Accurate and Real-time LiDAR Point Cloud Forecasting for Autonomous Driving

Soham Dasgupta
IIT Jodhpur
Jodhpur, IN
sohamd@iitj.ac.in

Kshitij Aphale
IIT Jodhpur
Jodhpur, IN
aphale.1@iitj.ac.in

Kaustab Pal
IIIT Hyderabad
Hyderabad, IN
kaustab21@gmail.com

Avinash Sharma
IIT Jodhpur
Jodhpur, IN
avinashsharma@iitj.ac.in

Abstract

Point cloud forecasting is a crucial task for the success of motion planning and state estimation problems. However, due to its complex nature and challenges in integrating with existing architectures, it remains an extremely difficult and interesting problem. LiDAR-based 3D sensing is one of the key components in modern autonomous driving systems and leads to scalability challenges as it yields large-scale point cloud data at a high frame rate. Range image representation of point cloud offers a compact representation of LiDAR data which enables applying powerful convolution network architectures for predicting future range images (and thereby point clouds). In this paper, we use range image representation and propose two different non-recurrent deep network models for point cloud forecasting. More specifically, we predict future point clouds from past observed point clouds by introducing a spatio temporal convolution (STC) block in the latent space of range images, thereby avoiding the use of RNNs to achieve much faster inference. The STC has two variants that use the temporal attention and InceptionNet blocks, respectively. We perform experiments on two publicly available datasets, namely KITTI and Nuscenes and report superior quantitative and qualitative results in comparison to SOTA methods while offering compact model with faster inference speeds.

CCS Concepts

• **Computing methodologies** → **Vision for robotics**.

Keywords

Range Sensing, 3D computer vision, Robotics

ACM Reference Format:

Soham Dasgupta, Kshitij Aphale, Kaustab Pal, and Avinash Sharma. 2024. Accurate and Real-time LiDAR Point Cloud Forecasting for Autonomous Driving. In *Indian Conference on Computer Vision Graphics and Image Processing (ICVGIP 2024), December 13–15, 2024, Bengaluru, India.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3702250.3702264

ICVGIP 2024, December 13–15, 2024, Bengaluru, India

ACM ISBN 979-8-4007-1075-9/24/12
https://doi.org/10.1145/3702250.3702264

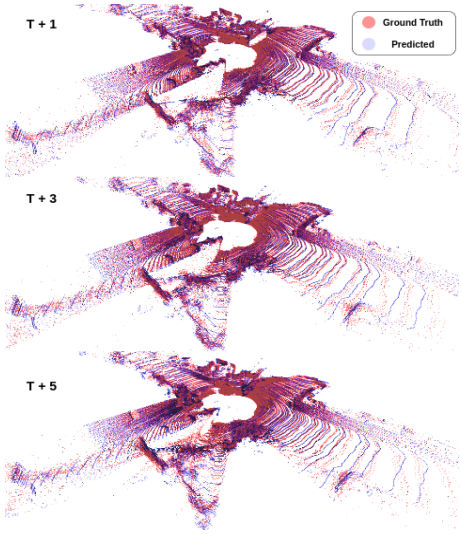


Figure 1: Point cloud forecasting with our "Model 2".

1 Introduction

Autonomous navigation is a rapidly evolving field with the potential to revolutionize various domains including transportation, exploration, and warehouse automation. By integrating a sophisticated array of sensors, computer vision algorithms, and deep learning models, autonomous navigation systems can interpret vast amounts of real-time data to construct accurate environmental maps, plan optimal routes, and execute precise movements. Among these sensor technologies, LiDAR plays a crucial role in autonomous navigation, serving as a key component for creating large-scale three-dimensional point clouds representing the surrounding environment in real-time. Apart from real-time sensing, predicting the future evolution of the objects in the scene greatly aids in localization & collision avoidance kind of tasks [17]. Thus, reliable point cloud forecasting [3, 11] is an important research challenge in autonomous driving.

Nevertheless, working with temporally evolving sequences of point clouds poses significant challenges. The unordered nature of point clouds in the spatial dimension (despite being temporally

ordered) and varying sampling sizes hinder the modeling of spatio-temporal coherence. As a result, conventional architectures for feature encoding (e.g., CNNs) and sequence processing (e.g., LSTMs) are unsuitable for direct application due to their inability to process spatially unordered data. Consequently, there have been a few breakthroughs in 3D computer vision where methods like PointNet [20] and its follow-up work PointNet++[21] have shown great success in feature extraction from point clouds.

Additionally, LiDAR point clouds data in autonomous navigation tasks have the additional challenge of having non-uniform sampling (farther regions have sparser sampling). This combined with a high degree of occlusions and sensor noise in real-world scenes makes it extremely challenging to capture the geometrical structures of objects and predict them in future timesteps. Another key challenge is that a full-scale LiDAR scan comprises over 100,000 points, thereby rendering feature extraction from these sequences a memory-intensive task and hence inhibiting the real-time performance.

Alternatively, range image representation of 3D point cloud (refer subsection 3.1) offers a compact solution to represent 3D data as a 2D matrix. This enables employing powerful convolution network architectures for predicting future range images (frames) and thus achieving point cloud forecasting. Although this representation suffers from some information loss while projecting 3D points to 2D image plane, it is still much easier to process & scale up with near real-time inference using well-established image-based deep learning architectures.

Recent methods for point cloud forecasting with range image representation [12, 16, 18] achieved impressive performance over traditional point cloud-based methods. TCNet [16] outlined a 3D CNN-based architecture to predict future frames and used masked range images to improve the prediction and improve noise suppression. Subsequently, PCPNet [12] proposed the use of semantic understanding of point cloud along with a transformer-based architecture to predict future frames. The recent method in [18] proposed to use LSTM-based architecture to learn the temporal coherence for range image prediction, with a much simpler and easy-to-converge model. However, these methods often produce extremely blurry results (loss of high-frequency details) and more importantly significant loss of details for small objects in the scene (refer to subsection 4.3). Additionally, the use of recurrent neural nets [18] for temporal regularization leads to compact model size but at the cost of slower inference time.

