GEOMETRY-BASED VIDEO PREDICTION WITH CAMERA MOTION FOR MOBILE ROBOTICS

BY

WEIHANG LIANG

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois Urbana-Champaign, 2022

Urbana, Illinois

Adviser:

Assistant Professor Katherine Driggs-Campbell

ABSTRACT

Video prediction is one of the fundamental research problems in computer vision, and it has a wide range of applications in planning and control for robotics. Recent learning-based approaches show promising results on various video datasets, and some have seen successful applications in planning robot arm motion. However, predicting future observations in a sequence given a set of past images remains a challenging task in mobile robotics, especially when the camera is in motion.

Early works in this area use deterministic approaches, which often yield visually unintuitive results due to the intrinsic variability of motion in the future. More recent works have adopted stochastic models to generate sharper future frames. However, most methods do not account for camera motion and perform poorly in scenarios with moving cameras when they are deployed on vehicles and mobile robots.

To solve the challenging task of video prediction on mobile platforms, we propose a geometry-based prediction framework that combines visual odometry prediction and view synthesis. Based on a sequence of observed frames, our method first predicts future camera poses and extracts the 3D geometry of the world, which are then jointly used to generate predicted future frames. Specifically, we train a recurrent visual odometry prediction model conditioned on raw RGB images with ground-truth pose labels. In addition, we train SynSin, a view synthesis method to generate 2D images from novel viewpoints using a 3D world representation. By combining these approaches, we demonstrate that our hierarchical deterministic approach outperforms previous stochastic works on the KITTI dataset.

To my parents, for their love and support.

ACKNOWLEDGMENTS

I would like to thank Professor Katherine Driggs-Campbell for all the support she gave me as my adviser. In 2019, I took my first class in college about robotics with her, where she taught me the fundamentals and introduced me to the exciting research projects about robotics. In 2020, I joined her lab. During my two years in her lab, she helped me tremendously in research and guided me through the challenges in my projects. Furthermore, she was always supportive and gave me help when I encountered difficulties in life. It would be impossible for me to complete my thesis and other projects without her help.

I would also like to thank Professor Sayan Mitra, Professor Roy Dong, Professor Saurabh Gupta, and Professor Liangyan Gui for their support and guidance in research and class.

I want to thank Neeloy Chakraborty, Pranav Sriram, Yixiao Fang, Zhe Huang, Peixin Chang, and Shuijing Liu for their collaboration and contribution to the IROS submission on video prediction. I want to also thank all my colleagues in the lab for their help and support.

Finally, I want to thank my parents. We have not seen each other for the entirety of my graduate school, but I can always feel their love and support however distant we are physically.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Motivation

Predicting future images in a video sequence conditioned on past images is one of the central problems in computer vision. The ability to predict future image observations of the world has a variety of applications in robotics, including visual planning and control [1, 2, 3, 4], apparent latency reduction [5, 6], and understanding the dynamics of the environment [7].

Early works in this field use deterministic models and have difficulty predicting sharp future frames due to the intrinsic stochasticity of motions of the objects in the images [2, 8]. More recently, stochastic frameworks encode the randomness in motion by learning a distribution of possible future images to model this uncertainty and sample from this distribution at test time to generate sharper images [9, 10, 11, 12].

However, most methods primarily focus on videos with static backgrounds. Thus, they fail to generate sharp predictions on long-term sequences collected with a moving camera. To handle camera motion, some works use

Figure 1.1: **One Predicted Sequence from Our Method.**
Ground-truth frames from the KITTI dataset [13] are compared against predicted frames from our approach.

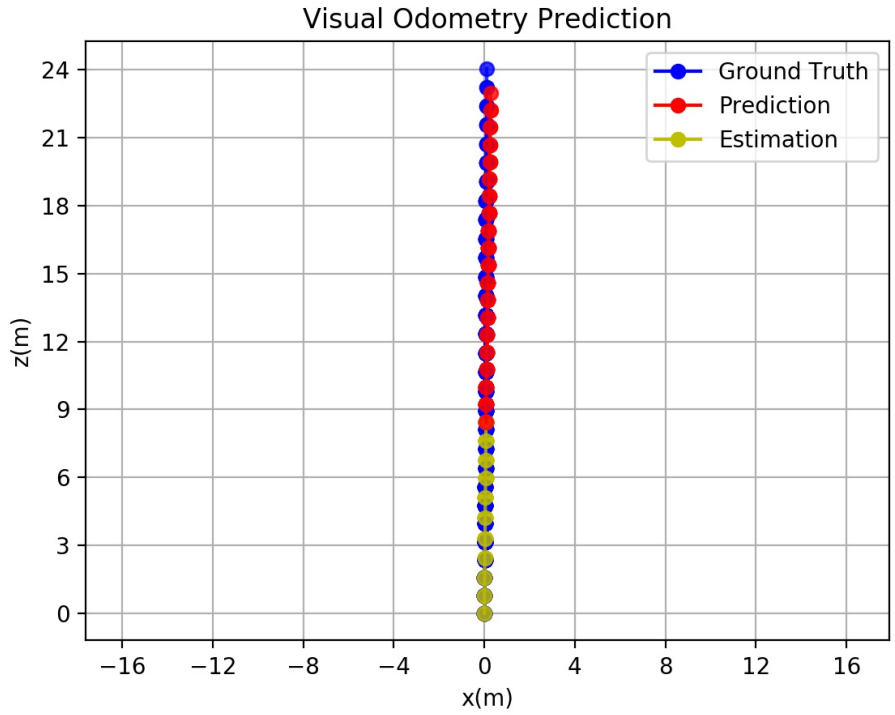

Figure 1.2: **One Predicted Pose Sequence from Our Method.** Ground-truth path of the camera is compared against the predicted path from our pose predictor.

autoregressive networks to model 2D optical flow in the video sequence and recursively generate future images from predictions, but only show marginal improvements [12, 14]. This problem arises from both the incapability of 2D dynamic models to capture 3D motion and the error accumulation in the autoregressive prediction process.

1.2 Contributions

In this thesis, to tackle video prediction with a moving camera on mobile robots, we propose a geometry-based approach with monocular visual odometry prediction and view synthesis. Contrary to past works that only model 2D dynamics [12, 14], we model the dynamics of the video as a camera moving in the 3D world. We estimate the camera poses of observed frames using an extension of DeepVO [15]. The encoded context in DeepVO is used to predict the future motion of the camera. Given the predicted camera poses,

we extract the 3D geometry of the world using SynSin [16] and generate the 2D prediction based on the new view position. The camera pose predictor and the view generator are trained separately using ground-truth poses and are combined together in testing. This two-stage training prevents the gradients of two tasks from distracting each other and thus improves the stability of training.

Our model generates sharper images in long sequences compared to state-of-the-art stochastic video prediction methods for two reasons. First, we explicitly model the 3D geometry and camera motion rather than the 2D pixel changes and optical flow induced by such motion. This 3D representation of the world captures the underlying dynamics more accurately than its 2D counterparts. Second, we mitigate error accumulation in the image space by replacing the autoregressive image prediction process with a hierarchical prediction approach. We predict high-level camera poses and use only a single 3D representation of the world extracted from the last observed frame to generate the future frames. In contrast to past autoregressive methods that accumulate errors in the image space during recursive prediction, our method predicts in the camera pose space which is much smaller and adds structures to the input images. Thus, our method is able to generate sharp future frames over long prediction horizons.

We present the following contributions: (1) We propose a novel framework for video prediction in moving camera scenarios. (2) By estimating and predicting the poses of the moving camera, our method captures the dynamics of the videos and is able to generate sharper long-term predictions than state-of-the-art baselines. (3) We demonstrate comparable performance to state-of-the-art baselines on various metrics and outperform them on the LPIPS metric [17].

1.3 Organization of the Thesis

This thesis is organized as follows:

In Chapter 2, we discuss related works of video prediction. We show the challenges of the prediction task and the solutions provided by traditional methods and recent learning-based methods as well as their limitations.

In Chapter 3, we discuss the problem formulation of the video prediction

task and details about our method. We introduce the key components of our prediction framework and how we combine them to solve the challenging task of video prediction with camera motion.

In Chapter 4, we explain our experiments and results. We first introduce the KITTI Visual Odometry Dataset [13], which we use to conduct our experiments. Then, we present the evaluation metrics that we use to assess our results. We introduce the three baselines that we compare our method against. We also explain the implementation details of our method. Finally, we analyze and present our qualitative and quantitative results and show our findings.

In Chapter 5, we conclude our work and discuss the limitations of our methods as well as potential future directions.

CHAPTER 2

RELATED WORK

In this chapter, we discuss work related to the task of video prediction. We present traditional methods which focus on building a 3D model of the environment. In addition, we also show the recent progress in learning-based methods as well as the challenges related to using learning-based approaches in the prediction task. Finally, we discuss the difficulty that comes with camera motion and present different approaches to tackle prediction with camera motion.

2.1 Traditional Methods

Traditional methods for video prediction take advantage of the scene geometry and construct a 3D model of the surroundings to leverage for prediction. Cobzas et al. propose a predictive display system that creates a scene model consisting of a dynamic texture and geometric model [5]. The system then uses the constructed 3D model to render new views at camera poses from the desired motion from the operator. As a result, the method is dependent on future motion input to compose future poses. Brudank uses a simpler model and splits the 3D scene into the ground plane and far plane and predicts them separately [6]. However, as shown in Figure 2.1, the model relies on the assumption that the scene only consists of mostly ground and background that is very far away from the camera (e.g., sky). When there are objects in the environment that is close to the camera, the ground-far-plane assumption no longer holds and will cause visual artifacts in the prediction process. In contrast, our method produces a detailed 3D point cloud for modeling the scene and does not rely on the assumption that the environment is simple.

Figure 2.1: **Examples of Scenarios that Ground-Far-Plane Assumption Holds and Fails.** In the top image, the assumption holds as the highway is flat and there are no vehicles close by. In the bottom image, the assumption fails as there are houses and a van close to the image which exceed two planes.

