.

[6] Bin Cheng, Inderjot Singh Saggu, Raunak Shah, Gaurav Bansal, and Dinesh Bharadia. S3net: Semantic-aware self-supervised depth estimation with monocular videos and synthetic data. In *European Conference on Computer Vision*, pages 52–69. Springer, 2020.

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[9] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.

[10] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2002–2011, 2018.

[11] Ravi Garg, Vijay Kumar Bg, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European conference on computer vision*, pages 740–756. Springer, 2016.

[12] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[13] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 270–279, 2017.

[14] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 3828–3838, 2019.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[16] Joao F Henriques and Andrea Vedaldi. Warped convolutions: Efficient invariance to spatial transformations. In *International Conference on Machine Learning*, pages 1461–1469. PMLR, 2017.

[17] Anthony Hu, Fergal Cotter, Nikhil Mohan, Corina Gurau, and Alex Kendall. Probabilistic future prediction for video scene understanding. In *European Conference on Computer Vision*, pages 767–785. Springer, 2020.

[18] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8981–8989, 2018.

[19] Tak-Wai Hui, Xiaoou Tang, and Chen Change Loy. A lightweight optical flow cnn-revisiting data fidelity and regularization. *arXiv preprint arXiv:1903.07414*, 2019.

[20] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017.

[21] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. *arXiv preprint arXiv:1506.02025*, 2015.

[22] Joel Janai, Fatma Guney, Anurag Ranjan, Michael Black, and Andreas Geiger. Unsupervised learning of multi-frame optical flow with occlusions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 690–706, 2018.

[23] Xiaojie Jin, Xin Li, Huaxin Xiao, Xiaohui Shen, Zhe Lin, Jimei Yang, Yunpeng Chen, Jian Dong, Luoqi Liu, Zequn Jie, et al. Video scene parsing with predictive feature learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5580–5588, 2017.

[24] Nal Kalchbrenner, Aäron Oord, Karen Simonyan, Ivo Danihelka, Oriol Vinyals, Alex Graves, and Koray Kavukcuoglu. Video pixel networks. In *International Conference on Machine Learning*, pages 1771–1779. PMLR, 2017.

[25] Kevin Karsch, Ce Liu, and Sing Bing Kang. Depth transfer: Depth extraction from video using non-parametric sampling. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2144–2158, 2014.

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

[27] Ismaël Koné and Lahsen Boulmane. Hierarchical resnext models for breast cancer histology image classification. In *International Conference Image Analysis and Recognition*, pages 796–803. Springer, 2018.

[28] Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. From big to small: Multi-scale local planar guidance for monocular depth estimation. *arXiv preprint arXiv:1907.10326*, 2019.

[29] Pauline Luc, Camille Couprie, Yann Lecun, and Jakob Verbeek. Predicting future instance segmentation by forecasting convolutional features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 584–599, 2018.

[30] Pauline Luc, Natalia Neverova, Camille Couprie, Jakob Verbeek, and Yann LeCun. Predicting deeper into the future of semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 648–657, 2017.

[31] Chenxu Luo, Zhenheng Yang, Peng Wang, Yang Wang, Wei Xu, Ram Nevatia, and Alan Yuille. Every pixel counts++: Joint learning of geometry and motion with 3d holistic understanding. *IEEE transactions on pattern analysis and machine intelligence*, 42(10):2624–2641, 2019.

[32] Zelun Luo, Boya Peng, De-An Huang, Alexandre Alahi, and Li Fei-Fei. Unsupervised learning of long-term motion dynamics for videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2203–2212, 2017.

[33] Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015.

[34] David Nilsson and Cristian Sminchisescu. Semantic video segmentation by gated recurrent flow propagation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6819–6828, 2018.

[35] Andrea Pilzer, Dan Xu, Mihai Puscas, Elisa Ricci, and Nicu Sebe. Unsupervised adversarial depth estimation using cycled generative networks. In *2018 International Conference on 3D Vision (3DV)*, pages 587–595. IEEE, 2018.

[36] Xiaojuan Qi, Zhengzhe Liu, Qifeng Chen, and Jiaya Jia. 3d motion decomposition for rgbd future dynamic scene synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7673–7682, 2019.