In this work, we propose a novel approach for point cloud forecasting from observed sequences of point clouds represented as LiDAR range images. Our main contribution in this paper is the introduction of a Spatio-Temporal Convolution (STC) block designed to operate in the latent space for learning to exploit spatio-temporal coherence in range images. More specifically, we employ an encoder-decoder architecture integrated with two variants, namely Temporal Attention [28] and InceptionNet [4] modules to effectively capture spatio-temporal dynamics. We perform extensive empirical analysis on publicly available datasets and report superior quantitative & qualitative results to demonstrate that our method (both variants) outperforms existing state-of-the-art techniques [18] in terms of both prediction accuracy as well as faster inference time.

2 Related Works

Point cloud forecasting is crucial for the success of motion planning algorithms [17]. Predicting future frames allows these algorithms to anticipate the motion of surrounding objects and plan accordingly. In the following section, we discuss existing literature critical to our problem, divided into two subparts relevant to our study: Point cloud forecasting and Spatio-temporal learning. For a more detailed related work, please refer to the *supplementary material*.

Point Cloud Forecasting : Modeling spatio-temporal information in point cloud forecasting has been a significant research direction. Initial efforts focused on predicting scene flows between successive point clouds to forecast future point cloud sequences [9, 37]. However, due to the high computational cost of scene flow methods, their application in downstream tasks like point cloud forecasting has often been limited. Recently methods have started forecasting point clouds directly from raw point cloud data [7, 12, 18, 32, 33]. Generally, 3D LiDAR point clouds can be represented in multiple forms, including explicit points [20, 21], voxels [15, 22], multi-view images [26, 35, 36], or range images [12, 16, 18]. Most recent point-voxel-based methods [7, 32, 33] are quite successful, however, they are extremely slow and consume a high amount of memory. In contrast, range image-based methods [12, 16, 18] often perform better than other representations in both computational efficiency and accuracy. However, despite decent performance in forecasting, these methods struggle with small objects like humans and trees, often producing blurrier range images. Recently, ATPPNet [18] used LSTM-based architectures to improve temporal consistency with a simpler, easier-to-converge model. However, even with improvements in inference speed and accuracy, the method still struggles to capture fine details. In this paper, we propose a range image-based representation-based point cloud forecasting method. Essentially converting the point clouds into range images, thereby reformulating point cloud forecasting as a spatio-temporal range image prediction problem.

Spatio-temporal Predictive Learning : As previously mentioned, range image prediction can be considered as a form of spatio-temporal prediction involving fixed sequences. The major challenge with our datasets is the high degree of noise and the presence of a moving camera, which sets them apart from other spatio-temporal datasets [24]. Due to these complexities, the problem can be reframed as a spatio-temporal learning task focused on predicting raw pixels from observed data [10, 25, 28, 31]. In the case of raw pixel prediction methods, many recurrent [10, 25, 30, 31] and non-recurrent methods [4, 28, 34] have been suggested. Although most methods are computationally expensive, methods like SIMVP [4] and TAU [28] have produced excellent results with relatively lower computational overhead. SIMVP [4] uses multiple inception blocks to model temporal information and leverages the large receptive field convolutions inside each inception block for accurate temporal predictions. On the other hand, TAU [28] separately models intra-frame and inter-frame attentions to capture correlations within and between frames. In this paper, we propose two encoder-decoder-based architectures, incorporating a middle STC block inspired by SIMVP [4] and TAU [28], that effectively capture temporal correlations between input frames to predict future time frames.

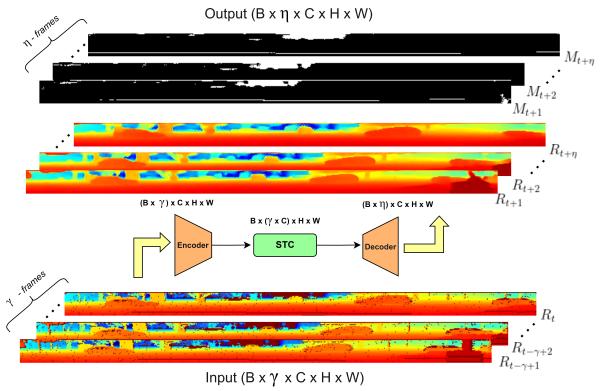


Figure 2: Overview of our proposed architecture.

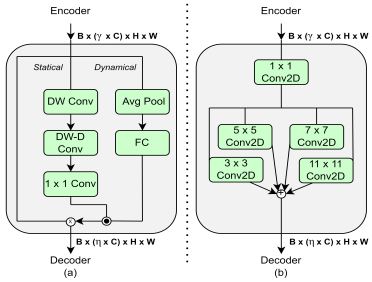


Figure 3: Overview of our STC blocks (a) Temporal Attention Block (Model 2) (b) Inception Block (Model 1).

3 Methodology

We aim to perform full-scale (non-downsampled) point cloud forecasting given a sequence of past point clouds. We propose to exploit recent advancements in convolution-based deep learning architectures for improved spatio-temporal feature extraction for this task.

3.1 Notations & Representation

The primary objective of our approach is to predict a future sequence of point clouds for a temporal window of η timesteps, utilizing observed data from γ timesteps of point clouds as input. Let the initial γ timesteps of the input point-cloud be denoted as $PC = \{P_{t-\gamma+1}, P_{t-\gamma+2}, \ldots, P_t\}$. We adopt range image representation for point cloud data. In order to obtain range image representation, we transform the input point clouds to spherical coordinates, followed by mapping them to image coordinates.

