Stochastic Future Prediction in Real World Driving Scenarios

by

Adil Kaan Akan

A Dissertation Submitted to the
Graduate School of Sciences and Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Computer Engineering

KOÇ ÜNİVERSİTESİ

June 3, 2022

Stochastic Future Prediction in Real World Driving Scenarios

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Adil Kaan Akan

and have found that it is complete and satisfactory in all respects,

and that any and all revisions required by the final

examining committee have been made.

Committee Members:

Assist. Prof. Fatma Güney (Advisor)

Assoc. Prof. Aykut Erdem

Prof. Jürgen Gall

Date:

To all my beloved ones

ABSTRACT

Stochastic Future Prediction in Real World Driving Scenarios
Adil Kaan Akan
Master of Science in Computer Engineering
June 3, 2022

Uncertainty plays a key role in future prediction. The future is uncertain. That means there might be many possible futures. A future prediction method should cover the whole possibilities to be robust. In autonomous driving, covering multiple modes in the prediction part is crucially important to make safety-critical decisions. Although computer vision systems have advanced tremendously in recent years, future prediction remains difficult today. Several examples are uncertainty of the future, the requirement of full scene understanding, and the noisy outputs space. In this thesis, we propose solutions to these challenges by modeling the motion explicitly in a stochastic way and learning the temporal dynamics in a latent space.

Firstly, we propose to use the motion information in the scene in stochastic video prediction. By separately modeling the appearance of the scene and the motion in the scene, the static and the dynamic parts are partitioned. The dynamic part is predicted by the explicit motion. Since the motion is predicted, the noisy pixel space is not used. We demonstrate the benefits of using the motion information scene explicitly.

Secondly, we propose to separate the scene into the 3D structure and the motion, which correspond to static and dynamic parts, in video prediction in driving scenarios. The static part is handled by the 3D structure and the motion of the ego-vehicle, whereas the dynamic part is handled by the remaining motion in the scene. The remaining motion is captured by explicitly conditioning the dynamic part on top of the static one. We demonstrate the improvements of the conditioning and structure-aware separation.

Finally, motivated by the compact representation, we propose a method for stochastic future prediction in Bird's-Eye View representation. Using a new formulation, we approach the problem as a state prediction rather than a trajectory prediction. For that purpose, we choose to use more powerful label-aware latent variables to generate more diverse and admissible future trajectories. Extensive evaluations show that both the diversity and the accuracy of the future trajectories significantly improved, especially in challenging cases of spatially far regions and temporally long spans.

ÖZETÇE

Yüksek Lisans Tez Başlığı
Adil Kaan Akan
Bilgisayar Mühendisliği, Yüksek Lisans
3 Haziran 2022

Belirsizlik gelecek tahmininde çok kritik bir rol oynamaktadır. Gelecek belirsizdir ve bu birçok gelecek ihtimali olduğunu gösterir. Bir gelecek tahmini yöntemi güvenilir olabilmek için bütün ihtimalleri kapsamalıdır. Otonom sürüşte emniyet açısından kritik kararlar verebilmek için tüm modları kapsamak aşırı derecede önemlidir. Bilgisayarlı görü yöntemleri son yıllarda çok hızlı şekilde ilerse de gelecek tahmini hala zor olmaya devam etmektedir. Bu zorluklardan birkaç tanesi geleceğin belirsizliği, tüm sahnenin anlaşılmasının gerekliliği ve piksellerin gürültülü olmasıdır. Bu tezde, bu problemlere hareket bilgisini modelleyerek ve zamansal dinamikleri öğrenerek çözümler sunuyoruz.

İlk olarak, sahnedeki gelecek bilgisinin açık bir şekilde işlenmesini sunarak video alanında gelecek tahmini yöntemi yapan bir yöntem sunuyoruz. Bu yöntem görünüşü ve hareketi ayrı ayrı modelleyerek sahnenin durağan ve hareketli kısımlarını ayırmaktadır. Bu sayede piksellerin gürültülü uzayından kaçarak hareket bilgisiyle tahmin yapmak mümkün olmuştur. Bu yöntemin kazançlarını detaylı bir şekilde göstermekteyiz.

İkinci olarak, sahneyi 3 boyutlu yapı ve artakalan hareket olacak şekilde ikiye ayırmayı sunuyoruz. Bu ayrışım ile sahnedeki durağan ve hareketli kısımları ayrı ayrı modelleyebiliyoruz. Durağan kısımlar 3 boyutlu yapıyla ve hareketli kısımlar ise artakalan hareket bilgisiyle tahmin edilmektedir. Artakalan hareket için önce 3 boyutlu yapıdaki hareketi tahmin edip bu hareket üstüne koşullanarak tahmin yapılmaktadır. Bu ayrışımın ve koşullanmanın etkilerini detaylı bir şekilde göstermekteyiz.

Son olarak, kompakt gösterimden etkilenerek, gelecek tahmini problemini kuşbakışı uzaya taşımayı sunuyoruz. Yeni bir formulasyon kullanarak gelecek tahmininden ziyade gelecek durum tahmini yapmanın problemi daha da kolaylaştıracağını sunmaktayız. Bu amaçla, gelecek bilgisinin kullanıldığı bir dağılım kullanarak daha dağınık ve kurallara uygun yörünge tahminleri yapılmaktadır. Yaptığımız detaylı deneyler ile bu yöntemin daha gerçekçi tahminler yaptığı gösterilmiştir.

ACKNOWLEDGMENTS

I would like to express my deepest appreciation and gratitude to Asst. Prof. Fatma Güney. This work would not have been possible without her continuous support and motivation. I also want to express my deepest gratitude for being not only my advisor but also my mentor. It has been a privilege to work with you.

I would like to thank the rest of my thesis committee: Assoc. Prof. Aykut Erdem, Prof. Jürgen Gall for agreeing to be in my committee and for evaluating my work.

Also, I would like to thank Assoc. Prof. Aykut Erdem and Assoc. Prof. Erkut Erdem for their contributions to this work.

I would like to thank all the members of AVG. With a special mention to Sadra Safadoust, for all their help and contribution to this work, çağan Selim Çoban, Ege Onat Özsüer, for guiding me throughout my studies.

Also, I would like to thank all of my friends. Especially, Berkay Uğur Şenocak for helpful discussions and code reviews, Batuhan Uluçeşme for his guidance in Koç University, Güneş Çepiç and Melih Kaan Kayapınar for their support, Metehan Gülpınar for his entertaining conversations in hard times, Salih Karagöz for his constructive feedback.

I would like to thank my mother, Dilek, my father, Mehmet, and my little sister, Ayça, for their unconditional support and the strength you have given to me. I always felt your support and motivation with me. It would have been very hard to complete this work with your belief in me.

Last but not least, I would like to thank Aleyna Gürkan, who became everything to me, supported me for everything I wanted to pursue, motivated me to stay at night for work, and tolerated me around deadlines. I honestly could not bear the burden without her endless encouragement, unwavering motivation, joyful and helpful accompany and warm love.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

PSNR	Peak Signal to Noise Ratio
SSIM	Structural Similarity
LPIPS	Learned Perceptual Image Patch Similarity
FVD	Frechet Video Distance
LSTM	Long Short Term Memory Networks
MLP	Multi Layer Perceptron
2D	2 Dimensional
3D	3 Dimensional
CNN	Convolutional Neural Networks

Chapter 1

INTRODUCTION

Videos can be defined as a set of image frames, and naturally, they contain much more information than images. The amount of information is not limited to pixels, and it also includes the temporal information in the videos, *namely* motion. The temporal information both eases and complicates the understanding of the videos. The information in the image level scales up when the number of image frames in the video increases. However, reasoning about the video gets more manageable since the amount of information increase with the number of image frames. For example, objects which are not fully visible in a few frames can be seen in the other frames. Videos include not only foreground objects that are moving but also background, which has complex information in most of the scenarios, e.g. driving scenarios. To understand the videos fully, one needs to figure out the environment, present agents or foreground objects that have an independent motion, relations between the objects, and motion of the background, if there is any. Moreover, the amount of information is not limited to these. The information at the pixel level is enormous because of the large resolution of the videos.

Since birth, we have a constant visual stream of data in our brains, and we naturally learn to process visual information like videos. As soon as we open our eyes to the world, we monitor the world around us and try to reason about it. We, humans, with this much experience, can reason about the videos without much effort. We can understand the objects moving around us and their relation to other objects. Our brain is capable of understanding the scene around us at the video level.

In autonomous driving, the ego-vehicle perceives the world as images at each time step. At each time step, the ego-vehicle takes pictures of the environment around it and makes several decisions according to the surrounding environment such as acceleration, stopping or turning. Before making these decisions, the ego-vehicle needs to understand the world around it, detect and track the moving objects, predict their future trajectories and plan the optimal route. For these tasks, the ego-vehicle needs to consider not only the information at the current time but also the past information. Since the ego-vehicle uses the image at each time steps, it has a stack of images captured at separate time steps, *a video*.

The amount of information in the autonomous driving is immense and to fully solve the problem, the ego-agent needs to be proficient at several sub-tasks such as detection, prediction and planning. Image level reasoning may not suitable to this end and video level understanding might be necessary to conquer the problem because the temporal information in the video can introduce a lot meaningful cues.

To capture all the information in the videos, prediction might be appropriate task. Humans can easily predict the future of the video because they can grasp all the information in the current and the past frames of the video. They can reason about the background and the objects, their motion, and their relation to other objects. Therefore, the prediction task might be the suitable task to fully understand because one needs to consume all the information until the current time before the prediction.

In this thesis, we focus on video future prediction in different representations. To predict the future, an agent first needs to understand its environment and its surrounding objects. Moreover, it must know the past states of both itself and surrounding objects. In order to predict the future, the agent should understand its current and past states, relate the environment and the surrounding objects to each other, then, conditioned on this information, and infer the future. Future prediction has been studied in different representations such as video frames [Akan et al., 2021, Denton and Fergus, 2018, Franceschi et al., 2020, Finn et al., 2016, Babaeizadeh et al., 2018, Lee et al., 2018, Akan et al., 2022, Castrejon et al., 2019, Villegas et al., 2019], key points [Villegas et al., 2017, Minderer et al., 2019] , coordinate space [Liu et al., 2021, Gu et al., 2021a, Zhao et al., 2020a, Gao et al., 2020a, Tang and Salakhutdinov, 2019a] and Birds-eye view [Hu et al., 2021, Akan and Güney, 2022, Hendy et al., 2020]. In the first part of the thesis, we propose a novel method for stochastic video prediction that exploits the motion history in the scene. In the second part, we propose another video prediction method that decomposes the scene into static and dynamic parts and models each part using domain knowledge of real-world driving datasets. In the third part, we propose a future instance segmentation method in Bird's-eye view representation from multiple camera images. In this part, we briefly analyze the problems.

1.1 Motivation

1.1.1 Video Prediction

Videos contain visual information enriched by motion. Motion is a useful cue for reasoning about human activities or interactions between objects in a video. Given a few initial frames of a video, our goal is to predict several frames into the future, as realistically as possible. By looking at a few frames, humans can predict what will happen next. Surprisingly, they can even attribute semantic meanings to random dots and recognize motion patterns [Johansson, 1973]. This shows the importance of motion to infer the dynamics of the video and to predict the future frames.

Video prediction can be defined as predicting future video frames, given a few initial ones. An example definition of the problem can be seen in Fig. 1.1.

Motion cues have been heavily utilized for future frame prediction in computer vision. A common approach is to factorize the video into static and dynamic components [Walker et al., 2015, Liu et al., 2017, Lu et al., 2017, Fan et al., 2019, Gao et al., 2019, Lotter et al., 2017, Jia et al., 2016, Vondrick and Torralba, 2017]. First, most of the previous methods are deterministic and fail to model the uncertainty of the future. Second, motion is typically interpreted as local changes from one frame to the next. However, changes in motion follow certain patterns when observed over some time interval. Consider scenarios where objects move with near-constant velocity, or humans repeating atomic actions in videos. Regularities in motion can be very informative for future frame prediction.

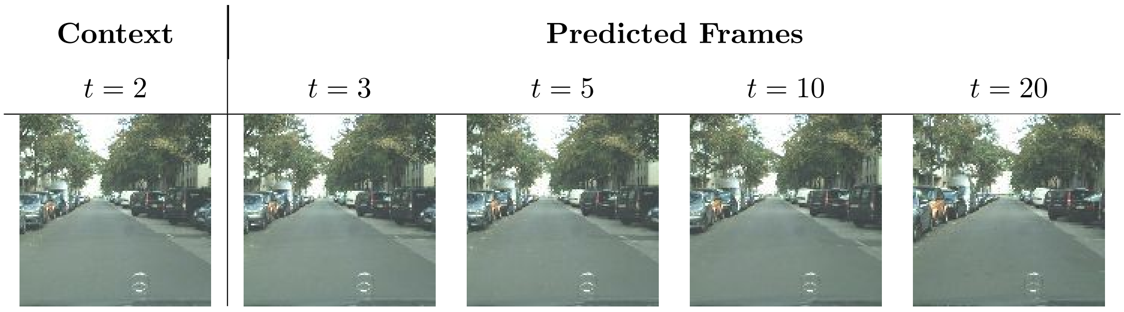

Figure 1.1: The goal in video prediction is to predict the next several frames into the future given a few initial frames. In this example, the first two frames are given and the model tries to predict the next 18 of them. The images are from Cityscapes [Cordts et al., 2016] dataset.

Video prediction methods predicts the target future frames directly in pixel space. That is, they encode both the motion and content of the video into latent variables or specific variables and decode those variable into images using neural networks. However, directly modeling latent variables to the image space is difficult because of the large resolution of the images and the number of possible values that a pixel can take. Several methods [Villegas et al., 2017, Minderer et al., 2019] first try to transfer the prediction task into a more meaningful space, e.g. key point space, and then from that space, future image frames are decoded. As we cover before, motion is a crucial information in a video. The motion in the video would ease the process of prediction. The future of the video can be predicted first transferring prediction task into the motion space and then, the target frames can be decoded, similar to key point approaches. Most of the motion in a video is global across the video rather than a local changes throughout the video. Therefore, by propagating the past motion information into the future steps can lead better predictions.

Stochastic methods have been proposed to model the inherent uncertainty of the future in videos. Earlier methods encode the dynamics of the video in stochastic latent variables which are decoded to future frames in a deterministic way [Denton and Fergus, 2018]. Instead, we first assume that both appearance and motion are encoded in the stochastic latent variables and decode them separately into appearance and motion predictions in a deterministic way. Inspired by the previous deterministic methods [Finn et al., 2016, Liu et al., 2017, Gao et al., 2019], we also estimate a mask relating the two. Both appearance and motion decoders are expected to predict the full frame but they might fail due to occlusions around motion boundaries. Intuitively, we predict a probabilistic mask from the results of the appearance and motion decoders to combine them into a more accurate final prediction. Our model learns to use motion cues in the dynamic parts and relies on appearance in the occluded regions.

In the first two parts of this thesis, we focus on the problem of stochastic video prediction. Different from classical video prediction, in stochastic video prediction, the aim is to generate different futures given the same initial frames. That is, the models learn the multimodality of the data and generate a diverse set of future frames. There are different challenges in this problem. For example, the most important challenge is the uncertainty of the future. Given a few initial frames of video, the future of the video is unknown to models, and there are multiple possible futures. Therefore, the models need to learn the underlying multimodal distribution to generate a set of diverse futures. In the first two chapters, we propose two different stochastic video prediction methods.

1.1.2 Bird's-Eye View Representation

Future prediction with sequential visual data has been studied from different perspectives. Stochastic video prediction methods, as we cover in operate in the pixel space and learn to predict future frames conditioned on the previous frames. These methods achieve impressive results by modeling the uncertainty of the future with stochasticity, however, long-term predictions in real-world sequences tend to be quite blurry due to the complexity of directly predicting pixels. A more practical approach is to consider a compact representation that is tightly connected to the modalities required for the downstream application. In self-driving, understanding the 3D properties of the scene and the motion of other agents plays a key role in future predictions. The bird's-eye view (BEV) representation meets these requirements by first fusing information from multiple cameras into a 3D point cloud and then projecting the points to the ground plane [Philion and Fidler, 2020]. This leads to a compact representation where the future location and motion of multiple agents in the scene can be reliably predicted. In this paper, we explore the potential of stochastic future prediction for self-driving to produce admissible and diverse results in long sequences with an efficient and compact BEV representation.

Future prediction from the BEV representation has been recently proposed in FIERY [Hu et al., 2021]. The BEV representations of past frames are first related in a temporal model and then used to learn two distributions representing the present and the future. Based on sampling from one of these distributions depending on train or test time, various future modalities are predicted with a recurrent model. For planning, long term multiple future predictions are crucial, however, the predictions of FIERY degrade over longer time spans due to the limited representation capability of a single distribution for increasing diversity in longer predictions. Following the official implementation, the predictions do not differ significantly based on random samples but converge to the mean, therefore lack diversity. We start from the same BEV representation and predict the same output modalities to be comparable. Differently, instead of two distributions for the present and the future, we propose to learn time-dependent distributions by predicting a residual change at each time step to better capture long-term dynamics. Furthermore, we show that by sampling a random variable at each time step, we can increase the diversity of future predictions while still being accurate and efficient. For efficiency, we use a state-space model [Murphy, 2023] instead of costly auto-regressive models.

The main idea behind the efficiency of the state-space model is to decouple the learning of dynamics and the generation of predictions [Franceschi et al., 2020]. We first learn a low dimensional latent space from the BEV representation to capture the dynamics and then learn to decode some output modalities from the predictions in that latent space. These output modalities show the location and the motion of the agents in the scene. We can train our dynamics model independently by learning to match the BEV representations of future frames that are predicted by our model to the ones that are extracted from a pre-trained BEV segmentation model [Philion and Fidler, 2020]. Through residual updates to the latent space, our model can capture the changes to the BEV representation over time. This way, the only information we use from the future is the BEV representation of future frames. Another option is to encode the future modalities to predict and provide the model with this encoded representation to learn a future distribution [Hu et al., 2021]. We experiment with both options in this paper. While providing labels in the future distribution leads to better performance, learning the dynamics becomes dependent on the labels. From the BEV predictions, we train a decoder to predict the location and the motion of future instances in a supervised manner. These output modalities increase the interpretability of the predicted BEV representations that can be used for learning a driving policy in future work.

In the last part of this thesis, we focus on future prediction in Bird's-Eye view (BEV) space. Instead of directly predicting the future in pixel space, we choose to use a more compact representation, namely BEV. In self-driving, understanding the 3D properties of the scene and the motion of other agents plays a key role in future predictions. The bird's-eye view (BEV) representation meets these requirements by first fusing information from multiple cameras into a 3D point cloud and then projecting the points to the ground plane [Philion and Fidler, 2020]. An example BEV representation can be seen in Fig. 1.2. This leads to a compact representation where the future location and motion of multiple agents in the scene can be reliably predicted. In the last chapter, we propose a novel stochastic future prediction method for self-driving utilizing BEV representation.

1.2 Proposed Methods and Models

First, we present a new method for stochastic video future frame prediction. We propose to include the motion history in the scene in order to generate more realistic future frames. The motion in the scene was continuous, and by learning to model the history of the motion, we can propagate it to future frames. We named our model SLAMP, which stands for "Stochastic Latent Appearance and Motion Prediction" because our model predicts future frames both in pixel space (**appearance**) and

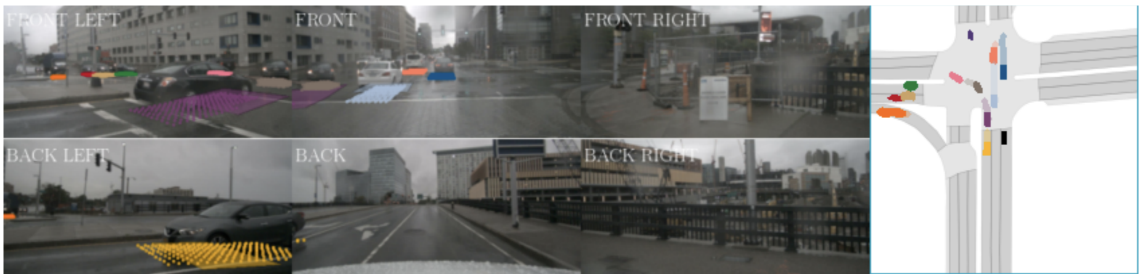

Figure 1.2: **Bird's-Eye View (BEV) Representation** BEV representation can be constructed from multiple cameras of the ego-vehicle. The images are from nuScenes dataset [Caesar et al., 2020] and for the BEV predictions and 2D projected points FIERY [Hu et al., 2021] is used.

motion space (**motion**).

Our major contribution is to exploit the existing motion history in the scene. With the explicit modeling of the motion history, our model predicts the motion between consecutive frames and predicts the future frames in the motion space. Besides in motion space, SLAMP predicts the future frames in pixel space. With these two predictions, SLAMP achieves a decomposition of the scene by attending pixel predictions for the static or occluded parts of the scene and motion predictions for the dynamic parts of the scene. SLAMP achieves state-of-the-art results on challenging real-world datasets with a dynamic background while performing on par with the previous methods on generic video prediction datasets such as KTH [Schüldt et al., 2004a] and BAIR [Ebert et al., 2017a].

Second, we present another novel method for stochastic video future frame prediction, specifically for real-world autonomous driving scenarios. We propose to model static parts of the scene with "Structure and Motion", e.g. Ego-motion, instead of pixel space appearance prediction. We name our model as SLAMP-3D because of its 3D-aware static extension to SLAMP.

The first contribution of SLAMP-3D is that we decompose the scene into static and dynamic components similar to SLAMP. Instead of using pixel space for the static part, we propose to model the structure and the ego-motion together by exploiting the domain knowledge [Zhou et al., 2017, Godard et al., 2019, Safadoust and Gney, 2021]. With this change in the static part, our work is much more suitable for the real-world driving datasets, KITTI [Geiger et al., 2012, Geiger et al., 2013] and Cityscapes [Cordts et al., 2016].

The camera motion in the scene can be explained by the structure and the ego-motion of the scene. If the scene is static, with no motion except the motion of the ego-vehicle, then camera motion can be explained perfectly with structure and ego-motion. However, in real-world scenarios, this assumption fails because there will be many independently moving objects whose motion cannot be explained by ego-motion. Therefore, as the second contribution in SLAMP-3D, we propose to condition the dynamic component on top of the static component to predict the residual flow, the remaining motion after the modeled ego-motion. As in the case of SLAMP, SLAMP-3D exploits the motion history and predicts the future frame in the motion space as an optical flow from the source frame to the target frame. Differently from SLAMP, the predicted optical flow is a *residual flow* which remains in the scene due to independently moving objects. With this conditioning, SLAMP-3D achieves improved results, especially for the foreground objects in the dynamic scenes.

Third, we present a novel method for future instance segmentation method in Bird's-eye view from multiple cameras. We propose to formulate the prediction task as learning temporal dynamics through stochastic residual updates to a latent representation of the state at each time step. We name our model "StretchBEV" because our method is much more capable than the existing ones (*streched*) both temporally and spatially.

The main contribution is formulating the problem as learning temporal dynamics. We propose to use a state-space model with stochastic residual updates to a latent representation at each time step. With this formulation, the prediction problem suffices to a state prediction problem, which we handle with stochastic residual updates. With its strong latent variable, *separated for each time step*, our methods can generate highly diverse future predictions while preserving the accuracy of the predictions. Our proposed approach clearly outperforms the state-of-the-art in various metrics evaluating the decoded predictions, especially by large margins in challenging cases of spatially far regions and temporally long spans. We also evaluate the diversity of the predictions both quantitatively and qualitatively and clearly show that our method outperforms the existing methods.

1.3 Contributions and Novelties

Our contributions are as follows:

- We propose "SLAMP", a novel method for stochastic video future frame prediction.
- We achieve comparable performance for generic video prediction datasets and outperform the existing ones in real-world scenarios.
- Our method not only predicts the future frame but also predicts the future optical flow without seeing the target frame.
- We propose "SLAMP-3D", a novel 3D aware stochastic video prediction method that can predict future frames of the video using decomposition of 3D-aware scene structure and residual motion.
- Our method achieves comparable performance on real-world driving datasets while having a much more inference speed
- Our method requires 1 second for generating 10 samples of a scene which is 40 $\times$ faster than the state-of-the-art method without a significant loss of performance.
- We propose StretchBEV, a novel method for predicting future instances and their segmentation masks.
- We clearly outperform state-of-the-art methods, especially by large margins in challenging cases of spatially far regions and temporally long spans.

- Our method has the same inference speed as the existing ones while having much more performance and diversity.

1.3.1 Publications

- Adil Kaan Akan, Fatma Güney, "StretchBEV: Stretching Future Instance Prediction Spatially and Temporally", Under Review
- Adil Kaan Akan, Sadra Safadoust, Erkut Erdem, Aykut Erdem, Fatma Güney, "Stochastic Video Prediction with Structure and Motion", Under Review
- Adil Kaan Akan, Sadra Safadoust, Erkut Erdem, Aykut Erdem, Fatma Güney, "SLAMP: Stochastic Latent Appearance and Motion Prediction", Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021.

1.4 Outline of the Thesis

Chapter 2 gives a brief background on autoregressive models and state-space models which we use in our proposed models.

Chapter 3 summarizes the previous work done on the future prediction, specifically on video and Bird's-Eye view representation.

Chapter 4 introduces the SLAMP model and its contribution, *motion history*. We first discuss the existing work done in the area of stochastic video prediction. Then, we present our method and its details. In the end, we demonstrate detailed experiments and comparison to existing state-of-the-art methods in this chapter.

Chapter 5 describes the SLAMP-3D and its decomposition strategy for 3D structure and residual motion. The details about its decomposition and conditioning are provided in this chapter. Moreover, we present detailed experiments and comparisons to existing work.

Chapter 6 explains the StretchBEV and its novel formulation of temporal dynamics updates. We first discuss the previous work done in Bird's-eye view representation space, then we explain our formulation. In the end, we show experimental results that achieve new state-of-the-art in various metrics.

Chapter 7 concludes this thesis with a brief summary and presents possible directions for future work.

Chapter 2

BACKGROUND

In this chapter, we cover methods of modelling of future prediction, specifically autoregressive models and state-space models. We introduce the basic concepts following the previous work [Denton and Fergus, 2018, Franceschi et al., 2020] since our proposed methods are building on top of them.

2.1 Autoregressive Models

For Autoregressive Models, we follow SVG [Denton and Fergus, 2018], where the future information is encoded into latent variable using a learned prior and posterior distribution pair.

SVG Architecture:

Denton and Fergus [Denton and Fergus, 2018] introduced SVG, an autoregressive model for stochastic video prediction. SVG encodes the future information into latent variables. They propose two different versions for their models, fixed-prior and learned-prior. In both versions, the posterior distribution is learned from the future time steps. However, the prior distribution changes depending on the version. In the fixed prior, the prior distribution is assumed to be a fixed standard Gaussian distribution. On the other hand, in the learned prior, the prior distribution is learnt from the previous and current time steps. They showed that learned prior outperforms the fixed one because the prior distribution has a flexibility to adapt according to the current time step. Below, we cover the steps of the algorithm.