2.2 Learning-based Methods

As with other progressions in learning-based computer vision, learning-based approaches show promising results in video prediction [18, 19, 20, 21, 22, 23]. However, the prediction results sometimes suffer from degraded quality such as blurriness [24, 25]. Two possible causes of this degradation are the averaging effect over all possible outcomes in the future [26] and compounding errors during recursive pixel-level predictions [27, 28].

2.2.1 Challenges with Averaging Effect

Imagine a car at an intersection with the choice to go left and right and we want to predict the image observation of the car at the next time step from a bird's eye view. If we do not know what action the car will take, then the future image observation is a distribution over the two actions. In this case, the network may predict an average of the two possible outcomes, resulting in a blurred image prediction as shown in Figure 2.2.

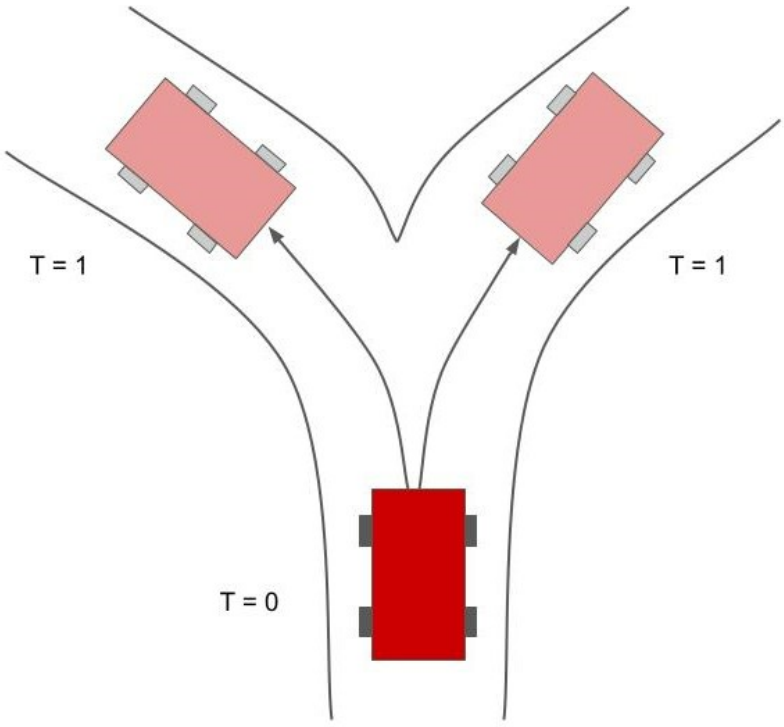

Figure 2.2: **Averaging Effect in Predicting Future Observation of a Car.**

To alleviate the averaging effect, previous work has extensively investigated the effect of stochasticity in video prediction. Modeling stochasticity helps generate sharper predictions than deterministic methods [9, 10]. By choosing one specific possibility at the generation time, the model avoids considering other possible outcomes and thus generates a sharper prediction. Others also use adversarial loss to generate realistic predictions by matching the prediction to real images [4].

2.2.2 Challenges with Compounding Errors

A common practice in multi-step video prediction is to recursively feed previously predicted frames as input in order to predict frames further into the future. Small errors in pixel-level prediction quickly accumulate in this autoregressive process as shown in Figure 2.3.

To prevent compounding errors, some approaches use hierarchical prediction and build a high-level structure on top of the raw pixels [29, 11]. Instead of directly predicting raw pixels, these approaches first convert observed frames into high-level representations and propagate those representations through recursive prediction. In multi-step prediction, these models can propagate within the high-level representation space until they reach the desired time step into the future, where they reconstruct the image by converting back from the propagated high-level representations.

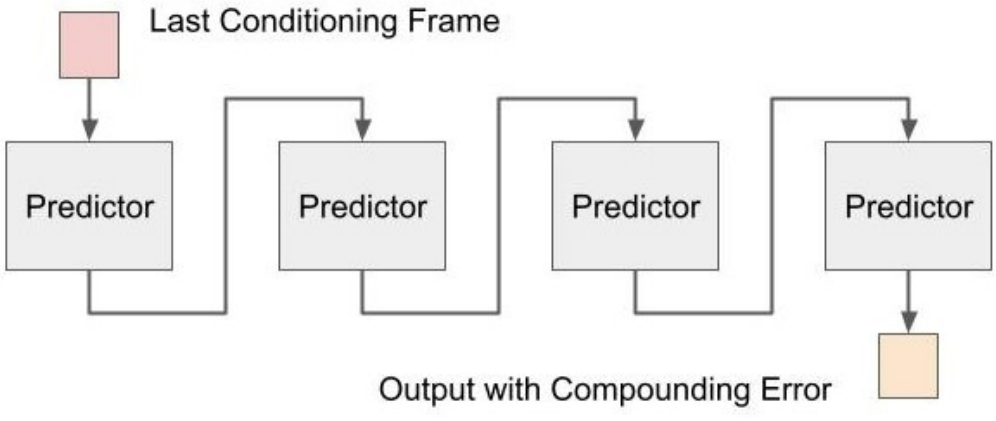

Figure 2.3: **Compounding Errors in Recursive Prediction Process.**

This process avoids the error accumulation in pixel space and thus leads to higher prediction quality. For example, extracting and predicting with human pose helps with predicting video sequences of human actions [29]. Another method by Franceschi et al. decouples frame synthesis from dynamics and predicts future frames through propagating latent states with learned dynamics [11]. This latent model shows advantages in modeling long-term dynamics over non-hierarchical methods.

2.2.3 Challenges with Camera Motion

All the aforementioned approaches have shown decent results on both artificial and real-world datasets [8, 30, 31]. However, the experiments are conducted on datasets with static backgrounds. Moreover, some approaches explicitly assume a static background, and thus cannot generalize to settings with a moving camera [29]. This assumption is especially problematic in mobile robot and autonomous vehicle applications, where there is constant camera motion. In contrast, our method predicts and utilizes the camera pose to generate future frames, which generalizes well to datasets with dynamic backgrounds caused by camera motions.

2.3 Video Prediction with Camera Motion

To overcome the challenges caused by a moving camera, recent methods try to explicitly model optical flow induced by camera motion. Akan et al. propose an extension to pixel-to-pixel stochastic models by adding a component that predicts future optical flow [12]. Then the method combines the pixel prediction and the prediction from optical flow using a learned mask. Sarkar et al. decompose the motion into camera and object motion by separately predicting the velocity and acceleration of the pixels [14]. Both approaches attempt to model the 2D projections of the dynamics and motion without explicitly modeling the 3D world. In contrast, we explicitly model the 3D world from the last observed image and use it to generate predictions.

Another line of work combines learning-based prediction with the geometry of the world. Mahjourian et al. propose an approach that uses the image frame, estimated frame depth, and ground-truth camera trajectory

Figure 2.4: **Optical Flow Caused by Camera Motion.** The camera is attached to a vehicle driving forward on the road. Pixels move toward the camera as shown by the arrows.

to render new predictions [32]. Such geometry-based approaches generate sharper images than pixel-to-pixel prediction models but also possess flaws. One of the biggest challenges that geometry-based methods face is imagining the unseen areas of the world, and the approach leaves those areas blank when they become disoccluded in the predicted images. Furthermore, another limitation of previous geometry-based approaches is that they rely on additional ground-truth information about future camera poses at test time. Such information is not available in the real world. In contrast, our method does not require ground-truth future poses of the camera at test time, and our view synthesis model is able to inpaint unseen parts of the world.

Finally, there is a concurrent work dealing with geometry-based video prediction using view synthesis [33]. While this work similarly combines view synthesis and pose prediction to generate future frames, we use a novel pose prediction module consisting of a state-of-the-art visual odometry model and a decoder Long Short-Term Memory (LSTM) network. Our method can condition on long-horizon, variable-length observations with our recurrent framework, while the concurrent work conditions its poses on a short, fixed sequence of input frames. Moreover, our pose predictor module also benefits from an auxiliary loss from the poses of conditioning frames in the training process.

CHAPTER 3

METHODOLOGY

In this chapter, we first formulate the video prediction problem. We then describe fundamental components of our proposed architecture including visual odometry, future camera pose prediction, and view synthesis. Finally, we present our combined hierarchical framework for geometry-based video prediction to solve the challenges that come with camera motion.

3.1 Problem Formulation

In the general video prediction problem, given a sequence of video frames $x_{1:t-1}$ from time 1 to $t-1$, we want to predict the next $T+1$ frames $\hat{x}_{t:t+T}$. In our work, we adopt a hierarchical approach towards the prediction problem. Instead of directly predicting the image frames $x_{t-1:t+T}$, we first predict the future relative camera transformations between frames, which are denoted as:

$$\hat{p}_{t:t+T} \triangleq \left\{ \hat{P}_i^{i+1} \right\}_{i=t}^{t+T-1}, \tag{3.1}$$

where $\hat{P}_i^{i+1}$ is the predicted transformation from frame x_i to frame x_{i+1}. Similarly, we denote the ground-truth relative camera transformations between all frames as $p_{1:t+T}$. We utilize the predicted relative transformations $\hat{p}_{t:t+T}$ along with the 3D geometry extracted from the last observed frame x_{t-1} to predict $\hat{x}_{t:t+T}$. During training, we assume to have access to ground-truth video frames $x_{1:t+T}$ and ground-truth relative transformations $p_{1:t+T}$. And during testing, we only use the conditioning video frames $x_{1:t-1}$ as input and does not rely on additional ground-truth information about relative transformations.