The input point cloud is transformed into spherical coordinates $SC = \{S_{t-\gamma+1}, S_{t-\gamma+2}, \ldots, S_t\}$, where $S_t = [r, \theta, \phi]$ given by Equation 1:

$$\begin{aligned} r &= \sqrt{x^2 + y^2 + z^2} \\ \theta &= \arccos\left(\frac{z}{r}\right) \\ \phi &= -\arctan 2(y, x) \end{aligned} \tag{1}$$

The derived spherical coordinates are then mapped into the image space by converting them to range image $RC = \{R_{t-\gamma+1}, R_{t-\gamma+2}, \ldots, R_t\}$, using Equation 2:

$$\begin{aligned} u &= \left\lfloor \left(\frac{\phi}{2\pi} + 0.5\right) \times W \right\rfloor \\ v &= \left\lfloor \left(1 - \frac{\theta + |\text{fov\_down}|}{\text{fov}}\right) \times H \right\rfloor \\ RC(u, v) &= r, \end{aligned} \tag{2}$$

where fov_down is the field of view of LiDAR in the downward direction and fov is the total field of view of LiDAR. This range image serves as input to our proposed model to predict the subsequent η frames $RC = \{R_{t+1}, R_{t+2}, \ldots, R_{t+\eta}\}$. The predicted frames are then concatenated along the temporal dimension to form a 4D tensor of shape (η, C, H, W) where $C = 1$ and contains the range distance r. The height and width (H, W) of the range image are set up depending on the field of view of the LiDAR sensor and the number of LiDAR lines. Notably, a quantization process occurs during spherical projection, when multiple points get projected onto the same pixel, the nearer point to the sensor is selected; if no points project onto a pixel, its value is set to zero. Although this leads to loss of information, the key advantage of range image representation is that now we can employ CNN-based architectures to efficiently exploit the spatial as well as temporal coherence in the data. The predicted range images are re-projected back into the point-cloud space, resulting in the predicted point cloud $PC = \{P_{t+1}, P_{t+2}, \ldots, P_{t+\eta}\}$. Additionally, the ground truth mask $M = \{M_{t-\gamma+1}, M_{t-\gamma+2}, \ldots, M_t\}$ is constructed by considering all non-zero pixels in the original range image, i.e, $M(u, v) \in \{0, 1\}$. This binary mask helps to retain focus on only those regions/pixels in the range image where we have a valid 3D point projection.

3.2 Proposed Model

We propose an encoder-decoder architecture with a Spatio Temporal Convolution (STC) block in the middle for computing the spatio-temporal aggregation of features in the latent space, to predict future range image frames as outlined in Figure 2. The encoder extracts multi-scale features in individual frames, while the STC block performs convolutions across time (i.e., across γxC channels) to capture temporal information. Finally, the decoder predicts the future range images.

Encoder Architecture: The encoder consists of multiple convolution blocks alternating between one and two strides. Each convolution block contains a 3x3 2D-convolution layer followed by a group normalization and a Leaky RELU [13] layer. The input to the encoder is a batch of 4D tensor (B, γ, C, H, W) where $C = 1$. The convolution happens across the spatial dimension $(B \times \gamma \times C \times H \times W)$, decomposing input tensors into hidden feature maps of shape $(B, \gamma, \hat{C}, \hat{H}, \hat{W})$, before passing on to the STC block.

STC Block: We use two different design choices for STC blocks inspired by SIMVP [4] and TAU [28] yielding two model names, *Model 1* and *Model 2* as shown in the Figure 3. The STC block performs spatial-temporal convoluting across $\gamma \times \hat{C}$ channels on $(\hat{H}, \hat{W})$.

The first design choice for the STC block (in Model 1) consists of multiple InceptionNet [27] blocks. The benefit of having multiple

InceptionNet blocks is the use of large receptive fields to model the large temporal sequences. Each InceptionNet block (Figure 3) consists of multiple parallel convolution blocks of 3x3, 5x5, 7x7 and 11x11 kernels. The larger kernels help in retrieving global motion (movement of the sensor) while the smaller kernels capture more local motions (movement of other vehicles). The inception net follows a multi-branch similar to U-Net [23] like architecture which helps in retrieving both global and local information. The input hidden representation is passed through multiple blocks from top to bottom without reductions of spatial dimension and the hidden information then moves from bottom to top. The module also contains skip connections similar to U-Net [23].

The second design choice for STC blocks (in Model 2) employs multiple temporal attention blocks. Each temporal attention block (Figure 3) decomposes the incoming channels into intra-frame static attentions and inter-frame dynamic attention units, using the learned attention weights to reweigh the inputs. The static attention consists of multiple depth-wise convolutions to model the large receptive field across time frames, while the dynamic attention block learns attention through a squeeze and excitation network [6]. The decomposition is important to capture inter and intra-frame correlations to produce more effective predictions.

Decoder Block: Both the STC modules return the same hidden dimension as its input $(B, \eta, \hat{C}, \hat{H}, \hat{W})$ which is then passed through the Decoder module to reconstruct the predicted range image. The Decoder method is similar to Encoder except it uses transpose convolution blocks instead to reconstruct the range image. Each decoder convolution block contains Convolution transpose followed by group normalization and leaky RELU [13]. The final convolution predicts two channels (B, η, C, H, W) where $C = 2$ which include the range image and its corresponding mask.

3.3 Loss Functions

We predict both future η range image frames and their corresponding masks. The masks are used to remove the unfounded predicted points, thereby avoiding regions corresponding to the sky, and the empty void pixels created by spherical projection. For optimization, we use three different types of losses. The range image loss L_{RV} is $L1$ loss calculated across all masked pixels between the ground truth and the masked predicted frame given by Equation 3:

$$L_{\text{RV}} = \frac{1}{\eta} \sum_{i=t+1}^{t+\eta} \left\| R_i - \hat{R}_i \right\|_1, \tag{3}$$

where R_i is the ground truth range image and $\hat{R}_i$ is the predicted range image. The mask loss L_{mask} is calculated with a Binary Cross Entropy (BCE) loss between predicted masks and ground truth mask values given by Equation 4

$$L_{\text{mask}} = \frac{1}{\eta HW} \sum_{s=t+1}^{t+\eta} \sum_{i=0}^{H-1} \sum_{j=0}^{W-1} M_s^{(i,j)} log(\hat{M}_s^{(i,j)}) + (1 - M_s^{(i,j)}) log(1 - \hat{M}_s^{(i,j)}), \tag{4}$$

where $M_s^{(i,j)}$ are the ground truth masks pixels and $\hat{M}_s^{(i,j)}$ are the predicted mask pixels. Finally, the model is fine-tuned on Chamfer distance loss [2] L_{CD} after the masked range image is reprojected back to point cloud space. The chamfer loss is given by Equation 5 :