[37] Rene Ranftl, Vibhav Vineet, Qifeng Chen, and Vladlen Koltun. Dense monocular depth estimation in complex dynamic scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4058–4066, 2016.

[38] MarcAurelio Ranzato, Arthur Szlam, Joan Bruna, Michael Mathieu, Ronan Collobert, and Sumit Chopra. Video (language) modeling: a baseline for generative models of natural videos. *arXiv preprint arXiv:1412.6604*, 2014.

[39] Josip Saric, Marin Orsic, Tonci Antunovic, Sacha Vrazic, and Sinisa Segvic. Warp to the future: Joint forecasting of features and feature motion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10648–10657, 2020.

[40] Selim Seferbekov, Vladimir Iglovikov, Alexander Buslaev, and Alexey Shvets. Feature pyramid network for multi-class land segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 272–275, 2018.

[41] Abhronil Sengupta, Yuting Ye, Robert Wang, Chiao Liu, and Kaushik Roy. Going deeper in spiking neural networks: Vgg and residual architectures. *Frontiers in neuroscience*, 13:95, 2019.

[42] Jiangxin Sun, Jiafeng Xie, Jian-Fang Hu, Zihang Lin, Jianhuang Lai, Wenjun Zeng, and Wei-shi Zheng. Predicting future instance segmentation with contextual pyramid convlstms. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 2043–2051, 2019.

[43] Sasha Targ, Diogo Almeida, and Kevin Lyman. Resnet in resnet: Generalizing residual architectures. *arXiv preprint arXiv:1603.08029*, 2016.

[44] Adam Terwilliger, Garrick Brazil, and Xiaoming Liu. Recurrent flow-guided semantic forecasting. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1703–1712. IEEE, 2019.

[45] Sudheendra Vijayanarasimhan, Susanna Ricco, Cordelia Schmid, Rahul Sukthankar, and Katerina Fragkiadaki. Sfm-net: Learning of structure and motion from video. *arXiv preprint arXiv:1704.07804*, 2017.

[46] Chaoyang Wang, Simon Lucey, Federico Perazzi, and Oliver Wang. Web stereo video supervision for depth prediction from dynamic scenes. In *2019 International Conference on 3D Vision (3DV)*, pages 348–357. IEEE, 2019.

[47] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[48] Junyuan Xie, Ross Girshick, and Ali Farhadi. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In *European Conference on Computer Vision*, pages 842–857. Springer, 2016.

[49] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[50] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1983–1992, 2018.

[51] Huizhong Zhou, Benjamin Ummenhofer, and Thomas Brox. Deeptam: Deep tracking and mapping. In *Proceedings of the European conference on computer vision (ECCV)*, pages 822–838, 2018.

[52] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1851–1858, 2017.

[53] Wei Zhuo, Mathieu Salzmann, Xuming He, and Miaomiao Liu. Indoor scene structure analysis for single image depth estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 614–622, 2015.

A Appendix

A.1 More Details of the Feature Forecasting Block

As discussed in section 3.1, the convGRU(cGRU) is employed to capture both intra and inter-level spatio-temporal dependencies. Here we discuss in detail the inter-level context sharing mechanism.

Mathematically, the block of cGRU for the features of the l-th pyramid level (denoted by $FB(\cdot)_l$) can be formulated as follows:

$$z_t^l = \sigma(W_z^l * X_t^l + H_z^l * Q_{t-1}^l + B_z^l) \qquad r_t^l = \sigma(W_r^l * X_t^l + H_r^l * Q_{t-1}^l + B_r^l) \tag{10}$$

$$\hat{Q}_t^l = tanh(H_q^l * (r_t^l \odot Q_{t-1}^l) + W_x^l * X_t^l + B_q^l) \qquad Q_t^l = (1 - z_t^l) \odot Q_{t-1}^l + z_t^l \odot \hat{Q}_t^l \tag{11}$$