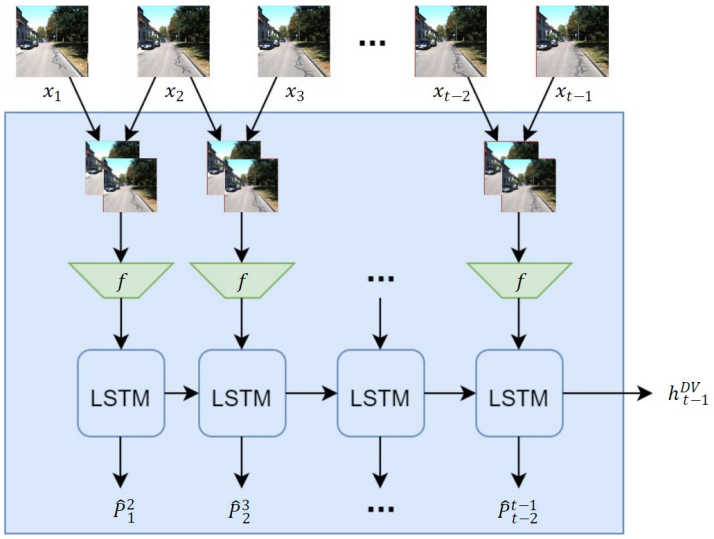

Figure 3.1: **DeepVO Architecture.** DeepVO takes pairs of monocular images in a sequence and outputs the relative pose between them.

3.2 Deep Visual Odometry

We first discuss a method for visual odometry used to estimate the relative changes in the camera poses of the conditioning frames $x_{1:t-1}$. We adapt DeepVO, a recurrent convolutional neural network (RCNN), to encode a sequence of raw RGB images and estimate the relative change in camera poses between pairs of consecutive frames [15]. Specifically, pairs of consecutive frames in $x_{1:t-1}$ are first stacked in the color channels to form the sequence $\bar{x} = \{x_{1:2}, x_{2:3}, ..., x_{t-2:t-1}\}$. Each stacked pair in $\bar{x}$ is encoded through a convolutional neural network (CNN) encoder to generate features $z = \{f(x_{1:2}), ..., f(x_{t-2:t-1})\}$. Each feature is then passed through a Long Short-Term Memory (LSTM) to sequentially predict relative pose transformations $\hat{p}_{1:t-1} = \{\hat{P}^2_1, ...\hat{P}^{t-1}_{t-2}\}$. The LSTM helps track the temporal changes in the motion. Formally, at step i, the previous hidden state h^{DV}_i from the LSTM and feature $f(x_{i:i+1})$ are fed into the LSTM to generate the relative pose $\hat{P}^{i+1}_i$ and the next hidden state h^{DV}_{i+1}:

$$\left(h^{DV}_{i+1}, \hat{P}^{i+1}_i\right) = \text{LSTM}\left(h^{DV}_i, f\left(x_{i:i+1}\right)\right). \tag{3.2}$$

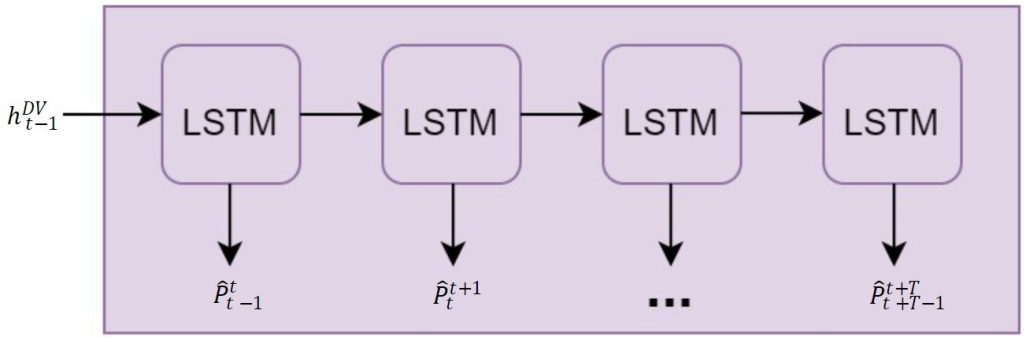

Figure 3.2: **Decoder LSTM Architecture.** The decoder LSTM initializes with the last hidden state h_{t-1}^{DV} from the LSTM in DeepVO and outputs pose prediction recursively.

3.3 Future Camera Pose Prediction

After DeepVO estimates the relative camera pose transformations $\hat{p}_{1:t-1}$ of the conditioning frames $x_{1:t-1}$, we use a decoder LSTM to predict future relative pose transformations $\hat{p}_{t:t+T}$ of the future camera frames $x_{t:t+T}$. First, our decoder LSTM is initialized with the final hidden state h_{t-1}^{DV} of the LSTM in the DeepVO network. This initialization process helps our decoder network to acquire rich contextual information from the conditioning frames. If we only use the relative pose estimations $\hat{p}_{1:t-1}$ of the conditioning frames, we may lose important visual context in the conditioning frames, e.g., a turn in the road ahead. Similar to [29], we generate sequences from our decoder solely from the hidden context in the last hidden state h_{t-1}^{DV} without any additional inputs. Our model outputs more accurate sequences because this formulation avoids the error accumulation that occurs in other recursive approaches where the model feeds past pose predictions back as input. Taking the last hidden state h_{t-1}^{DV} from DeepVO as input, the decoder LSTM outputs prediction sequence of future relative pose transformations $\hat{p}_{t:t+T}$.

At the first prediction step t, the decoder LSTM uses the initial hidden state h_{t-1}^{DV} to predict the relative pose transformation $\hat{P}_{t-1}^{t}$ between frames x_{t-1} and x_t and generate the next hidden state h_t^{dec}. The decoder LSTM similarly generates the rest of the predicted future transformations $\hat{p}_{t+1:t+T}$ as follows:

$$\left(h_{i+1}^{dec}, \hat{P}_{i}^{i+1}\right) = \text{LSTM}\left(h_{i}^{dec}\right). \tag{3.3}$$

The pose estimations from DeepVO and the future pose predictions from our decoder LSTM constitute the complete sequence of generated relative pose transformations $\hat{p}_{1:t+T}$ for both observed and future video frames. When training DeepVO and the future pose decoder on a sequence of image and pose data, we first convert each generated relative predicted pose transformation from the outputs into absolute transformations with respect to x_1:

$$\left\{\hat{P}_i^{i+1}\right\}_{i=1}^{t+T-1} \rightarrow \left\{\hat{P}_1^{j}\right\}_{j=2}^{t+T}. \tag{3.4}$$

We denote the sequence of converted transformation predictions as:

$$\hat{p}_{1:t+T}^{1} \triangleq \left\{\hat{P}_1^{i+1}\right\}_{i=1}^{t+T-1}, \tag{3.5}$$

where $\hat{P}_1^i$ is the predicted transformation from frame x_1 to frame x_i. The ground-truth pose sequence is similarly converted into absolute coordinates with respect to x_1. We denote the converted ground-truth transformations as $p_{1:t+T}^{1}$. Converting to absolute transformations with respect to the first frame allows the network to learn the global context of the sequence instead of only focusing on the relative poses between consecutive frames. Small errors in the relative pose output can be corrected in the next estimation or prediction by the global context from absolute transformations.

Similar to [15], we then compute the mean squared error (MSE) between $\hat{p}_{1:t+T}^{1}$ and the ground-truth poses $p_{1:t+T}^{1}$ as follows:

$$\mathcal{L}_{PP} = \left\|\hat{\boldsymbol{\zeta}} - \boldsymbol{\zeta}\right\|_2^2 + 100 \cdot \left\|\hat{\boldsymbol{\phi}} - \boldsymbol{\phi}\right\|_2^2, \tag{3.6}$$

where the pair $\left(\boldsymbol{\zeta}, \hat{\boldsymbol{\zeta}}\right)$ constitutes the ground-truth translation and predicted translation respectively from $p_{1:t+T}^{1}$ and $\hat{p}_{1:t+T}^{1}$. Similarly, $\left(\boldsymbol{\phi}, \hat{\boldsymbol{\phi}}\right)$ represents the ground-truth rotation and predicted rotation sequences respectively. When training, the gradients from the error between $p_{1:t-1}^{1}$ and $\hat{p}_{1:t-1}^{1}$ only backpropagate through the DeepVO network, acting as an auxiliary loss to improve pose prediction accuracy. The gradients from the error between $p_{t:t+T}^{1}$ and $\hat{p}_{t:t+T}^{1}$ backpropagate through both our decoder LSTM network and DeepVO. Through this training process, our pose predictor learns to capture the motion of the camera and the visual context from conditioning frames and predict the motion for future video frames.

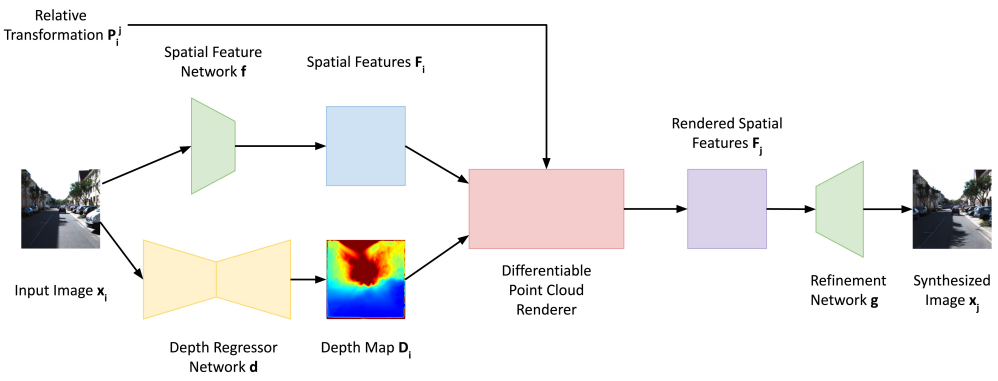

Figure 3.3: **SynSin Architecture.** SynSin takes a monocular image and a relative pose as input and outputs a new view at the pose.

3.4 View Synthesis

We use SynSin, an end-to-end view synthesis framework, to generate a sequence of images at new camera views based on a single RGB image and a sequence of relative pose transformations [16]. Given an input RGB frame x_i and a relative pose transformation P_i^j, we synthesize the RGB frame $\hat{x}_j$, which corresponds to the image of the scene from the new viewpoint.