$$L_{\text{CD}} = \frac{1}{\eta} \sum_{i=t+1}^{t+\eta} \left[\frac{1}{2} \left(\frac{1}{|P_i|} \sum_{p_i \in P_i} \min_{\hat{p}_i \in \hat{P}_i} \|p_i - \hat{p}_i\|_2 + \frac{1}{|\hat{P}_i|} \sum_{\hat{p}_i \in \hat{P}_i} \min_{p_i \in P_i} \|\hat{p}_i - p_i\|_2 \right) \right], \tag{5}$$

where P_i are the ground truth point cloud and $\hat{P}_i$ is the predicted masked point cloud. One thing to note here is that due to the slow parallelism of the algorithm, the calculation of chamfer distance is quite slow. Hence we only use chamfer distance to fine-tune on a small number of epochs to improve the results of our model. The total net loss is given by Equation 6 :

$$L_{\text{Total}} = w_{\text{RV}} \cdot L_{\text{RV}} + w_{\text{M}} \cdot L_{\text{M}} + w_{\text{CD}} \cdot L_{\text{CD}}, \tag{6}$$

where w_{RV}, w_{mask} and w_{CD} are the weights associated with each loss. We have observed that training initially with range view and mask loss provides a good initialization. While further fine-tuning on chamfer distance loss [2] provides a better 3D awareness, attending to the 3D points.

4 Experiments and Discussions

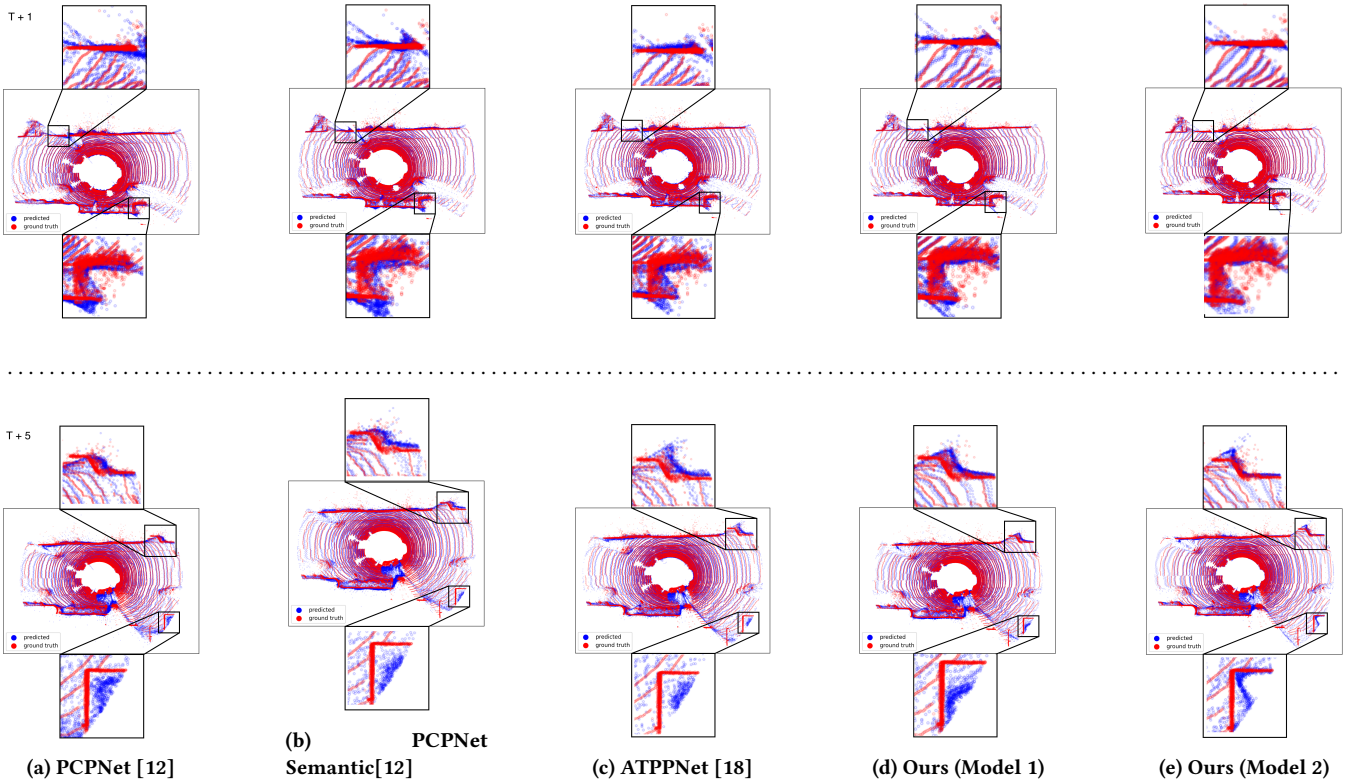
4.1 Dataset

For the quantitative analysis, we compare our results on two publicly available datasets, KITTI [5] and NuScenes [1]. For qualitative analysis, we provide bird-eye view (BEV) and range view (RV) images derived from both KITTI [5] and NuScenes [1] datasets. Additionally, we also test the robustness of our model on a downstream task of localization using [29].

4.2 Experimental Setup

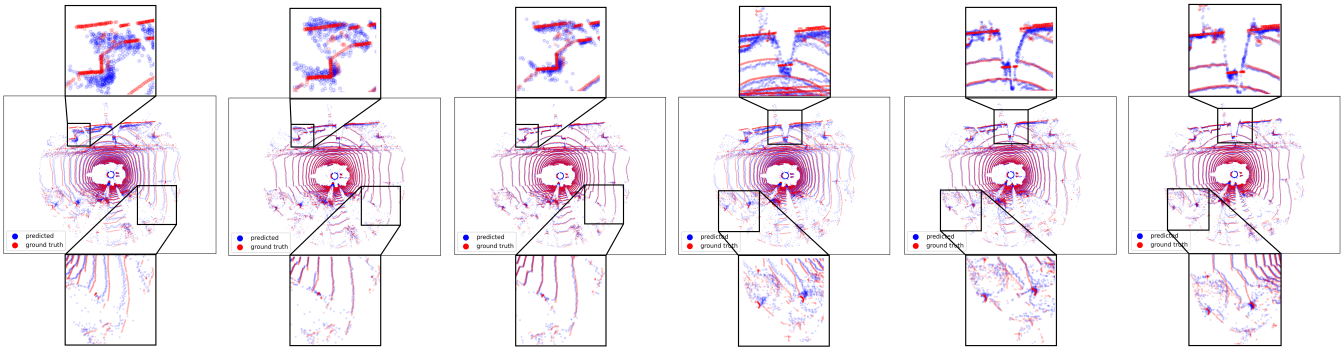
We use the observed point clouds from past $\gamma = 5$ timesteps as input and predicted the future $\eta = 5$ timesteps' point clouds as output. The models were initially trained for 100 epochs using only L_{RV} and L_{mask}, followed by fine-tuning with L_{CD} for an additional 10 epochs. The training was performed using the Adam optimizer [8] along with an exponential decay scheduler. The experiments were executed on two Nvidia RTX A30 GPUs, with each model requiring approximately 100 hours to train.