SVG encodes each image into a feature space using CNNs.

$$\begin{aligned} \mathbf{h}_{t-1} &= Enc(\mathbf{x}_{t-1}) \qquad (2.1) \\ \mathbf{h}_t &= Enc(\mathbf{x}_t) \end{aligned}$$

where $\mathbf{x}_{1:T}$ are the frames of the video, *Enc* is CNN-based encoder and, $h_{1:T}$ are the encoded features of the corresponding video frame. Then, from the encoded features of current and future time steps, two separate distribution pairs, prior and posterior, are learnt.

$$\begin{aligned}\mu_\psi(t), \sigma_\psi(t) &= \mathrm{LSTM}_\psi(h_{t-1}) \\ \mu_\phi(t), \sigma_\phi(t) &= \mathrm{LSTM}_\phi(h_t)\end{aligned} \tag{2.2}$$

where μ_ψ, σ_ψ are neural networks learning the prior distribution and μ_ϕ, σ_ϕ are neural networks learning the posterior distribution. After sampling the latent variable from the corresponding distribution, posterior in training and prior in the evaluation, the next frame's features are predicted by a separate model from combination of previous frame's features, $\mathbf{h}_{t-1}$ and the latent variable, $\mathbf{z}_t$. Note that in the fixed prior version, the prior distribution is not learnt. Therefore, the networks, μ_ψ, σ_ψ, do not exist and the latent variable is sampled from $\mathcal{N}(0, 1)$.

$$\mathbf{h}_t = \mathrm{LSTM}_\theta(\mathbf{h}_{t-1}, \mathbf{z}_t) \tag{2.3}$$

After predicting the next frame's features, it is straightforward to decode it using CNNs into next frame.

$$\mathbf{x}_t = Dec(\mathbf{h}_t) \tag{2.4}$$

where Dec is a CNN-based decoder and decodes the features into video frames.

The whole process is repeated for each time step, where the model uses its predictions for the unseen steps. This means that the error accumulates throughout the generation process and towards to end, the generated video frames get blurry.

2.2 State-Space Models

As opposed to interleaving process of auto-regressive models, state-space models separate the frame generation from the modelling of dynamics [Gregor and Besse, 2018]. State-of-the-art method SRVP [Franceschi et al., 2020] proposes a state-space model for video generation with deterministic state transitions representing residual change between the frames. In this chapter, we follow SRVP [Franceschi et al., 2020], where they consider the problem of stochastic video prediction as a distribution of possible future frames given the conditioning frames of a video, and briefly summarize the approach.

SRVP Architecture:

Franceschi et al. [Franceschi et al., 2020] introduced SRVP, a state-space model for stochastic video prediction model. SRVP encodes the scene into a stochastic latent variable and uses time-independent residual updates to create a different state representation for each time step. The SRVP separates the content and the dynamic by using a separate deterministic content network to encode the scene details and using it for the decoding of future frames. In this way, the latent state will learn the dynamics, e.g. motion in the scene, and the content part will learn the details of the scene.

In each time step, SRVP [Franceschi et al., 2020] creates a state variable, $\mathbf{y}_t$, by using a stochastic residual network to update previous states into future ones.

$$\mathbf{y}_{t+\Delta t} = \mathbf{y}_t + \Delta t \cdot f_\theta\big(\mathbf{y}_t, \mathbf{z}_{\lfloor t \rfloor + 1}\big). \tag{2.5}$$

where $\mathbf{y}_t$ is the state representation at time t, Δt is a hyperparameter that defines the size of the updates, $\mathbf{z}$ is the latent variable sampled from prior or posterior distribution according to train or inference time. The prior distribution is learnt from current state, $\mathbf{y}_t$ whereas the posterior distribution is learnt from the future video frames, $\mathbf{x}_{t+1:T}$.

$$\mathbf{z}_t \sim \mathcal{N}(\mu_\theta(\mathbf{y}_t), \sigma_\theta(\mathbf{y}_t)\mathrm{I}) \tag{2.6}$$

where μ_θ and σ_θ are the neural networks learning the prior distribution from the current state.

As stated before, SRVP decompose the content and dynamics into separate spaces. SRVP learns a specific content variable, which is purely deterministic and stays the same throughout the video. That is, they assume the background is static and does not change. Since the content variable is available during the decoding of each frame, the network forces the latent variables to focus on the dynamics rather than the content of the video.

$$\mathbf{w} = \mathbf{c}_{\psi}(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K) \tag{2.7}$$

where $\mathbf{w}$ is the content variable which stays the same throughout the video, $\mathbf{c}_{\psi}$ is a neural network calculating the content variable, $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_K$ are the corresponding image frames.

After creating state representation for each time step and calculating the content variable, SRVP decodes all of them into video frames, e.g. images, which can be seen in (4.8).

$$\mathbf{x}_t = g_{\theta}\big(\mathbf{y}_t, \mathbf{w}\big). \tag{2.8}$$

where $\mathbf{x}_t$ is the image frame at time t, g_{θ} is the decoder parameterized by θ, $\mathbf{y}_t$ is the state representation at time t, $\mathbf{w}$ is the content feature extracted by deterministic content network.

2.3 Comparison between Autoregressive and State-Space Models

As we cover both models in this chapter, we briefly compare them in this section.

Autoregressive models use their predictions to propagate into the future. To predict further into the future, models need to use their predictions. This brings two drawbacks. First, the predictions are not 100% correct. Each prediction has an error, even if the error is too small. Since the predictions are fed back into the models, the errors accumulate as the model rolls out into the future. Second, their run-time performance is slow compared to state-space models. Since the model needs to use its predictions further into the future, it first needs to generate the previous time steps. For example, to generate two steps into the future, the model needs to generate one step into the future. This sequential mechanism costs their run-time performance.

State-space models move the problem of prediction into a different state representation than the input representation, which is much smaller than the input most of the time. In this way, the prediction is made on the state-space, which is much faster due to the small dimensionality of the state-space. When all the states are predicted, the states are decoded into the input or output representation, which simply separates the prediction and generation. However, the problem with state-space models is that they assume the background static. For example, SRVP generates a deterministic content variable that stays the same throughout the video, see (2.7). The assumption of static background holds for several datasets such as KTH [Schüldt et al., 2004b] and BAIR [Ebert et al., 2017b]. However, in real-world scenarios, this assumption breaks. For example, in the driving scenarios, the background is not still, moving according to the ego-motion. As we show in Chapter 4, SRVP performs poorly on two autonomous driving datasets with moving background.

Chapter 3

LITERATURE REVIEW

3.1 Future Prediction in Video

In this section, we cover the previous work focusing on video prediction. We first focus on the previous work specifically in Section 3.1.1, then we cover the work done on video decomposition, and related works on 3D structure and motion in Sections 3.1.2 and 3.1.3, respectively.

3.1.1 Stochastic Video Prediction

SV2P [Babaeizadeh et al., 2018] and SVG [Denton and Fergus, 2018] are the first to model the stochasticity in video sequences using latent variables. The input from past frames are encoded in a posterior distribution to generate the future frames. In a stochastic framework, learning is performed by maximizing the likelihood of the observed data and minimizing the distance of the posterior distribution to a prior distribution, either fixed [Babaeizadeh et al., 2018] or learned from previous frames [Denton and Fergus, 2018]. Since time-variance is proven crucial by these previous works, we sample a latent variable at every time step. Sampled random variables are fed to a frame predictor, modelled recurrently using an LSTM. Typically, each distribution, including the prior and the posterior, is modeled with a recurrent model such as an LSTM.

There are various extensions to the basic formulation of stochastic video generation. Villegas et al. [Villegas et al., 2019] replace the linear LSTMs with convolutional ones at the cost of increasing the number of parameters. Castrejon et al. [Castrejon et al., 2019] introduce a hierarchical representation to model latent variables at different scales which increases the complexity. Karapetyan et al. [Karapetyan et al., 2022] proposes usage of hierarchical recurrent networks to generate high quality video frames using multi-scale architectures. Assouel et al. [Assouel et al., 2022] introduce Variational Independent Modules to learn entity-centric representations and their transitions throughout the video. Babaeizadeh et al. [Babaeizadeh et al., 2021] over-parameterize an existing architecture to first overfit to the training set, and then use data augmentation to generalize to the validation or test sets. Lee et al. [Lee et al., 2018] incorporate an adversarial loss into the stochastic framework to generate sharper images, at the cost of less diverse results. We also use convolutional LSTMs to generate diverse and sharp-looking results without any adversarial losses by first reducing the spatial resolution to reduce the cost. Future prediction is typically performed in the pixel space but there are other representations such as keypoints [Villegas et al., 2017, Minderer et al., 2019] , coordinates [Gu et al., 2021a, Gao et al., 2020a] and birds-eye view [Hu et al., 2021, Akan and Güney, 2022].

Optical flow has been used before in future prediction [Li et al., 2018, Lu et al., 2017]. Li et al. [Li et al., 2018] generate future frames from a still image by using optical flow generated by an off-the-shelf model, whereas we compute flow as part of prediction. Lu et al. [Lu et al., 2017] use optical flow for video extrapolation and interpolation without modeling stochasticity. Long-term video extrapolation results show the limitation of this work in terms of predicting future due to relatively small motion magnitudes considered in extrapolation. Differently from flow, Xue et al. [Xue et al., 2016] model the motion as image differences using cross convolutions.

3.1.2 Video Decomposition

The previous work explored motion cues for video generation either explicitly with optical flow [Walker et al., 2015, Walker et al., 2016, Liang et al., 2017, Liu et al., 2017, Lu et al., 2017, Fan et al., 2019, Gao et al., 2019] or implicitly with temporal differences [Lotter et al., 2017] or pixel-level transformations [Jia et al., 2016, Vondrick and Torralba, 2017]. There are some common factors among these methods such as using recurrent models [Shi et al., 2015, Lotter et al., 2017, Fan et al., 2019], specific processing of dynamic parts [Jia et al., 2016, Liang et al., 2017, Fan et al., 2019, Gao et al., 2019], utilizing a mask [Finn et al., 2016, Liu et al., 2017, Gao et al., 2019], and adversarial training [Vondrick and Torralba, 2017, Lu et al., 2017]. In our models, we also use recurrent models, predict a mask, and separately process motion, but in a stochastic way.

Specifically, state-space model SRVP [Franceschi et al., 2020] learns a content variable from the first few frames which remains unchanged while predicting the future frames. As we will show in the next chapter, Chapter 4, the content variable in SRVP cannot handle changes in the background. In addition to pixel space, SLAMP separately models changes in motion and keeps track of a motion history. This reduces the role of the pixel decoder to recover occlusions, e.g. around motion boundaries. We propose to decompose the scene as static and dynamic where the static part not only considers the ego-motion but also the structure. This allows us to differentiate the motion in the background from the motion in the foreground which leads to a more meaningful scene decomposition for driving. Disentanglement of explicit groups of factors has been explored before for generating 3D body models depending on the body pose and the shape [Skafte and Hauberg, 2019]. Inspired by the conditioning of the body pose on the shape, in SLAMP-3D, we condition the motion of the dynamic part on the static part to achieve a disentanglement between the two types of motion in driving.

3.1.3 Structure and Motion

Our approach, SLAMP-3D is related to view synthesis [Garg et al., 2016] where the target view can be synthesized by warping a source view based on the depth estimation from the target view [Jaderberg et al., 2015]. The difference between the synthesized target view and the original one can be used for self-supervised training. Monocular depth estimation approaches [Zhou et al., 2017, Zhan et al., 2018, Wang et al., 2018, Bian et al., 2019, Mahjourian et al., 2018, Godard et al., 2019, Guizilini et al., 2020] generalize view synthesis to adjacent frames by also estimating the relative pose from one frame to the next. These methods can successfully model the structure and motion in the static part of the scene. However, the motion of independently moving objects remains as a source of error. More recent works [Yin and Shi, 2018, Zou et al., 2018, Chen et al., 2019, Ranjan et al., 2019, Luo et al., 2019] estimate the residual optical flow to model the motion of independently moving objects. In addition, some of these methods model the consistency between depth and optical flow [Zou et al., 2018], and also motion segmentation [Ranjan et al., 2019, Luo et al., 2019, Safadoust and Gney, 2021]. In SLAMP-3D, we similarly learn the decomposition of the world into static and dynamic parts but in longer sequences by exploiting the history from previous frames to predict future frames with uncertainty.

3.2 Future Prediction in Bird's-Eye View

In this section, we first compare the autoregressive models and state-space models in Section 3.2.1. Then, we focus on the future prediction in driving scenarios in Section 3.2.2.

Our work, StretchBEV introduced in Chapter 6, fits into the stochastic future prediction framework, producing long term, diverse predictions, however, we predict future in the Bird's-Eye view (BEV) space instead of the noisy pixel space.

3.2.1 Autoregressive Models vs. State-Space Models

As introduced in Section 2.3, autoregressive method have several problems such as error accumulation and low runtime speed. The performance of auto-regressive methods can be improved by increasing the network capacity [Villegas et al., 2019] or introducing a hierarchy into the latent variables [Castrejon et al., 2019], which also increase the complexity of these methods. Due to complexity of predicting pixels, another group of work moves away from the pixel space to the keypoints [Minderer et al., 2019] or to the motion space by incorporating motion history as we propose in both SLAMP and SLAMP-3D. Our proposed approach follows a similar strategy by performing future prediction in the BEV representation, but more efficiently by avoiding auto-regressive predictions.

Auto-regressive strategy requires encoding the predictions, leading to high computational cost and creates a tight coupling between the temporal dynamics and the generation process [Gregor and Besse, 2018, Rubanova et al., 2019].

On the other hand, the state-space models (SSM) break this coupling by separating the learning of dynamics from the generation process, resulting in a computationally more efficient approach. Furthermore, learned states can be directly used in reinforcement learning [Gregor and Besse, 2018] and can be interpreted [Rubanova et al., 2019], making SSMs particularly appealing for self-driving. Earlier SSMs with variational deep inference suffer from complicated inference schemes and typically target low-dimensional data [Krishnan et al., 2017, Karl et al., 2017]. An efficient training strategy with a temporal model based on residual updates is proposed for high-dimensional video prediction in the state of the art SSM [Franceschi et al., 2020]. We adapt a similar residual update strategy for predicting future BEV representations. We also experimentally show that the content variable for the static part of the scene is not as effective in the BEV space as it is in pixel space [Franceschi et al., 2020].

3.2.2 Future Prediction in Driving Scenarios

The typical approach to the prediction problem in self-driving is to first perform detection and tracking, and then do the trajectory prediction [Chai et al., 2020, Hong et al., 2019]. In these methods, errors are propagated at each step. There are some recent methods [Luo et al., 2018, Casas et al., 2018, Casas et al., 2020, Djuric et al., 2021] which directly address the prediction problem from the sensory input including LiDAR, radar, and other sensors. These methods also typically rely on an HD map of the environment. Due to high performance and efficiency of end-to-end approach, we follow a similar approach for future prediction but using cameras only and without relying on HD maps.

Despite their efficiency, most of the previous work focus on the most likely prediction [Casas et al., 2018] or only models the uncertainty regarding the ego-vehicle's trajectory [Casas et al., 2020, Djuric et al., 2021]. The motion prediction methods which consider the behavior of all the agents in the scene typically assume a top-down rasterised representation as input, e.g. Argoverse setting [Chang et al., 2019]. Even then, multiple future prediction problem is typically addressed by generating a fixed number of predictions [Gao et al., 2020b, Liang et al., 2020], for example by estimating the likelihood of multiple target locations [Zhao et al., 2020b, Gu et al., 2021b]. There are some exceptions [Tang and Salakhutdinov, 2019b, Sriram et al., 2020, Huang et al., 2020] which directly address the diversity aspect with a probabilistic framework. These works, especially the ones using a deep variational framework [Tang and Salakhutdinov, 2019b, Sriram et al., 2020] are similar to our approach in spirit, however, they operate in the coordinate space by assuming the availability of a top-down map where locations of agents are marked. In this thesis, we aim to learn this top-down BEV representation from multiple cameras by also segmenting the agents in the scene.

FIERY [Hu et al., 2021] is the first to address probabilistic future prediction from multiple cameras. However, future predictions are limited both in terms of diversity and length considering the typical video prediction setting. In Chapter 6, we propose a probabilistic future prediction method that can generate diverse predictions extending to different temporal horizons with a stochastic temporal dynamics model.

Chapter 4

SLAMP: STOCHASTIC LATENT APPEARANCE AND MOTION PREDICTION

4.1 Introduction

Motion is an important clue in the videos. The temporal connection between the frames is crucial for improving the performance of any task in video representation. Most of the time, the motion in the video stays the same or does not change much throughout the video duration. For example, if a person walks at the beginning of a video, then most probably, the person will continue walking. With this insight, we introduce the concept of motion history. The motion history corresponds to explicitly modeling the past motion in the video and propagating it to the future of the video. In this way, the future frames are generated using the motion information in the video.

In this chapter, we introduce SLAMP, which is a stochastic future prediction method for videos. In SLAMP, we exploit the *motion history* concept and generate the future frames using the motion information.

Our method is summarized as follows: We first sample latent variables from two separate distributions, one for appearance and one for motion, which is learnt from the images. The latent variables contain the information from the future time steps, similar to SVG [Denton and Fergus, 2018]. After, we simultaneously generate the future target frame in both the pixel space and the motion space using the sampled latent variables. The pixel space corresponds to that generating the target frame as an image. On the other hand, the motion space corresponds to generating the motion from the current frame to the target frame, then synthesizing the target frame. So, SLAMP predicts the future frame in two different modalities. In the end, a mask is predicted to combine both predictions in the best way.

Using the motion history in a stochastic way boosts performance in challenging real-world driving scenarios with a moving background. One example case can be seen in Fig. 4.1. In the high-speed scenarios, our model can preserve the shape of the car (top example in Fig. 4.1), and in dense environments, our model can conserve the details of the car, such as the license plate.

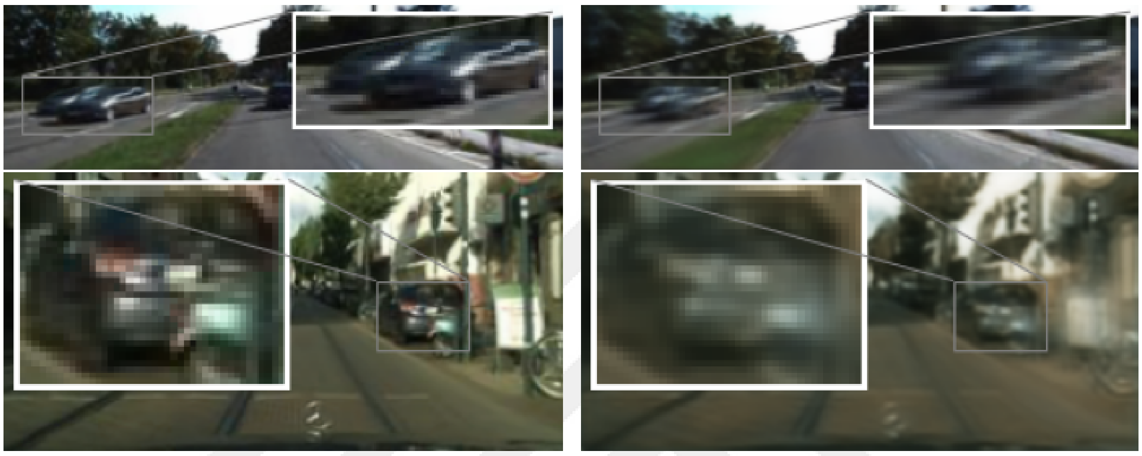

Figure 4.1: Comparison of the first prediction frames (11th) SLAMP (**left**) vs. state-of-the-art method, SRVP [Franceschi et al., 2020] (**right**) on KITTI [Geiger et al., 2013] (**top**) and Cityscapes [Cordts et al., 2016] (**bottom**) datasets. Our method can predict both foreground and background objects better than SRVP. Full sequence predictions can be seen in Appendix.

In this chapter, we first explain our approach in detail in Section 4.2, then we validate our claims on motion history with the experiments presented in Section 4.3.

4.2 Methodology

4.2.1 Stochastic Video Prediction

Given the previous frames $\mathbf{x}_{1:t-1}$ until time t, our goal is to predict the target frame $\mathbf{x}_t$. For that purpose, we assume that we have access to the target frame $\mathbf{x}_t$ during training and use it to capture the dynamics of the video in stochastic latent variables $\mathbf{z}_t$. By learning to approximate the distribution over $\mathbf{z}_t$, we can decode the future frame $\mathbf{x}_t$ from $\mathbf{z}_t$ and the previous frames $\mathbf{x}_{1:t-1}$ at test time.

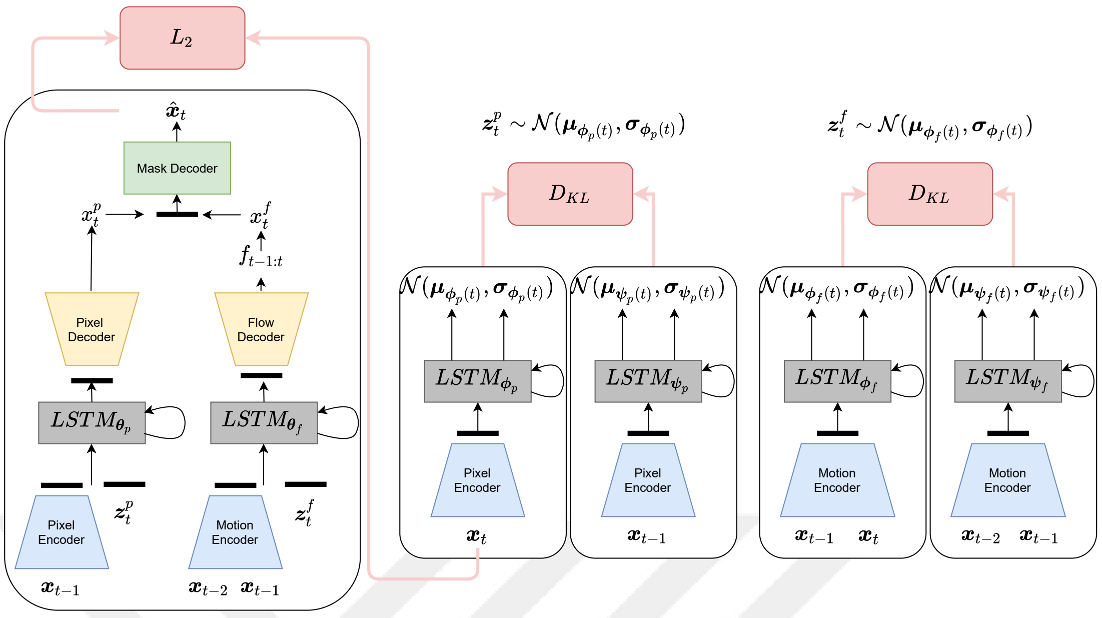

Figure 4.2: **SLAMP.** This figure shows the components of our SLAMP model including the prediction model, inference and learned prior models for pixel and then flow from left to right. Observations $\mathbf{x}_t$ are mapped to the latent space by using a pixel encoder for appearance on each frame and and a motion encoder for motion between consecutive frames. The blue boxes show encoders, yellow and green ones decoders, gray ones recurrent posterior, prior, and predictor models, and lastly red ones show loss functions during training. Note that L_2 loss is applied three times for appearance prediction $\mathbf{x}_t^p$, motion prediction $\mathbf{x}_t^f$, and the combination of the two $\hat{\mathbf{x}}_t$ according to the mask prediction $\mathbf{m}(\mathbf{x}_t^p, \mathbf{x}_t^f)$. We only show L2 loss between the actual frame $\mathbf{x}_t$ and the final predicted frame $\hat{\mathbf{x}}_t$ in the figure. For inference, only the prediction model and learned prior models are used.

Using all the frames including the target frame, we compute a posterior distribution $q_\phi(\mathbf{z}_t|\mathbf{x}_{1:t})$ and sample a latent variable $\mathbf{z}_t$ from this distribution at each time step. The stochastic process of the video is captured by the latent variable $\mathbf{z}_t$. In other words, it should contain information accumulated over the previous frames rather than only condensing the information on the current frame. This is achieved by encouraging $q_\phi(\mathbf{z}_t|\mathbf{x}_{1:t})$ to be close to a prior distribution $p(\mathbf{z})$ in terms of KL-divergence. The prior can be sampled from a fixed Gaussian at each time step or can be learned from previous frames up to the target frame $p_\psi(\mathbf{z}_t|\mathbf{x}_{1:t-1})$. We prefer the latter one as it is shown to work better by learning a prior that varies across time [Denton and Fergus, 2018].

The target frame $\mathbf{x}_t$ is predicted based on the previous frames $\mathbf{x}_{1:t-1}$ and the latent vectors $\mathbf{z}_{1:t}$.

In practice, we only use the latest frame $\mathbf{x}_{t-1}$ and the latent vector $\mathbf{z}_t$ as input and dependencies from further previous frames are propagated with a recurrent model. The output of the frame predictor $\mathbf{g}_t$

contains the information required to decode $\mathbf{x}_t$.

Typically, $\mathbf{g}_t$ is decoded to a fixed-variance Gaussian distribution whose mean is the predicted target frame $\hat{\mathbf{x}}_t$ [Denton and Fergus, 2018].

4.2.2 SLAMP

We call the predicted target frame, appearance prediction $\mathbf{x}_t^p$ in the pixel space. In addition to $\mathbf{x}_t^p$, we also estimate optical flow $\mathbf{f}_{t-1:t}$ from the previous frame $t-1$ to the target frame t. The flow $\mathbf{f}_{t-1:t}$ represents the motion of the pixels from the previous frame to the target frame. We reconstruct the target frame $\mathbf{x}_t^f$ from the estimated optical flow via differentiable warping [Jaderberg et al., 2015]. Finally, we estimate a mask $\mathbf{m}(\mathbf{x}_t^p, \mathbf{x}_t^f)$ from the two frame estimations to combine them into the final estimation $\hat{\mathbf{x}}_t$:

$$\hat{\mathbf{x}}_t = \mathbf{m}(\mathbf{x}_t^p, \mathbf{x}_t^f) \odot \mathbf{x}_t^p + (\mathbf{1} - \mathbf{m}(\mathbf{x}_t^p, \mathbf{x}_t^f)) \odot \mathbf{x}_t^f \tag{4.1}$$

where $\odot$ denotes element-wise Hadamard product and $\mathbf{x}_t^f$ is the result of warping the source frame to the target frame according to the estimated flow field $\mathbf{f}_{t-1:t}$. Especially in the dynamic parts with moving objects, the target frame can be reconstructed accurately using motion information. In the occluded regions where motion is unreliable, the model learns to rely on the appearance prediction. The mask prediction learns a weighting between the appearance and the motion predictions for combining them.

We call this model SLAMP-Baseline because it is limited in the sense that it only considers the motion with respect to the previous frame while decoding the output.

In SLAMP, we extend the stochasticity in the appearance space to the motion space as well. This way, we can model appearance changes and motion patterns in the video explicitly and make better predictions of future. Fig. 4.2 shows an illustration of SLAMP (see Appendix for SLAMP-Baseline).

In order to represent appearance and motion, we compute two separate posterior distributions $q_{\phi_p}(\mathbf{z}_t^p|\mathbf{x}_{1:t})$ and $q_{\phi_f}(\mathbf{z}_t^f|\mathbf{x}_{1:t})$, respectively. We sample two latent variables $\mathbf{z}_t^p$ and $\mathbf{z}_t^f$ from these distributions in the pixel space and the flow space. This allows a decomposition of the video into static and dynamic components. Intuitively, we expect the dynamic component to focus on changes and the static to what remains constant from the previous frames to the target frame. If the background is moving according to a camera motion, the static component can model the change in the background assuming that it remains constant throughout the video, e.g. ego-motion of a car.