where W^l, H^l are the convolution kernels controlling the information propagation along the update gate (z_t) and reset gate (r_t). Q_t denotes the predicted forecasted feature ($F_{rgb}^{t+k} = Q_t$) and X_t denotes the input pyramidal features. σ is a sigmoid function and B is the bias term. As per figure 2(d), it can be observed that the hidden states of the top level are passed on to the lower levels. This enhances the higher-level contexts by incorporating lower-level contextual details with it. Overall, for a certain cell in the forecasting block $FB(.)_l$, it accepts information from other cells including the feature of the l-th level block Q at the current time step t, output of previous time step Q_{t-1}^l, and hidden states of higher levels forecasting blocks from previous time steps $\{Q_{t-1}^{l-k}, Q_{t-1}^{l-k+1}, .., Q_{t-1}^{l-1}\}$. For the cGRU of the l-th level, the information propagation with pathways can be formulated as follows:

$$Q_t^l = (1 - z_t^l) \odot Q \Uparrow_{t-1}^l + z_t^l \odot \hat{Q}_t^l \qquad Q \Uparrow_{t-1}^l = Q_{t-1}^l + \sum \mathbb{A}^{l-1,l} \odot \phi(Q_t^{l-1}) \tag{12}$$

$$\mathbb{A}^{l-1,l} = \sigma(W_{l-1,l} * \phi(Q_t^{l-1}) + B_{l-1,l}) \tag{13}$$

where $W_{l-1,l}$ is the weight parameter facilitating the inter-context information transmission from a cGRU in $FB(.)_{l-1}$ to a cGRU in $FB(.)_l$. Here $\phi(.)$ denotes the upsampling operator to interpolate bi-linearly to match the dimensions of Q_{t-1}^{l-1} with Q_{t-1}^l. The attention vector $\mathbb{A}^{l-1,l}$ is used to selectively transfer the context from the higher levels to the lower levels, such that the forecasted features are not plagued by features representing occluded objects. The effectiveness of this inter-GRU context sharing is illustrated in Table 2 (of main paper) where we compare two variants of the forecasting block : 1) ConvGRU-Intra : This variant does not have any intra-level sharing, , only inter-level context sharing 2) ConvGRU-Inter : This variant does not have any inter-level sharing, , same level context sharing only. Of these two variants, it can be observed that ConvGRU-Inter actually has more performance drop suggesting that selectively passing higher level context is necessary for a good forecast.

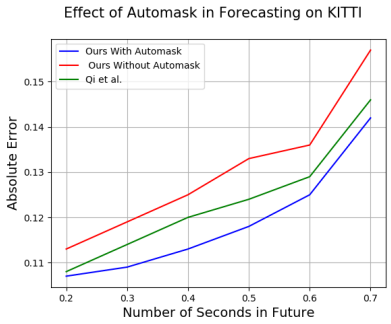

Figure 5: **Effect of Automasking:** Illustrates the importance of using Automasking in refining the depth forecast over multiple seconds into the future. We compared our method with and without the Automask as we ll as with a exisiting depth forecasting aprroach [36] on KITTI dataset. The plot shows that our approach with automask is stable as the scene evolves, however ours without automask is quite unstable as highlighted in quantitative evaluation in Table S1.

A.2 Details on Pose Channel Attention Block

It highlights the relative importance of the certain regions in target and source views and enhances the channels with more effective information such as uncertainties introduced by moving objects, occlusions, and incomplete Lambertian surfaces. Given a local feature $x \in \mathbb{R}^{C \times H \times W}$, where C, H, W denote channel, height, width. We independently convolute x through 3 $conv$ blocks with 3×3 filter to obtain 3 feature outputs denoted by $Q \in \mathbb{R}^{C \times H \times W}$, $K \in \mathbb{R}^{C \times HW}$, $V \in \mathbb{R}^{HW \times C}$. K and V are dot producted (elementwise multiplication), and fed to a softmax operation to regress

channel attention ω. Finally, a dot product is performed between Q and ω to obtain the output of channel attention $x^{'}$ as follows:

$$\omega = softmax(F(K) \odot F(V)), x^{'} = F(Q) \odot \omega \tag{14}$$

where $F(.)$ denotes 3×3 convolution layer. The effectiveness of the proposed channel attention based reconstruction block is demonstrated in the experiment section (section 4.2 of main paper). The output of the PCAB is passed through 2 fully-connected layers to obtain the relative pose $P_{so \to ta}$ which corresponds to the estimated pose for the forecasted depth at $t + k$.

A.3 Details of self-supervised objective functions

We discuss in detail the loss components that drive our network.