While comparing with SOTA, we re-train the nearest baseline ATPPNet [18] in the same experimental settings as used for training our models. However, for other models [3, 11, 12, 16] we relied on the available numbers in the existing literatures. We compare our model with four range image-based point cloud forecasting methods, namely TCNet [16], PCPNet [12], PCPNet Semantic [12] and ATPPNet [18] while conducting quantitative analysis. Furthermore, we also compare our results with the point-based methods (PointLSTM [3] and MoNet [11]) on a downsampled KITTI [5] point cloud of 65536 points similar to the evaluation conducted by [18]. For more details about our experimental setup, please refer to our *supplementary material.*



(a) PCPNet [12] (b) PCPNet Semantic[12] (c) ATPPNet [18] (d) Ours (Model 1) (e) Ours (Model 2)

Figure 4: Qualitative results on KITTI dataset [5].



(a) ATPPNet T1 [18] (b) Ours (Model 1) T1 (c) Ours (Model 2) T1 (d) ATPPNet T5 [18] (e) Ours (Model 1) T5 (f) Ours (Model 2) T5

Figure 5: Qualitative results on NuScenes dataset [1].

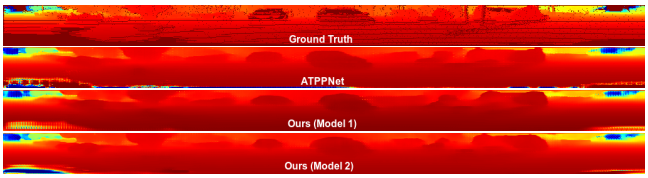


Figure 6: Qualitative visualization of the predicted range image on KITTI dataset [5] (color coding: Red closer depth & Blue is farther depth).

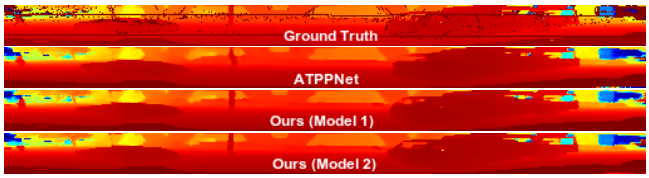


Figure 7: Qualitative visualization of the predicted range image on NuScenes dataset [1] (color coding: Red closer depth & Blue is farther depth).

Table 1: Quantitative comparison on KITTI dataset [5].

Predicted timestep	Chamfer Distance Loss ↓						Range Image Loss ↓					
	TCNet [16]	PCPNet [12]	PCPNet Semantic [12]	ATPPNet [18]	Model 1	Model 2	TCNet [16]	PCPNet [12]	PCPNet Semantic [12]	ATPPNet [18]	Model 1	Model 2
$T+1$	0.2530	0.2520	0.2420	0.2527	0.2228	**0.1940**	0.5540	0.5430	0.5030	0.4437	0.4285	**0.3951**
$T+2$	0.3090	0.3010	0.2980	0.3886	0.2756	**0.2401**	0.6710	0.6620	0.6200	0.5544	0.5350	**0.4854**
$T+3$	0.3770	0.3620	0.3540	0.3653	0.3347	**0.2876**	0.7790	0.7730	0.7270	0.6539	0.6369	**0.5754**
$T+4$	0.4480	0.4350	0.4270	**0.3179**	0.3994	0.3480	0.8780	0.8720	0.8250	0.7486	0.7281	**0.6622**
$T+5$	0.5470	0.5140	0.5030	0.4722	0.4754	**0.4171**	0.9740	0.9730	0.9200	0.8426	0.8177	**0.7548**
Mean	0.3870	0.3730	0.3650	0.3594	0.3415	**0.2974**	0.7710	0.7650	0.7190	0.6486	0.6293	**0.5746**

Table 2: Quantitative comparison on NuScenes dataset [1].

Predicted timestep	Chamfer Distance Loss ↓			Range Image Loss ↓		
	ATPPNet [18]	Model 1	Model 2	ATPPNet [18]	Model 1	Model 2
$T+1$	0.5576	0.5104	**0.4696**	0.4851	0.4319	**0.3913**
$T+2$	0.7052	0.6353	**0.6164**	0.5387	0.4773	**0.4428**
$T+3$	0.6618	0.6103	**0.5837**	0.5998	0.5283	**0.5033**
$T+4$	0.6292	0.5680	**0.5362**	0.6610	0.5799	**0.5612**
$T+5$	0.7695	0.6986	**0.6921**	0.7197	0.6292	**0.6160**
Mean	0.6647	0.6045	**0.5796**	0.6009	0.5293	**0.5029**

4.3 Qualitative Results

We render both (color-coded) range images as well as birds-eye-view (BEV) of 3D point clouds for visualizing the performance of methods.

In Figure 6, we show the color-coded 2D rendering of range images for a specific frame from a sequence in the KITTI dataset. We can observe that in comparison to ATPPNet [18], both our models (Model 1 and Model 2) yield sharper results consisting of better high-frequency details in the scene. While comparing Model 1 and Model 2, the latter produces much more accurate details with higher edge-preserving qualities. The pattern remains consistent when we observe range images in the NuScenes dataset [1] in Figure 7. Although these images are more blurry (due less number of LiDAR sensing channels) compared to their KITTI [5] counterparts, the results from Model 1 and Model 2 appear sharper than those produced by ATPPNet [18].