The Motion History: The latent variable $\mathbf{z}_t^f$ should contain motion information accumulated over the previous frames rather than local temporal changes between the last frame and the target frame. We achieve this by encouraging $q_{\phi_f}(\mathbf{z}_t^f|\mathbf{x}_{1:t})$ to be close to a prior distribution in terms of KL-divergence. Similar to [Denton and Fergus, 2018], we learn the motion prior conditioned on previous frames up to the target frame: $p_{\psi_f}(\mathbf{z}_t^f|\mathbf{x}_{1:t-1})$. We repeat the same for the static part represented by $\mathbf{z}_t^p$ with posterior $q_{\phi_p}(\mathbf{z}_t^p|\mathbf{x}_{1:t})$ and the learned prior $p_{\psi_p}(\mathbf{z}_t^p|\mathbf{x}_{1:t-1})$.

4.2.3 Variational Inference

For our basic formulation (SLAMP-Baseline), the derivation of the loss function is straightforward and provided in Appendix. For SLAMP, the conditional joint probability corresponding to the graphical model in Fig. 4.3 is:

$$p(\mathbf{x}_{1:T}) = \prod_{t=1}^{T} p(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \mathbf{z}_t^p, \mathbf{z}_t^f) \\ p(\mathbf{z}_t^p|\mathbf{x}_{1:t-1}, \mathbf{z}_{t-1}^p) \;\; p(\mathbf{z}_t^f|\mathbf{x}_{1:t-1}, \mathbf{z}_{t-1}^f) \tag{4.2}$$

The true distribution over the latent variables $\mathbf{z}_t^p$ and $\mathbf{z}_t^f$ is intractable. We train time-dependent inference networks $q_{\phi_p}(\mathbf{z}_t^p|\mathbf{x}_{1:T})$ and $q_{\phi_f}(\mathbf{z}_t^f|\mathbf{x}_{1:T})$ to approximate the

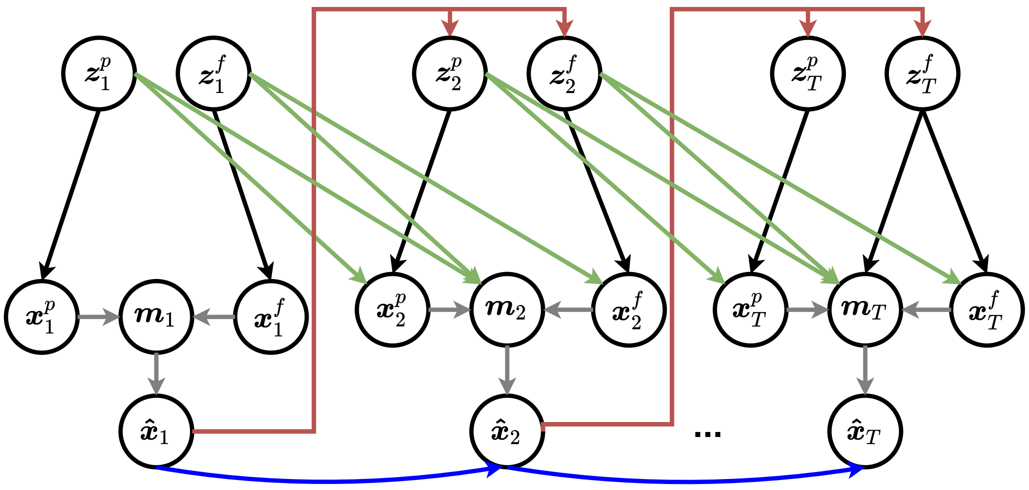

Figure 4.3: **Generative Model of SLAMP.** The graphical model shows the generation process of SLAMP with motion history. There are two separate latent variables for appearance $\mathbf{z}_t^p$ and motion $\mathbf{z}_t^f$ generating frames $\mathbf{x}_t^p$ and $\mathbf{x}_t^f$ (black). Information is propagated between time-steps through the recurrence between frame predictions (blue), corresponding latent variables (green), and from frame predictions to latent variables (red). The final prediction $\hat{\mathbf{x}}_t$ is a weighted combination of the $\mathbf{x}_t^p$ and $\mathbf{x}_t^f$ according to the mask $\mathbf{m}(\mathbf{x}_t^p, \mathbf{x}_t^f)$. Note that predictions at a time-step depend on all of the previous time-steps recurrently, but only the connections between consecutive ones are shown for clarity.

true distribution with conditional Gaussian distributions. In order to optimize the likelihood of $p(\mathbf{x}_{1:T})$, we need to infer latent variables $\mathbf{z}_t^p$ and $\mathbf{z}_t^f$, which correspond to uncertainty of static and dynamic parts in future frames, respectively. We use a variational inference model to infer the latent variables.

Since $\mathbf{z}_t^p$ and $\mathbf{z}_t^f$ are independent across time, we can decompose Kullback-Leibler terms into individual time steps. We train the model by optimizing the variational lower bound (see Appendix for the derivation):

$$\begin{aligned}
\log p_{\boldsymbol{\theta}}(\mathbf{x}) &\geq \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}_p,\boldsymbol{\phi}_f,\boldsymbol{\psi}_p,\boldsymbol{\psi}_f}(\mathbf{x}_{1:T}) \\
&= \sum_t \mathbb{E}_{\substack{\mathbf{z}^p_{1:t}\sim q_{\boldsymbol{\phi}_p} \\ \mathbf{z}^f_{1:t}\sim q_{\boldsymbol{\phi}_f}}} \log p_{\boldsymbol{\theta}}(\mathbf{x}_t|\mathbf{x}_{1:t-1},\mathbf{z}^p_{1:t},\mathbf{z}^f_{1:t}) \\
&\qquad - \beta\Big[D_{\mathrm{KL}}\big(q(\mathbf{z}^p_t|\mathbf{x}_{1:t})\;||\;p(\mathbf{z}^p_t|\mathbf{x}_{1:t-1})\big) + D_{\mathrm{KL}}\big(q(\mathbf{z}^f_t|\mathbf{x}_{1:t})\;||\;p(\mathbf{z}^f_t|\mathbf{x}_{1:t-1})\big)\Big]
\end{aligned} \tag{4.3}$$

The likelihood $p_{\boldsymbol{\theta}}$, can be interpreted as an L_2 penalty between the actual frame $\mathbf{x}_t$ and the estimation $\hat{\mathbf{x}}_t$ as defined in (4.1). We apply the L_2 loss to the predictions of appearance and motion components as well.

The posterior terms for uncertainty are estimated as an expectation over $q_{\boldsymbol{\phi}_p}(\mathbf{z}^p_t|\mathbf{x}_{1:t})$, $q_{\boldsymbol{\phi}_f}(\mathbf{z}^f_t|\mathbf{x}_{1:t})$. As in [Denton and Fergus, 2018], we also learn the prior distributions from the previous frames up to the target frame as $p_{\boldsymbol{\psi}_p}(\mathbf{z}^p_t|\mathbf{x}_{1:t-1})$, $p_{\boldsymbol{\psi}_f}(\mathbf{z}^f_t|\mathbf{x}_{1:t-1})$. We train the model using the re-parameterization trick [Kingma and Welling, 2014]. We classically choose the posteriors to be factorized Gaussian so that all the KL divergences can be computed analytically.

4.2.4 Architecture

We encode the frames with a feed-forward convolutional architecture to obtain appearance features at each time-step. In SLAMP, we also encode consecutive frame pairs into a feature vector representing the motion between them. We then train linear LSTMs to infer posterior and prior distributions at each time-step from encoded appearance and motion features.

Stochastic video prediction model with a learned prior [Denton and Fergus, 2018] is a special case of our baseline model with a single pixel decoder, we also add motion and mask decoders. Next, we describe the steps of the generation process for the dynamic part.

At each time step, we encode $\mathbf{x}_{t-1}$ and $\mathbf{x}_t$ into $\mathbf{h}^f_t$, representing the motion from the previous frame to the target frame. The posterior LSTM is updated based on the $\mathbf{h}^f_t$:

$$\begin{aligned}
\mathbf{h}^f_t &= \mathrm{MotionEnc}(\mathbf{x}_{t-1},\mathbf{x}_t) \\
\boldsymbol{\mu}_{\boldsymbol{\phi}_f(t)},\boldsymbol{\sigma}_{\boldsymbol{\phi}_f(t)} &= \mathrm{LSTM}_{\boldsymbol{\phi}_f}(\mathbf{h}^f_t)
\end{aligned} \tag{4.4}$$

For the prior, we use the motion representation $\mathbf{h}^f_{t-1}$ from the previous time step, i.e. the motion from the frame $t-2$ to the frame $t-1$, to update the prior LSTM:

$$\begin{aligned} \mathbf{h}^f_{t-1} &= \text{MotionEnc}(\mathbf{x}_{t-2}, \mathbf{x}_{t-1}) \\ \boldsymbol{\mu}_{\boldsymbol{\psi}_f(t)}, \boldsymbol{\sigma}_{\boldsymbol{\psi}_f(t)} &= \text{LSTM}_{\boldsymbol{\psi}_f}(\mathbf{h}^f_{t-1}) \end{aligned} \tag{4.5}$$

At the first time-step where there is no previous motion, we assume zero-motion by estimating the motion from the previous frame to itself.

The predictor LSTMs are updated according to encoded features and sampled latent variables:

$$\begin{aligned} \mathbf{g}^f_t &= \text{LSTM}_{\boldsymbol{\theta}_f}(\mathbf{h}^f_{t-1}, \mathbf{z}^f_t) \\ \boldsymbol{\mu}_{\boldsymbol{\theta}_f} &= \text{FlowDec}(\mathbf{g}^f_t) \end{aligned} \tag{4.6}$$

There is a difference between the train time and inference time in terms of the distribution the latent variables are sampled from. At train time, latent variables are sampled from the posterior distribution. At test time, they are sampled from the posterior for the conditioning frames and from the prior for the following frames. The output of the predictor LSTMs are decoded into appearance and motion predictions separately and combined into the final prediction using the mask prediction (Eq. (4.1)).

4.3 Experiments

We evaluate the performance of the proposed approach and compare it to the previous methods on three standard video prediction datasets including Stochastic Moving MNIST, KTH Actions [Schüldt et al., 2004a] and BAIR Robot Hand [Ebert et al., 2017a]. We specifically compare our baseline model (SLAMP-Baseline) and our model (SLAMP) to SVG [Denton and Fergus, 2018] which is a special case of our baseline with a single pixel decoder, SAVP [Lee et al., 2018], SV2P [Babaeizadeh et al., 2018], and lastly to SRVP [Franceschi et al., 2020]. We also compare our model to SVG [Denton and Fergus, 2018] and SRVP [Franceschi et al., 2020] on two different challenging real world datasets, KITTI [Geiger et al., 2012, Geiger et al.,

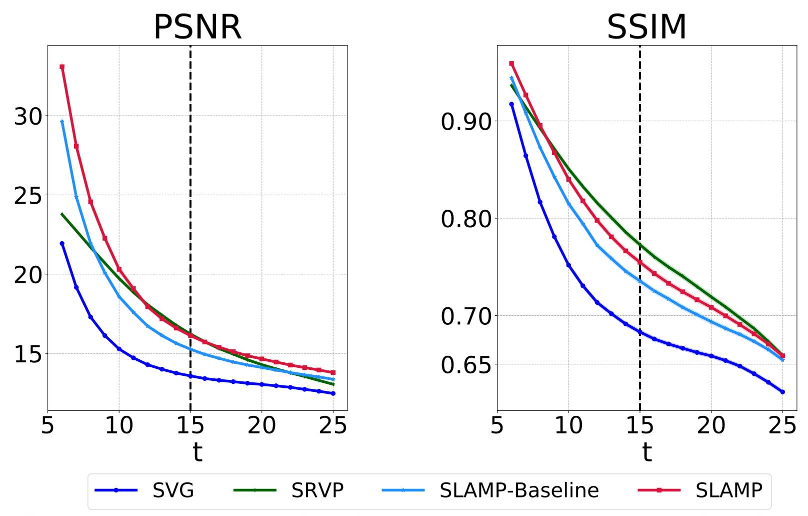

Figure 4.4: **Quantitative Results on MNIST.** This figure compares SLAMP to SLAMP-Baseline, SVG [Denton and Fergus, 2018], and SRVP [Franceschi et al., 2020] on MNIST in terms of PSNR (**left**) and SSIM (**right**). SLAMP clearly outperforms our baseline model and SVG, and performs comparably to SRVP. Vertical bars mark the length of the training sequences.

2013] and Cityscapes [Cordts et al., 2016], with moving background and complex object motion. We follow the evaluation setting introduced in [Denton and Fergus, 2018] by generating 100 samples for each test sequence and report the results according to the best one in terms of average performance over the frames. Our experimental setup including training details and parameter settings can be found in Appendix. We also share the code for reproducibility.

Table 4.1: **FVD Scores on the KTH and BAIR.**This table compares all the methods in terms of FVD scores with their 95%-confidence intervals over five different samples from the models. Our model is the second best on KTH and among top three methods on BAIR.

Dataset	SV2P	SAVP	SVG	SRVP	SLAMP - Baseline	SLAMP
KTH	636 ± 1	374 ± 3	377 ± 6	**222 ± 3**	236 ± 2	<u>228</u> ± 5
BAIR	965 ± 17	**152 ± 9**	255 ± 4	<u>163</u> ± 4	245 ± 5	—

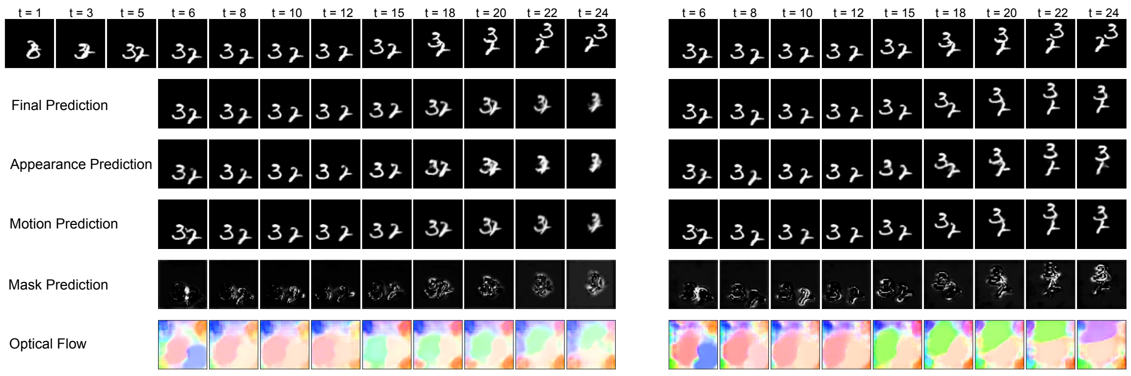

Figure 4.5: **SLAMP-Baseline (left) vs. SLAMP (right) on MNIST.** The top row shows the ground truth, followed by the frame predictions by the final, the appearance, the motion, and the last two rows show the mask and the optical flow predictions with false coloring. In this challenging case with bouncing and collisions, the baseline confuses the digits and cannot predict last frames correctly whereas SLAMP can generate predictions very close to the ground truth by learning smooth transitions in the motion history, as can be seen from optical flow predictions.

Evaluation Metrics: We compare the performance using three frame-wise metrics and a video-level one. Peak Signal-to-Noise Ratio (PSNR), *higher better*, based on L_2 distance between the frames penalizes differences in dynamics but also favors blur predictions. Structured Similarity (SSIM), *higher better*, compares local patches to measure similarity in structure spatially. Learned Perceptual Image Patch Similarity (LPIPS) [Zhang et al., 2018], *lower better*, measures the distance between learned features extracted by a CNN trained for image classification. Frechet Video Distance (FVD) [Unterthiner et al., 2019], *lower better*, compares temporal dynamics of generated videos to the ground truth in terms of representations computed for action recognition.

Stochastic Moving MNIST: This dataset contains up to two MNIST digits moving linearly and bouncing from walls with a random velocity as introduced in [Denton and Fergus, 2018]. Following the same training and evaluation settings as in the previous work, we condition on the first 5 frames during training and learn to predict the next 10 frames. During testing, we again condition on the first 5 frames

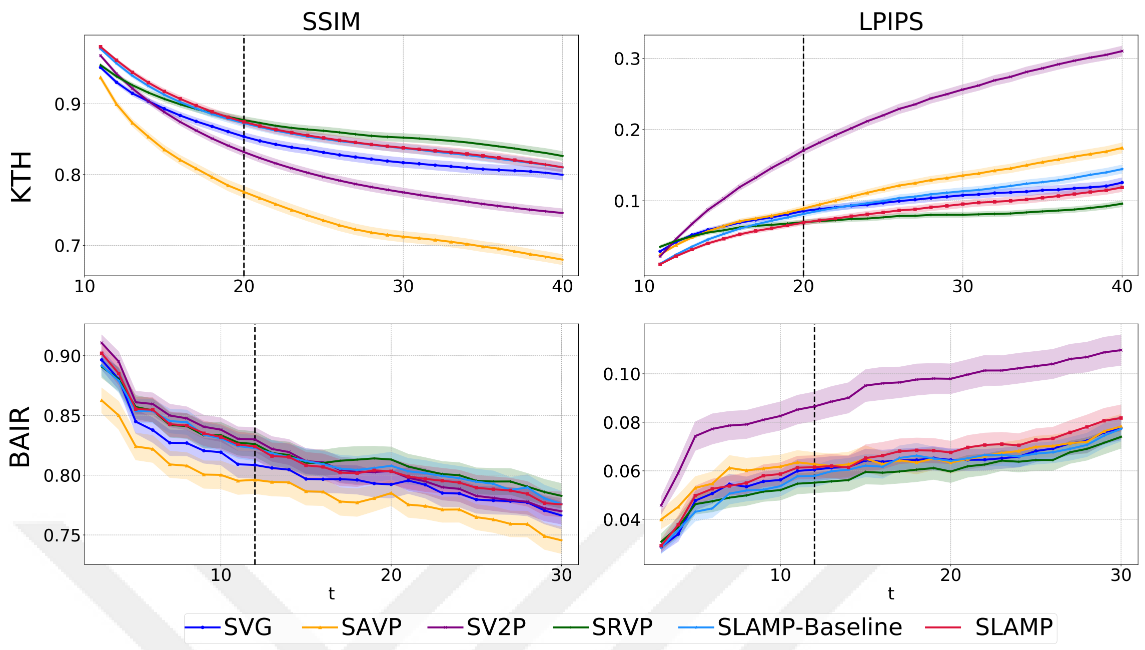

Figure 4.6: **Quantitative Results on KTH and BAIR.** We compare our results to previous work in terms of PSNR, SSIM, and LPIPS metrics with respect to the time steps on KTH (**top**), and BAIR (**bottom**) datasets, with 95%-confidence intervals. Vertical bars mark the length of training sequences. SLAMP outperforms previous work including SVG [Denton and Fergus, 2018], SAVP [Lee et al., 2018], SV2P [Babaeizadeh et al., 2018] and performs comparably to the state of the art method SRVP [Franceschi et al., 2020] on both datasets.

but predict the next 20 frames.

Fig. 4.4 shows quantitative results on MNIST in comparison to SVG [Denton and Fergus, 2018] and SRVP [Franceschi et al., 2020] in terms of PSNR and SSIM, omitting LPIPS as in SRVP. Our baseline model with a motion decoder (SLAMP-Baseline) already outperforms SVG on both metrics. SLAMP further improves the results by utilizing the motion history and reaches a comparable performance to the state of the art model SRVP. This shows the benefit of separating the video into static and dynamic parts in both state-space models (SRVP) and auto-regressive models (ours, SLAMP). This way, models can better handle challenging cases such as crossing digits as shown next.

We qualitatively compare SLAMP to SLAMP-Baseline on MNIST in Fig. 4.5.

Table 4.2: **Results on MNIST.** This table compares the results of SLAMP and SLAMP-Baseline to the previous work on MNIST dataset. Following the previous work, we report the results as the mean and the 95%-confidence interval in terms of PSNR and SSIM. Bold and underlined scores indicate the best and the second best performing method, respectively.

Models	PSNR	SSIM
SVG [Denton and Fergus, 2018]	14.50 ± 0.04	0.7090 ± 0.0015
SRVP [Franceschi et al., 2020]	$\underline{16.93 \pm 0.07}$	**0.7799** ± **0.0020**
SLAMP-Baseline	16.83 ± 0.06	0.7537 ± 0.0018
SLAMP	**18.07** ± **0.08**	$\underline{0.7736 \pm 0.0019}$

The figure shows predictions of static and dynamic parts as appearance and motion predictions, as well the final prediction as the combination of the two. According to the mask prediction, the final prediction mostly relies on the dynamic part shown as black on the mask and uses the static component only near the motion boundaries. Moreover, optical flow prediction does not fit the shape of the digits but expands as a region until touching the motion region of the other digit. This is due to the uniform black background. Moving a black pixel in the background randomly is very likely to result in another black pixel in the background, which means zero-loss for the warping result. Both models can predict optical flow correctly for the most part and resort to the appearance result in the occluded regions. However, continuity in motion is better captured by SLAMP with the colliding digits whereas the baseline model cannot recover from it, leading to blur results, far from the ground truth. Note that we pick the best sample for both models among 100 samples according to LPIPS.

KTH Action Dataset:

KTH dataset contains real videos where people perform a single action such as walking, running, boxing, etc. in front of a static camera [Schüldt et al., 2004a]. We

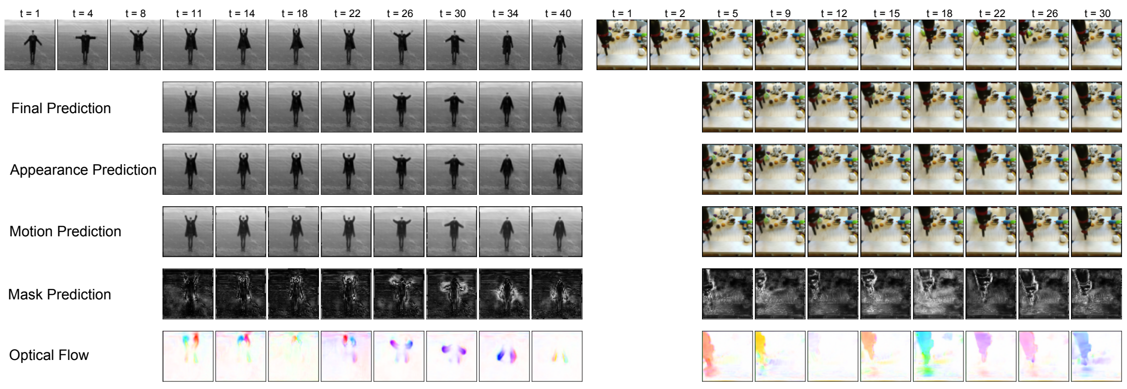

Figure 4.7: **Qualitative Results on KTH and BAIR.** We visualize the results of SLAMP on KTH (**left**) and SLAMP-Baseline on BAIR (**right**) datasets. The top row shows the ground truth, followed by the frame predictions by the final, the appearance, the motion, and the last two rows show the mask and the optical flow predictions. The mask prediction combines the appearance prediction (white) and the motion prediction (black) into the final prediction. Note that results on BAIR are without motion history because there are only two conditioning frames, i.e. one flow field to condition on.

expect our model with motion history to perform very well by exploiting regularity in human actions on KTH. Following the same training and evaluation settings used in the previous work, we condition on the first 10 frames and learn to predict the next 10 frames. During testing, we again condition on the first 10 frames but predict the next 30 frames.

Fig. 4.6 and Table 5.4 show quantitative results on KTH in comparison to previous approaches. Both our baseline and SLAMP models outperform previous approaches and perform comparably to SRVP, in all metrics including FVD. A detailed visualization of all three frame predictions as well as flow and mask are shown in Fig. 4.7. Flow predictions are much more fine-grained than MNIST by capturing fast motion of small objects such as hands or thin objects such as legs (see Appendix). The mask decoder learns to identify regions around the motion boundaries which cannot be matched with flow due to occlusions and assigns more weight to the appearance prediction in these regions.

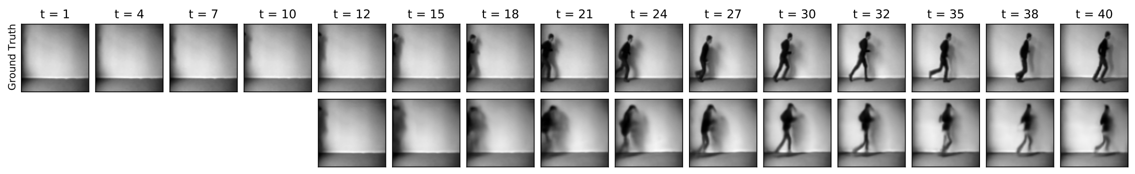

Figure 4.8: **Subject Appearing after the Conditioning Frames.** This figure shows a case where the subject appears after conditioning frames on KTH with ground truth (**top**) and a generated sample by our model (**bottom**). This shows our model's ability to capture dynamics of the dataset by generating samples close to the ground truth, even conditioned on empty frames.

Table 4.3: **Results on KTH.** This table compares the results of SLAMP and SLAMP-Baseline to the previous work on KTH dataset. Following the previous work, we report the results as the mean and the 95%-confidence interval in terms of PSNR, SSIM, and LPIPS. Bold and underlined scores indicate the best and the second best performing method, respectively.

Models	PSNR	SSIM	LPIPS
SV2P [Babaeizadeh et al., 2018]	28.19 ± 0.31	0.8141 ± 0.0050	0.2049 ± 0.0053
SAVP [Lee et al., 2018]	26.51 ± 0.29	0.7564 ± 0.0062	0.1120 ± 0.0039
SVG [Denton and Fergus, 2018]	28.06 ± 0.29	0.8438 ± 0.0054	0.0923 ± 0.0038
SRVP [Franceschi et al., 2020]	**29.69 ± 0.32**	**0.8697 ± 0.0046**	**0.0736 ± 0.0029**
SLAMP-Baseline	29.20 ± 0.28	0.8633 ± 0.0048	0.0951 ± 0.0036
SLAMP	<u>29.39 ± 0.30</u>	<u>0.8646 ± 0.0050</u>	<u>0.0795 ± 0.0034</u>

On KTH, the subject might appear after the conditioning frames. These challenging cases can be problematic for some previous work as shown in SRVP [Franceschi et al., 2020]. Our model can generate samples close to the ground truth despite very little information on the conditioning frames as shown in Fig. 4.8. The figure shows the best sample in terms of LPIPS, please see Appendix for a diverse set of samples with subjects of various poses appearing at different time steps.

BAIR Robot Hand: This dataset contains videos of a robot hand moving and pushing objects on a table [Ebert et al., 2017a]. Due to uncertainty in the movements of the robot arm, BAIR is a standard dataset for evaluating stochastic video prediction models. Following the training and evaluation settings used in the previous work, we condition on the first 2 frames and learn to predict the next 10 frames. During testing, we again condition on the first 2 frames but predict the next 28 frames.