To capture the scene semantics, the input frame x_i is first passed through a spatial feature network f and a depth regressor network d in parallel to generate a set of spatial features F_i and estimated depth map D_i respectively. The spatial feature network is adapted from BigGAN [34] and uses ResNet [35] as building blocks. The depth regressor network uses the U-Net architecture [36] and outputs depth maps of the same spatial resolution as the input images. The spatial features F_i and depth map D_i are then projected to a 3D point cloud of spatial features C.

This point cloud C is then passed through a differentiable point cloud renderer to generate the spatial features F_j from the new viewpoint according to the relative transformation P_i^j. The rendered spatial features F_j that correspond to the new viewpoint are then passed through a refinement network g, which consists of ResNet blocks similar to the spatial feature network f. The refinement network g inpaints of the dis-occluded regions and corrects local errors from the rendering process. Finally, the refinement network outputs our final predicted frame $\hat{x}_j$.

SynSin can be trained end-to-end on pairs of monocular RGB images and the relative poses between the pairs. It does not require direct supervision from ground-truth depth labels. As a result, SynSin can generalize better to real-world datasets and applications as complete depth information is rarely available in real-world scenarios. The SynSin architecture is trained with the following loss function [16]:

$$\mathcal{L}_{SS} = \lambda_{GAN}\mathcal{L}_{GAN} + \lambda_{l1}\mathcal{L}_{l1} + \lambda_c\mathcal{L}_c, \tag{3.7}$$

where $\mathcal{L}_{GAN}$, $\mathcal{L}_{l1}$, and $\mathcal{L}_c$ refer to the discriminator loss, L1 loss and content loss respectively. The gradients from the loss function backpropragtes through the whole SynSin architecture as every component of SynSin, including the point cloud renderer, is differentiable.

3.5 Combined Architecture

We combine the visual odometry, future pose prediction, and view synthesis components to create our proposed framework. Given a set of conditioning frames $x_{1:t-1}$, we concatenate pairs of consecutive frames to form $\bar{x} = \{x_{1:2}, x_{2:3}, ..., x_{t-2:t-1}\}$. Each element in $\bar{x}$ is passed through DeepVO to generate a set of predicted relative pose transformations $\hat{p}_{1:t-1}$ between the conditioning frames and hidden states $h^{DV}_{1:t-1}$. We then initialize our decoder LSTM using the last hidden state h^{DV}_{t-1} from DeepVO and sequentially generate a set of future pose predictions $\hat{p}_{t:t+T}$ based only on the context from the hidden state h^{DV}_{t-1}.

Using the future pose predictions $\hat{p}_{t:t+T}$, we then generate the set of predicted frames $\hat{x}_{t:t+T}$ in two steps. First, we convert the relative transformations in $\hat{p}_{t:t+T}$ to transformations with respect to last conditioning frame x_{t-1}. This produces the converted transformations:

$$\left\{\hat{P}^i_{i-1}\right\}^{t+T}_{i=t} \rightarrow \left\{\hat{P}^j_{t-1}\right\}^{t+T}_{j=t}. \tag{3.8}$$

We then feed the converted transformation predictions and the last conditioning frame x_{t-1} through SynSin to generate $\hat{x}_{t:t+T}$. Our combined architecture is visualized in Fig 3.4. Contrary to past works, our work does not generate a sequence of video frames recursively, i.e., use a predicted frame

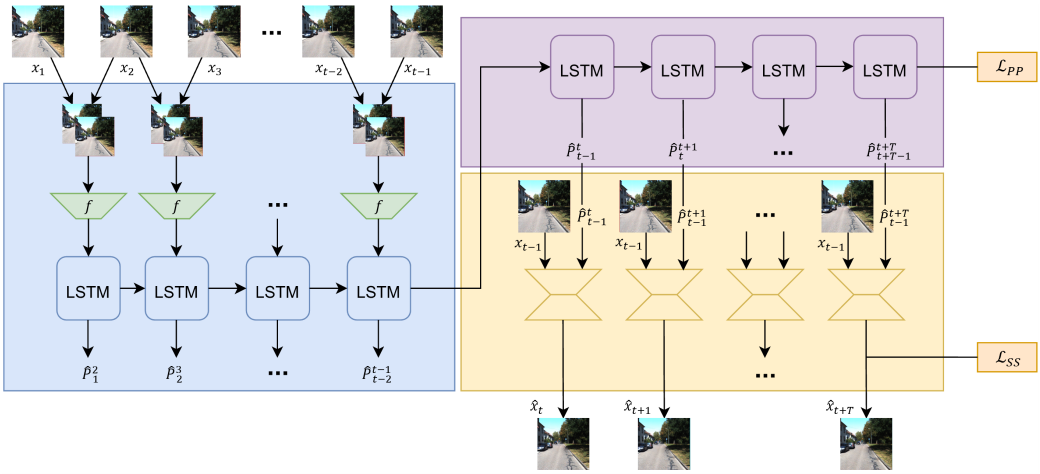

Figure 3.4: **Architecture Overview.** (a) A sequence of conditioning video frames $x_{1:t-1}$ is passed in as an input. (b) DeepVO (in blue) predicts the relative pose transformations $\hat{p}_{1:t-1}$ of the conditioning images. (c) The future pose predictor (in purple) takes the final hidden state of DeepVO as input and predicts future transformations $\hat{p}_{t:t+T}$. (d) SynSin (in yellow) conditions on the final ground-truth input frame and the sequence of converted future poses that are relative to frame x_{t-1} to generate predicted images $\hat{x}_{t:t+T}$.

$\hat{x}_{i-1}$ to generate frame $\hat{x}_i$ [9, 12]. To predict a future image $x_{t+\tau}$, we only use the predicted pose $\hat{P}_{t-1}^{t+\tau}$ and the last conditioning frame x_{t-1} to synthesize the image. In other words, we only predict in image space once for every future image we generate. This design choice mitigates the accumulation of prediction error across long sequences and thus helps with the overall performance of our model.

CHAPTER 4

EXPERIMENTS

In this chapter, we first introduce the KITTI dataset that we use to conduct our experiments. We evaluate the performance of our framework on the KITTI dataset and compare it against state-of-the-art video prediction methods. We discuss the metrics used for evaluation and the baseline methods and ablations that we compare to. We discuss the implementation details for our method and the three baseline methods. We present our experimental results and discuss their implications. Finally, we show the limitations and failure cases of our method.

4.1 KITTI Dataset

The KITTI Visual Odometry dataset is used to train and evaluate our method and baselines [37]. The dataset is collected on a vehicle equipped with stereo cameras and LIDAR driving in different environments. The dataset consists of 22 sequences, and we use the subset of 10 sequences that are publicly available online and have ground-truth pose trajectories. We only use monocular RGB image sequences and the odometry labels from the dataset.

For preprocessing, the raw images are center-cropped and resized to a square 256x256 resolution to reduce the computation need. We train and test our adapted DeepVO network and our decoder LSTM network on the 10 cropped monocular image sequences as well as the ground-truth pose labels. Similarly, SynSin is trained independently on the same subset of images and pose labels with the same center-crop preprocessing.

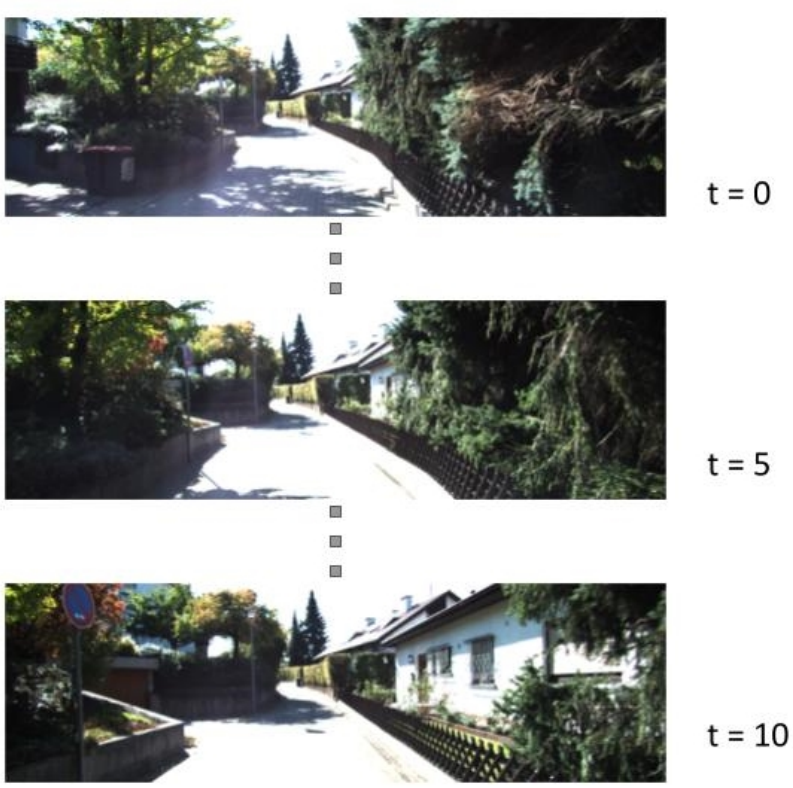

Figure 4.1: **An Example of KITTI Sequence.** A typical sequence in the KITTI dataset [13] where the vehicle drives forward on the road.

4.2 Evaluation Metrics

We assess the performance of our framework and baselines with the following evaluation metrics:

1. **Peak Signal-To-Noise Ratio (PSNR)**: PSNR computes a pixel-wise distance between two images a and b, closely related to mean squared error (MSE) as:

$$\text{PSNR}(a,b) = 10\log_{10}\left(\frac{R^2}{\text{MSE}(a,b)}\right), \tag{4.1}$$

   where R is the maximum possible value of a pixel in an image. The higher the PSNR, the better the similarity score.

2. **Structural Similarity (SSIM)**: SSIM computes the similarity between two images as a mean of the similarity within local windows of

Figure 4.2: **Center-Crop Process.** Note that the resolution of the raw KITTI image can change but we always resize it down to 256 × 256.

the images, accounting for luminance, contrast, and structure [38]. A higher SSIM value typically indicates more similar images.