Figure 4 and Figure 5 show the visualization of birds-eye-view (BEV) images of predicted 3D point clouds on KITTI [5] and Nuscenes [1] datasets, respectively. We can infer superior quality of point cloud forecasting using our models in comparison to SOTA method (ATPPNet) with significantly less noise while capturing finer details with better edge preserving qualities.

While comparing Model 1 and Model 2, we again observe better point cloud prediction for latter compared to former when comparing dense point cloud datasets such as KITTI [5] (shown in Figure 4). This disparity in performance can be attributed to attention mechanism in Model 2 enabling it to capture long-range spatio-temporal dependency. A similar trend is observed when comparing on sparser datasets such as Nuscenes [1] (shown in Figure 5), where we observe Model 2 to slightly outperform Model 1. Nevertheless, both our models consistently outperform ATPPNet [18].

4.4 Quantitative Results

We can infer from Table 1 that, our models achieved superior performance over best performing SOTA method i.e., ATPPNet [18] with an overall reduction of 4.98% & 17.20% in the mean Chamfer Distance and a decrease of 2.98% & 11.41% in the mean range image loss on the KITTI dataset for proposed Model 1 and Model 2, respectively. In the case of the Nuscenes dataset, as reported in Table 2, we observe an overall decrease of 9.04% & 12.79% in the mean chamfer distance prediction and a decrease of 11.92% & 16.31% in the mean range image loss for Model 1 and Model 2, respectively when compared to ATPPNet [18].

In the case of downsampled point cloud, as reported in Table 3 we notice a decrease of 7.04% in the mean chamfer distance for our Model 2, however, we notice a slight increase of 6.71% Model 1 when compared to the nearest baseline [18].

Additionally, we provide quantitative comparison of the model size and inference time of our method with existing methods and reported results in Table 4. We attained an average inference time (per frame) of 53.65ms and 19.53ms for Model 1 and Model 2 respectively which in comparison to ATPPNet, is a speedup of 1.95x and 5.37x respectively when compared using KITTI [5] dataset. In regard to model size, our main Model 2 is also much smaller in comparison to ATPPNet being almost half in size, however, Model 1 is slightly bigger in terms of model size. Nevertheless, it is important to note that faster inference is of critical importance in the case of autonomous driving scenarios.

4.5 Effect on Downstream Tasks

F-LOAM [29] (Fast LiDAR Odometry and Mapping) is a real-time method for Odometry and Mapping using range measurements. We adopt F-LOAM [29] to estimate the motion of the vehicles using the ground truth and the predicted point cloud from our method as well as one of our baseline [18]. We try the experiment on sequence 10 of KITTI [5] dataset. We perform two sets of experiments. In the

Table 3: Quantitative Ccomparison with Baselines on downsampled KITTI Point Clouds [5].

	Chamfer Distance ↓							
Predicted timestep	PointLSTM [3]	MoNet [11]	TCNet [16]	PCPNet [12]	PCPNet Semantic [12]	ATPPNet [18]	Model 1	Model 2
$T+1$	0.3320	0.2780	0.2900	0.2771	0.2725	0.2219	0.2524	**0.2201**
$T+2$	0.5610	0.4090	0.3570	0.3977	0.3987	0.2821	0.3114	**0.2715**
$T+3$	0.8100	0.5490	0.4410	0.3827	0.3870	0.3526	0.3760	**0.3239**
$T+4$	1.0540	0.6920	0.5220	0.3327	0.3295	0.4284	0.4490	**0.3898**
$T+5$	1.2990	0.8420	0.6290	0.4760	0.4831	0.5119	0.5285	**0.4651**
Mean	0.8110	0.5540	0.4480	0.4220	0.3742	0.3594	0.3835	**0.3341**

Table 4: Comparison of models with total parameters and inference time on KITTI [5] and Nuscenes [1] Datasets.

	Total Parameters	Inference time (KITTI)	Inference time (Nuscenes)
TCNet [16]	17.0M	46.28ms	-
PCPNet [12]	22.6M	**18.59ms**	-
ATPPNet [18]	11.0M	104.85ms	82.33ms
Model 1	13.6M	53.65ms	17.20ms
Model 2	**5.5M**	19.53ms	**9.64ms**

Table 5: F-LOAM [29] pose error.

	LOAM Pose Error ↓		
Predicted timestep	ATPPNet [18]	Model 1	Model 2
$T+1$	0.0981	0.0873	**0.0806**
$T+2$	0.1404	0.1126	**0.0998**
$T+3$	0.1930	0.1476	**0.1260**
$T+4$	0.2537	0.1873	**0.1590**
$T+5$	0.3217	0.2302	**0.1971**
Mean	0.2014	0.1530	**0.1325**

first one we used a temporal window of 10 point clouds (5 observed and 5 predicted) from ATPPNet [18], Model 1, and Model 2 and calculate the pose error as given by : $L_{loam} = ||p_T - \hat{p}_T||_2^2$, where $\hat{p}_T$ is the motion estimate from the predicted point cloud and p_T is the motion estimate from the ground truth point cloud. The results obtained are reported in Table 5. We notice a sharp improvement in localization using predicted point clouds from our methods (Model 1 & Model 2) resulting in an improvement of 24% and 34.2%, over the mean F-LOAM disparity score, respectively.

As part of the second experiment, we aim to perform a qualitative analysis by recovering a continuous trajectory of the vehicle by considering predictions from predicted point clouds instead of partially observed ones. We first predict future point clouds from an observed sequence with a temporal window of 5 timesteps for the **entire sequence ten** of KITTI dataset [5]. These predictions were then concatenated into continuous sequences for $T+1$ and $T+5$ time steps essentially obtaining a temporal window of 1192 frames, covering almost the entire sequence ten of the KITTI [5] dataset. We then employed F-LOAM [29] to predict the ego vehicle's trajectory from the sequences of predicted point clouds ($T+1$ and $T+5$) and compared these predictions with the ground truth trajectory predicted using the ground truth point clouds. A visualization of ego trajectory in Figure 8 demonstrates a significant improvement in trajectory prediction for Models 1 and 2 as compared to ATPPNet [18], particularly at $T+5$. Both of these experiments demonstrate superior performance of our method for downstream motion estimation task.