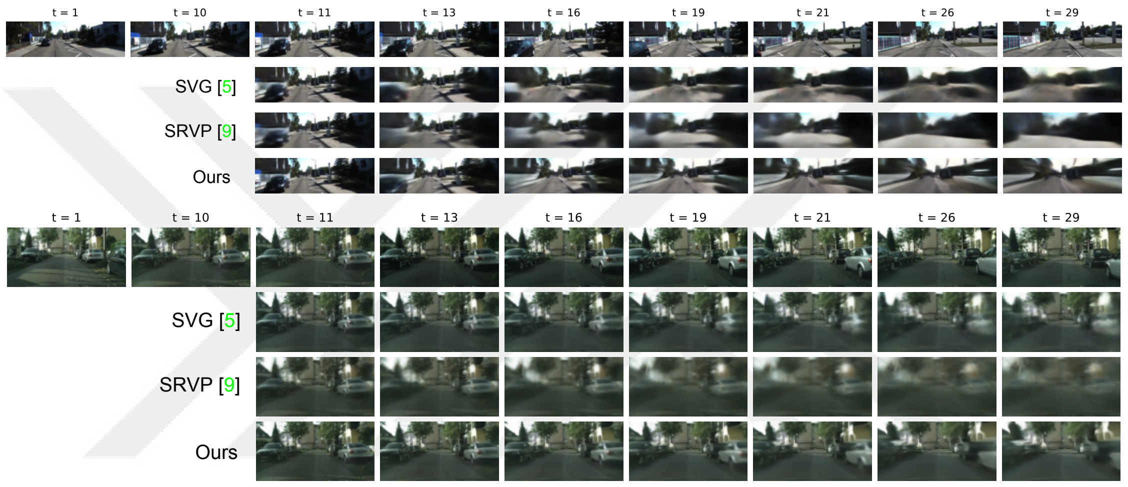

Figure 4.9: **Qualitative Comparison on KITTI and Cityscapes.** We compare SLAMP to SVG [Denton and Fergus, 2018] and SRVP [Franceschi et al., 2020] on KITTI (**top**) and Cityscapes (**bottom**). Our model can better capture the changes due to ego-motion thanks to explicit modeling of motion history.

We show quantitative results on BAIR in Fig. 4.6 and Table 5.4. Our baseline model achieves comparable results to SRVP, outperforming other methods in all metrics except SV2P [Babaeizadeh et al., 2018] in PSNR and SAVP [Lee et al., 2018] in FVD. With 2 conditioning frames only, SLAMP cannot utilize the motion history and performs similarly to the baseline model on BAIR (see Appendix). This is simply due to the fact that there is only one flow field to condition on, in other words, no motion history. Therefore, we only show the results of the baseline model on this dataset.

Table 4.4: **Results on BAIR.** This table compares the results of SLAMP and SLAMP-Baseline to the previous work on BAIR dataset. Following the previous work, we report the results as the mean and the 95%-confidence interval in terms of PSNR, SSIM, and LPIPS. Bold and underlined scores indicate the best and the second best performing method, respectively.

Models	PSNR	SSIM	LPIPS
SV2P [Babaeizadeh et al., 2018]	**20.39 ± 0.27**	0.8169 ± 0.0086	0.0912 ± 0.0053
SAVP [Lee et al., 2018]	18.44 ± 0.25	0.7887 ± 0.0092	0.0634 ± 0.0026
SVG [Denton and Fergus, 2018]	18.95 ± 0.26	0.8058 ± 0.0088	0.0609 ± 0.0034
SRVP [Franceschi et al., 2020]	19.59 ± 0.27	**0.8196 ± 0.0084**	**0.0574 ± 0.0032**
SLAMP-Baseline	19.60 ± 0.26	$\underline{0.8175 \pm 0.0084}$	$\underline{0.0596 \pm 0.0032}$
SLAMP	$\underline{19.67 \pm 0.26}$	0.8161 ± 0.0086	0.0639 ± 0.0037

Real-World Driving Datasets: We perform experiments on two challenging autonomous driving datasets: KITTI [Geiger et al., 2012, Geiger et al., 2013] and Cityscapes [Cordts et al., 2016] with various challenges. Both datasets contain everyday real-world scenes with complex dynamics due to both background and foreground motion. KITTI is recorded in one town in Germany, while Cityscapes is recorded in 50 European cities, leading to higher diversity.

Cityscapes primarily focuses on semantic understanding of urban street scenes, therefore, contains a larger number of dynamic foreground objects compared to KITTI. However, motion lengths are larger on KITTI due to lower frame rate. On both datasets, we condition on 10 frames and predict 10 frames into the future to train our models. Then at test time, we predict 20 frames conditioned on 10 frames.

As shown in Table 4.5, SLAMP outperforms both methods on all of the metrics on both datasets, which shows its ability to generalize to the sequences with moving background. Even SVG [Denton and Fergus, 2018] performs better than the state of the art SRVP [Franceschi et al., 2020] in the LPIPS metric for KITTI and on

Table 4.5: **Results with a Moving Background.** We evaluate our model SLAMP in comparison to SVG and SRVP on KITTI [Geiger et al., 2013] and Cityscapes [Cordts et al., 2016] datasets by conditioning on 10 frames and predicting 20 frames into the future.

Models	PSNR (↑)	SSIM (↑)	LPIPS (↓)
SVG [Denton and Fergus, 2018]	12.70 ± 0.70	0.329 ± 0.030	<u>0.594</u> ± 0.034
SRVP [Franceschi et al., 2020]	<u>13.41</u> ± 0.42	<u>0.336</u> ± 0.034	0.635 ± 0.021
SLAMP	**13.46 ± 0.74**	**0.337 ± 0.034**	**0.537 ± 0.042**

KITTI [Geiger et al., 2012, Geiger et al., 2013]

Models	PSNR (↑)	SSIM (↑)	LPIPS (↓)
SVG [Denton and Fergus, 2018]	20.42 ± 0.63	<u>0.606</u> ± 0.023	<u>0.340</u> ± 0.022
SRVP [Franceschi et al., 2020]	<u>20.97</u> ± 0.43	0.603 ± 0.016	0.447 ± 0.014
SLAMP	**21.73 ± 0.76**	**0.649 ± 0.025**	**0.2941 ± 0.022**

Cityscapes [Cordts et al., 2016]

both SSIM and LPIPS for Cityscapes, which shows the limitations of SRVP on scenes with dynamic backgrounds. We also perform a qualitative comparison to these methods in Fig. 4.9. SLAMP can better preserve the scene structure thanks to explicit modeling of ego-motion history in the background.

Visualization of Latent Space: We visualize stochastic latent variables of the dynamic component on KTH compared to the static latent variables. We provide both the static and the dynamic components for comparison. As can be seen from Fig. 4.10, static variables on the right are more scattered and do not form clusters according to semantic classes as in the dynamic variables on the left. This shows that our model can learn video dynamics according to semantic classes with separate modeling of the dynamic component.

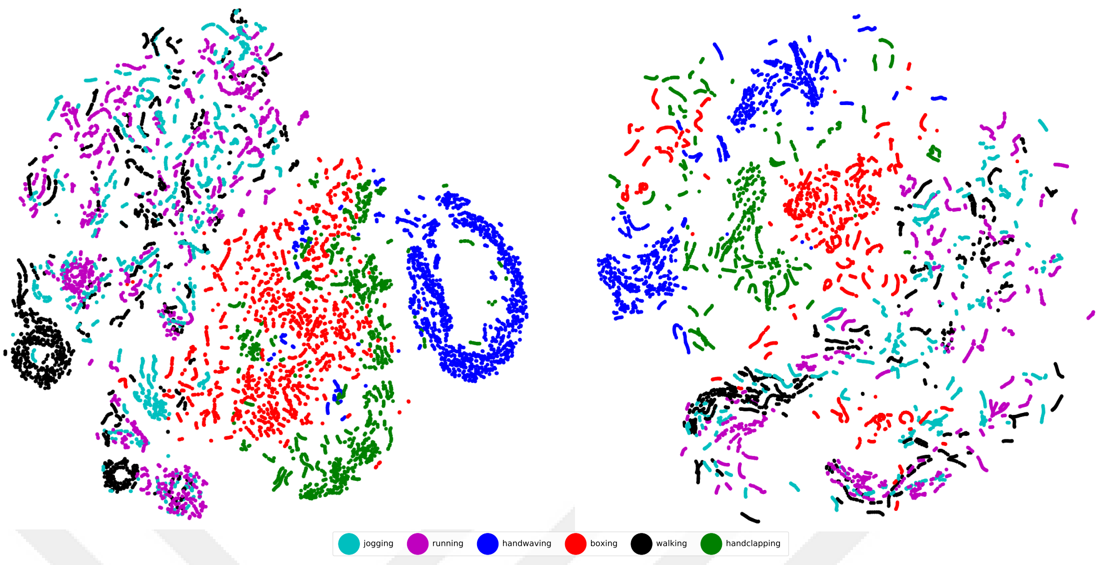

Figure 4.10: **Dynamic (left) vs. Static (right) Latent Variables**. This figure shows the T-SNE visualization of dynamic and static latent variables on 300 test videos from the KTH dataset. In dynamic latent variables, different classes with similar repetitive movements such as walking, running, and jogging are clustered together. However, in static latent variables, points are more scattered and do not form clusters according to semantic actions.

4.4 Using a different architecture

In this section, we show the superiority of SLAMP's decoding scheme by using a different method, namely SRVP [Franceschi et al., 2020]. We integrate SLAMP's extra decoder into SRVP's state-space model in order to find out whether SLAMP's decoding can be extended into other architecture or not.

SRVP Architecture: Franceschi et al. [Franceschi et al., 2020] introduced SRVP, a state-space model for stochastic video prediction model. SRVP encodes the scene into a stochastic latent variable and uses time-independent residual updates to create a different state representation for each time step. The SRVP separates the content and the dynamic by using a separate deterministic content network to encode the scene details and using it for the decoding of future frames. In this way, the latent state will learn the dynamics, e.g. motion in the scene, and the content part will learn the details of the scene.

In each time step, SRVP [Franceschi et al., 2020] creates a state variable, by_t, by using a stochastic residual network to update previous states into future ones.

$$\mathbf{y}_{t+\Delta t} = \mathbf{y}_t + \Delta t \cdot f_\theta\big(\mathbf{y}_t, \mathbf{z}_{\lfloor t \rfloor + 1}\big). \tag{4.7}$$

where by_t is the state representation at time t, Δt is a hyperparameter that defines the size of the updates, $\mathbf{z}$ is the latent variable sampled from prior or posterior distribution according to train or inference time. After creating state representation for each time step, SRVP decodes all of them into video frames, e.g. images, which can be seen in (4.8).

$$\mathbf{x}_t = g_\theta\big(\mathbf{y}_t, \mathbf{w}\big). \tag{4.8}$$

where bx_t is the image frame at time t, g_θ is the decoder parameterized by θ, $\mathbf{y}_t$ is the state representation at time t, $\mathbf{w}$ is the content feature extracted by deterministic content network.

Addition of SLAMP's decoder:

By following SLAMP, we add another decoder, *motion decoder* g_θ^f, into SRVP to decode the motion from the previous frame to the next frame. After the motion, e.g. optical flow is predicted, we use it to warp the previous image into the next ones. In motion decoder, we use two consecutive state representations, e.g. to decode the motion from $t-1$ to t, we use $\mathbf{y}_{t-1}$ and $\mathbf{y}_t$. Moreover, we use a content vector similar to the original decoder of SRVP.

So, the motion decoder of SRVP works as follows:

$$\mathbf{f}_{t-1:t} = g_\theta^f\big(\mathbf{y}_{t-1}, \mathbf{y}_t, \mathbf{w}\big). \tag{4.9}$$
$$\mathbf{x}_t = \text{Warp}(\mathbf{x}_{t-1}, \mathbf{f}_{t-1:t})$$

where $\mathbf{f}_{t-1:t}$ is the optical flow from time $t-1$ to t, g_θ^f is the motion decoder, $\mathbf{y}$'s are the representations, Warp is a function for differentiable warping [Jaderberg et al., 2015].

The goal is to learn the motion dynamics of the scene by explicitly decoding them. In the end, *new* SRVP architecture will have two predictions coming from

Table 4.6: **SRVP++ Comparisons on generic video prediction datasets** This table shows the quantitative results comparing the SRVP++ with SVG [Denton and Fergus, 2018], the original SRVP [Franceschi et al., 2020] and SLAMP. **Direct** refers to direct masking strategy whereas **Mask** refer to binary mask strategy. Following the previous work, we report the results as the mean and the 95%-confidence interval in terms of PSNR, SSIM, and LPIPS on all the datasets except LPIPS on MNIST.

	Models	PSNR (↑)	SSIM (↑)	LPIPS (↓)
MNIST	SVG	14.50 ± 0.04	0.7090 ± 0.0015	—
	SRVP	16.93 ± 0.07	**0.7799 ± 0.0020**	**0.1129 ± 0.0010**
	SLAMP	18.07 ± 0.08	0.7736 ± 0.0019	0.1282 ± 0.0006
	SRVP++ - Direct	**18.14 ± 0.08**	0.7634 ± 0.0019	0.1323 ± 0.0012
	SRVP++ - Mask	16.57 ± 0.05	0.7367 ± 0.0018	0.1597 ± 0.0011
KTH	SVG	28.06 ± 0.29	0.8438 ± 0.0054	0.0923 ± 0.0038
	SRVP	**29.69 ± 0.32**	**0.8697 ± 0.0046**	**0.0736 ± 0.0029**
	SLAMP	29.39 ± 0.30	0.8646 ± 0.0050	0.0795 ± 0.0034
	SRVP++ - Direct	28.04 ± 0.26	0.8541 ± 0.0053	0.0948 ± 0.0034
	SRVP++ - Mask	28.76 ± 0.30	0.8612 ± 0.0053	0.0836 ± 0.0034
BAIR	SVG	18.95 ± 0.26	0.8058 ± 0.0088	0.0609 ± 0.0034
	SRVP	19.59 ± 0.27	0.8196 ± 0.0084	**0.0574 ± 0.0032**
	SLAMP	19.67 ± 0.26	0.8161 ± 0.0086	0.0639 ± 0.0037
	SRVP++ - Direct	19.74 ± 0.26	**0.8206 ± 0.0084**	0.0598 ± 0.0033
	SRVP++ - Mask	19.01 ± 0.26	0.8089 ± 0.0087	0.0663 ± 0.0037

two different branches. We employ the same strategy as SLAMP's for the combination, see (4.1). Moreover, we propose a different combination strategy. Instead of predicting a binary mask to combine predictions of each branch, the mask decoder will directly output the final image. The intuition is to let the network decide the pixels instead of explicitly selecting a pixel. We ablate this combination strategy in our experiments. See the appendix for the overall architectural figure of SRVP++.

Experiments: In order to analyze the proposed decoding scheme, we compare the new SRVP, which we name SRVP++, with SVG, SRVP, and SLAMP.

We first show results on generic video prediction datasets, namely MNIST, KTH [Schüldt et al., 2004a] and BAIR [Ebert et al., 2017b]. According to the results on Table 4.6, SRVP++ decoding strategy does not improve the results. We suspect that this is due to the static background of the generic video prediction datasets. SRVP's assumption of static background holds for these datasets. However, it fails for the real-world driving datasets with moving background, as it is explained before in Section 4.3.

We further evaluate the models on challenging real-world datasets, KITTI [Geiger et al., 2012, Geiger et al., 2013] and Cityscapes [Cordts et al., 2016] following SLAMP. Since the static background assumption of SRVP fails on the real-world driving datasets with moving background, SRVP++ performs better than SRVP. The additional motion decoder can learn the motion in the scene and break the assumption of SRVP. By warping the previous frames, SRVP++ can generate higher-quality images than SRVP. The results in Table 4.7 show that SLAMP's decoding scheme can be applied to different methods and improve their performance on challenging real-world datasets.

4.5 Conclusion

We presented a stochastic video prediction framework to decompose video content into appearance and dynamic components. Our baseline model with deterministic motion and mask decoders outperforms SVG, which is a special case of our baseline model. Our model with motion history, SLAMP, further improves the results and reaches the performance of the state-of-the-art method SRVP on the previously used datasets. Moreover, it outperforms both SVG and SRVP on two real-world autonomous driving datasets with dynamic background and complex motion. We show that motion history enriches the model's capacity to predict the future, leading

Table 4.7: **SRVP++ Comparisons on real-world driving datasets** This table shows the quantitative results comparing the SRVP++ with SVG [Denton and Fergus, 2018], the original SRVP [Franceschi et al., 2020] and SLAMP on KITTI [Geiger et al., 2012, Geiger et al., 2013] and Cityscapes [Cordts et al., 2016]. **Direct** refers to direct masking strategy whereas **Mask** refer to binary mask strategy. Following the previous work, we report the results as the mean and the 95%-confidence interval in terms of PSNR, SSIM, and LPIPS on all the datasets.

	Models	PSNR (↑)	SSIM (↑)	LPIPS (↓)
KITTI	SVG	12.70 ± 0.70	0.329 ± 0.030	0.594 ± 0.034
	SRVP	13.41 ± 0.42	0.336 ± 0.034	0.635 ± 0.021
	SLAMP	13.46 ± 0.74	<u>0.337 ± 0.034</u>	0.537 ± 0.042
	SRVP++ - Direct	**14.09 ± 0.49**	**0.345 ± 0.023**	<u>0.505 ± 0.020</u>
	SRVP++ - Mask	<u>14.01 ± 0.50</u>	0.331 ± 0.024	**0.455 ± 0.020**
Cityscapes	SVG	20.42 ± 0.63	<u>0.606</u> ± 0.023	0.340 ± 0.022
	SRVP	<u>20.97</u> ± 0.43	0.603 ± 0.016	0.447 ± 0.014
	SLAMP	**21.73 ± 0.76**	**0.649 ± 0.025**	**0.2941 ± 0.022**
	SRVP++ - Direct	20.11 ± 0.42	0.596 ± 0.017	0.331 ± 0.011
	SRVP++ - Mask	20.35 ± 0.39	0.587 ± 0.015	<u>0.314 ± 0.008</u>

to better predictions in challenging cases.

Moreover, we show that the decoding scheme of SLAMP can be used in other video prediction methods, e.g. SRVP [Franceschi et al., 2020], and improve the performance on real-world datasets with a moving background.

Our model with motion history cannot realize its full potential in standard settings of stochastic video prediction datasets. A fair comparison is not possible on BAIR due to the small number of conditioning frames. BAIR holds a great promise with changing background but infrequent, small changes are not reflected in current evaluation metrics.

Chapter 5

STOCHASTIC VIDEO PREDICTION WITH STRUCTURE AND MOTION

5.1 Introduction

The world observed from a moving vehicle can be decomposed into a static part that moves only according to the vehicle's motion, or the ego-motion, and a dynamic part that contains independently moving objects. With this separation, the decomposition of the static and the dynamic parts is inevitable. The static part of the video does not have any motion; however, they move in the video according to the motion of the vehicle. This part can easily be explained by the domain knowledge following the previous work on structure and motion [Zhou et al., 2017, Godard et al., 2019]. The static part is modeled by both the 3D structure and the ego-motion. On the other hand, the motion of the objects that move independently cannot be explained by ego-motion. There should be a separate mechanism to model the remaining motion in the scene after applying the ego-motion. The most suitable option is to use optical flow on top of the ego-motion for modeling the independently moving objects.

The key factor in this design is that the optical flow should correspond to residual or remaining motion in the scene after applying the ego-motion. To do this, the dynamic part, which corresponds to the independently moving objects, should be conditioned on the static part. In this way, the dynamic part will be aware of the static part, e.g. ego-motion, and can be modeled as a residual motion in the scene. Following previous work on video decomposition [Skafte and Hauberg, 2019], which conditions the body pose on the shape, we condition the motion of the dynamic part on the static part to achieve a disentanglement between the two types of motion in driving.

In this chapter, we propose SLAMP-3D. SLAMP-3D decomposes the driving video into static and dynamic parts. The static part is handled by the 3D structure and the ego-motion. On top of this, the dynamic part is covered with the residual flow using the conditioning of the dynamic on top of the static part. With this separation, the performance of future prediction in real-world scenes with moving background and independently moving foreground objects on two real-world driving datasets, KITTI [Geiger et al., 2012, Geiger et al., 2013] and Cityscapes [Cordts et al., 2016]. Furthermore, conditioning the object motion on the ego-motion improves the results, especially for foreground objects in dynamic scenes of Cityscapes.

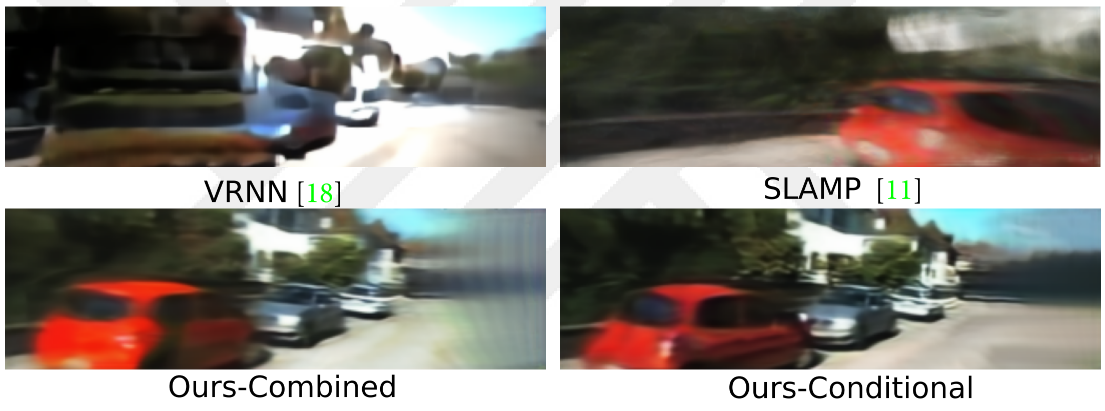

Figure 5.1: **Future Prediction while Turning.** We compare the future prediction results of previous methods (**top**) to ours (**bottom**) while the vehicle makes a right turn. While previous methods fail due to a less frequent scenario in the dataset, our method can generate a better future prediction due to explicit modeling of the structure and ego-motion. This figure shows a single frame after the conditioning frames, please see Fig. 5.8 for the whole sequence.

With this conditioned decomposition, SLAMP-3D can predict the future frames even in the most challenging cases, such as turning. In Fig. 5.1, SLAMP-3D can preserve the structure of the red car easily while the other methods fail to do so.

The overall architecture of SLAMP-3D can be seen in the Fig. 5.2.

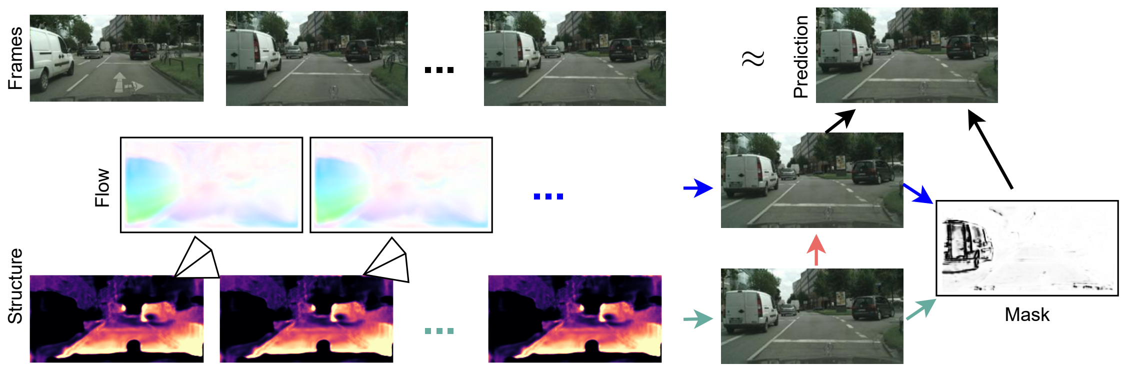

Figure 5.2: **Model Overview.** For predicting future frames in a video (**top**), we predict depth on each frame and camera pose, and optical flow between consecutive frames. We model the static part of the scene with depth and pose (**bottom**). Conditioned on the static part, we model the remaining motion due to moving foreground objects with the flow in the dynamic parts of the scene (**middle**). We obtain the final prediction (**top-right**) by combining the static and the dynamic predictions based on the predicted mask (**right**, dynamic in black). The moving car is missing in depth because it is predicted by the dynamic. The arrow from the static to the final prediction is omitted for clarity.

5.2 Methodology

We investigate the effects of decomposing the scene into static and dynamic parts for stochastic video prediction, inspired by and built on top of SLAMP. We assume that the background or the static part of the scene, moves only according to the motion of the ego-vehicle and can be explained by the ego-motion and the scene structure. We first predict future depth and ego-motion conditioned on a few given frames. This is different than SLAMP which predicts the static part in the pixel-space. By using the predictions of the structure and the ego-motion, we synthesize the static part of the scene. We model the remaining motion due to independently moving objects, the dynamic part of the scene, as the residual flow on top of the ego-motion. At the end, we combine the static and the dynamic predictions with a learned mask to generate the final predictions.

5.2.1 Stochastic Video Prediction (SVP)

The same notation given in Section 4.2.1 is preserved throughout this chapter.

5.2.2 From SLAMP to SLAMP-3D

In this chapter, we propose to extend SLAMP, explained in Chapter 4, to SLAMP-3D by incorporating scene structure into the estimation. Similar to SLAMP, we decompose the scene into static and dynamic components where the static component focus on the changes in the background due to camera motion and the dynamic component on the remaining motion in the scene due to independently moving objects. There are two major differences with respect to SLAMP. First, in this work, we model the static parts of the scene in the motion space as well to better represent the changes in the background due to the ego-motion of the vehicle. Second, we condition the prediction of dynamic component on the static to predict the residual motion in the scene, i.e. the remaining motion after the scene moves according to ego-motion due to the independently moving objects.

Similar to SLAMP, we compute two separate distributions, $q_{\phi_s}\left(\mathbf{z}_t^s \mid \mathbf{x}_{1:t}\right)$ and $q_{\phi_d}\left(\mathbf{z}_t^d \mid \mathbf{x}_{1:t}\right)$, for static and dynamic components, respectively. This way, both distributions focus on different parts of the scene which decomposes scene into static and dynamic parts. Since the camera motion applies both to the background and to the objects, dynamic component only needs to learn the residual motion. We achieve this by introducing an explicit conditioning of the dynamic latent variable $\mathbf{z}_t^d$ on the static latent variable $\mathbf{z}_t^s$:

$$\begin{aligned} p\left(\mathbf{z}_t^s \mid \mathbf{x}_{1:t}\right) &\approx q_{\phi_s}\left(\mathbf{z}_t^s \mid \mathbf{x}_{1:t}\right) \quad \text{and} \\ p\left(\mathbf{z}_t^d \mid \mathbf{x}_{1:t}\right) &\approx q_{\phi_d}\left(\mathbf{z}_t^d \mid \mathbf{x}_{1:t}, \mathbf{z}_t^s\right) \end{aligned} \tag{5.1}$$

With two latent variables, we extend the stochasticity to the motion space, separately for the ego-motion and the object motion. Furthermore, we condition the dynamic component on the static to define the object motion as residual motion that remains after explaining the scene according to camera motion, similar to the disentanglement of the body pose and the shape in [Skafte and Hauberg, 2019].