3. **Learned Perceptual Image Patch Similarity (LPIPS)**: LPIPS measures the perceptual similarity between patches in two images via a trained convolutional neural network [17]. Generally, a lower LPIPS score indicates more perceptually similar images.

While Peak Signal-to-Noise Ratio (PSNR) is a classic metric for indicating image similarity, it is not necessarily informative in its assessment of the visual quality of the images. As PSNR is closely linked to MSE, blurry and often imperceptible images may obtain a higher PSNR compared to sharper, more visually interpretable and similar results. As such, metrics like SSIM and, more recently, LPIPS provide a more meaningful evaluation of images through the explicit consideration of perceptual similarity. Particularly, LPIPS has been shown to be far more indicative of similarity in terms of human perception compared to other classic metrics, including SSIM [17].

4.3 Baselines

We compare our method against three state-of-the-art methods in video prediction: SVG [9], SRVP [11], and SLAMP [12].

4.3.1 SVG

SVG [9] is a stochastic video generation model that consists of two components: (1) a recurrent frame prediction model and (2) a recurrent learned prior model that captures uncertainty in the conditioning frames through stochastic latent variables. For inference, the prediction model uses samples from the distribution in the learned prior model and the encoded conditioning frames to generate prediction outputs.

4.3.2 SRVP

SRVP [11] is a hierarchical method that uses a latent model to capture the underlying dynamics of the video. Unlike SVG and other autoregressive stochastic prediction models, SRVP does not feed prediction frames back to the model to generate new predictions. Instead, SRVP propagates through its learned latent dynamic model with a residual update rule. For generating image predictions, SRVP uses only the latent state but not previous prediction results. As a result, SRVP decouples the frame synthesis process from the temporal dynamics. SRVP is thus less prone to compounding error in the multi-step video prediction process.

4.3.3 SLAMP

To overcome the challenges from camera motion, SLAMP [12] uses an extension of the SVG architecture and adds a pixel motion prediction branch to the SVG prediction model. In addition to predicting the pixel appearance, SLAMP also predicts the future optical flow. SLAMP captures the uncertainty in appearance and motion by modeling them in separate stochastic latent variables. SLAMP then samples from these latent variables to generate appearance and motion predictions. The optical flow from the motion

prediction is used to reconstruct the image prediction through warping. Finally, SLAMP applies a learned mask on images from both the appearance and motion predictors to generate the final prediction output. By modeling the pixel motion explicitly in a new prediction branch, SLAMP can capture the pixel dynamics with camera motion better than SVG.

4.4 Ablations

In addition to comparing our proposed method against the aforementioned baselines, we use ground-truth 3D pose inputs for the view synthesis module as ablations for testing maximum theoretical performance given a perfect pose predictor. In the KITTI dataset, the camera is rigidly attached to a vehicle driving on the ground, so we also perform an ablation using only 2D poses (i.e., 2D positions and yaw headings). Compared to using full 3D poses, using 2D poses lose information about possible changes in elevation and angular movements in the pitch and roll axes.

4.5 Implementation Details

For our method and the three baseline methods, we train and test on the 10 publicly available sequences from the KITTI Odometry dataset [37]. We use six sequences for training, two for validation, and two for testing. For preprocessing, we center-crop and resize the images to the resolution of 256 $\times$ 256. We condition all prediction models on 10 input frames and predict the next 10 frames in training. For testing, we let the models predict the next 20 frames given 10 input frames.

The three baseline models only accept lower resolution input including 64 $\times$ 64. Therefore, we extend the image encoder and decoder by adding additional convolutional and pooling layers to accept 256 $\times$ 256 input. For a fair comparison, we use the same extension on SVG and SLAMP since they are very similar in terms of network architecture.

While the three baseline models only use image data to train, our method uses ground-truth poses in addition to the images from the dataset to train the visual odometry prediction module and the view synthesis module. We

train both modules separately and combine them at the test time. At the test time, our method uses only conditioning image sequences as input and does not rely on ground-truth pose labels. It is also theoretically possible to train both modules together with only image data in an end-to-end fashion.

For the visual odometry prediction module, instead of predicting 3D camera poses, we predict in 2D because the vehicle is driving on the ground in the KITTI dataset. We observe that the visual odometry prediction model learns to predict 2D poses better as they are simpler compared to full 3D poses. Using 2D poses results in better performance for the combined model. Compared to the original DeepVO LSTM network, we reduce the hidden state feature size from 1000 features to 50 features and the number of recurrent layers from 2 layers to 1 layer. We discover that a small and shallow recurrent network helps with the overall performance. We use the same parameters in the decoder LSTM to make sure the sizes of hidden states match. The matching sizes allow us to use the last hidden state from the LSTM in DeepVO to directly initialize the decoder LSTM for pose prediction. We feed zero input to the decoder LSTM to make sure it only relies on the information from the hidden state.

In the view synthesis module, we first estimate the depth and extract features from the last observed frame. We use inverse depth to handle the long-tail distribution of depth in the outdoor KITTI scenes. The module then uses the depth and extracted features to create a 3D point cloud representation of the scene geometry. Using the relative transformation between the camera pose in the last observation and predicted future camera poses, we render new 2D views as our image predictions.

4.6 Results

We compare the performance of our method against the baseline methods on the KITTI dataset. We show that our method outperforms the baselines both qualitatively in terms of visual quality and quantitatively on the LPIPS metric. We also discuss the implications of our results and the limitations of our method.

4.6.1 Qualitative Results

Qualitatively, our method produces sharp and realistic predictions into the future. An example of a generated sequence from the baselines and our method can be seen in Figure 4.4. Compared to the baselines, our method generates predictions that remain sharp even at long time horizons, whereas the prediction image quality of our baselines quickly degrades over time and suffers from blurriness and visual artifacts. While our method also has imperfections in depth estimation and image reconstruction, there are significantly fewer visual artifacts and the resulting images are of much higher visual quality. Unlike the baselines, our method uses a hierarchical approach and only generates image prediction once at the desired time step. As a result, our method does not suffer as much from blurriness and error accumulation in the image space as the autoregressive baselines. Even with errors in the visual odometry prediction process, the resulting images remain sharp, though from a slightly different viewpoint.