5 Ablation Studies

In this section, we present a detailed analysis of each component of our model and also highlight some of the rationale behind the design choices that we have adopted.

Impact of loss functions: We perform ablative experiment by removing L_{CD}, L_{RV} and L_{mask} and report results in Table 6. The results indicate deterioration in performance while removing any of the loss function (i.e., increase in chamfer distance values). Specifically, we observe higher noise due to the removal of L_{CD} in the results. On the other hand, removing L_{mask} results in points being inpainted in the empty pixels of range image which increases redundant points in the projected point clouds and consequently increase the overall chamfer distance score. However, interestingly the removal of L_{CD} leads to better range image prediction since fine-tuning the model on L_{CD} leads to the model becoming more 3D aware which affects its corresponding 2D predictions. For relevant qualitative evaluation refer to our *supplementary material: Figure S1 and S2.*

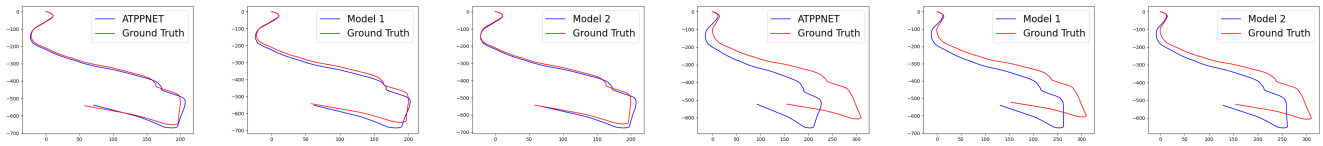
Effect on sequence length: In Table 7, we report the results by varying the temporal window size (input and output number of frames vary together). We observe that by using fewer frames our models yield better forecasting for short duration as listed in Table 7. However, increasing the window size results in an overall performance drop due to the higher uncertainty associated with future states as well as higher variance in spatio-temporal dimensions. Thus, we adopt a window size of 5 for reporting results as it strikes the perfect balance between accuracy and long-term forecasting.

Table 6: Results of ablation study on the impact of different losses in the model.

	Chamfer Distance ↓						Range View Loss ↓					
	Original		without L_{CD}		without L_{mask}		Original		without L_{CD}		without L_{mask}	
Predicted timestep	Model 1	Model 2	Model 1	Model 2	Model 1	Model 2	Model 1	Model 2	Model 1	Model 2	Model 1	Model 2
$T+1$	0.2228	0.1940	0.3547	0.3160	0.3065	0.2685	0.4285	0.3951	0.4120	0.3811	0.4586	0.4231
$T+2$	0.2756	0.2401	0.5096	0.4425	0.4091	0.3498	0.5350	0.4854	0.5130	0.4700	0.5590	0.5133
$T+3$	0.3347	0.2876	0.6729	0.5779	0.3912	0.3464	0.6369	0.5754	0.6106	0.5591	0.6572	0.6039
$T+4$	0.3994	0.3480	0.8410	0.7399	0.3535	0.3071	0.7281	0.6622	0.7013	0.6452	0.7494	0.6917
$T+5$	0.4754	0.4171	1.0356	0.9199	0.4681	0.4100	0.8177	0.7548	0.7935	0.7343	0.8442	0.7835
Mean	0.3415	0.2974	0.6828	0.5993	0.3857	0.3364	0.6293	0.5746	0.6061	0.5579	0.6537	0.6031

Table 7: Results of ablation study on performance changes on varying window size.

	Chamfer Distance ↓						Range View Loss ↓					
	3		5		7		3		5		7	
Predicted timestep	Model 1	Model 2	Model 1	Model 2	Model 1	Model 2	Model 1	Model 2	Model 1	Model 2	Model 1	Model 2
$T+1$	0.2341	0.2023	0.2228	0.1940	0.2729	0.2281	0.4049	0.3645	0.4285	0.3951	0.4566	0.4242
$T+2$	0.2669	0.2289	0.2756	0.2401	0.3971	0.3307	0.5128	0.4606	0.5350	0.4854	0.5658	0.5176
$T+3$	0.2970	0.2537	0.3347	0.2876	0.5387	0.4652	0.6307	0.5683	0.6369	0.5754	0.6621	0.6081
$T+4$	-	-	0.3994	0.3480	0.4583	0.4057	-	-	0.7281	0.6622	0.7447	0.6908
$T+5$	-	-	0.4754	0.4171	0.3343	0.2761	-	-	0.8177	0.7548	0.8220	0.7708
$T+6$	-	-	-	-	0.4704	0.3946	-	-	-	-	0.9063	0.8534
$T+7$	-	-	-	-	0.6346	0.5571	-	-	-	-	0.9986	0.9460
Mean	0.2660	0.2283	0.3415	0.2974	0.4438	0.3797	0.5161	0.4645	0.6293	0.5746	0.7366	0.6873



(a) ATPPNet T1 [18] **(b) Ours (Model 1) T1** **(c) Ours (Model 2) T1** **(d) ATPPNet T5 [18]** **(e) Ours (Model 1) T5** **(f) Ours (Model 2) T5**

Figure 8: Trajectory prediction using F-LOAM [29] on KITTI Dataset [5].

6 Conclusions and Future Work

Our approach achieves high-fidelity point-cloud prediction with significantly faster inference than the original baselines. However, it suffers from excessive blurring and loss of high-frequency details. Incorporating more scene context, such as fusing RGB images from cameras or fusing the semantics [12], could address these issues. Along with that incorporating additional extrinsic information, such as data from IMU sensors, could further improve the accuracy of point cloud forecasting. Additionally, refining the loss functions, possibly through weak supervision from a discriminator [14] or gradient-based losses [19], could help preserve high-frequency details. Finally, we believe that constructing a global canonical structure, from which future steps can be derived, as explored in [7], offers another interesting direction for future research.

References

[1] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. 2020. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11621–11631.

[2] Haoqiang Fan, Hao Su, and Leonidas J Guibas. 2017. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 605–613.