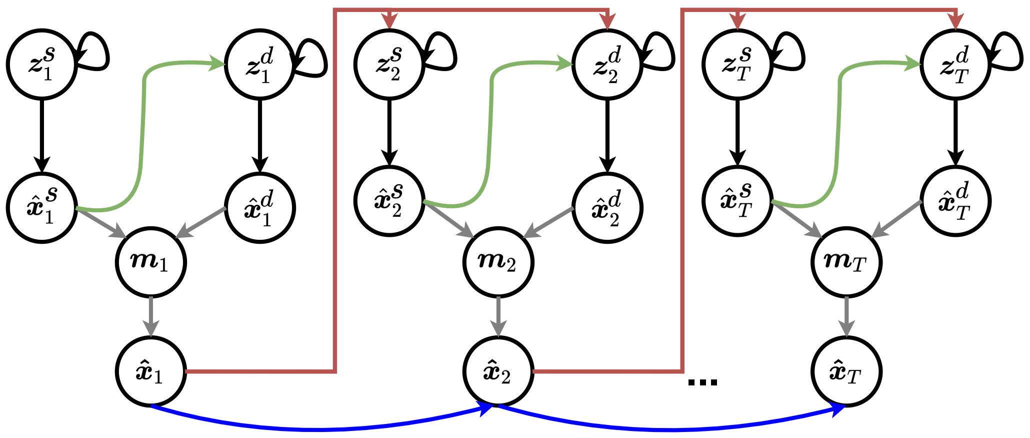

Figure 5.3: **Graphical Model of SLAMP-3D.** The graphical model shows the generation process of our model with static and dynamic latent variables $\mathbf{z}_t^s$ and $\mathbf{z}_t^d$ generating frames $\hat{\mathbf{x}}_t^s$ and $\hat{\mathbf{x}}_t^d$. Information is propagated between time-steps through the recurrence between frame predictions (blue), corresponding latent variables, and from frame predictions to latent variables (red). The dependency of dynamic latent variables on static is shown in green.

Two Types of Motion History: Similar to SLAMP, we model the motion history in SLAMP-3D but by disentangling it to ego-motion and residual motion. Essentially, we learn two separate motion histories for the static and the dynamic components. The latent variables $\mathbf{z}_t^s$ and $\mathbf{z}_t^d$ contain motion information accumulated over the previous frames rather than local temporal changes between the last frame and the target frame. This is achieved by encouraging each posterior, $q_{\phi_s}\left(\mathbf{z}_t^s \mid \mathbf{x}_{1:t}\right)$ and $q_{\phi_d}\left(\mathbf{z}_t^d \mid \mathbf{x}_{1:t}, \mathbf{z}_t^s\right)$, to be close to a prior distribution in terms of KL-divergence which can be seen in (5.1). Similar to SVG [Denton and Fergus, 2018] and SLAMP, we learn each prior distribution from the previous frames up to the target frame, $p_{\psi_s}\left(\mathbf{z}_t^s \mid \mathbf{x}_{1:t-1}\right)$ and $p_{\psi_d}\left(\mathbf{z}_t^d \mid \mathbf{x}_{1:t-1}\right)$. Note that we learn separate posterior and prior distributions for the static and the dynamic components. The static contains depth and pose encoding, while the dynamic contains the flow encoding conditioned on the static as explained next.

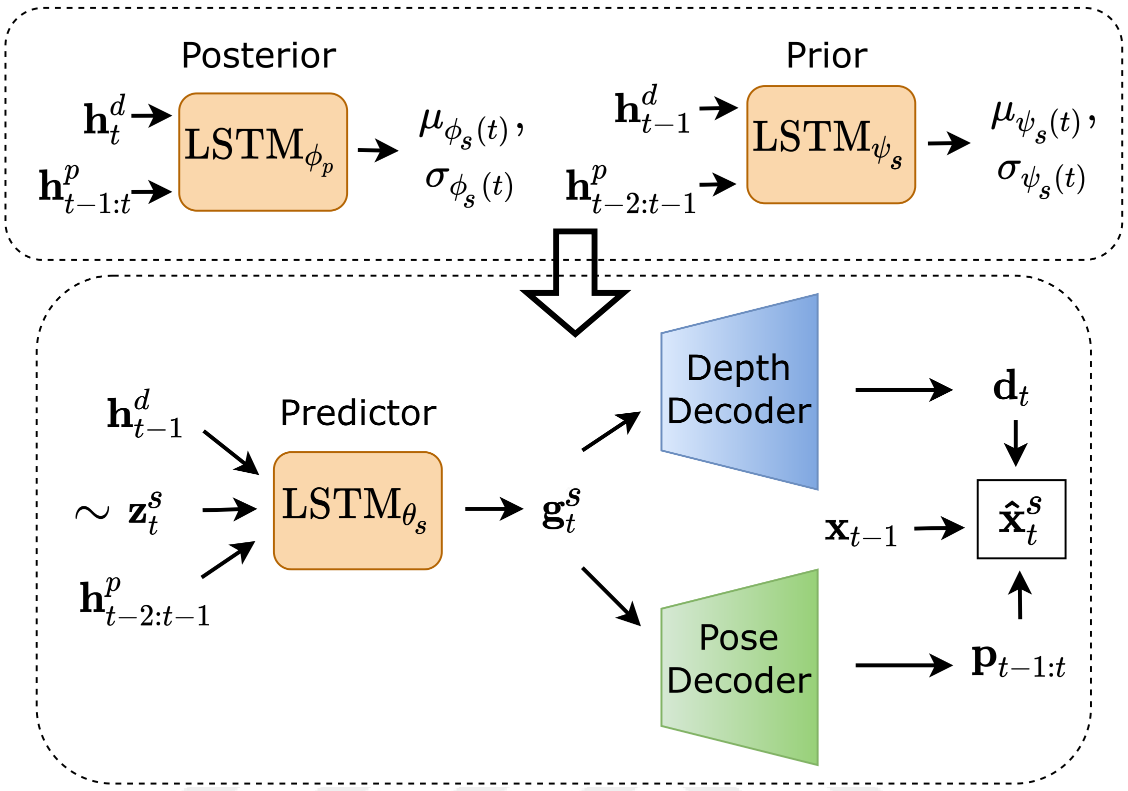

Figure 5.4: **Static Architecture.** The architecture shows the sampling of the static latent variable $\mathbf{z}^s_t$ from the posterior $\boldsymbol{\phi}_s$ or the prior $\boldsymbol{\psi}_s$, which is then used to predict $\mathbf{g}^s_t$, the future representation of the static part. The future depth $\mathbf{d}_t$ and pose $\mathbf{p}_{t-1:t}$ are decoded and used to generate the static prediction $\hat{\mathbf{x}}^s_t$. The architecture shows the similar procedure for the dynamic part using optical flow with the flow decoder instead of depth and pose.

5.2.3 SLAMP-3D

Inspired by the previous work on unsupervised learning of depth and ego-motion [Zhou et al., 2017, Godard et al., 2019, Casser et al., 2019, Guizilini et al., 2020, Safadoust and Gney, 2021], flow [Jason et al., 2016, Ren et al., 2017, Meister et al., 2018], as well as the relating of the two [Zou et al., 2018, Yin and Shi, 2018, Ranjan et al., 2019, Chen et al., 2019], we reconstruct the target frame from the static and the dynamic components at each time-step.

Depth and Pose: For the static component which moves only according to the

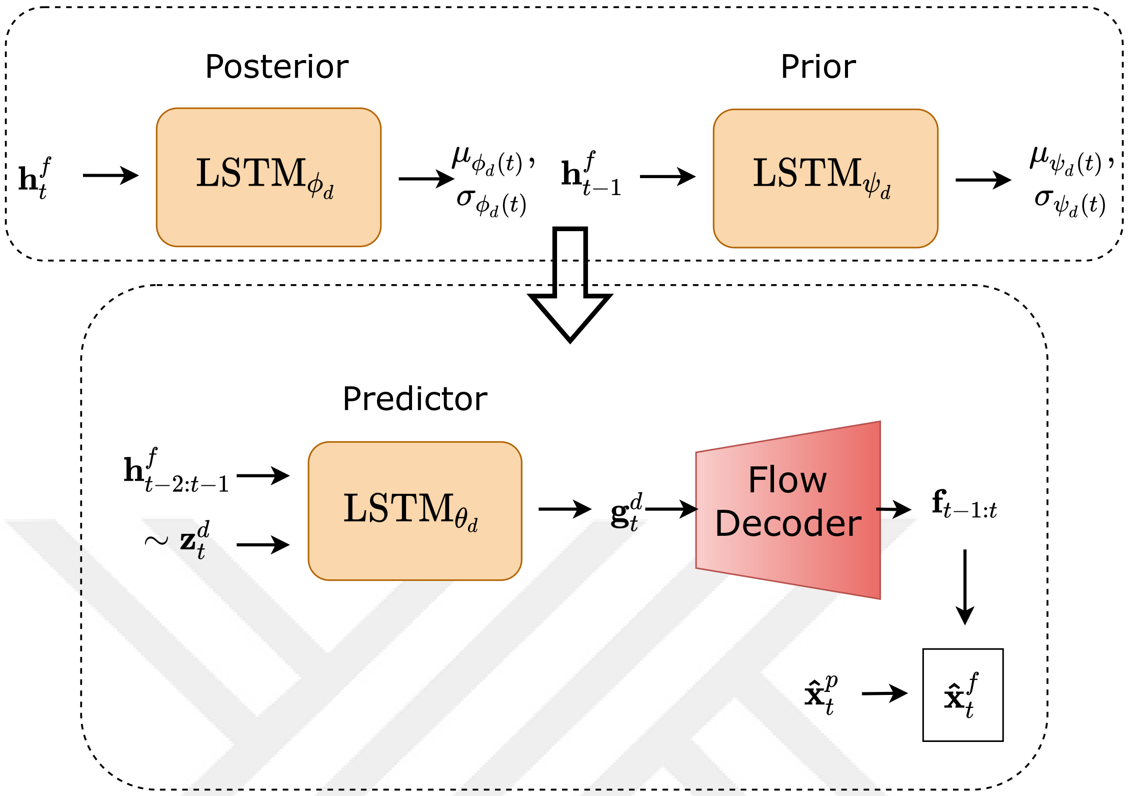

Figure 5.5: **Dynamic Architecture.** The architecture in shows the sampling of the static latent variable $\mathbf{z}_t^d$ from the posterior $\boldsymbol{\phi}_d$ or the prior $\boldsymbol{\psi}_d$, which is then used to predict $\mathbf{g}_t^d$, the future representation of the dynamic part. The future optical flow $\mathbf{f}_{t-1:t}$ is decoded and used to generate the dynamic prediction $\hat{\mathbf{x}}_t^d$.

camera motion, we estimate the relative camera motion or the camera pose $\mathbf{p}_{t-1:t}$ from the previous frame $t-1$ to the target frame t. The pose $\mathbf{p}_{t-1:t}$ represents the 6-degrees of freedom rigid motion of the camera from the previous frame to the target frame. We also estimate the depth of the target frame, $\mathbf{d}_t$, and reconstruct the target frame as $\hat{\mathbf{x}}_t^s$ from the previous frame $\mathbf{x}_{t-1}$ using the estimated pose $\mathbf{p}_{t-1:t}$ and depth $\mathbf{d}_t$ via differentiable warping [Jaderberg et al., 2015]. Note that we predict the depth and pose a priori, without actually seeing the target frame in contrast to the previous unsupervised monocular depth approaches that use the target frame to predict depth and pose [Zhou et al., 2017, Godard et al., 2019, Casser et al., 2019, Guizilini et al., 2020].

Residual Flow: We estimate optical flow $\mathbf{f}_{t-1:t}$ as the remaining motion from the reconstruction of the target frame $\hat{\mathbf{x}}_t^s$ to the target frame $\mathbf{x}_t$. The flow $\mathbf{f}_{t-1:t}$ represents the motion of the pixels belonging to objects that move independently from the previous frame to the target frame. We reconstruct the target frame as $\hat{\mathbf{x}}_t^d$ from the static prediction, $\hat{\mathbf{x}}_t^s$, by using the estimated optical flow $\mathbf{f}_{t-1:t}$. Explicit conditioning of the optical flow on the static prediction allows the network to learn the remaining motion in the scene from the source frame to the target frame which corresponds to the motion that cannot be explained by the camera pose. As in the case of depth and pose prediction, we predict the flow from the reference frame to the target frame without actually seeing the target frame. In other words, we predict future motion based on the motion history, which is different than regular optical flow approaches that use the target frame for prediction [Jason et al., 2016, Ren et al., 2017, Meister et al., 2018].

Note that the role of optical flow in SLAMP-3D is different than SLAMP. In SLAMP-3D, the optical flow is predicted conditioned on the static component. This way, the optical flow in SLAMP-3D corresponds to the residual flow as opposed to full flow as in the case of SLAMP. Therefore, we warp the *static component's prediction* instead of the previous frame as in the case of SLAMP.

Combining Static and Dynamic Predictions: Similar to SLAMP, we use the Hadamard product to combine the static prediction $\hat{\mathbf{x}}_t^s$ and the dynamic prediction $\hat{\mathbf{x}}_t^d$ into the final prediction $\hat{\mathbf{x}}_t$, showed in (5.2). The static parts in the background can be reconstructed accurately using the depth and camera pose estimation. For the dynamic parts with moving objects, the target frame can be reconstructed accurately using the residual motion prediction. The mask prediction learns a weighting between the static and the dynamic predictions for combining them into the final prediction.

$$\hat{\mathbf{x}}_t = \mathbf{m}(\mathbf{x}_t^s, \mathbf{x}_t^d) \odot \mathbf{x}_t^s + (\mathbf{1} - \mathbf{m}(\mathbf{x}_t^s, \mathbf{x}_t^d)) \odot \mathbf{x}_t^d \tag{5.2}$$

5.2.4 Variational Inference

Removing time dependence for clarity, Eq. (6.3) expresses the conditional joint probability corresponding to the graphical model shown in Fig. 5.3:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \iint p(\mathbf{x} \mid \mathbf{z}^s, \mathbf{z}^d)\; p(\mathbf{z}^s)\; p(\mathbf{z}^d)\; \mathrm{d}\mathbf{z}^s\; \mathrm{d}\mathbf{z}^d \tag{5.3}$$

The true distribution over the latent variables $\mathbf{z}_t^s$ and $\mathbf{z}_t^d$ is intractable. We assume that the dynamic component $\mathbf{z}_t^d$ depends on the static component $\mathbf{z}_t^s$ and train time-dependent inference networks $q_{\boldsymbol{\phi}_s}\left(\mathbf{z}_t^s \mid \mathbf{x}_{1:T}\right)$ and $q_{\boldsymbol{\phi}_d}\left(\mathbf{z}_t^d \mid \mathbf{x}_{1:T}, \mathbf{z}_t^s\right)$ to approximate the true distribution with conditional Gaussian distributions. In order to optimize the likelihood of $p\left(\mathbf{x}_{1:T}\right)$, we need to infer latent variables $\mathbf{z}_t^s$ and $\mathbf{z}_t^d$, which correspond to uncertainty in static and dynamic parts of future frames, respectively. We use a variational inference model to infer the latent variables.

At each time-step, $\mathbf{z}_t^d$ depends on $\mathbf{z}_t^s$ but each is independent across time. Therefore, we can decompose Kullback-Leibler terms into individual time steps. We train the model by optimizing the variational lower bound as shown in (6.6) where each $\mathbf{z}_{1:t}^{\cdot}$ is sampled from the respective posterior distribution $q_{\boldsymbol{\phi}_{\cdot}}$ (see Appendix for the derivation).

$$\begin{aligned} \log p_{\boldsymbol{\theta}}(\mathbf{x}) \geq & \mathbb{E}_{q_s(\mathbf{z}^s \mid \mathbf{x})}\left[\mathbb{E}_{q_d(\mathbf{z}^d \mid \mathbf{z}^s, \mathbf{x})}\left[\log p(\mathbf{x} \mid \mathbf{z}^s, \mathbf{z}^d)\right]\right] \\ & - \mathbb{E}_{q_s(\mathbf{z}^s \mid \mathbf{x})}\left[D_{KL}(q_d(\mathbf{z}^d \mid \mathbf{z}^s, \mathbf{x}) \mid\mid p(\mathbf{z}^d))\right] - D_{KL}(q_s(\mathbf{z}^s \mid \mathbf{x}) \mid\mid p(\mathbf{z}^s)) \end{aligned} \tag{5.4}$$

The likelihood $p_{\boldsymbol{\theta}}$ can be interpreted as minimizing the difference between the actual frame $\mathbf{x}_t$ and the prediction $\hat{\mathbf{x}}_t$ as defined in Eq. (4.1) by changing p and f to s and d. We apply the reconstruction loss to the predictions of static and dynamic components as well. The posterior terms for uncertainty are estimated as an expectation over $q_{\boldsymbol{\phi}_s}\left(\mathbf{z}_t^s \mid \mathbf{x}_{1:t}\right)$ and $q_{\boldsymbol{\phi}_d}\left(\mathbf{z}_t^d \mid \mathbf{x}_{1:t}, \mathbf{z}_t^s\right)$. For the second term, we use sampling to approximate the expectation as proposed in [Skafte and Hauberg, 2019]. As in [Denton and Fergus, 2018], we also learn the prior distributions from the previous frames up to the target frame as $p_{\boldsymbol{\psi}_s}\left(\mathbf{z}_t^s \mid \mathbf{x}_{1:t-1}\right)$ and $p_{\boldsymbol{\psi}_d}\left(\mathbf{z}_t^d \mid \mathbf{x}_{1:t-1}\right)$. We train the model using the re-parameterization trick [Kingma and Welling, 2014] and choose the posteriors to be factorized Gaussian so that all the KL divergences can be computed analytically. We apply the optimal variance estimate as proposed in σ-VAE [Rybkin et al., 2021] to learn an optimal value for the hyper-parameter corresponding to the weight of the KL term.

5.2.5 Architecture

We encode the frames with a feed-forward convolutional architecture to obtain frame-wise features at each time-step. Our goal is to reduce the spatial resolution of the frames in order to reduce the complexity of the model. On top of this shared representation, we have a depth head and a pose head to learn an encoding of depth at each frame, $\mathbf{h}_t^d$, and an encoding of pose for each consecutive frame pair, $\mathbf{h}_{t-1:t}^p$. As shown in Fig. 5.4, we train convolutional LSTMs to infer the static distribution at each time-step from the encoding of depth and pose:

$$\begin{aligned}
&\mathbf{e}_{t-1},\ \mathbf{e}_t = \mathrm{ImgEnc}\left(\mathbf{x}_{t-1}\right),\ \mathrm{ImgEnc}\left(\mathbf{x}_t\right) \\
&\mathbf{h}_t^d,\ \mathbf{h}_{t-1:t}^p = \mathrm{DepthEnc}\left(\mathbf{e}_t\right),\quad \mathrm{PoseEnc}\left(\mathbf{e}_{t-1},\mathbf{e}_t\right) \\
&\mathbf{z}_t^s \sim \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\phi}_s(t)},\boldsymbol{\sigma}_{\boldsymbol{\phi}_s(t)}\right) \quad \text{where} \\
&\boldsymbol{\mu}_{\boldsymbol{\phi}_s(t)},\ \boldsymbol{\sigma}_{\boldsymbol{\phi}_s(t)} = \mathrm{LSTM}_{\boldsymbol{\phi}_s}\left(\mathbf{h}_t^d,\ \mathbf{h}_{t-1:t}^p\right)
\end{aligned} \tag{5.5}$$

Eq. (5.5) only shows the posterior distribution, similar steps are followed for the static prior by using the depth and pose representations from the previous time-step.

At the first time-step where there is no previous pose estimation, we assume zero-motion by estimating the pose from the previous frame to itself. We learn a static frame predictor to generate the target frame based on the encoding of depth and pose and the static latent variable $\mathbf{z}_t^s$:

$$\mathbf{g}_t^s = \mathrm{LSTM}_{\boldsymbol{\theta}_s}\left(\mathbf{h}_{t-1}^d, \mathbf{h}_{t-2:t-1}^p, \mathbf{z}_t^s\right) \tag{5.6}$$

Then, the depth and pose estimations are decoded from $\mathbf{g}_t^s$. Based on the depth and pose estimations, the static prediction $\hat{\mathbf{x}}_t^s$ is reconstructed by inverse warping the previous frame $\mathbf{x}_{t-1}$.

We learn an encoding of the remaining motion, $\mathbf{h}_t^f$, in the dynamic parts from the output of the static frame predictor $\mathbf{g}_t^s$ to the encoding of the target frame $\mathbf{x}_t$.

We train convolutional LSTMs to infer dynamic posterior and prior distributions at each time-step from the encoded residual motion. The posterior LSTM is updated based on the $\mathbf{h}_t^f$ and the latent variable $\mathbf{z}_t^d$ is sampled from the posterior:

$$\begin{aligned}
&\mathbf{h}_t^f = \text{MotionEnc}\left(\mathbf{g}_t^s, \mathbf{e}_t\right) \\
&\mathbf{z}_t^d \sim \mathcal{N}\left(\boldsymbol{\mu}_{\boldsymbol{\phi}_d(t)}, \boldsymbol{\sigma}_{\boldsymbol{\phi}_d(t)}\right) \quad \text{where} \\
&\boldsymbol{\mu}_{\boldsymbol{\phi}_d(t)},\ \boldsymbol{\sigma}_{\boldsymbol{\phi}_d(t)} = \text{LSTM}_{\boldsymbol{\phi}_d}\left(\mathbf{h}_t^f\right)
\end{aligned} \tag{5.7}$$

For the dynamic prior, we use the motion representation from the previous time step to update the prior LSTM and sample the $\mathbf{z}_t^d$ from it. Similar to the pose above, we assume zero-motion at the first time-step where there is no previous motion. The dynamic predictor LSTM is updated according to encoded features and sampled latent variables:

$$\mathbf{g}_t^d = \text{LSTM}_{\boldsymbol{\theta}_d}\left(\mathbf{h}_{t-1}^f, \mathbf{z}_t^d\right) \tag{5.8}$$

During training, the latent variables are sampled from the posterior distribution. At test time, they are sampled from the posterior for the conditioning frames and from the prior for the following frames.

After we generate target time's features for static and dynamic components, $\mathbf{g}_t^s$ and $\mathbf{g}_t^d$, respectively, we decode them into depth and pose for the static component, optical flow for the dynamic component. We first predict the depth values of the target frame without seeing the frame and the pose from the previous frame to the target frame using the static features, $\mathbf{g}_t^s$. Then, we warp the previous frame $\mathbf{x}_{t-1}$ using the predicted depth and pose to obtain the static prediction for the target frame, $\hat{\mathbf{x}}_t^s$. For the dynamic part, we predict the residual flow from the dynamic features, $\mathbf{g}_t^d$. Then, we warp the static frame prediction $\hat{\mathbf{x}}_t^s$, using the predicted residual flow to obtain the dynamic prediction, $\hat{\mathbf{x}}_t^d$. Finally, static and dynamic predictions are combined into the final prediction, $\hat{\mathbf{x}}_t$, using the predicted mask as shown in (5.2).

5.3 Experiments

Implementation Details: We first process each image with a shared backbone [Simonyan and Zisserman, 2015, He et al., 2016] to reduce the spatial resolution. We add separate heads for extracting features for depth, pose, and flow. We learn the static distribution based on depth and pose features, and dynamic distribution based on flow features with two separate ConvLSTMs. Based on the corresponding latent variables, we learn to predict the next frame's static and dynamic representation with another pair of ConvLSTMs. We decode depth and pose from the static representation and flow from the dynamic. We obtain the static frame prediction by warping the previous frame according to depth and pose, and dynamic prediction by warping the static prediction according to flow. We train the model using the negative log likelihood loss.

Datasets:

We perform experiments on two challenging autonomous driving datasets, KITTI [Geiger et al., 2012, Geiger et al., 2013] and Cityscapes [Cordts et al., 2016]. Both datasets contain everyday real-world scenes with complex dynamics due to both background and foreground motion. We train our model on the training set of the Eigen split on KITTI [Eigen et al., 2014]. We apply the pre-processing followed by monocular depth approaches and remove the static scenes [Zhou et al., 2017]. In addition, we use the depth ground-truth on KITTI to validate our depth predictions. Cityscapes primarily focuses on semantic understanding of urban street scenes, therefore contains a larger number of dynamic foreground objects compared to KITTI. However, motion lengths are larger on KITTI due to lower frame-rate. On both datasets, we condition on 10 frames and predict 10 frames into the future to train our models. Then, at test time, we predict 20 frames conditioned on 10 frames.

Evaluation Metrics: We compare the video prediction performance using four different metrics: Peak Signal-to-Noise Ratio (PSNR) based on the L_2 distance between the frames penalizes differences in dynamics but also favors blurry predictions. Structured Similarity (SSIM) compares local patches to measure similarity in structure spatially. Learned Perceptual Image Patch Similarity (LPIPS) [Zhang et al., 2018] measures the distance between learned features extracted by a CNN trained for image classification. Frechet Video Distance (FVD) [Unterthiner et al., 2019], lower better, compares temporal dynamics of generated videos to the ground truth in terms of representations computed for action recognition.

5.3.1 Ablation Study

Table 5.1: **Baselines.** Results of our method (*Conditional*) by removing the dynamic part (*Depth-Only*) and by removing the conditioning of the dynamic part on the static (*Combined*).

Models	PSNR (↑)	SSIM (↑)	LPIPS (↓)
Depth-Only	13.02 ± 0.44	0.301 ± 0.021	0.523 ± 0.014
Combined	**14.45 ± 0.35**	0.378 ± 0.019	0.533 ± 0.016
Conditional	14.32 ± 0.33	**0.383 ± 0.020**	**0.501 ± 0.016**

KITTI [Geiger et al., 2012, Geiger et al., 2013]

Models	PSNR (↑)	SSIM (↑)	LPIPS (↓)
Depth-Only	19.97 ± 0.48	0.580 ± 0.016	0.445 ± 0.014
Combined	21.00 ± 0.41	0.631 ± 0.014	0.309 ± 0.009
Conditional	**21.43 ± 0.43**	**0.643 ± 0.014**	**0.306 ± 0.009**

Cityscapes [Cordts et al., 2016]

We first examine the importance of each contribution on KITTI and Cityscapes in Table 6.1. We start with a simplified version of our model called *Depth-Only* by removing the dynamic latent variables and the flow decoder. This model inherits the weakness of depth and ego-motion estimation methods by assuming a completely

Table 5.2: **Quantitative Results on KITTI.** We compare our *Combined* and *Conditional* models to the other video prediction approaches on KITTI [Geiger et al., 2012, Geiger et al., 2013]. The best results are shown in bold and the second best underlined.

Models	PSNR (↑)	SSIM (↑)	LPIPS (↓)
SVG [Denton and Fergus, 2018]	12.70 ± 0.70	0.329 ± 0.030	0.594 ± 0.034
SRVP [Franceschi et al., 2020]	13.41 ± 0.42	0.336 ± 0.034	0.635 ± 0.021
SLAMP	13.46 ± 0.74	0.337 ± 0.034	0.537 ± 0.042
Improved-VRNN [Castrejon et al., 2019]	14.15 ± 0.47	<u>0.379</u> ± 0.023	**0.372 ± 0.020**
Ours-Combined	**14.45 ± 0.35**	0.378 ± 0.019	0.533 ± 0.016
Ours-Conditional	<u>14.32</u> ± 0.33	**0.383 ± 0.020**	0.501 ± 0.016

static scene. Therefore, it cannot model the motion of the dynamic objects but it still performs reasonably well, especially on KITTI, since the background typically covers a large portion of the image. In the next two models, we include dynamic latent variables. We evaluate the importance of conditioning of dynamic variables on the static, *Combined* versus *Conditional*. For the *Combined*, we independently model static and dynamic latent variables and simply combine them in the end with the predicted mask. For the *Conditional*, we conditioned the dynamic latent variable on the static latent variable. First, both models improve the results compared to the *Depth-Only* case. This confirms our intuition about modelling the two types of motion in the scene separately. The two perform similarly on KITTI but the *Conditional* outperforms the *Combined* on Cityscapes due to larger number of moving foreground objects.

5.3.2 Quantitative Results

We trained and evaluated stochastic video prediction methods on KITTI and Cityscapes by using the same number of conditioning frames including SVG [Denton and Fergus, 2018], SRVP [Franceschi et al., 2020], SLAMP, and Improved-VRNN [Castrejon et al., 2019]. We optimized their architectures for a fair comparison (see Appendix).