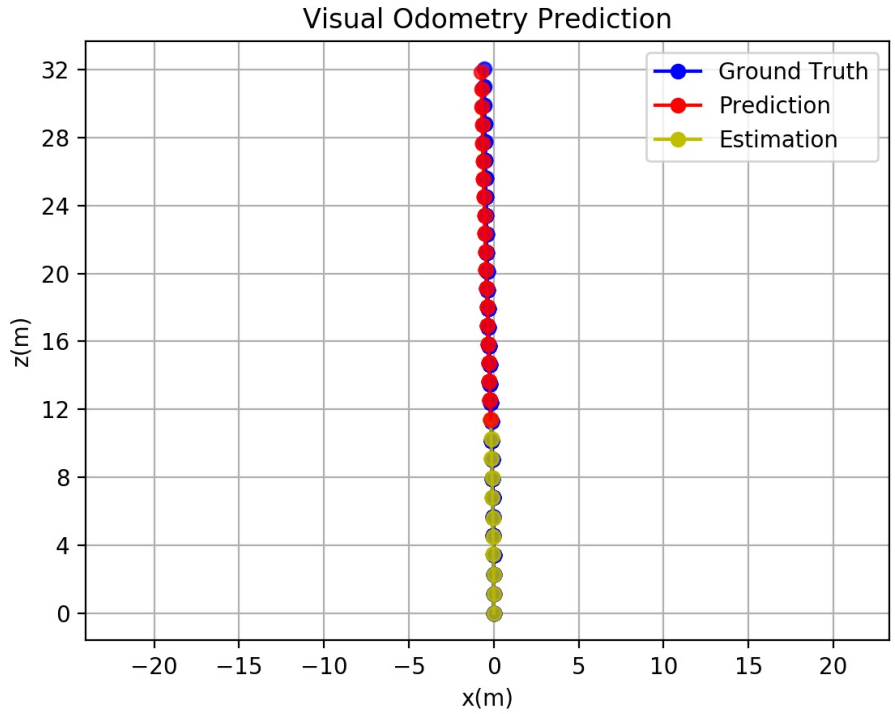

Figure 4.3: **Visual Odometry Prediction Results from Our Method.**

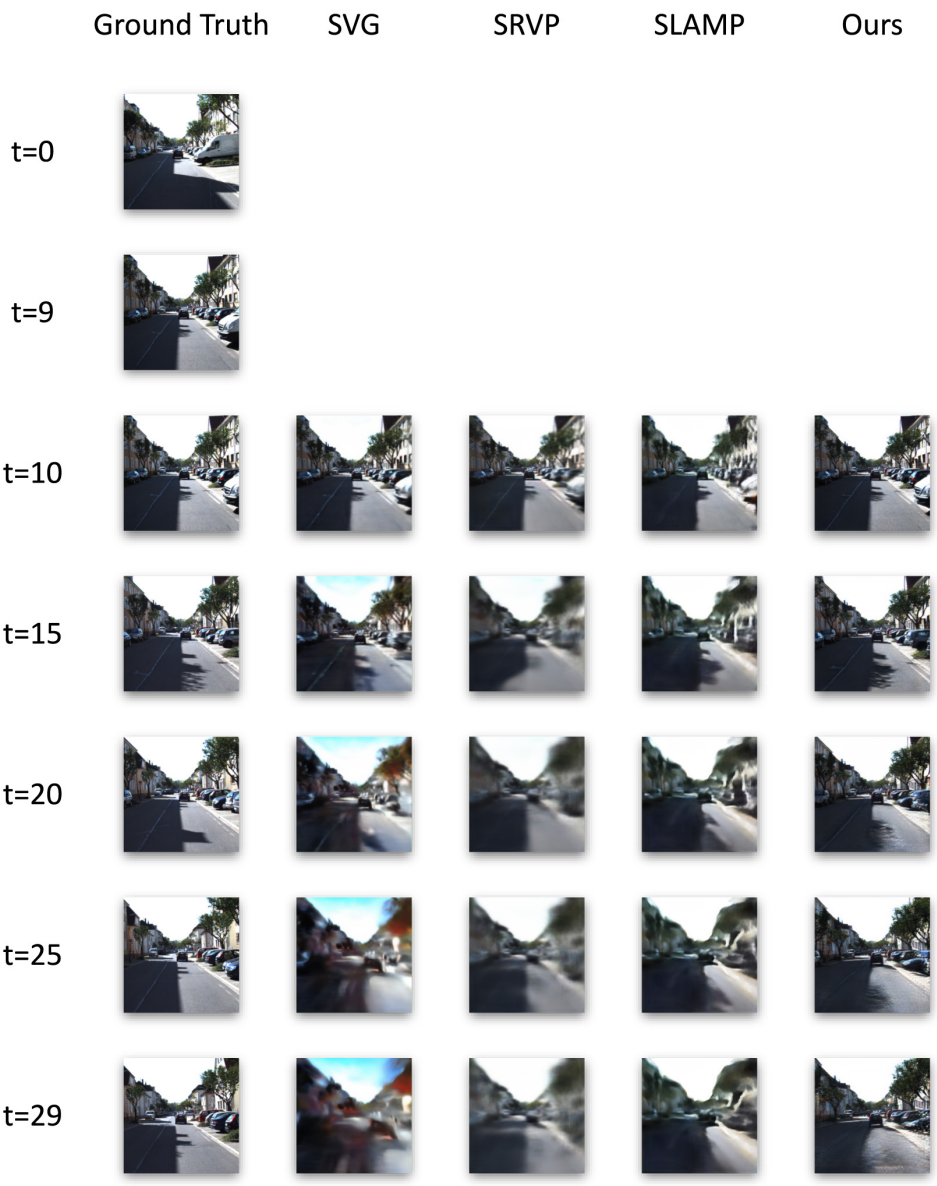

Figure 4.4: **Predicted Sequences from SVG, SRVP, SLAMP, and Our Method.**

4.6.2 Quantitative Results

Table 4.1 showcases the performance of each of the baselines, ablations, and our method on the test set of the KITTI Odometry Dataset. We found similar results for SVG, SRVP, and SLAMP to their original respective works on the PSNR, SSIM, and LPIPS metrics. Our method reaches slightly worse performance than the state-of-the-art results for PSNR and SSIM metrics but significantly outperforms them on the LPIPS metric. The high LPIPS score suggests that the future frames generated by our model have a far higher perceptual similarity to the ground-truth frames. We attribute the high performance of our model on the LPIPS metric to two key factors, a more accurate model of the pixel motion and a hierarchical approach that avoids error accumulation in image space.

In the dataset, the motion of the camera in the surrounding 3D world induces the movements in 2D pixels. Unlike the baselines models that only model 2D pixels and optical flow, our method models the underlying camera motion and 3D geometry. Thus, our model can better capture the dynamics of pixel motion and thus achieves higher visual quality than the 2D-only baselines.

Table 4.1: **PSNR, SSIM, and LPIPS results for SVG [9], SRVP [11], and SLAMP [12], our method, and ablations on the KITTI dataset.** ↑ denotes higher values are better while ↓ means lower values are better. Stochastic baselines are tested with both sampling one result as well as taking the best result over 100 samples. Ablation results are found by feeding ground-truth (GT) poses from KITTI into SynSin.

Models	PSNR (↑)	SSIM (↑)	LPIPS (↓)
SVG - 1 Sample	12.244 ± 0.076	0.396 ± 0.003	0.663 ± 0.004
SVG - 100 Samples	12.268 ± 0.076	0.397 ± 0.003	0.661 ± 0.004
SRVP - 1 Sample	12.482 ± 0.068	0.407 ± 0.003	0.727 ± 0.003
SRVP - 100 Samples	$\mathbf{13.504 \pm 0.070}$	$\mathbf{0.433 \pm 0.003}$	0.698 ± 0.003
SLAMP - 1 Sample	12.539 ± 0.080	0.402 ± 0.003	0.595 ± 0.004
SLAMP - 100 Samples	12.714 ± 0.082	0.408 ± 0.003	0.588 ± 0.004
Ours	11.301 ± 0.091	0.337 ± 0.004	$\mathbf{0.452 \pm 0.006}$
GT 2D Pose + SynSin	12.727 ± 0.108	0.386 ± 0.004	0.380 ± 0.006
GT 3D Pose + SynSin	12.897 ± 0.107	0.393 ± 0.004	0.380 ± 0.006

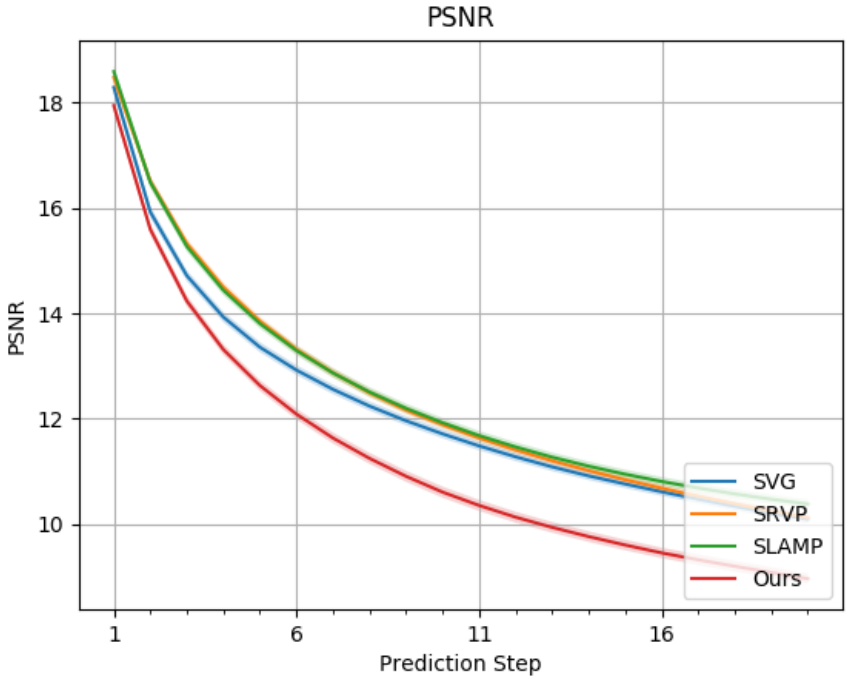

Figure 4.5: **The Average PSNR over Time during Testing.**

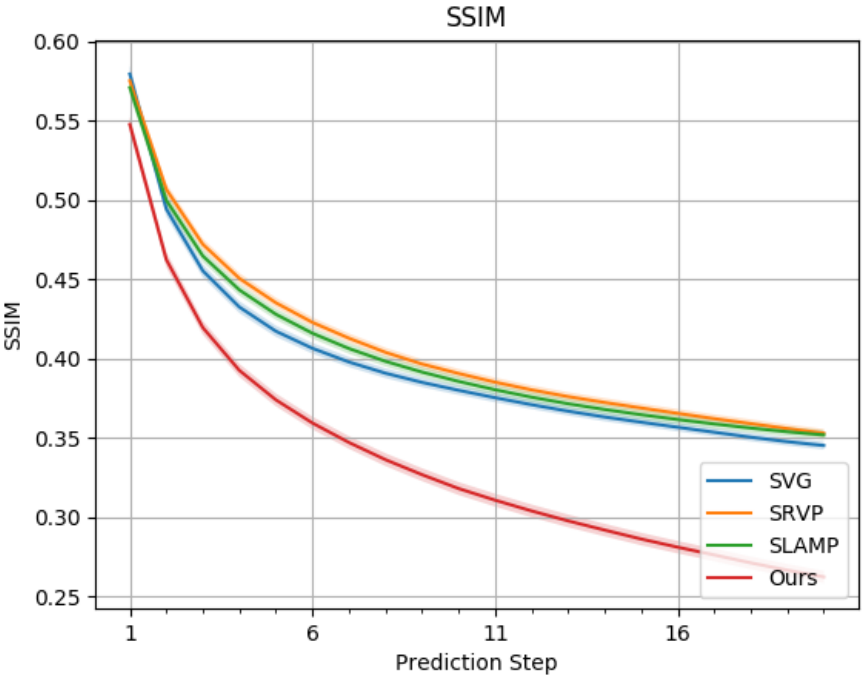

Figure 4.6: **The Average SSIM over Time during Testing.**

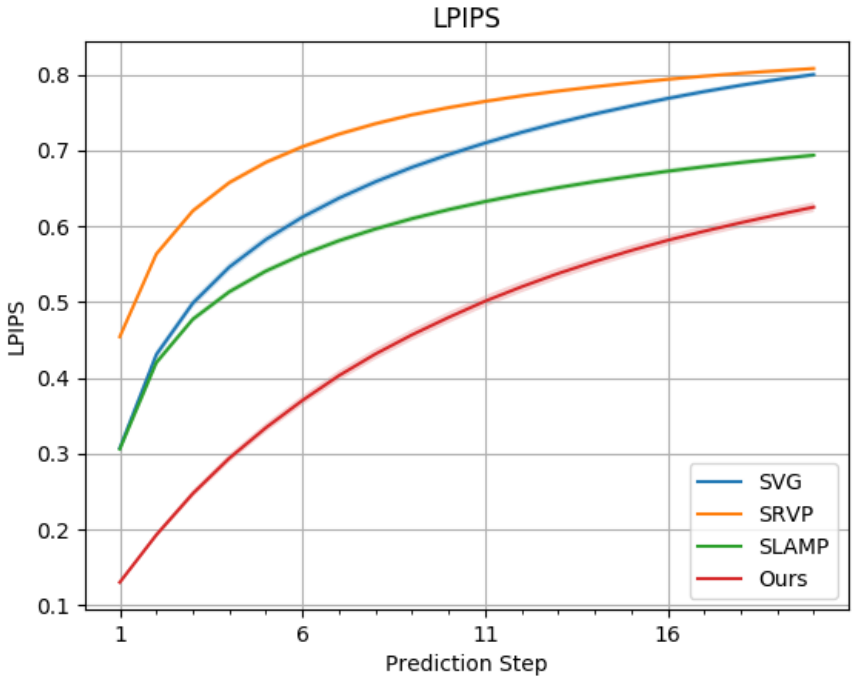

Figure 4.7: **The Average LPIPS over Time during Testing.**

In addition, our method uses a hierarchical approach when generating image predictions multiple steps into the future. In our method, we first predict the future camera poses using only the information from the last hidden state in DeepVO. At the desired time step, we use the predicted pose and the last conditioning frame to generate the new predicted view. In contrast, SVG and SLAMP feed prediction as input recursively in order to generate long-term predictions. Prediction error in image space accumulates in this autoregressive process, while our method avoids such error accumulation by using conditioning frames as input and only predicting once to generate the final output.