[3] Hehe Fan and Yi Yang. 2019. PointRNN: Point recurrent neural network for moving point cloud processing. *arXiv preprint arXiv:1910.08287* (2019).

[4] Zhangyang Gao, Cheng Tan, Lirong Wu, and Stan Z. Li. 2022. SimVP: Simpler Yet Better Video Prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 3170–3180.

[5] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*. 3354–3361. https://doi.org/10.1109/CVPR.2012.6248074

[6] Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-Excitation Networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7132–7141. https://doi.org/10.1109/CVPR.2018.00745

[7] Tarasha Khurana, Peiyun Hu, David Held, and Deva Ramanan. 2023. Point cloud forecasting as a proxy for 4d occupancy forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1116–1124.

[8] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014). https://api.semanticscholar.org/CorpusID:6628106

[9] Xingyu Liu, Charles R. Qi, and Leonidas J. Guibas. 2019. FlowNet3D: Learning Scene Flow in 3D Point Clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[10] William Lotter, Gabriel Kreiman, and David Cox. 2016. Deep predictive coding networks for video prediction and unsupervised learning. *arXiv preprint arXiv:1605.08104* (2016).

[11] Fan Lu, Guang Chen, Zhijun Li, Lijun Zhang, Yinlong Liu, Sanqing Qu, and Alois Knoll. 2021. Monet: Motion-based point cloud prediction network. *IEEE Transactions on Intelligent Transportation Systems* 23, 8 (2021), 13794–13804.

[12] Zhen Luo, Junyi Ma, Zijie Zhou, and Guangming Xiong. 2023. PCPNet: An Efficient and Semantic-Enhanced Transformer Network for Point Cloud Prediction. *IEEE Robotics and Automation Letters* 8, 7 (2023), 4267–4274. https://doi.org/10.1109/LRA.2023.3281937

[13] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. In *Proc. ICML*, Vol. 30. 3.

[14] Michaël Mathieu, Camille Couprie, and Yann LeCun. 2015. Deep multi-scale video prediction beyond mean square error. *CoRR* abs/1511.05440 (2015). https://api.semanticscholar.org/CorpusID:205514

[15] Daniel Maturana and Sebastian Scherer. 2015. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 922–928.

[16] Benedikt Mersch, Xieyuanli Chen, Jens Behley, and Cyrill Stachniss. 2022. Self-supervised point cloud prediction using 3d spatio-temporal convolutional networks. In *Conference on Robot Learning*. PMLR, 1444–1454.

[17] Mohd Omama, Sripada V. S. Sundar, Sandeep Chinchali, Arun Kumar Singh, and K. Madhava Krishna. 2022. Drift Reduced Navigation with Deep Explainable Features. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 6316–6323. https://doi.org/10.1109/IROS47612.2022.9981330

[18] Kaustab Pal, Aditya Sharma, Avinash Sharma, and K Madhava Krishna. 2024. ATPPNet: Attention based Temporal Point cloud Prediction Network. *arXiv preprint arXiv:2401.17399* (2024).

[19] Sylvain Paris, Samuel W Hasinoff, and Jan Kautz. 2011. Local Laplacian filters: Edge-aware image processing with a Laplacian pyramid. *ACM Trans. Graph.* 30, 4 (2011), 68.

[20] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 652–660.

[21] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems* 30 (2017).

[22] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. 2017. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3577–3586.

[23] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi (Eds.). Springer International Publishing, Cham, 234–241.

[24] Christian Schuldt, Ivan Laptev, and Barbara Caputo. 2004. Recognizing human actions: a local SVM approach. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, Vol. 3. IEEE, 32–36.

[25] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems* 28 (2015).

[26] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. 2015. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*. 945–953.

[27] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, D. Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2014. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014), 1–9. https://api.semanticscholar.org/CorpusID:206592484

[28] Cheng Tan, Zhangyang Gao, Lirong Wu, Yongjie Xu, Jun Xia, Siyuan Li, and Stan Z Li. 2023. Temporal attention unit: Towards efficient spatiotemporal predictive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 18770–18782.

[29] Han Wang, Chen Wang, Chun-Lin Chen, and Lihua Xie. 2021. F-loam: Fast lidar odometry and mapping. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 4390–4396.

[30] Yunbo Wang, Lu Jiang, Ming-Hsuan Yang, Li-Jia Li, Mingsheng Long, and Li Fei-Fei. 2018. Eidetic 3D LSTM: A model for video prediction and beyond. In *International conference on learning representations*.

[31] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu. 2017. Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. *Advances in neural information processing systems* 30 (2017).

[32] Xinshuo Weng, Junyu Nan, Kuan-Hui Lee, Rowan McAllister, Adrien Gaidon, Nicholas Rhinehart, and Kris M Kitani. 2022. S2net: Stochastic sequential pointcloud forecasting. In *European Conference on Computer Vision*. Springer, 549–564.

[33] Xinshuo Weng, Jianren Wang, Sergey Levine, Kris Kitani, and Nicholas Rhinehart. 2021. Inverting the Pose Forecasting Pipeline with SPF2: Sequential Pointcloud Forecasting for Sequential Pose Forecasting. In *Proceedings of the 2020 Conference on Robot Learning (Proceedings of Machine Learning Research, Vol. 155)*, Jens Kober, Fabio Ramos, and Claire Tomlin (Eds.). PMLR, 11–20. https://proceedings.mlr.press/v155/weng21a.html

[34] Ziru Xu, Yunbo Wang, Mingsheng Long, Jianmin Wang, and M KLiss. 2018. PredCNN: Predictive Learning with Cascade Convolutions.. In *IJCAI*. 2940–2947.

[35] Ze Yang and Liwei Wang. 2019. Learning relationships for multi-view 3D object recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*. 7505–7514.

[36] Tan Yu, Jingjing Meng, and Junsong Yuan. 2018. Multi-view harmonized bilinear network for 3d object recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 186–194.

[37] Mingliang Zhai, Xuezhi Xiang, Ning Lv, and Xiangdong Kong. 2021. Optical flow and scene flow estimation: A survey. *Pattern Recognition* 114 (2021), 107861. https://doi.org/10.1016/j.patcog.2021.107861