(a) **Masked Photometric(L_{mpe})** : Photometric loss is a standard L1 loss calculated between the reconstructed view $I_{so \to ta}$ and target view I_{ta} denoted as L_{pe}. We formulate the mask photometric loss L_{mpe} by performing the dot product with mask L_{pe} as follows:

$$L_{mpe} = \sum_{s} (A_s . L_{pe}), s \in (so \to ta, ta \to so) \tag{15}$$

(b) **Dissimilarity (L_{ds}):** This is resilient to outliers as well as being differentiable. It calculates the dissimilarity in source and target views. We define it as follows:

$$L_{ds} = \frac{1 - SSIM(\hat{v}_{so}, v_{ta})}{2} \tag{16}$$

(c) **Smoothness (L_{sm}):** This term mainly contributes to the quality of the disparity map. Applying such regularization enforces the DeFNet to produce sharp edge distribution at pixels that change rapidly, while produces smooth depth in continuous regions.

$$L_{sm} = \sum_{k \epsilon x,y} \sum_{i,j} \left| \partial_k D^{i,j} \right| \exp^{-\left| \partial_k v_{ta}^{i,j} \right|} \tag{17}$$

where $D^{i,j}$ is the mean-normalized inverse depth map. i, j denote pixel index value of v_{ta}. ∂_x and ∂_y denote gradients in the x and y directions, respectively.
(d) **Pose Consistency (L_{pc}):**It a simple L1 loss that ensures the 'past' and 'future' inter-frame translations are consistent with each other. It is represented as

$$L_{pc} = \left\| P_{t,t+k} - P_{t+2k,t+k} \right\|_1 \tag{18}$$

.

A.4 Evaluation Metrics

There has been a lot of depth estimation metrics to quantitatively evaluate the estimated depth from the models. For more meaningful analysis of predicted depth maps and complete comparison of different algorithms, we use the same metrics as [9, 53] used, namely Absolute Relative Error, Linear Root Mean Square Error, log scale invariant RMSE. Given a predicted depth map d_p and its corresponding ground truth d_g, T is the total number of pixels and n is the current pixel. The evaluation metrics are represented mathematically as follows

$$AE = \frac{1}{T} \sum_{n} \frac{|d_g - d_p|}{d_g}$$

$$RMSE = \sqrt{\frac{1}{T} \sum_{n} (d_g - d_p)^2}$$

$$Log\ RMSE = \frac{1}{T} \sum_{n} (\log d_g - \log d_p + \alpha(d_g, d_p))^2$$

A.5 More Quantitative Analysis

The mid-term forecast results are compared in Table S1. Drawing similar observation from short-term forecasting, our method outperforms existing baselines and approaches. In KITTI, our method gains by 3% in Abs Rel over [36] while beating the adapted baselines [42, 14] by $\sim 10-27\%$. Without any surprise, [42] performs the worst among all and very similar to the variant of ours without PCAB and Automask (last row of Table S1) which justifies our claim of the usefulness of ego-motion branch. In Cityscapes, we observe almost a similar trend like that of KITTI but in some metrics [17] performed better. This is because of the complexity of the cityscapes dataset where there are multiple person object and occlusion, suggesting the need to handle them for better quality forecast. Our variant without PCAB and Automask (last row of Table S1) performs the worst in this dataset. It is interesting to note that most of the methods without ego-motion perform worse at mid-forecast, which highlights the need to address motion for longer forecasts.