For both baselines and our method, the metrics become worse as we predict further into the future as shown in Figure 4.5, Figure 4.6, and Figure 4.7. The degradation in the quality of predictions is expected as the image observations become harder to predict over time.

Even though SRVP performs the best in terms of PSNR and SSIM, it actually shows the worst performance among all methods in the LPIPS metric. This observation suggests that higher PSNR and SSIM may not directly correlate to higher perceptual similarity. This finding is further supported when comparing the prediction results qualitatively.

Compared to the ablations with ground-truth poses, our method shows comparable performance. In addition, using only 2D poses shows similar performance as using 3D poses, which supports the usage of 2D poses in the visual odometry prediction.

4.6.3 One-shot Prediction Performance and Stochasticity

Unlike the stochastic baselines, our model is fully deterministic. Stochastic prediction models benefit from modeling the intrinsic uncertainty in motion. A common practice is to use multiple inferences in test time and choose the best sample by comparing with the ground-truth future frames. In real-life applications, ground-truth future images will not be available as predictions are only valuable when there is no ground-truth information. In those scenarios, it is thus impossible to determine which of the generated predictions from the multiple inferences fits the best. In other words, models need to have good one-shot performance, and sampling multiple times is infeasible in practice. In addition, compared to methods that use multiple samples, our method only requires one inference for predicting the future and reduces computing time, which is an important factor in real-time applications in robotics.

Theoretically, stochastic models should benefit from sampling multiple times since the larger number of samples would cover a wider distribution of future outcomes, making it more likely to match the actual future frame. However, we observe minimal gain when using 100 samples in SVG and SLAMP when compared to using one sample. This observation suggests that stochastic models may not benefit from having stochasticity in our experiment scenarios.

4.6.4 Interpretability

Unlike the other methods that rely on 2D models and dynamics, our method has better interpretability due to its 3D world model and hierarchical design. Instead of predicting 2D pixels and optical flow, we explicitly predict camera motion in the 3D world. The camera motion and 3D geometry generated by our model is more easily interpretable than latent models like SRVP. The poses generated by our high-level pose predictor and 3D geometry generated by our view synthesis module are also useful in applications like robotics and autonomous driving.

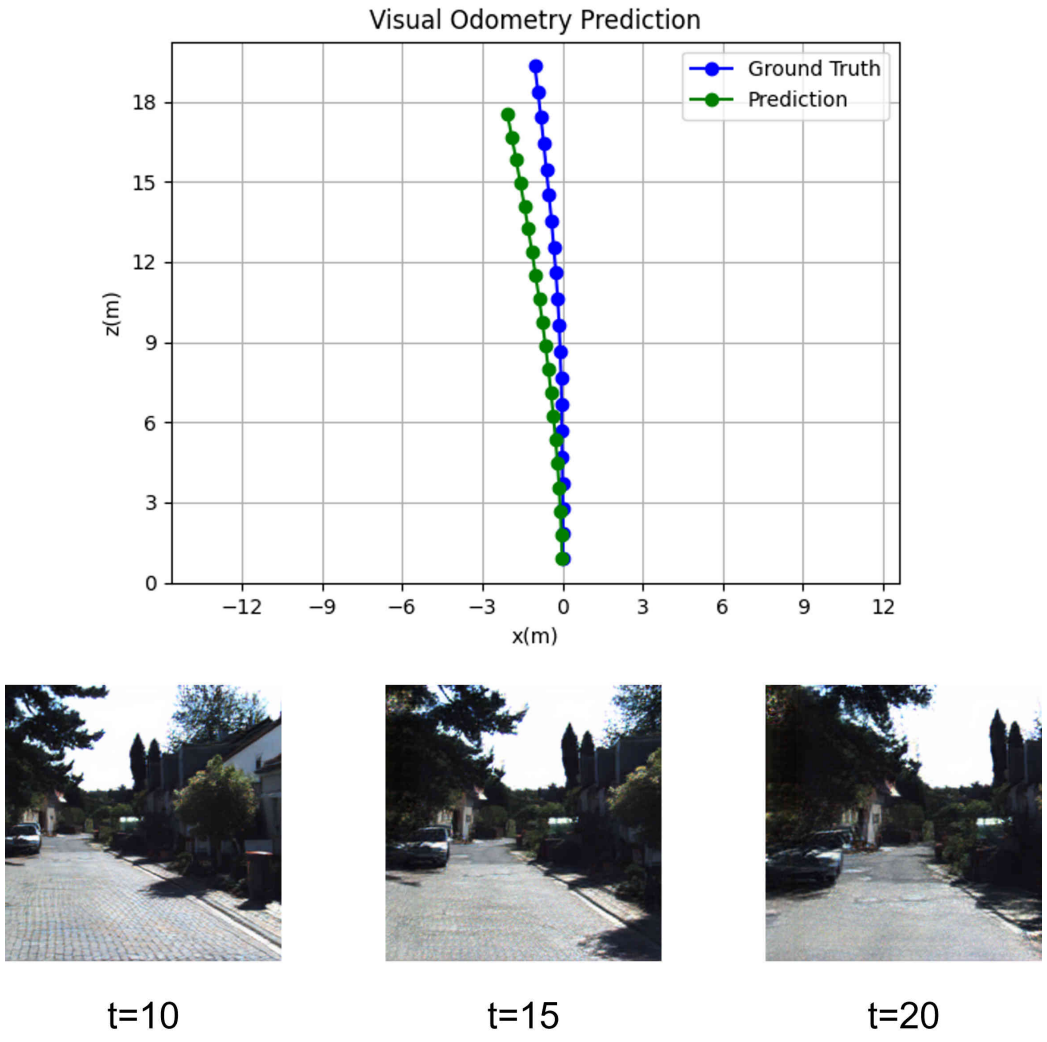

Figure 4.8: **Our Predicted Pose and Images on a Curved Path.**

In Figure 4.8, we compare the visual odometry prediction module against the ground-truth camera pose. In the scene, the model is able to successfully predict the slight left turn based on the conditioning frames. The model learns the semantic meanings from the scenes and generates predictions that correspond to the surroundings.

4.6.5 Limitations

In the KITTI dataset, most parts of the image are static, while there are some dynamic objects including other vehicles and pedestrians. Since our method only uses the last conditioning frame to construct the 3D model of the scene, the 3D model is inherently static as we cannot extract object motion from a single frame. As a result, our method cannot model dynamic objects in the scene. For instance, if a bike is approaching the vehicle, the method is not able to predict the movement so the bike stays in place in the predicted frame as if it is static as shown in Figure 4.9.

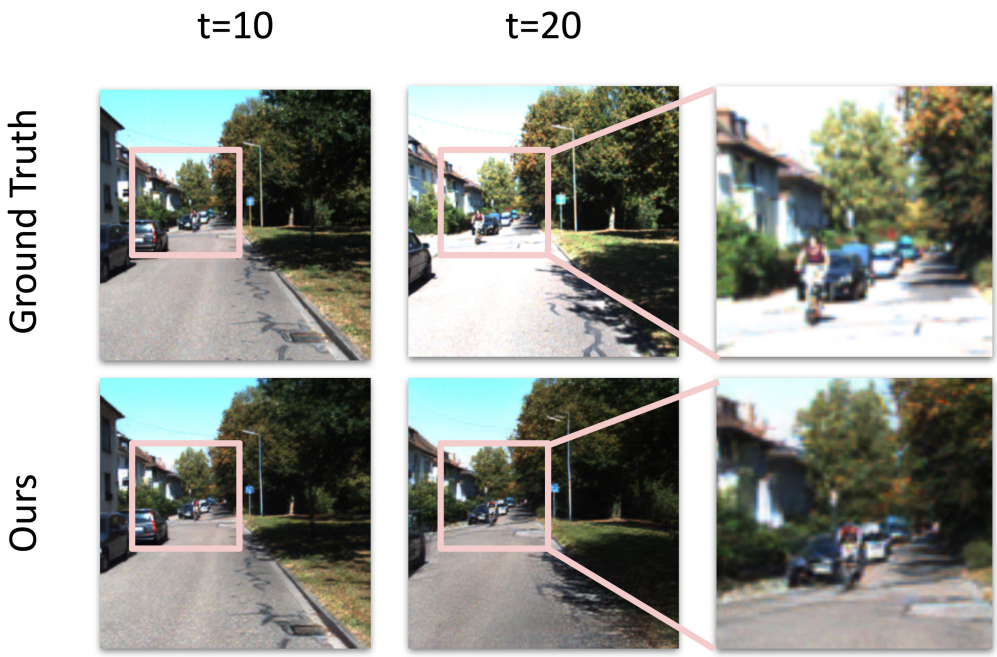

Figure 4.9: **Failure Case when Predicting Dynamic Objects.** The bike is approaching the camera, but it is fixed in space in our prediction frame.