Table 5.3: **Quantitative Results on Cityscapes.** We compare our *Combined* and *Conditional* models to the other video prediction approaches on Cityscapes [Cordts et al., 2016]. The best results are shown in bold and the second best underlined.

Models	PSNR (↑)	SSIM (↑)	LPIPS (↓)
SVG [Denton and Fergus, 2018]	20.42 ± 0.63	0.606 ± 0.023	0.340 ± 0.022
SRVP [Franceschi et al., 2020]	20.97 ± 0.43	0.603 ± 0.016	0.447 ± 0.014
SLAMP [Akan et al., 2021]	**21.73 ± 0.76**	**0.649 ± 0.025**	<u>0.294</u> ± 0.022
Improved-VRNN [Castrejon et al., 2019]	21.42 ± 0.67	0.618 ± 0.020	**0.260 ± 0.014**
Ours-Combined	21.00 ± 0.41	0.631 ± 0.014	0.309 ± 0.009
Ours-Conditional	<u>21.43</u> ± 0.43	<u>0.643</u> ± 0.014	0.306 ± 0.009

Frame-Level Evaluations: We report the frame-level evaluation results on Table 5.2 and Table 5.3.Recent methods including SLAMP, Improved-VRNN [Castrejon et al., 2019], and our models clearly outperform SVG [Denton and Fergus, 2018] and SRVP [Franceschi et al., 2020] in terms of all three metrics. Improved-VRNN [Castrejon et al., 2019] [1] achieves that by using five times more parameters compared to our models (57M vs. 308M). A recent study shows the importance of attacking the under-fitting issue in video prediction [Babaeizadeh et al., 2021]. The results can be improved by over-parameterizing the model and using data augmentation to prevent over-fitting. This finding is complementary to other approaches including ours, however, it comes at a cost in terms of run-time. The time required to generate 10 samples is 40 seconds for Improved-RNN compared to 1 second, or significantly less in case of SRVP, for other methods.

SLAMP achieves the top-performing results on Cityscapes in terms of PSNR and SSIM by explicitly modelling the motion history. The separation between the pixel and the motion space is achieved by learning a separate distribution for each with a slight increase in complexity compared to SVG. We follow a similar decomposition as

[1]The results of Improved-VRNN on Cityscapes is different from the ones in the original paper since we retrained their model on a similar resolution to ours with the same number of conditioning frames as ours.

static and dynamic but we differentiate between the two types of motion in driving. As a result, our models clearly outperform SLAMP on KITTI where the camera motion is large due to low frame-rates. In addition, our models can produce reliable depth predictions for the static part of the scene (Table 5.7). In SLAMP, we report results on generic video prediction datasets such as MNIST, KTH [Schüldt et al., 2004a] and BAIR [Ebert et al., 2017a] as well. In SLAMP-3D, we focus on driving scenarios by utilizing the domain knowledge for a better modelling of the static part. In case of SLAMP, the pixel decoder only focuses on cases that cannot be handled by the flow decoder, e.g. occlusions. Whereas in our case, the static models the whole background, leading not only to a better segmentation of the background in the mask but also to better results in the background on KITTI, as shown in Table 5.5.

Table 5.4: **FVD Scores on KITTI and Cityscapes.**

Dataset	KITTI	Cityscapes
SVG [Denton and Fergus, 2018]	1733 ± 198	870 ± 95
SRVP [Franceschi et al., 2020]	1792 ± 190	1409 ± 138
SLAMP	1585 ± 154	796 ± 89
VRNN [Castrejon et al., 2019]	**1022 ± 145**	**658 ± 80**
Ours-Combined	1463 ± 186	793 ± 86
Ours-Conditional.	1297 ± 142	789 ± 84

Video-Level Evaluations: We also use a video level evaluation metric, FVD [Unterthiner et al., 2019], for a video-level comparison. According to the results in Table 5.4, VRNN performs the best in terms of FVD. Our models, both combined and conditional, outperform all the other methods and approach the performance of VRNN with a nearly 40 times shorter inference time.

Foreground and Background Evaluations: We compare the prediction performance separately in the foreground and the background regions of the scene in Table 5.5 and Table 5.6 on KITTI and Cityscapes, respectively. We use off-the-shelf

Table 5.5: **Foreground and Background Evaluations on KITTI.** We use off-the-shelf semantic segmentation network to identify the foreground regions, mask all the other regions and compute the metrics separately in the foreground and the background.

	Models	PSNR (↑)	SSIM (↑)	LPIPS (↓)
Foreground	SVG [Denton and Fergus, 2018]	26.47 ± 0.31	0.9147 ± 0.0026	0.1183 ± 0.0027
	SRVP [Franceschi et al., 2020]	27.28 ± 0.31	0.9162 ± 0.0026	0.1244 ± 0.0029
	SLAMP	26.93 ± 0.32	0.9154 ± 0.0026	0.1125 ± 0.0026
	Improved-VRNN [Castrejon et al., 2019]	26.86 ± 0.27	0.9192 ± 0.0025	**0.0877 ± 0.0020**
	Ours-Combined	**27.44 ± 0.29**	**0.9193 ± 0.0025**	0.1093 ± 0.0024
	Ours-Conditional	26.80 ± 0.29	0.9161 ± 0.0026	0.1046 ± 0.0023
Background	SVG [Denton and Fergus, 2018]	13.30 ± 0.07	0.4003 ± 0.0034	0.5227 ± 0.0031
	SRVP [Franceschi et al., 2020]	13.97 ± 0.06	0.4080 ± 0.0033	0.5498 ± 0.0028
	SLAMP	14.09 ± 0.08	0.4089 ± 0.0032	0.4707 ± 0.0029
	Improved-VRNN [Castrejon et al., 2019]	14.82 ± 0.07	0.4417 ± 0.0037	**0.3380 ± 0.0031**
	Ours-Combined	**15.11 ± 0.06**	0.4455 ± 0.0032	0.4712 ± 0.0028
	Ours-Conditional	15.02 ± 0.05	**0.4507 ± 0.0031**	0.4451 ± 0.0028

semantic segmentation models to extract the objects in the scene and assume some of the semantic classes as the foreground. Specifically, we use the pre-trained model by [Zhu et al., 2019, Reda et al., 2018], which is the best model on KITTI semantic segmentation leaderboard with code available (the second-best overall), to obtain the masks on this dataset. On Cityscapes, we use the pre-trained model by [Tao et al., 2020] for similar reasons. We choose the following classes as the foreground objects: "person, rider, car, truck, bus, train, motorcycle, bicycle" and consider the remaining pixels as the background. For foreground results, we extract the foreground regions based on the segmentation result and ignore all the other pixels in the background by assigning the mean color, gray. We do the opposite masking for the background region by setting all of the foreground regions to gray, and calculate the metrics again.

Table 5.6: **Foreground and Background Evaluations on Cityscapes.** We use off-the-shelf semantic segmentation network to identify the foreground regions, mask all the other regions and compute the metrics separately in the foreground and the background.

	Models	PSNR (↑)	SSIM (↑)	LPIPS (↓)
Foreground	SVG [Denton and Fergus, 2018]	30.63 ± 0.14	0.9483 ± 0.0006	0.0676 ± 0.0005
	SRVP [Franceschi et al., 2020]	30.85 ± 0.14	0.9474 ± 0.0006	0.0763 ± 0.0006
	SLAMP	<u>31.71</u> ± 0.15	<u>0.9536</u> ± 0.0005	0.0577 ± 0.0005
	Improved-VRNN [Castrejon et al., 2019]	30.65 ± 0.14	0.9495 ± 0.0006	**0.0554 ± 0.0004**
	Ours-Combined	31.52 ± 0.14	<u>0.9536</u> ± 0.0005	<u>0.0568</u> ± 0.0004
	Ours-Conditional	**31.77 ± 0.14**	**0.9542 ± 0.0005**	0.0579 ± 0.0005
Background	SVG [Denton and Fergus, 2018]	21.24 ± 0.04	0.6699 ± 0.0014	0.2620 ± 0.0011
	SRVP [Franceschi et al., 2020]	21.96 ± 0.04	0.6757 ± 0.0014	0.3358 ± 0.0012
	SLAMP	**22.66 ± 0.05**	**0.7087 ± 0.0014**	<u>0.2341</u> ± 0.0011
	Improved-VRNN [Castrejon et al., 2019]	<u>22.52</u> ± 0.07	0.6811 ± 0.0017	**0.2155 ± 0.0013**
	Ours-Combined	21.87 ± 0.04	0.6935 ± 0.0013	0.2496 ± 0.0009
	Ours-Conditional	22.36 ± 0.05	<u>0.7026</u> ± 0.0013	0.2426 ± 0.0009

On Cityscapes, our *Combined* and *Conditional* models achieve the best results in terms of PSNR and SSIM metrics in the foreground regions. The *Conditional* outperforms the *Combined*, showing the importance of conditioning of dynamic latent variables on the static for modelling the independent motion of foreground objects. However, SLAMP and Improved-VRNN outperform our models in the background. Improved-VRNN's performance is especially impressive in terms of LPIPS, consistently in all regions on both datasets. On KITTI, our *Combined* model performs the best in foreground regions in terms of PSNR and SSIM. For background regions, our *Combined* and *Conditional* models are the two best-performing models in terms of PSNR and SSIM. In summary, we validate our two claims; first, to better model the motion history of foreground objects on the more dynamic Cityscapes and second, to better model the larger motion in the background on KITTI due to a smaller frame rate.

Evaluation of Depth Predictions: In order to validate the quality of our depth

Table 5.7: **Evaluation of Depth Predictions.** We compare depth predictions from our *Combined* and *Conditional* models to the state-of-the-art unsupervised monocular depth prediction method Monodepth2 [Godard et al., 2019] on KITTI. We predict future depth without seeing the target frame.

Models	Abs Rel (↓)	Sq Rel (↓)	RMSE (↓)	RMSE$_{log}$ (↓)	$\delta < 1.25$ (↑)	$\delta < 1.25^2$ (↑)	$\delta < 1.25^3$ (↑)
Ours-Combined	0.204	2.388	7.232	0.289	0.729	0.892	0.949
Ours-Conditional	0.221	3.173	7.474	0.298	0.724	0.887	0.944
Monodepth2 [Godard et al., 2019]	**0.136**	**1.095**	**5.369**	**0.213**	**0.836**	**0.946**	**0.977**

predictions, we evaluate them on KITTI with respect to a state-of-the-art monocular depth estimation approach Monodepth2 [Godard et al., 2019] by training it on a similar resolution to ours (320 × 96). Monodepth2 serves as an upper-bound for the performance of our models because we predict future depth based on previous frame predictions without actually seeing the target frame. As can be seen from Table 5.7, both our models perform reasonably well in all metrics [Eigen et al., 2014] in comparison to Monodepth2.

Run-time comparisons: We compare the methods in terms of their run-time in Table 5.8. We measure the time needed to generate 10 samples and report the results in seconds as the average over the dataset. There is a trade off between the run-time and the prediction performance of the models. Improved-VRNN has the largest run-time by far due to its hierarchical latent space, with nearly 40 seconds of processing time for ten samples. The performance of Improved-VRNN is impressive, especially in terms of both LPIPS and FVD, however costly in terms of both memory and run-time, therefore not applicable to autonomous driving. Our models, both Combined and Conditional, have a similar run-time on KITTI and only a slight increase in run-time compared to vanilla stochastic video generation model SVG [Denton and Fergus, 2018]. The state-space model SRVP [Franceschi et al., 2020] leads to the best run-time by removing the need for autoregressive predictions but at the cost of low performance on real-world datasets.

Table 5.8: **Run-time Comparison of the Models.** We measure the time needed to generate 10 samples and report the results in seconds as the average over the dataset. We only measure the GPU processing time and discard the other operations.

Models	**KITTI**	**Cityscapes**
SVG [Denton and Fergus, 2018]	1.04	0.90
SLAMP	2.03	1.44
SRVP [Franceschi et al., 2020]	0.11	0.12
Improved VRNN [Castrejon et al., 2019]	39.25	41.59
Ours-Combined	1.02	1.16
Ours-Conditional	1.03	1.21

5.3.3 Qualitative Results

We visualize the results of our model including the intermediate predictions on KITTI (Fig. 5.6) as well as the final prediction in comparison to previous methods on Cityscapes (Fig. 5.7). Please see Appendix for more results on both datasets. As can be seen from Fig. 5.6, our model can predict structure and differentiate between the two types of motion in the scene. First, the camera motion in the background is predicted (*Ego-motion*) and the motion of the foreground object is predicted as residual on top of it (*Residual Flow*). The scene decomposition into static and dynamic and separate modeling of motion in each allow our model to generate better predictions of future in dynamic scenes. Fig. 5.7 shows a case where our model can correctly estimate the change in the shape and the size of the two vehicles moving in comparison to previous approaches.

We evaluate the diversity of predictions qualitatively in Fig. 5.9. For this purpose, we visualize the standard deviations of generated frames over 100 samples for Improved-VRNN [Castrejon et al., 2019] and our *Conditional* model. The higher the standard deviation at a pixel, the higher the uncertainty is at that pixel due to differences in samples. While the uncertainty is mostly uniform over the image for Improved-VRNN, our model can pinpoint it to the foreground regions thanks to our separate modeling of the motion history in the background and the foreground regions.

5.4 Conclusion and Future Work

We introduced a conditional stochastic video prediction model by decomposing the scene into static and dynamic in driving videos. We showed that separate modelling of foreground and background motion leads to better future predictions. Our method is among the top-performing methods overall while still being efficient. The modelling of the static part with domain knowledge is justified on KITTI with large camera motion. The separate modelling of foreground objects as residual motion leads to better results in dynamic scenes of Cityscapes. Our visualizations show that flow prediction can capture the residual motion of foreground objects on top of ego-motion thanks to our conditional model.

We found that modelling the stochasticity in terms of underlying factors is beneficial for autonomous driving. From this point forward, we see three important directions to improve stochastic video prediction in autonomous driving. Since the uncertainty is mostly due to foreground objects, stochastic methods can focus on foreground objects with more sophisticated models. Next, there is a limitation in our model due to the warping operation. Our model can predict the future motion of a car visible in the conditioning frames but it cannot predict a car appearing after that. For that, we need other ways of obtaining self-supervision without warping.

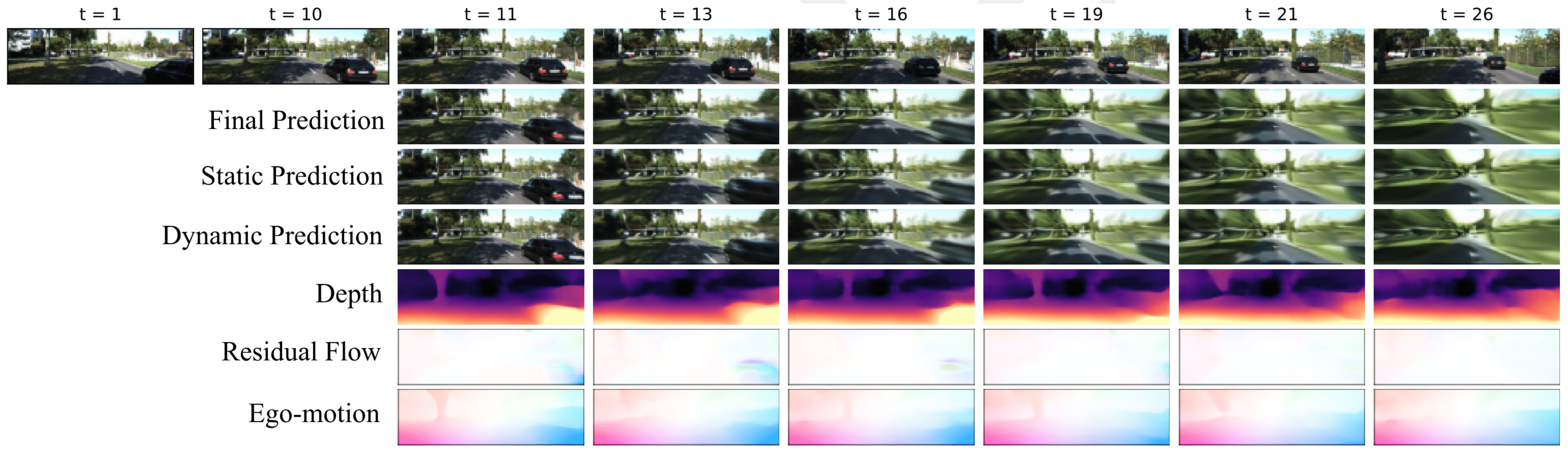

Figure 5.6: **Conditional Model on KITTI.** The top row shows the ground truth, followed by the frame predictions by the final, the static, and the dynamic. The last three rows show depth, residual flow, and flow due to ego-motion. The structure and camera motion can explain the static part of the scene and the independent motion of the moving car on the left is captured by the residual flow.

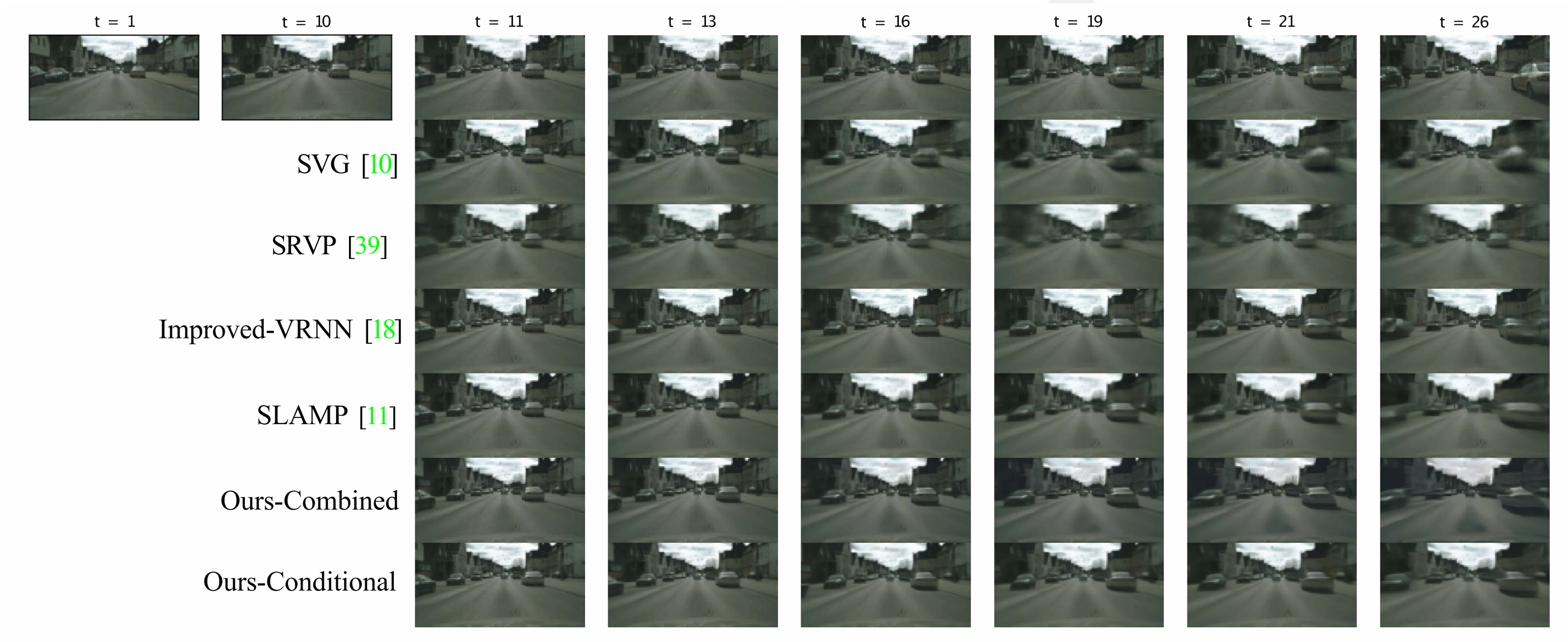

Figure 5.7: **Comparison to Other Methods on Cityscapes.** Our *Conditional* model can better predict the motion of the vehicles moving on the left and on the right thanks to the separate modeling of the motion history of background and foreground regions.

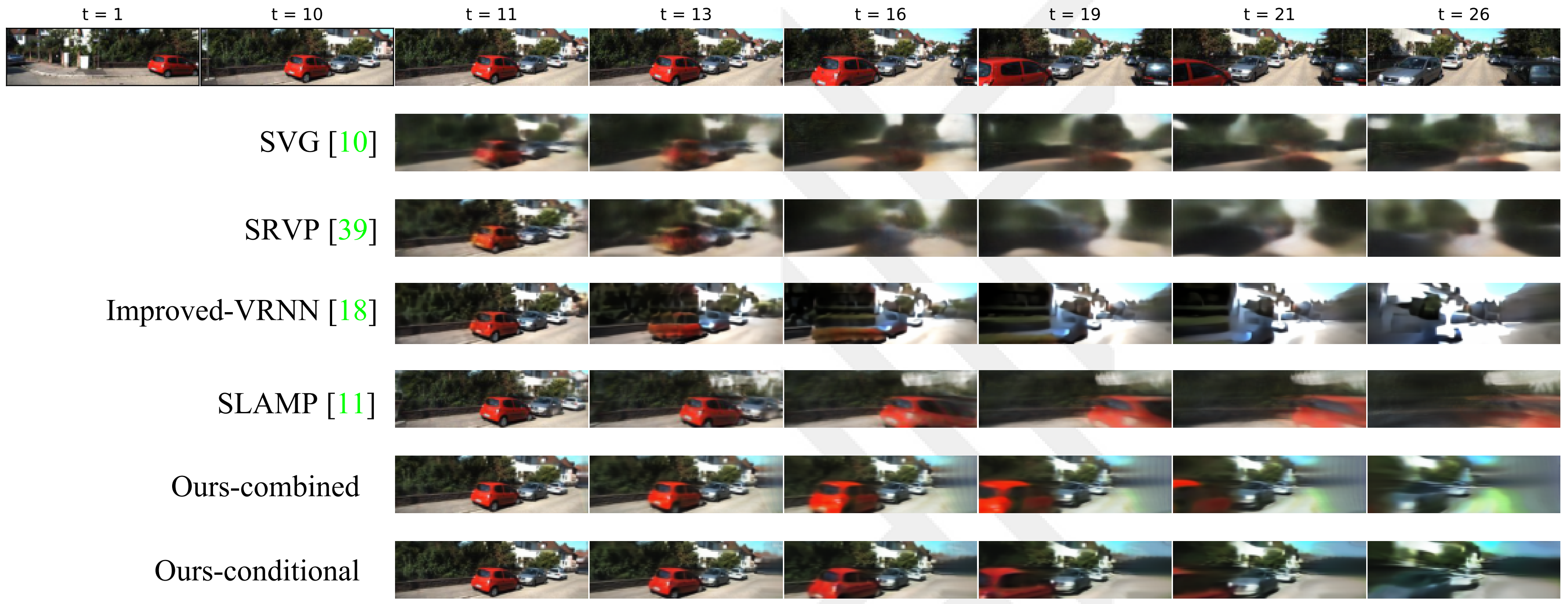

Figure 5.8: **Difficult Turning Case on KITTI.** Our *Conditional* and *Combined* model can better predict future after the conditioning frames. Our models, especially the conditional model, can better preserve the red car in their predictions for a longer time. All of the other methods fail in the later time steps due to the difficulty of predicting future in the turning case which is not frequent in the dataset.

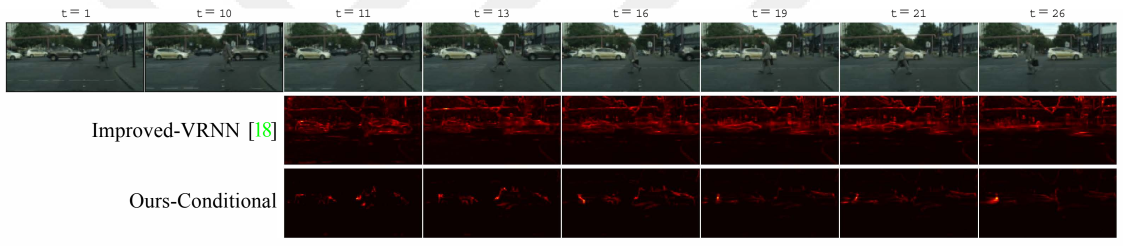

Figure 5.9: **Diversity of Samples on Cityscapes.** We compare the standard deviations of 100 samples generated by Improved-VRNN [Castrejon et al., 2019] and our *Conditional* model. Our model can pinpoint uncertain regions around the foreground objects in comparison to more uniform results of Improved-VRNN.

Chapter 6

STRETCHBEV: STRETCHING FUTURE INSTANCE PREDICTION SPATIALLY AND TEMPORALLY

6.1 Introduction

Sequential visual data has been used before in autonomous driving tasks. However, the pixel space of the images or videos is noisy and contains a lot of redundant information. Instead, a more compact representation, which is much more suitable for the downstream tasks of autonomous driving, can be used. For example, understanding the 3D structure of the world and the motion of the agents around it can help the tasks in driving scenarios. The bird's-eye view (BEV) representation meets these requirements by first fusing information from multiple cameras into a 3D point cloud and then projecting the points to the ground plane [Philion and Fidler, 2020]. This leads to a compact representation where the agents, the lanes, and all the necessary information are preserved, and the redundant ones are dismissed.

In this chapter, we explore the potential of stochastic future prediction for self-driving in BEV representation. The aim is to generate admissible and diverse results in long sequences with an efficient and compact BEV representation.

Future prediction from the BEV representation has been recently proposed in FIERY [Hu et al., 2021]. Their method has several drawbacks. First, the lack of diversity might be a problem due to two distributions representing the present and the future. These two distributions may not be enough to model the diversity because the predictions of FIERY degrade over longer time spans due to the limited representation capability of a single distribution for increasing diversity in longer predictions. For example, for planning, long-term multiple future predictions are crucial.

Following FIERY, we start from the same BEV representation and predict the same output modalities to be comparable. Differently, instead of two distributions for the present and the future, we propose to learn time-dependent distributions by predicting a residual change at each time step to better capture long-term dynamics. Furthermore, we show that by sampling a random variable at each time step, we can increase the diversity of future predictions while still being accurate and efficient. For efficiency, we use a state-space model [Murphy, 2023] instead of costly auto-regressive models.

6.2 Methodology

6.2.1 A Compact Representation for Future Prediction

Modern self-driving vehicles are typically equipped with multiple cameras observing the scene from multiple viewpoints. Placing cameras on the vehicle is cheap but processing information even from a single camera can be quite expensive. The traditional approach in computer vision is to extract low-level and semantic cues from these cameras and then fuse them into a holistic scene representation to perform prediction and planning. Recent success of end-to-end methods in driving has led to a rethinking of this approach. Furthermore, building and maintaining HD maps require a significant effort which is expensive and hard to scale. A better approach is to learn a geometrically consistent scene representation which can also mark the location, the motion, and even the semantics of the dynamic objects in the scene.

The bird's-eye view (BEV) representation initially proposed in [Philion and Fidler, 2020], takes image $\mathbf{x}_t^i$ at time t from each camera $i \in \{1, \cdots, 6\}$ and fuses them into a compact BEV representation $\mathbf{s}_t$. This is achieved by encoding each image and also by predicting a distribution over the possible depth values. The BEV features are obtained by weighting the encoded image features according to depth probabilities predicted. These features are first lifted to 3D by using known camera intrinsic and extrinsic parameters and then the height dimension is pooled over to project the features into the bird's-eye view. This results in the state representation $\mathbf{s}_t$ that we use for future prediction as explained next.