KITTI								
Method	Supervision	Higher is better			Lower is better			
		$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	Abs Rel	Sq Rel	RMSE	RMSE-Log
Oracle	-	0.873	0.954	0.981	0.108	0.660	3.756	0.211
Copy Last	-	0.559	0.791	0.844	0.228	3.01	7.824	0.389
Qi et al. [36]	Supervised	-	-	-	0.129	0.819	4.372	0.234
Sun et al. [42]	Supervised	0.696	0.805	0.824	0.159	0.982	5.462	0.308
Goddard et al. [14]	Unsupervised	0.768	0.904	0.916	0.138	0.894	4.911	0.249
Ours	Unsupervised	**0.794**	**0.919**	**0.943**	**0.125**	**0.798**	**4.113**	**0.227**
Ours w/o Automask	Unsupervised	0.774	0.898	0.919	0.136	0.872	4.924	0.284
Ours w/o PCAB	Unsupervised	0.782	0.907	0.928	0.134	0.851	4.845	0.245
Ours w/o PCAB + Automask	Unsupervised	0.705	0.817	0.874	0.144	0.915	6.271	0.302
Ours w/o Flow Forecast	Unsupervised	0.746	0.906	0.928	0.135	0.845	4.711	0.298
Cityscapes								
Method	Supervision	Higher is better			Lower is better			
		$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	Abs Rel	Sq Rel	RMSE	RMSE-Log
Oracle	-	0.812	0.928	0.952	0.148	1.451	5.923	0.258
Copy Last	-	0.511	0.781	0.802	0.304	5.006	8.319	0.517
Qi et al. [36]	Supervised	0.718	0.857	0.881	0.224	3.015	7.661	0.394
Hu et al. [17]	Supervised	0.735	0.896	**0.928**	0.195	**1.712**	**6.375**	0.299
Sun et al. [42]	Supervised	0.695	0.817	0.842	0.259	4.115	7.842	0.428
Goddard et al. [14]	Unsupervised	0.724	0.853	0.882	0.211	2.478	7.266	0.357
Ours	Unsupervised	**0.742**	**0.900**	0.927	**0.192**	1.719	6.388	**0.298**
Ours w/o Automask	Unsupervised	0.713	0.851	0.874	0.241	3.101	7.884	0.417
Ours w/o PCAB	Unsupervised	0.729	0.865	0.892	0.207	2.280	6.919	0.321
Ours w/o PCAB + Automask	Unsupervised	0.680	0.804	0.829	0.262	4.247	7.976	0.472
Ours w/o Flow Forecast	Unsupervised	0.740	0.876	0.910	0.210	2.081	7.249	0.376

Table 2: **Quantitative Evaluation** of our DeFNet model with existing approaches and baselines on KITTI val (eigen split) and Cityscapes dataset for mid-term forecasting.

A.6 Qualitative Analysis

In Figure S2 and S3, we present qualitative comparisons to multiple previous works and baselines on KITTI and in Figure S4, we present qualitative comparisons on Cityscapes dataset. In KITTI, we observe that our method produces sharper predictions for thin structures and complex shapes such as people. Goddard et al [14] and Qi et al. [36] loses geometric consistency in Mid Term forecasting whereas our method excels in preserving the shape. This highlights that ego-motion network contributes with scene geometrics for future forecasts. In Cityscapes, we observe that in shorter forecasts, our method produces slightly blurred forecast – this explains the difficulty of the scenes in the dataset. We also observe that in low-illumination region of the image our method struggles to get a good forecast. Both the baselines [36, 14] also fails to preserve shape and scene geometry. It is interesting to note that in mid-forecast when the car moves further away from the scene, our method still can maintain the shape and structure, however, the baselines [36, 14] fails drastically as the object moves away – this shows the importance of multi-scale forecasting over single scale and also highlights the importance of handling motion for longer forecast which is consistent with our previous findings.

A.7 Further Analysis

• **Is Pose Really Making the difference?** To validate the role of pose in estimating scene geometrics of the forecasts, we perform experiments in Table S2 where we compare with a copy baseline that directly copies the $[R|T]_{t-1,t}$ similar to [36]. We also compare with the depth forecasting method