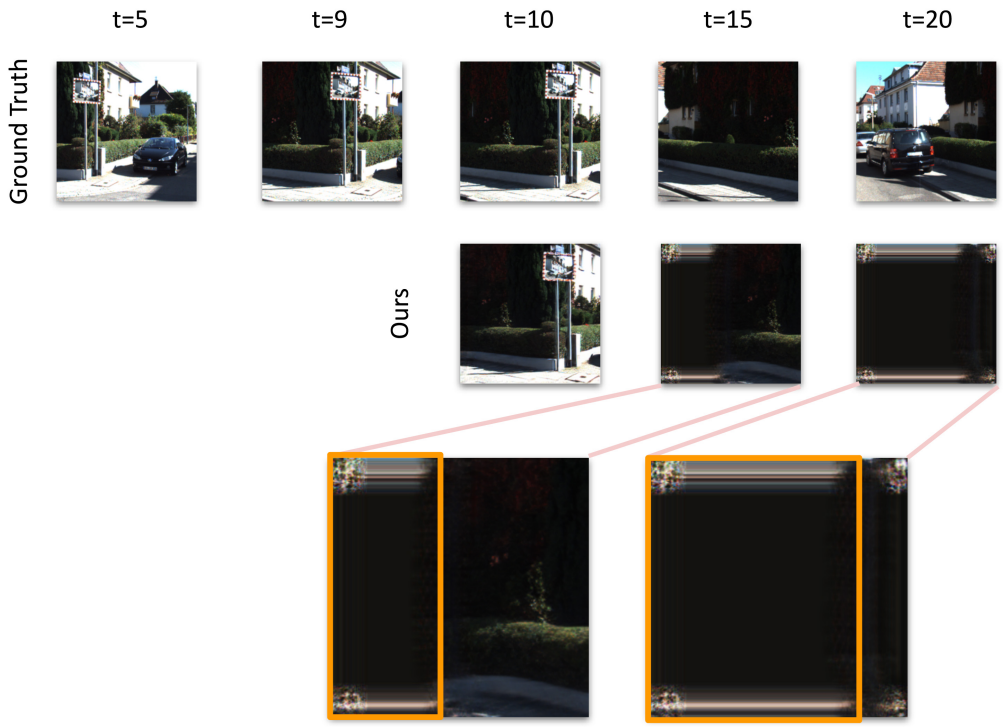

Figure 4.10: **Failure Case when Inpainting Large Unseen Areas.** The vehicle is turning left in the sequence, resulting in large unseen areas on the left of the prediction frames, which our method fails to inpaint.

There are also cases where the vehicle makes large turns into environments that do not overlap with the conditioning frames. Even though the refinement network in our SynSin view synthesis module can account for small unseen areas, it fails to inpaint large regions as seen in Figure 4.10.

CHAPTER 5

CONCLUSIONS

We propose a novel geometry-based approach for addressing the issue of camera motion in mobile robotics, integrating monocular visual odometry prediction and view synthesis to produce a hierarchical video prediction framework. Our model captures the camera motion across the conditioning frames and predicts future camera poses, which are used in tandem with the last conditioning frames to generate future frames. Due to the capability of our framework to account for camera motion, our model outperforms state-of-the-art video prediction methods such as SVG, SRVP, and SLAMP in terms of LPIPS metric and visual quality on the KITTI dataset. We believe that the approach to account for camera motion in our method is generalizable and can be integrated into other models and frameworks to enhance performance in scenarios with non-stationary cameras. However, our method also has limitations including the inability to predict dynamic object movements in the scene as well as the failure to inpaint large unseen areas. Recent view synthesis methods show promising results in capturing dynamic environments from multiple views [39, 40]. There is also success in inpainting large parts of unseen areas using generative methods [41]. Possible future works could explore (1) extracting the motion of dynamic objects and higher-quality 3D geometry from multiple conditioning frames in the video sequence by using dynamic multi-view view synthesis approaches, (2) improving the inpainting capability of our model by using better generative methods, and (3) enhancing the pose prediction framework to more accurately generate future poses.

REFERENCES

[1] K. Pertsch, O. Rybkin, F. Ebert, S. Zhou, D. Jayaraman, C. Finn, and S. Levine, "Long-horizon visual planning with goal-conditioned hierarchical predictors," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 17 321–17 333, 2020.

[2] C. Finn, I. Goodfellow, and S. Levine, "Unsupervised learning for physical interaction through video prediction," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 29, 2016.

[3] C. Finn and S. Levine, "Deep visual foresight for planning robot motion," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2786–2793.

[4] A. X. Lee, R. Zhang, F. Ebert, P. Abbeel, C. Finn, and S. Levine, "Stochastic adversarial video prediction," *arXiv preprint arXiv:1804.01523*, 2018.

[5] D. Cobzas and M. Jagersand, "Tracking and predictive display for a remote operated robot using uncalibrated video," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2005, pp. 1847–1852.

[6] M. Brudnak, "Predictive displays for high latency teleoperation," in *Proc. NDIA Ground Veh. Syst. Eng. Technol. Symp.*, 2016, pp. 1–16.

[7] J. Oh, X. Guo, H. Lee, R. L. Lewis, and S. Singh, "Action-conditional video prediction using deep networks in atari games," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 28, 2015.

[8] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised Learning of Video Representations using LSTMs," in *International Conference on Machine Learning (ICML)*, 2015, pp. 843–852.

[9] E. Denton and R. Fergus, "Stochastic video generation with a learned prior," in *International Conference on Machine Learning (ICML)*, 2018, pp. 1174–1183.

[10] M. Babaeizadeh, C. Finn, D. Erhan, R. H. Campbell, and S. Levine, "Stochastic variational video prediction," *arXiv preprint arXiv:1710.11252*, 2017.

[11] J.-Y. Franceschi, E. Delasalles, M. Chen, S. Lamprier, and P. Gallinari, "Stochastic latent residual video prediction," in *International Conference on Machine Learning (ICML)*, 2020, pp. 3233–3246.

[12] A. K. Akan, E. Erdem, A. Erdem, and F. Güney, "Slamp: Stochastic latent appearance and motion prediction," in *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 14 728–14 737.

[13] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[14] M. Sarkar, D. Ghose, and A. Bala, "Decomposing camera and object motion for an improved video sequence prediction," in *NeurIPS 2020 Workshop on Pre-registration in Machine Learning*, 2021, pp. 358–374.

[15] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 2043–2050.

[16] O. Wiles, G. Gkioxari, R. Szeliski, and J. Johnson, "Synsin: End-to-end view synthesis from a single image," *arXiv, arXiv:1912.08804*, vol. abs/1912.08804, 2019. [Online]. Available: http://arxiv.org/abs/1912.08804

[17] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *CVPR*, 2018.

[18] M. Oliu, J. Selva, and S. Escalera, "Folded recurrent neural networks for future video prediction," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 716–731.

[19] J.-T. Hsieh, B. Liu, D.-A. Huang, L. F. Fei-Fei, and J. C. Niebles, "Learning to decompose and disentangle representations for video prediction," *Advances in neural information processing systems*, vol. 31, 2018.

[20] R. Villegas, D. Erhan, H. Lee et al., "Hierarchical long-term video prediction without supervision," in *International Conference on Machine Learning*. PMLR, 2018, pp. 6038–6046.

[21] Y. Ye, M. Singh, A. Gupta, and S. Tulsiani, "Compositional video prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 10 353–10 362.

[22] B. Wu, S. Nair, R. Martin-Martin, L. Fei-Fei, and C. Finn, "Greedy hierarchical variational autoencoders for large-scale video prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2318–2328.

[23] H. Wu, Z. Yao, J. Wang, and M. Long, "Motionrnn: A flexible model for video prediction with spacetime-varying motions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 435–15 444.

[24] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," *arXiv preprint arXiv:1511.05440*, 2015.

[25] W. Byeon, Q. Wang, R. K. Srivastava, and P. Koumoutsakos, "Contextvp: Fully context-aware video prediction," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 753–769.

[26] S. Oprea, P. Martinez-Gonzalez, A. Garcia-Garcia, J. A. Castro-Vargas, S. Orts-Escolano, J. Garcia-Rodriguez, and A. Argyros, "A review on deep learning techniques for video prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[27] W. Lotter, G. Kreiman, and D. Cox, "Deep predictive coding networks for video prediction and unsupervised learning," *arXiv preprint arXiv:1605.08104*, 2016.

[28] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee, "Decomposing motion and content for natural video sequence prediction," *arXiv preprint arXiv:1706.08033*, 2017.

[29] R. Villegas, J. Yang, Y. Zou, S. Sohn, X. Lin, and H. Lee, "Learning to generate long-term future via hierarchical prediction," in *International Conference on Machine Learning (ICML)*, 2017, pp. 3560–3569.

[30] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local SVM approach," in *International Conference on Pattern Recognition (ICPR)*, vol. 3, 2004, pp. 32–36.

[31] F. Ebert, C. Finn, A. X. Lee, and S. Levine, "Self-Supervised Visual Planning with Temporal Skip Connections," in *Conference on Robot Learning (CoRL)*, 2017, pp. 344–356.

[32] R. Mahjourian, M. Wicke, and A. Angelova, "Geometry-based next frame prediction from monocular video," in *IEEE Intelligent Vehicles Symposium (IV)*, 2017, pp. 1700–1707.

[33] K. E. Ak, Y. Sun, and J. H. Lim, "Robust multi-frame future prediction by leveraging view synthesis," in *2021 IEEE International Conference on Image Processing (ICIP)*, 2021, pp. 2693–2697.

[34] A. Brock, J. Donahue, and K. Simonyan, "Large scale gan training for high fidelity natural image synthesis," *arXiv preprint arXiv:1809.11096*, 2018.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[36] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[37] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[38] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

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

[40] C. Gao, A. Saraf, J. Kopf, and J.-B. Huang, "Dynamic view synthesis from dynamic monocular video," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 5712–5721.

[41] A. Liu, R. Tucker, V. Jampani, A. Makadia, N. Snavely, and A. Kanazawa, "Infinite nature: Perpetual view generation of natural scenes from a single image," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 458–14 467.