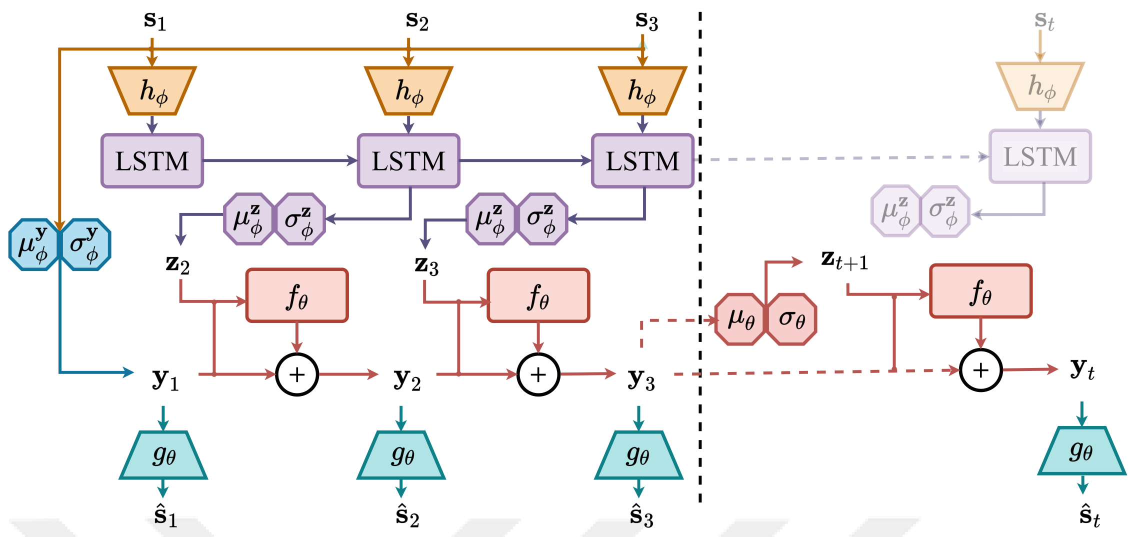

Figure 6.1: **Architecture for Learning Temporal Dynamics.** This figure shows the inference procedure of our model StretchBEV. We start with the first $k = 3$ conditioning frames where we sample the stochastic latent variables from the posterior distribution (purple). On the right, we show the prediction at a step t after the conditioning frames where we sample from the learned future distribution (red). The dashed vertical line marks the conditioning frames.

6.2.2 Learning Temporal Dynamics in BEV

Notation: In our formulation, $\mathbf{s}_{1:T}$ denotes a sequence of BEV feature maps representing the state of the vehicle and its environment for T time steps. In stochastic future prediction, the goal is to predict possible future states $\hat{\mathbf{s}}_{k+1:T}$ conditioned on the state in the first k time steps. Precisely, we condition on the first $k = 3$ steps and predict future in varying lengths from 4 to 12 steps ahead.

Differently from previous work on stochastic video prediction [Denton and Fergus, 2018, Franceschi et al., 2020], the BEV state $\mathbf{s}_t$ that is input to our stochastic prediction framework is the intermediate representations in a high dimensional space rather than a video frame in the pixel space as explained in Section 6.2.1. Similarly, the predicted output $\hat{\mathbf{s}}_t$ represents the predictions of future in the same high dimensional space. We decode these high dimensional future predictions into various output modalities $\hat{\mathbf{o}}_t$ such as future instance segmentation and motion as explained in Section 6.2.4. While these modalities need to be trained in a supervised manner in contrast to typical self-supervised stochastic video prediction frameworks [Denton and Fergus, 2018, Franceschi et al., 2020], they provide interpretability which is critical in self-driving. Furthermore, using these modalities in the posterior in addition to the future state representations improves the results significantly as we show in our experiments.

Stochastic Residual Dynamics:

Following [Franceschi et al., 2020], we learn the changes in the states through time with stochastic residual updates to a sequence of latent variables. For each state $\mathbf{s}_t$, there is a corresponding latent variable $\mathbf{y}_t$ generating it, independent of the previous states (Fig. 6.1). Each $\mathbf{y}_{t+1}$ only depends on the previous $\mathbf{y}_t$ and a stochastic variable $\mathbf{z}_{t+1}$. The randomness is introduced by the stochastic latent variable $\mathbf{z}_{t+1}$ which is sampled from a normal distribution learned from the previous state's latent variable only:

$$\mathbf{z}_{t+1} \sim \mathcal{N}\left(\mu_\theta(\mathbf{y}_t), \sigma_\theta(\mathbf{y}_t)\,\mathbf{I}\right) \tag{6.1}$$

Given $\mathbf{z}_{t+1}$, the dependency between the latent variables $\mathbf{y}_t$ and $\mathbf{y}_{t+1}$ is deterministic through the residual update:

$$\mathbf{y}_{t+1} = \mathbf{y}_t + f_\theta\left(\mathbf{y}_t, \mathbf{z}_{t+1}\right) \tag{6.2}$$

where f_θ is a small CNN to learn the residual updates to $\mathbf{y}_t$. We learn the distribution of future states from the corresponding latent variable as a normal distribution with constant diagonal variance: $\hat{\mathbf{s}}_t \sim \mathcal{N}(g_\theta(\mathbf{y}_t))$. The first latent variable is inferred from the conditioning frames by assuming a standard Gaussian prior: $\mathbf{y}_1 \sim \mathcal{N}\left(\mathbf{0}, \mathbf{I}\right)$.

On the Content Variable: In video prediction, a common practice is to represent the static parts of the scene with a content variable which allows the model to focus on the moving parts. On the contrary to the state of the art in video prediction [Franceschi et al., 2020], the content variable does not improve the results in our case (see Appendix), therefore we omit it in our formulation here. This can be attributed to the details in the background that are confusing for learning dynamics while operating in the pixel space, but in our case, most of these details are already suppressed in the BEV representation.

On Diversity: In contrast to a present and a future distribution in FIERY [Hu et al., 2020, Hu et al., 2021], there is a distribution learned at each time step in our model. This corresponds to sampling stochastic random variables at each time step as opposed to sampling once to represent all the future frames. This is the key property which allows our model to produce diverse predictions in long sequences. By sampling from a learned distribution at each time step, our model learns to represent the complex dynamics of future frames, even for predictions further away from the conditioning frames. Furthermore, our model does not need a separate temporal block for learning the dynamics prior to learning these distributions. The dynamics are learned through the temporal evolution of latent variables by also considering the randomness of the future predictions with stochastic random variables at each time step. This not only increases the diversity of predictions but also alleviates the need for a separate temporal block, e.g. with 3D convolutions. Note that our formulation is still efficient, almost the same inference time as FIERY (see Appendix), because the latent variables are low dimensional and each state is generated independently.

Moreover, FIERY uses only a single vector of latent variables which is expanded to the spatial grid to generate futures states probabilistically. Therefore, it uses the same stochastic noise in all the coordinates of the grid. However, in our model, we have a separate random variable at each coordinate of the grid to model the uncertainty spatially as well. Training FIERY with a separate random variable at each location of the grid results in diverging loss values.

6.2.3 Variational Inference and Architecture

Following the generative process in [Franceschi et al., 2020], the joint probability of the BEV states $\mathbf{s}_{1:T}$, the output modalities $\mathbf{o}_t$, and the latent variables $\mathbf{z}_{1:T}$ and $\mathbf{y}_{1:T}$

is as follows:

$$p\left(\mathbf{s}_{1:T}, \mathbf{o}_{1:T}, \mathbf{z}_{2:T}, \mathbf{y}_{1:T}\right) = p\left(\mathbf{y}_1\right) \prod_{t=2}^{T} p\left(\mathbf{z}_t, \mathbf{y}_t | \mathbf{y}_{t-1}\right) \prod_{t=1}^{T} p\left(\mathbf{o}_t | \mathbf{s}_t\right) p\left(\mathbf{s}_t | \mathbf{y}_t\right) \tag{6.3}$$

$$p\left(\mathbf{z}_t, \mathbf{y}_t | \mathbf{y}_{t-1}\right) = p\left(\mathbf{y}_t | \mathbf{y}_{t-1}, \mathbf{z}_t\right) p\left(\mathbf{z}_t | \mathbf{y}_{t-1}\right) \tag{6.4}$$

The relationship between $\mathbf{y}_t$ and $\mathbf{y}_{t-1}$ in $p\left(\mathbf{y}_t | \mathbf{y}_{t-1}, \mathbf{z}_t\right)$ (6.4) is deterministic through the stochastic latent residual as formulated in (6.2). Similarly for the term $p\left(\mathbf{o}_t | \mathbf{s}_t\right)$ in (6.3), the output modalities $\mathbf{o}_t$ is learned from $\mathbf{s}_t$ with a deterministic decoder in a supervised manner (Section 6.2.4).

Our goal is to maximize the likelihood of the BEV states extracted from the frames (Section 6.2.1) and the corresponding output modalities $p\left(\mathbf{s}_{1:T}, \mathbf{o}_{1:T}\right)$. For that purpose, we learn a deep variational inference model q parametrized by ϕ which is factorized as follows:

$$\begin{aligned} q_{Z,Y} &\triangleq q\left(\mathbf{z}_{2:T}, \mathbf{y}_{1:T} | \mathbf{s}_{1:T}, \mathbf{o}_{1:T}\right) \\ &= q\left(\mathbf{y}_1 | \mathbf{s}_{1:k}\right) \prod_{t=2}^{T} q\left(\mathbf{z}_t | \mathbf{s}_{1:t}, \mathbf{o}_{1:t}\right) q\left(\mathbf{y}_t | \mathbf{y}_{t-1}, \mathbf{z}_t\right) \end{aligned} \tag{6.5}$$

where $k = 3$ is the number of conditioning frames and $q\left(\mathbf{y}_t | \mathbf{y}_{t-1}, \mathbf{z}_t\right)$ is equal to $p\left(\mathbf{y}_t | \mathbf{y}_{t-1}, \mathbf{z}_t\right)$ with the residual update as explained above. We obtain two versions of our model by keeping (StretchBEV-P) or removing (StretchBEV) the dependency of $\mathbf{z}_t$ on $\mathbf{o}_{1:t}$ in the posterior in (6.5). We refer the reader to Appendix for the derivation of the following evidence lower bound (ELBO):

$$\begin{aligned} \log p\left(\mathbf{s}_{1:T}, \mathbf{o}_{1:T}\right) &\geq \mathcal{L}\left(\mathbf{s}_{1:T}, \mathbf{o}_{1:T}; \theta, \phi\right) \\ &\triangleq - D_{\mathrm{KL}}\left(q\left(\mathbf{y}_1 | \mathbf{s}_{1:k}\right) \,||\, p\left(\mathbf{y}_1\right)\right) \\ &\quad + \mathbb{E}_{\left(\tilde{\mathbf{z}}_{2:T}, \tilde{\mathbf{y}}_{1:T}\right) \sim q_{Z,Y}} \left[\sum_{t=1}^{T} \log p\left(\mathbf{s}_t | \tilde{\mathbf{y}}_t\right) p\left(\mathbf{o}_t | \mathbf{s}_t\right) \right. \\ &\quad \left. - \sum_{t=2}^{T} D_{\mathrm{KL}}\Big(q\left(\mathbf{z}_t | \mathbf{s}_{1:t}, \mathbf{o}_{1:t}\right) \,||\, p\left(\mathbf{z}_t | \tilde{\mathbf{y}}_{t-1}\right)\Big) \right] \end{aligned} \tag{6.6}$$

where D_{KL} denotes the Kullback-Leibler (KL) divergence, θ and ϕ represent model and variational parameters, respectively. Following the common practice, we choose $q\left(\mathbf{y}_1 | \mathbf{s}_{1:k}\right)$ and $q\left(\mathbf{z}_t | \mathbf{s}_{1:t}, \mathbf{o}_{1:t}\right)$ to be factorized Gaussian for analytically computing the

KL divergences and use the re-parametrization trick [Kingma and Welling, 2014] to compute gradients through the inferred variables. Then, the resulting objective function is maximizing the ELBO as defined in (6.6) and minimizing the supervised losses for the output modalities (Section 6.2.4).

We provide a summary of the steps in our temporal model as shown in Fig. 6.1. We start by fusing the images $\mathbf{x}_t^i$ at time t from each camera i into the BEV state $\mathbf{s}_t$ at each time step as explained in Section 6.2.1.

1. The resulting BEV states are still in high resolution, $\mathbf{s}_t \in \mathbb{R}^{C \times H \times W}$ where $(H, W) = (200, 200)$. Therefore, we first process them with an encoder h_ϕ to reduce the spatial resolution to 50×50.

2. The first latent variable $\mathbf{y}_1$ is inferred using a convolutional neural network on the first three encoded states.

3. The stochastic latent variable $\mathbf{z}_t$ is inferred at each time step from the respective encoded state, using a recurrent neural network which is a combination of a ConvGRU and convolutional blocks.

4. The residual change in the dynamics is predicted with f_θ based on both the previous state dynamics $\mathbf{y}_t$ and the stochastic latent variable $\mathbf{z}_{t+1}$ and added to $\mathbf{y}_t$ to obtain $\mathbf{y}_{t+1}$.

5. From each $\mathbf{y}_t$, the state $\hat{\mathbf{s}}_t$ is predicted in the original resolution with g_θ.

6. Finally, the output modalities $\hat{\mathbf{o}}_t$ are decoded from the state prediction $\hat{\mathbf{s}}_t$.

6.2.4 Decoding Future Predictions

Based on the predictions of the future states at each time step, we train a supervised decoder to output semantic segmentation, instance center, offsets, and future optical flow, similar to the previous work [Cheng et al., 2020, Hu et al., 2021]. The decoding function is a deterministic neural network that can be trained either jointly with the dynamics or independently, e.g. later for interpretability. The output modalities show both the location and the motion of instances at each time step. The motion predicted as future flow is used to track instances. We use the same supervised loss functions for each modality as the FIERY [Hu et al., 2021]. Although the decoding of future predictions is not necessary for planning and control, for example when training a driving agent to act on predictions, these predictions provide interpretability and allow to compare the methods in terms of various metrics evaluating each modality.

6.3 Experiments

6.3.1 Dataset and Evaluation Setting

We evaluate the performance of the proposed approach and compare to the state of the art method, FIERY [Hu et al., 2021], on the nuScenes dataset [Caesar et al., 2020]. On the nuScenes, there are 6 cameras with overlapping views which provide the ego vehicle with a complete view of its surroundings. The nuScenes dataset consists of 1000 scenes with 20 seconds long at 2 frames per second.

We first follow the training and the evaluation setting proposed in FIERY [Hu et al., 2021] for comparison by using 1.0 second of past context to predict 2.0 seconds of future context. Given the sampling rate of 2 frames per second, this setting corresponds to predicting 4 future frames conditioned on 3 past frames. We call this setting *short* in terms of temporal length and define two more settings for longer temporal predictions. In the *mid* and *long* settings, we double and triple the number of future frames to predict, i.e. 8 and 12, respectively, that corresponds to 4.0 and 6.0 seconds into the future. These settings are closer to the stochastic video prediction setup [Denton and Fergus, 2018, Franceschi et al., 2020, Akan et al., 2021] where there are typically many more frames to predict than the conditioning frames for measuring diversity and the performance of the models further away from the conditioning frames. Note that *short* and *long* refer to temporal length in our evaluations as opposed spatial coverage as defined in the previous work [Hu et al., 2021]. We also evaluate in terms of spatial coverage but call it *near* (30m$\times$30m) and *far* (100m $\times$100m) for clarity.

6.3.2 Training Details

Our models follow the input and output setting proposed in the previous work [Hu et al., 2021]. We process 6 camera images at a resolution of 224×480 pixels for each frame and construct the BEV state of size $200 \times 200 \times 64$. We further process the states into a smaller spatial resolution (50×50) for efficiency before learning the temporal dynamics but increase it back to the initial resolution afterwards. Given the predicted states, we use the same decoder architecture as the FIERY to decode the object centers, the segmentation masks, the instance offsets, and the future optical flow at a resolution of 200×200 pixels. We provide the details of the architecture in Appendix and we will release the code upon publication.

In our approach, learning temporal dynamics and decoding output modalities are separated from each other. Therefore, we can pre-train the temporal dynamics part without using the labels for the output modalities. In pre-training, our objective is to learn to match the future states that are extracted using a pre-trained BEV segmentation model [Philion and Fidler, 2020], conditioned on the past states. This approach is more similar to self-supervised stochastic video prediction methods [Denton and Fergus, 2018, Franceschi et al., 2020]. Furthermore, this way, learning of temporal dynamics can be improved by using camera sequences only as input which can be easily collected in large quantities. Then, we fine-tune the temporal dynamics with a smaller learning rate (see Appendix for details) while learning to decode the output modalities in a supervised manner. The alternative is to jointly train the temporal dynamics and supervised decoding without pre-training. We present the results of our model StretchBEV with and without pre-training.

StretchBEV does not use the labels ($\mathbf{o}_t$) for learning the temporal dynamics, it only uses them in the supervised loss to decode the output modalities. In our full model StretchBEV-P, we encode output modalities following FIERY and use them in the posterior for learning the temporal dynamics. During training, we sample the stochastic latent variables from the posterior and learn to minimize the difference between the posterior and the future distribution. During inference, we sample from the posterior in the conditioning frames and sample from the learned future distribution in the following steps as shown in Fig. 6.1.

6.3.3 Metrics

We use two different metrics for evaluating the decoded modalities, one frame level and another video level, that are also used in the previous work [Hu et al., 2021]. The first is Intersection over Union (IoU) to measure the quality of the segmentation at each frame. The second is Video Panoptic Quality (VPQ) to measure the quality of the segmentation and consistency of the instances through the video:

$$\text{VPQ} = \sum_{t=0}^{H} \frac{\sum_{(p_t,q_t)\in TP_t} \text{IoU}(p_t, q_t)}{|TP_t| + \frac{1}{2}|FP_t| + \frac{1}{2}|FN_t|} \tag{6.7}$$

where H is the temporal horizon considered, TP_t is the number of True Positives, FP_t the number of False Positives, and FN_t the number of False Negatives at a time step t. We report both metrics for both spatially near and far regions and temporally short, mid, and long spans as explained in Section 6.3.1.

We evaluate the diversity quantitatively in terms of Generalized Energy Distance (D_{GED}) [Szkely and Rizzo, 2017] by using $(1 - \text{VPQ})$ as the distance as proposed in FIERY [Hu et al., 2021].

6.3.4 Ablation Study

In Table 6.1, we evaluate the effect of different versions of our model using IoU and VPQ metrics in the short temporal setting to be comparable to the previous work FIERY [Hu et al., 2021]. We reproduced their results as shown in the row *Reproduced*. In the first part of the table (StretchBEV), we show the results without explicitly using the labels for future prediction. In that case, labels are only used for decoding the output modalities and back-propagated to future prediction through decoding. Although this introduces a two-stage training, we believe that reporting results using this separation is important for future work to focus on future prediction with more unlabelled data. We measure the effect of pre-training by learning to match our future predictions to the results of a pre-trained model [Philion and Fidler, 2020] in terms of the BEV state representation. Pre-training allows our model

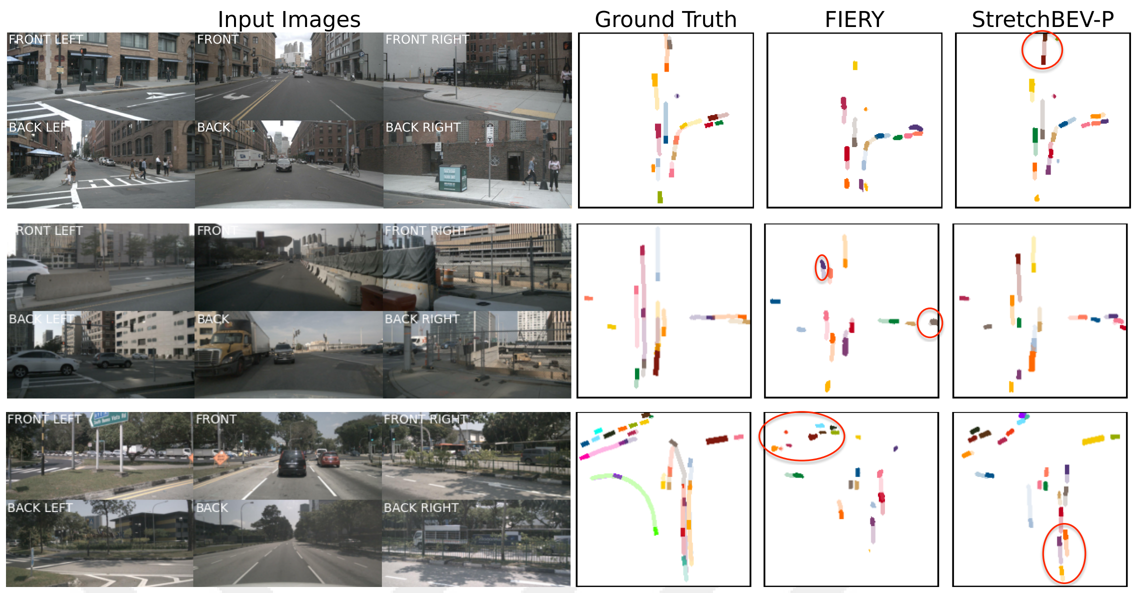

Figure 6.2: **Qualitative Comparison over Different Temporal Horizons.** In this figure, we qualitatively compare the results of our model StretchBEV-P **(right)** to the ground truth **(left)** and FIERY [Hu et al., 2021] **(middle)** over short **(top)**, mid **(middle)**, and long **(bottom)** temporal horizons. Each color represents an instance of a vehicle with its trajectory trailing in the same color transparently.

	Pre-training	Posterior w/labels	IoU (↑)		VPQ (↑)	
			Near	Far	Near	Far
StretchBEV	—	—	53.3	35.8	41.7	26.0
	✓		55.5	37.1	46.0	29.0
FIERY [Hu et al., 2021]	—	✓	**59.4**	36.7	50.2	29.4
Reproduced			58.8	35.8	50.5	29.0
StretchBEV-P	—	✓	58.1	**52.5**	**53.0**	**47.5**

Table 6.1: **Ablation Study.** In this table, we present the results for the two versions of our model with (StretchBEV-P) and without (StretchBEV) using the labels for the output modalities in the posterior while learning the temporal dynamics and also show the effect of pre-training for the latter in comparison to FIERY [Hu et al., 2021] and our reproduced version of their results (Reproduced).

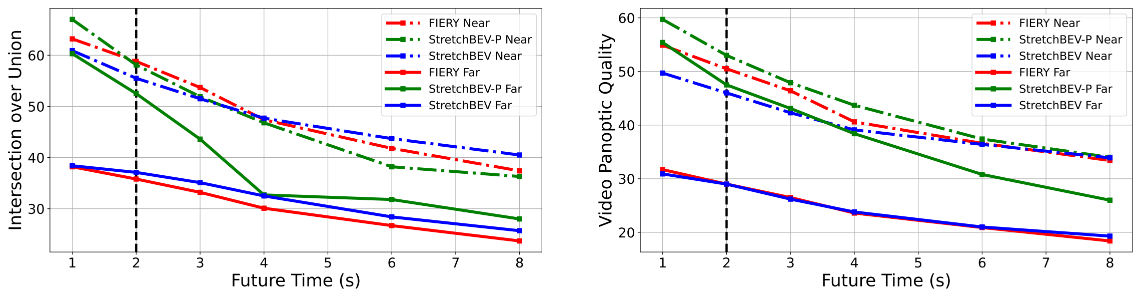

Figure 6.3: **Evaluation over Different Temporal Horizons.** We plot the performance of our models StretchBEV and StretchBEV-P in comparison to FIERY [Hu et al., 2021] over a range of temporal horizons from 1 second to 8 seconds in terms of IoU **(left)** and VPQ **(right)** for spatially far **(solid)** and near **(dashed)** regions separately. The vertical dashed line marks the training horizon.

to learn the dynamics before decoding and improves the results significantly in each metric.

In the second half of the Table 6.1, we report the results using the labels in future prediction by explicitly feeding their encoding to the posterior distribution with the same encoding used in [Hu et al., 2021] to learn the future distribution. The difference between StretchBEV and StretchBEV-P is that the first has access to the BEV encoding of future predictions while the latter has access to both the BEV encoding and the encoding of the output modalities to predict in the posterior distribution. As can be seen from the results, both FIERY and our model using the labels in the future distribution perform better. This shows the importance of using a more direct and accurate information about future while learning the posterior. Compared to FIERY, our model can use the labels in the conditioning frames during inference and improves the results, especially in spatially far regions and in terms of VPQ, which point to a higher quality in our predictions stretching spatially and temporally over the video.

6.3.5 Temporally Long Predictions

In longer temporal horizons, future prediction becomes increasingly difficult. This is mainly due to increasing uncertainty of future further away from conditioning frames. In Fig. 6.3, we present the results over different temporal horizons for our model with pre-training without using the labels in the posterior (StretchBEV), FIERY [Hu et al., 2021], and our model by using the labels in the posterior (StretchBEV-P). There is a separate plot for IoU on the left and for VPQ on the right with respect to the future time steps predicted, ranging from 1s to 8s. The vertical line in 2s marks the training horizon. In Appendix, we provide a table for the results over short, mid, and long temporal spans.

The negative effect of uncertain futures on each metric can be observed from the results of all the methods degrading from shorter to longer temporal spans. Our models perform better than FIERY in longer temporal spans. This is due to better handling of uncertainty with stochastic latent residual variables. Our method StretchBEV-P outperforms FIERY by significant margins, especially in terms of far VPQ in longer temporal horizons, showing consistent predictions in the overall scene throughout the video. This can be attributed to the difficulty of locating small vehicles in spatially far regions. Since StretchBEV-P has access to the labels via the posterior in the conditioning frames, it learns the temporal dynamics to correctly propagate them to the future frames, while StretchBEV and FIERY struggle to locate the instances in the first place. FIERY learns a single distribution for present and future each, therefore we cannot utilize the labels in the conditioning frames with FIERY. The results of StretchBEV outperforming the other two methods in terms of near IOU in longer temporal spans is promising for future prediction methods with less supervision.

In Fig. 6.2, we qualitatively compare the performance of our model StretchBEV-P on the right to FIERY in the middle over short, mid, and long temporal horizons in each row. In the first row, our model predicts the future trajectories that are more similar to the ground truth shown on the left. For example, FIERY fails to predict the trajectory of the vehicle in front (marked with red circle). In the second row, our model correctly segments the vehicles, whereas FIERY misses several vehicles far on the right and also, predicts a vehicle that does not exist (in purple on the top left). In the third row, our model predicts the future trajectories of the vehicles correctly while FIERY misses some of the vehicles (marked with red circles). The challenging case of a vehicle turning on the left (green in ground truth) is missed by both models. Some of the vehicles are not visible on the input images, e.g. the back camera in the long temporal horizon.

6.3.6 Segmentation

Methods	Segmentation IoU
Fishing-Cam [Hendy et al., 2020]	30.0
Fishing-LiDAR [Hendy et al., 2020]	44.3
FIERY [Hu et al., 2021]	57.3
StretchBEV	<u>58.8</u>
StretchBEV-P	**65.7**

Table 6.2: **Comparison of Semantic Segmentation Prediction.** In this table, we compare the predictions of our models, StretchBEV and StretchBEV-P for semantic segmentation to other BEV prediction methods in terms of IoU using the setting proposed in [Hendy et al., 2020], i.e. 32.0m $\times$ 19.2m at 10cm resolution over 2s future.