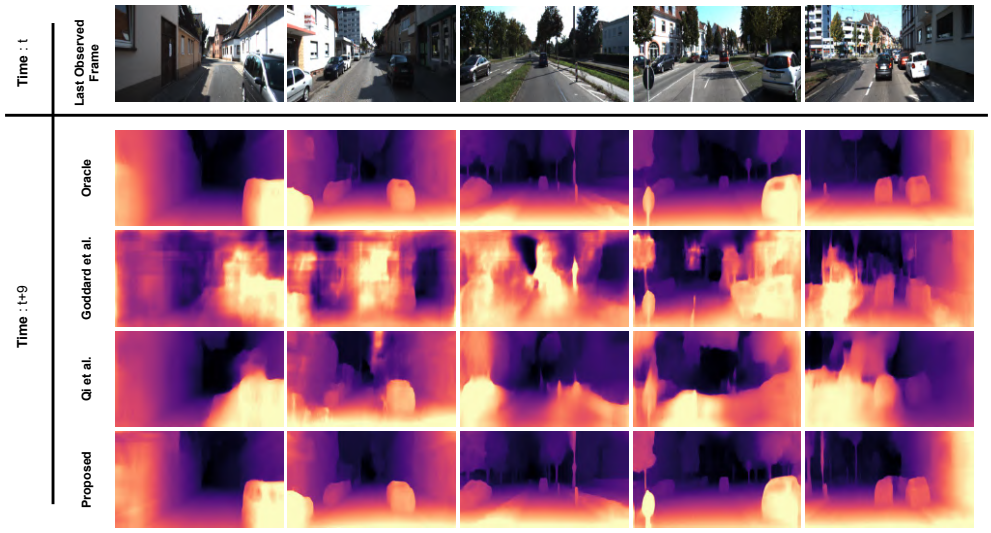

Figure 6: **Qualitative Illustration on KITTI:** Illustrates the mid-term forecast on KITTI

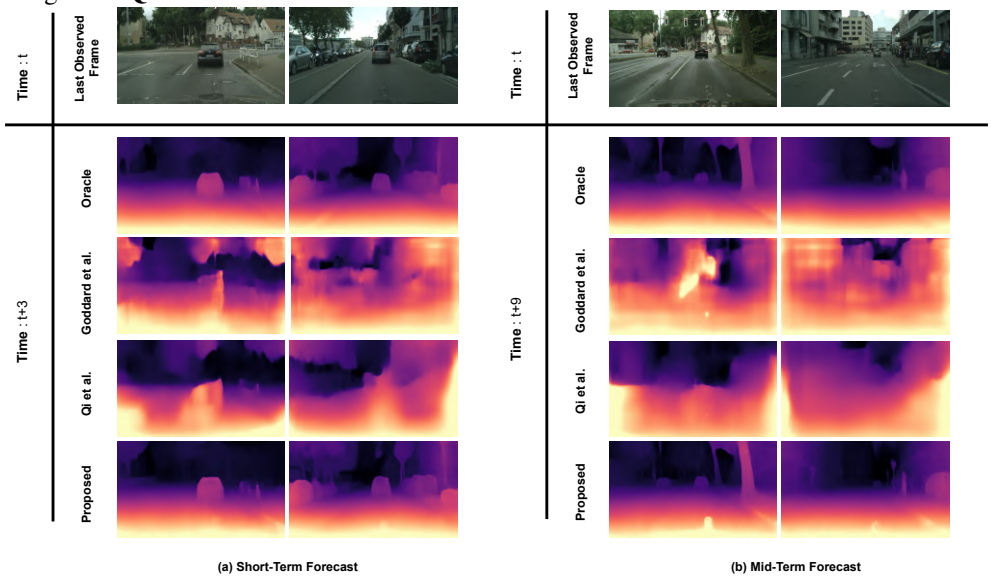

Figure 7: **Qualitative Illustration on Cityscapes:** Illustrates the (a) short-term forecast and (b) mid-term forecast on Cityscapes

proposed by Qi et al. [36] and validate these methods in mid-term forecast setting on KITTI dataset. Similar to [36], our ego-motion module reduces the mean angle error (RAE) by 28% and RMAE by 38% highlighting the importance of using the PCAB in ego-motion network and also suggesting that estimating the 6-DOF pose for the future unseen depth is indeed important. A plausible reason for this improvement is the choice of the source and target views during the training phase of the ego-motion network. During training, we pass 3 images to the ego-motion network – i) two source views ii) one target view. The source view in ideal scenarios can be selected as adjacent views of the target. However, we carefully chose the views at time step "t" and "t+2k" as source and view at "t+k" as the target. It is interesting to note that we did not use the frames at "t+k" and "t+2k" to train the depth forecasting network, hence by estimating the pose $P_{t,t+k}/P_{t+k,t+2k}$ we are actually infusing some meaningful information about the future scene geometrics into the DeFNet. This might be the major contributor to the improvement of our method over the existing approaches as seen from our experiment in Table S2 and also from the qualitative results in Figure S2,S3,S4.

• **Choice of Encoder Backbone** Table S3 explores influence of single frame to the forecasting accuracy. We consider 4 different backbones for rgb encoder and 3 different backbones for flow encoder for evaluating the quality of depth forecasts. For rgb encoder, the backbone based on ResNet-101 has a better performance gain of 4% over backbone with VGG due to more deeper layers.