The previous work on bird's-eye view segmentation typically focuses on single image segmentation task with a couple of exceptions focusing on prediction. In Table 6.2, we compare our methods, StretchBEV and StretchBEV-P, to two BEV segmentation prediction methods [Hendy et al., 2020, Hu et al., 2021] using their setting with 32.0m $\times$ 19.2m at 10cm resolution. Both methods predict 2s into the future which corresponds to our short temporal setting. FIERY [Hu et al., 2021] outperforms the previous method [Hendy et al., 2020] even when using LiDAR, and our method significantly outperforms both methods.

6.3.7 Diversity

	Generalized Energy Distance (↓)					
	Short		Mid		Long	
	Near	Far	Near	Far	Near	Far
FIERY [Hu et al., 2021]	106.09	140.36	118.74	147.26	127.18	152.38
StretchBEV	103.97	132.38	114.11	138.15	119.01	142.51
StretchBEV-P	**82.04**	**85.51**	**94.02**	**98.45**	**101.90**	**109.12**

Table 6.3: **Quantitative Evaluation of Diversity.** This table compares the results of our models to the reproduced results of FIERY [Hu et al., 2021] in terms of Generalized Energy Distance based on VPQ (lower better) for evaluating diversity.

We quantitatively evaluate diversity by computing D_{GED} over 10 samples and show the results in Table 6.3 for our models, StretchBEV and StretchBEV-P and FIERY (our reproduced version). Both our models outperforms FIERY with lower distance scores, demonstrating higher levels of diversity in the samples quantitatively. The difference is especially apparent in spatially far regions. For qualitative comparison, in Fig. 6.4, we visualize three samples from FIERY (left) and our model (right) over short, mid, and long temporal spans from top to bottom. While FIERY generates almost the same predictions in all three samples, our model can generate diverse predictions of future (marked with red). In the first row, our model can predict the turning behavior of the green vehicle at different speeds. In the second row, our model learns to adjust the speed of nearby vehicles proportionally, as in the case of purple, blue, and gray vehicles in the middle. Similarly, in the third row, our model can generate different predictions for the moving vehicles in the middle.

Run-time Comparison: We compare the inference speed of our model StretchBEV-P and FIERY [Hu et al., 2021] by measuring the average time needed to process a validation example in inference over 250 forward passes. Both models have almost the same inference speed (FIERY: 0.6436 seconds/example vs. StretchBEV-P:

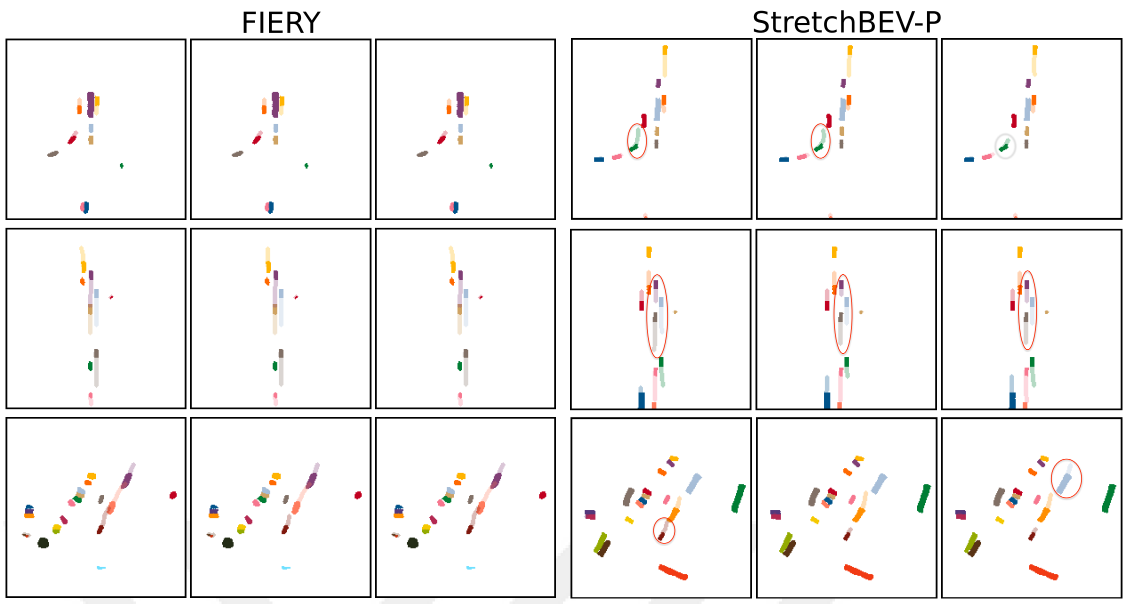

Figure 6.4: **Qualitative Comparison of Diversity.** In this figure, we visualize random samples from FIERY [Hu et al., 2021] **(left)** and our model StretchBEV-P **(right)** over short **(top)**, mid **(middle)**, and long **(bottom)** temporal horizons.

0.6469 seconds/example). Although our model processes each time step separately, it does not introduce any drawbacks in terms of speed and its inference speed is almost the same as FIERY.

6.4 Conclusion

We introduced StretchBEV, a stochastic future instance prediction method that improve over the state of the art, especially in challenging cases, with more diverse predictions. We proposed two versions of our method with and without the labels for output modalities explicitly in the posterior while learning the dynamics. Both models improve the state of the art in spatially far regions and over temporally long horizons. Using labels in the posterior significantly improves the results in almost all metrics but introduces a dependency on the availability of labels in the conditioning frames during inference.

Chapter 7

CONCLUSION

This chapter provides a brief summary, explain the limitations of our work and .

7.1 Summary

In this thesis, we propose two novel methods for stochastic video prediction and another method for future instance segmentation in Bird's-eye view representation.

In Chapter 4, we introduced SLAMP, which can predict future frames of a video, given a few initial ones. SLAMP achieves state-of-the-art results on challenging real-world datasets with a moving background while performing on par with the other methods on the generic video prediction datasets. SLAMP can decompose video content into appearance and dynamic components, each of which is modeled separately with different latent variables. Moreover, we showed that motion history enriches the model's capacity to predict the future, leading to better predictions in challenging cases.

In Chapter 5, we have shown that using domain knowledge can improve the performance in stochastic video prediction. We proposed SLAMP-3D, a conditional stochastic video prediction model, by decomposing the scene into static and dynamic in driving videos. We showed that separate modeling of foreground and background motion leads to better future predictions. Also, by conditioning the dynamic latent variable on the static one, SLAMP-3D can predict the residual motion of foreground objects on top of ego-motion, which is predicted by the static part. We demonstrated that our model is among the top-performing methods overall while still being efficient.

In Chapter 6, we have stated that by formulating the problem of future instance segmentation as a learning temporal dynamics through stochastic residual updates to a latent representation in each time step, both the performance and the diversity can be improved. We introduced StretchBEV, which is able to perform better than previous methods with a powerful posterior distribution, especially in challenging cases of spatially far regions and temporally long spans. We showed that our model achieves state-of-the-art results while still being efficient in terms of run-time performance.

7.2 Limitations and Possible Future Directions

In all of the methods we propose, there is no hierarchy. Introducing hierarchy for both latent variables and the predictors might increase the performance. In the literature, hierarchic methods have been used [Castrejon et al., 2019] and the results are improved using the hierarchy. Thus, we believe that introducing hierarchy in the SLAMP's architecture may improve the performance in both generic video prediction and real world datasets.

In SLAMP-3D, using the domain knowledge, we model video as the ego-motion and the residual motion in the scene. Since both parts warp the previous frames to synthesize the target frame, our model can predict the future motion of a car visible in the conditioning frames but it cannot predict a car appearing after that. This limitation can be solved by introducing a separate branch which predicts the target frame in pixel space. However, this improvement is against the motivation of SLAMP-3D. Moreover, we could not use any of the well-known tricks in view synthesis such as multi-scale, forward-backward consistency check, 3D convolutions, and better loss functions [Godard et al., 2019, Guizilini et al., 2020] due to computational reasons. Any improvement there can yield performance gains for stochastic video prediction with structure and motion.

Our StretchBEV has a powerful posterior distribution that leads to remarkable results. Although using labels in the posterior significantly improves the results in almost all metrics, it introduces a dependency on the availability of labels in the conditioning frames during inference. Future work on learning dynamics should focus on closing the gap between the two approaches, for example with scheduled sampling.

StretchBEV method can be interpreted as a Neural-ODE [Chen et al., 2018] because of its residual update dynamics. In our model, we use only one update in between time steps but in future, we plan to explore increasing the number of updates in between time step as done in the previous work [Franceschi et al., 2020]. Moreover, we think that focusing on objects more by following object-centric approaches might lead to better predictions.

One might explore driving policies that can utilize stochastic future predictions. Learned latent states at each time step can be directly fed into a policy learning algorithm, e.g. as states in deep reinforcement learning. Furthermore, these states can be interpreted via supervised decoding into various future modalities that StretchBEV predicts.

BIBLIOGRAPHY

[Akan et al., 2021] Akan, A. K., Erdem, E., Erdem, A., and Guney, F. (2021). Slamp: Stochastic latent appearance and motion prediction. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV).*

[Akan and Güney, 2022] Akan, A. K. and Güney, F. (2022). Stretchbev: Stretching future instance prediction spatially and temporally. *arXiv preprint arXiv:2203.13641.*

[Akan et al., 2022] Akan, A. K., Safadoust, S., Erdem, E., Erdem, A., and Güney, F. (2022). Stochastic video prediction with structure and motion. *arXiv preprint arXiv:2203.10528.*

[Assouel et al., 2022] Assouel, R., Castrejon, L., Courville, A., Ballas, N., and Bengio, Y. (2022). VIM: Variational independent modules for video prediction. In *First Conference on Causal Learning and Reasoning.*

[Babaeizadeh et al., 2018] Babaeizadeh, M., Finn, C., Erhan, D., Campbell, R. H., and Levine, S. (2018). Stochastic variational video prediction. In *Proc. of the International Conf. on Learning Representations (ICLR).*

[Babaeizadeh et al., 2021] Babaeizadeh, M., Saffar, M. T., Nair, S., Levine, S., Finn, C., and Erhan, D. (2021). Fitvid: Overfitting in pixel-level video prediction. *arXiv preprint arXiv:2106.13195.*

[Bian et al., 2019] Bian, J., Li, Z., Wang, N., Zhan, H., Shen, C., Cheng, M.-M., and Reid, I. (2019). Unsupervised scale-consistent depth and ego-motion learning from monocular video. In *Advances in Neural Information Processing Systems (NeurIPS).*

[Caesar et al., 2020] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. (2020). nuscenes: A multimodal dataset for autonomous driving. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Casas et al., 2020] Casas, S., Gulino, C., Liao, R., and Urtasun, R. (2020). Spagnn: Spatially-aware graph neural networks for relational behavior forecasting from sensor data. In *Proc. IEEE International Conf. on Robotics and Automation (ICRA).*

[Casas et al., 2018] Casas, S., Luo, W., and Urtasun, R. (2018). Intentnet: Learning to predict intention from raw sensor data. In *Proc. Conf. on Robot Learning (CoRL).*

[Casser et al., 2019] Casser, V., Pirk, S., Mahjourian, R., and Angelova, A. (2019). Depth prediction without the sensors: Leveraging structure for unsupervised learning from monocular videos. In *Proc. of the Conf. on Artificial Intelligence (AAAI).*

[Castrejon et al., 2019] Castrejon, L., Ballas, N., and Courville, A. (2019). Improved conditional vrnns for video prediction. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV).*

[Chai et al., 2020] Chai, Y., Sapp, B., Bansal, M., and Anguelov, D. (2020). Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In *Proc. Conf. on Robot Learning (CoRL).*

[Chang et al., 2019] Chang, M.-F., Lambert, J. W., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D., and Hays, J. (2019). Argoverse: 3d tracking and forecasting with rich maps. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Chen et al., 2018] Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud,

D. K. (2018). Neural ordinary differential equations. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[Chen et al., 2019] Chen, Y., Schmid, C., and Sminchisescu, C. (2019). Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*.

[Cheng et al., 2020] Cheng, B., Collins, M. D., Zhu, Y., Liu, T., Huang, T. S., Adam, H., and Chen, L.-C. (2020). Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation. In *CVPR*.

[Cordts et al., 2016] Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. (2016). The cityscapes dataset for semantic urban scene understanding. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[Denton and Fergus, 2018] Denton, E. and Fergus, R. (2018). Stochastic video generation with a learned prior. In *Proc. of the International Conf. on Machine learning (ICML)*.

[Djuric et al., 2021] Djuric, N., Cui, H., Su, Z., Wu, S., Wang, H., Chou, F., Martin, L. S., Feng, S., Hu, R., Xu, Y., Dayan, A., Zhang, S., Becker, B. C., Meyer, G. P., Vallespi-Gonzalez, C., and Wellington, C. K. (2021). Multixnet: Multi-class multistage multimodal motion prediction. In *Proc. IEEE Intelligent Vehicles Symposium (IV)*.

[Ebert et al., 2017a] Ebert, F., Finn, C., Lee, A. X., and Levine, S. (2017a). Self-supervised visual planning with temporal skip connections. In *1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings*.

[Ebert et al., 2017b] Ebert, F., Finn, C., Lee, A. X., and Levine, S. (2017b). Self-supervised visual planning with temporal skip connections. In *Proc. Conf. on Robot Learning (CoRL).*

[Eigen et al., 2014] Eigen, D., Puhrsch, C., and Fergus, R. (2014). Depth map prediction from a single image using a multi-scale deep network. In *Advances in Neural Information Processing Systems (NeurIPS).*

[Fan et al., 2019] Fan, H., Zhu, L., and Yang, Y. (2019). Cubic lstms for video prediction. In *Proc. of the Conf. on Artificial Intelligence (AAAI).*

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

[Franceschi et al., 2020] Franceschi, J.-Y., Delasalles, E., Chen, M., Lamprier, S., and Gallinari, P. (2020). Stochastic latent residual video prediction. In *Proc. of the International Conf. on Machine learning (ICML).*

[Gao et al., 2019] Gao, H., Xu, H., Cai, Q.-Z., Wang, R., Yu, F., and Darrell, T. (2019). Disentangling propagation and generation for video prediction. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV).*

[Gao et al., 2020a] Gao, J., Sun, C., Zhao, H., Shen, Y., Anguelov, D., Li, C., and Schmid, C. (2020a). Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11525–11533.

[Gao et al., 2020b] Gao, J., Sun, C., Zhao, H., Shen, Y., Anguelov, D., Li, C., and Schmid, C. (2020b). Vectornet: Encoding HD maps and agent dynamics from vectorized representation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Garg et al., 2016] Garg, R., Bg, V. K., Carneiro, G., and Reid, I. (2016). Unsupervised CNN for single view depth estimation: Geometry to the rescue. In *Proc. of the European Conf. on Computer Vision (ECCV).*

[Geiger et al., 2013] Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research (IJRR).*

[Geiger et al., 2012] Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Godard et al., 2019] Godard, C., Mac Aodha, O., Firman, M., and Brostow, G. J. (2019). Digging into self-supervised monocular depth estimation. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV).*

[Gregor and Besse, 2018] Gregor, K. and Besse, F. (2018). Temporal difference variational auto-encoder. In *Proc. of the International Conf. on Learning Representations (ICLR).*

[Gu et al., 2021a] Gu, J., Sun, C., and Zhao, H. (2021a). Densetnt: End-to-end trajectory prediction from dense goal sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15303–15312.

[Gu et al., 2021b] Gu, J., Sun, C., and Zhao, H. (2021b). Densetnt: End-to-end trajectory prediction from dense goal sets. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV).*

[Guizilini et al., 2020] Guizilini, V., Ambrus, R., Pillai, S., Raventos, A., and Gaidon, A. (2020). 3D packing for self-supervised monocular depth estimation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[He et al., 2016] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Hendy et al., 2020] Hendy, N., Sloan, C., Tian, F., Duan, P., Charchut, N., Xie, Y., Wang, C., and Philbin, J. (2020). FISHING net: Future inference of semantic heatmaps in grids. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) Workshops.*

[Hong et al., 2019] Hong, J., Sapp, B., and Philbin, J. (2019). Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Hu et al., 2020] Hu, A., Cotter, F., Mohan, N., Gurau, C., and Kendall, A. (2020). Probabilistic future prediction for video scene understanding. In *Proc. of the European Conf. on Computer Vision (ECCV).*

[Hu et al., 2021] Hu, A., Murez, Z., Mohan, N., Dudas, S., Hawke, J., Badrinarayanan, V., Cipolla, R., and Kendall, A. (2021). FIERY: Future instance segmentation in bird's-eye view from surround monocular cameras. In *Proceedings of the International Conference on Computer Vision (ICCV).*

[Huang et al., 2020] Huang, X., McGill, S. G., DeCastro, J. A., Fletcher, L., Leonard, J. J., Williams, B. C., and Rosman, G. (2020). Diversitygan: Diversity-aware vehicle motion prediction via latent semantic sampling. *IEEE Robotics and Automation Letters (RA-L).*

[Jaderberg et al., 2015] Jaderberg, M., Simonyan, K., Zisserman, A., and kavukcuoglu, k. (2015). Spatial transformer networks. In *Advances in Neural Information Processing Systems (NeurIPS).*

[Jason et al., 2016] Jason, J. Y., Harley, A. W., and Derpanis, K. G. (2016). Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In *Proc. of the European Conf. on Computer Vision (ECCV) Workshops.*

[Jia et al., 2016] Jia, X., De Brabandere, B., Tuytelaars, T., and Gool, L. V. (2016). Dynamic filter networks. In *Advances in Neural Information Processing Systems (NeurIPS).*

[Johansson, 1973] Johansson, G. (1973). Visual perception of biological motion and a model for its analysis. *Perception & Psychophysics*, 14(2):201–211.

[Karapetyan et al., 2022] Karapetyan, A., Villar-Corrales, A., Boltres, A., and Behnke, S. (2022). Video prediction at multiple scales with hierarchical recurrent networks. *arXiv preprint arXiv:2203.09303.*

[Karl et al., 2017] Karl, M., Soelch, M., Bayer, J., and van der Smagt, P. (2017). Deep variational bayes filters: Unsupervised learning of state space models from raw data. In *Proc. of the International Conf. on Learning Representations (ICLR).*

[Kingma and Welling, 2014] Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In *Proc. of the International Conf. on Learning Representations (ICLR).*

[Krishnan et al., 2017] Krishnan, R. G., Shalit, U., and Sontag, D. (2017). Structured inference networks for nonlinear state space models. In *Proc. of the Conf. on Artificial Intelligence (AAAI).*

[Lee et al., 2018] Lee, A. X., Zhang, R., Ebert, F., Abbeel, P., Finn, C., and Levine, S. (2018). Stochastic adversarial video prediction. *arXiv.org.*

[Li et al., 2018] Li, Y., Fang, C., Yang, J., Wang, Z., Lu, X., and Yang, M.-H. (2018). Flow-grounded spatial-temporal video prediction from still images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 600–615.

[Liang et al., 2020] Liang, M., Yang, B., Hu, R., Chen, Y., Liao, R., Feng, S., and Urtasun, R. (2020). Learning lane graph representations for motion forecasting. In *Proc. of the European Conf. on Computer Vision (ECCV).*

[Liang et al., 2017] Liang, X., Lee, L., Dai, W., and Xing, E. P. (2017). Dual motion gan for future-flow embedded video prediction. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV).*

[Liu et al., 2021] Liu, Y., Zhang, J., Fang, L., Jiang, Q., and Zhou, B. (2021). Multimodal motion prediction with stacked transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7577–7586.

[Liu et al., 2017] Liu, Z., Yeh, R. A., Tang, X., Liu, Y., and Agarwala, A. (2017). Video frame synthesis using deep voxel flow. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV).*

[Lotter et al., 2017] Lotter, W., Kreiman, G., and Cox, D. (2017). Deep predictive coding networks for video prediction and unsupervised learning. In *Proc. of the International Conf. on Learning Representations (ICLR).*

[Lu et al., 2017] Lu, C., Hirsch, M., and Scholkopf, B. (2017). Flexible spatio-temporal networks for video prediction. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Luo et al., 2019] Luo, C., Yang, Z., Wang, P., Wang, Y., Xu, W., Nevatia, R., and Yuille, A. (2019). Every pixel counts++: Joint learning of geometry and motion with 3d holistic understanding. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 42(10).

[Luo et al., 2018] Luo, W., Yang, B., and Urtasun, R. (2018). Fast and furious: Real time end-to-end 3D detection, tracking and motion forecasting with a single convolutional net. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Mahjourian et al., 2018] Mahjourian, R., Wicke, M., and Angelova, A. (2018). Unsupervised learning of depth and ego-motion from monocular video using 3d geometric constraints. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Meister et al., 2018] Meister, S., Hur, J., and Roth, S. (2018). Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *Proceedings of the AAAI Conference on Artificial Intelligence.*

[Minderer et al., 2019] Minderer, M., Sun, C., Villegas, R., Cole, F., Murphy, K. P., and Lee, H. (2019). Unsupervised learning of object structure and dynamics from videos. In *Advances in Neural Information Processing Systems (NeurIPS).*

[Murphy, 2023] Murphy, K. P. (2023). *Probabilistic Machine Learning: Advanced Topics.* MIT Press.

[Philion and Fidler, 2020] Philion, J. and Fidler, S. (2020). Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *Proc. of the European Conf. on Computer Vision (ECCV).*

[Ranjan et al., 2019] Ranjan, A., Jampani, V., Balles, L., Kim, K., Sun, D., Wulff, J., and Black, M. J. (2019). Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Reda et al., 2018] Reda, F. A., Liu, G., Shih, K. J., Kirby, R., Barker, J., Tarjan, D., Tao, A., and Catanzaro, B. (2018). Sdc-net: Video prediction using spatially-displaced convolution. In *Proc. of the European Conf. on Computer Vision (ECCV).*

[Ren et al., 2017] Ren, Z., Yan, J., Ni, B., Liu, B., Yang, X., and Zha, H. (2017). Unsupervised deep learning for optical flow estimation. In *Proc. of the Conf. on Artificial Intelligence (AAAI).*

[Rubanova et al., 2019] Rubanova, Y., Chen, R. T. Q., and Duvenaud, D. K. (2019).

Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[Rybkin et al., 2021] Rybkin, O., Daniilidis, K., and Levine, S. (2021). Simple and effective vae training with calibrated decoders. In *Proc. of the International Conf. on Machine learning (ICML)*.

[Safadoust and Gney, 2021] Safadoust, S. and Gney, F. (2021). Self-supervised monocular scene decomposition and depth estimation. In *International Conference on 3D Vision (3DV)*, pages 627–636.

[Schüldt et al., 2004a] Schüldt, C., Laptev, I., and Caputo, B. (2004a). Recognizing human actions: A local svm approach. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[Schüldt et al., 2004b] Schüldt, C., Laptev, I., and Caputo, B. (2004b). Recognizing human actions: A local svm approach. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*.

[Shi et al., 2015] Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., kin Wong, W., and chun Woo, W. (2015). Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[Simonyan and Zisserman, 2015] Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *Proc. of the International Conf. on Learning Representations (ICLR)*.

[Skafte and Hauberg, 2019] Skafte, N. and Hauberg, S. r. (2019). Explicit disentanglement of appearance and perspective in generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[Sriram et al., 2020] Sriram, N. N., Liu, B., Pittaluga, F., and Chandraker, M.

(2020). SMART: Simultaneous multi-agent recurrent trajectory prediction. In *Proc. of the European Conf. on Computer Vision (ECCV).*

[Szkely and Rizzo, 2017] Szkely, G. J. and Rizzo, M. L. (2017). The energy of data. *Annual Review of Statistics and Its Application*, 4(1):447–479.

[Tang and Salakhutdinov, 2019a] Tang, C. and Salakhutdinov, R. R. (2019a). Multiple futures prediction. *Advances in Neural Information Processing Systems*, 32:15424–15434.

[Tang and Salakhutdinov, 2019b] Tang, Y. C. and Salakhutdinov, R. (2019b). Multiple futures prediction. In *Advances in Neural Information Processing Systems (NeurIPS).*

[Tao et al., 2020] Tao, A., Sapra, K., and Catanzaro, B. (2020). Hierarchical multiscale attention for semantic segmentation. *arXiv preprint arXiv:2005.10821.*

[Unterthiner et al., 2019] Unterthiner, T., van Steenkiste, S., Kurach, K., Marinier, R., Michalski, M., and Gelly, S. (2019). Towards accurate generative models of video: A new metric & challenges. *arXiv.org.*

[Villegas et al., 2019] Villegas, R., Pathak, A., Kannan, H., Erhan, D., Le, Q. V., and Lee, H. (2019). High fidelity video prediction with large stochastic recurrent neural networks. In *Advances in Neural Information Processing Systems (NeurIPS).*

[Villegas et al., 2017] Villegas, R., Yang, J., Zou, Y., Sohn, S., Lin, X., and Lee, H. (2017). Learning to generate long-term future via hierarchical prediction. In *international conference on machine learning*, pages 3560–3569. PMLR.

[Vondrick and Torralba, 2017] Vondrick, C. and Torralba, A. (2017). Generating the future with adversarial transformers. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Walker et al., 2016] Walker, J., Doersch, C., Gupta, A., and Hebert, M. (2016). An uncertain future: Forecasting from static images using variational autoencoders. In *Proc. of the European Conf. on Computer Vision (ECCV).*

[Walker et al., 2015] Walker, J., Gupta, A., and Hebert, M. (2015). Dense optical flow prediction from a static image. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV).*

[Wang et al., 2018] Wang, C., Miguel Buenaposada, J., Zhu, R., and Lucey, S. (2018). Learning depth from monocular videos using direct methods. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Xue et al., 2016] Xue, T., Wu, J., Bouman, K. L., and Freeman, W. T. (2016). Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *Advances In Neural Information Processing Systems.*

[Yin and Shi, 2018] Yin, Z. and Shi, J. (2018). GeoNet: Unsupervised learning of dense depth, optical flow and camera pose. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Zhan et al., 2018] Zhan, H., Garg, R., Saroj Weerasekera, C., Li, K., Agarwal, H., and Reid, I. (2018). Unsupervised learning of monocular depth estimation and visual odometry with deep feature reconstruction. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Zhang et al., 2018] Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Zhao et al., 2020a] Zhao, H., Gao, J., Lan, T., Sun, C., Sapp, B., Varadarajan, B., Shen, Y., Shen, Y., Chai, Y., Schmid, C., et al. (2020a). Tnt: Target-driven trajectory prediction. *arXiv preprint arXiv:2008.08294.*

[Zhao et al., 2020b] Zhao, H., Gao, J., Lan, T., Sun, C., Sapp, B., Varadarajan, B., Shen, Y., Shen, Y., Chai, Y., Schmid, C., Li, C., and Anguelov, D. (2020b). TNT: target-driven trajectory prediction. In *Proc. Conf. on Robot Learning (CoRL).*

[Zhou et al., 2017] Zhou, T., Brown, M., Snavely, N., and Lowe, D. G. (2017). Unsupervised learning of depth and ego-motion from video. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Zhu et al., 2019] Zhu, Y., Sapra, K., Reda, F. A., Shih, K. J., Newsam, S., Tao, A., and Catanzaro, B. (2019). Improving semantic segmentation via video propagation and label relaxation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).*

[Zou et al., 2018] Zou, Y., Luo, Z., and Huang, J.-B. (2018). DF-Net: Unsupervised joint learning of depth and flow using cross-task consistency. In *Proc. of the European Conf. on Computer Vision (ECCV).*