The backbone with ResNext-101 alone performs slightly inferior to our FPN variant in short forecast but drops drastically on mid-forecast. This advantage is largely due to the presence of multi-scale features which improves the receptive fields over both short and mid-forecast which otherwise is unavailable for single scale features in VGG, ResNet-101 and ResNext-101. Due to the power of hierarchical pyramidal structure in FPN, we use the ResNext + FPN variant as our rgb-encoder. For flow encoder, FlowCNN performs a little worse than LiteFlowNet and LiteFlowNet2 in short forecast (around 9%) but has a major drop (around 15%) in mid-forecast. This is again due to the inability of the network to handle features at different scales unlike LiteFlowNet/LiteFlowNet2.0 which are multi-scale networks. It is interesting to observe that LiteFlowNet2.0 although uses less parameters than LiteFlowNet performs almost similar in short but is slightly better in mid-forecast as it handles the flow information better due to extra regularization. Thus we consider LiteFlowNet2.0 as our flow forecasting backbone.

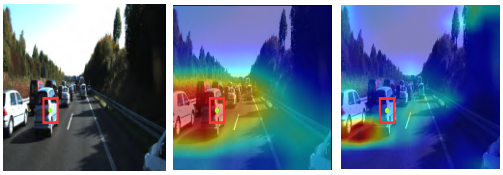

Figure 3 **Illustration** of the effect of receptive field for multi-level forecasting vs single level forecast

Method	Short Term	Mid Term
	Abs Rel	Abs Rel
F2F [29]	0.112	0.131
ConvLSTM [42]	0.108	0.124
ConvGRU (Ours)	**0.107**	**0.122**
ConvGRU-Intra	0.109	0.128
ConvGRU-Inter	0.111	0.133

Table S3: **Validation of the design choice** of our forecasting block on KITTI dataset. We compare our cGRU along with Inter-Context and Intra-Context variants with other baselines

Method	Mid Term	
	RAE ↓	RMSE ↓
Copy	0.502	0.045
Qi et al[36]	0.411	0.029
Ours	**0.320**	**0.018**

Table S2: **Future frame pose evaluation on KITTI dataset** RAE means relative angle error for the rotation component. RMSE represents root mean square error for the translation component. ↓ means the lower the better

Encoder	Backbone	Short Term	Mid Term
		Abs Rel	Abs Rel
RGB	VGG [41]	0.119	0.141
	ResNet-101 [43]	0.114	0.135
	ResNext-101 [27]	0.109	0.126
	ResNext-101+FPN [40](Ours)	**0.107**	**0.125**
Flow	FlowCNN [44]	0.117	0.145
	LiteFlowNet [18]	0.107	0.126
	LiteFlowNet2.0 [19](Ours)	**0.107**	**0.125**

Table S3: **Influence of Backbone on Forecasting Performance** We compare our choice with the existing backbones for rgb and flow over both short/mid forecast on KITTI dataset.

•**Single Level Forecast vs Multi-level Forecast:** Top-down architectures is beneficial in capturing more contextual information as seen in Mask-RCNN. To verify this, we conducted a experiment by replacing our cGRU with F2F [29] and ConvLSTM [42] as shown in Tab S3. As compared to F2F which employs a CNN to predict features at each level, Conv-LSTMs effectively capture intra-level spatio-temporal context, which is important for estimating depth. In section 3.1, we introduced a convGRU as our forecasting block to capture intra and inter-level context among the pyramidal hierarchies. From Tab S3 we can see that using $F2F$ in our forecasting block drops the performance by $5/9\%$ in short/mid-forecasts. It is interesting to note that ConvLSTM performs slightly inferior to cGRU in both short and mid-forecast. This is consistent with our findings in Tab 1. This is probably because convLSTMs use more gates than convGRUs which may not work well for self-supervised setting. We also validated a cGRU with only intra-level information sharing, however, we observe that the performance drops by $2/6\%$ in short/mid forecast which indicates that a single cGRU is not capable of forecasting better depths. When we do a inter-level information sharing, the performance drops even more ($\sim 4/11\%$), which clearly shows the benefit of using multi-scale forecasting in capturing global context.