

A Latent Variable Model State Estimation System for Image Sequences

Nils Kornfeld

*Institute of Transportation Systems
German Aerospace Center
Berlin, Germany
nils.kornfeld@dlr.de*

Zachary Feng

*Department of Mathematics and Statistics
McGill University
Montreal, Canada
zachary.feng@mail.mcgill.ca*

Abstract—Self-driving cars need to be able to assess and understand the state of their surroundings. To achieve this goal, it is necessary to construct a model which holds information about the state of the environment based on sensor measurements. In common state estimation systems like Kalman filters, it is necessary to explicitly model state transitions and the observation process. These models have to match the internal dynamics of the observed system as closely as possible to yield reliable estimation results. In this work, we propose a method that can learn an approximation of the internal dynamics of a system, without the need to explicitly model these processes. Our system even works on highly complex data like frames of a video sequence. The approach is based on a latent variable model with a continuous hidden state space. To deal with the fact that the estimated processes are sequential, we use recurrent neural networks. As an example to show the potential of this system, resulting predicted future frames of short video sequences are shown. The proposed system shows a general approach for state estimation without the need for any knowledge about the underlying state transition or observation processes.

Index Terms—artificial intelligence, predictive models, predictive encoding, artificial neural networks

I. INTRODUCTION

State estimation needs to be applied in many areas of science and engineering. It plays an increasingly important role in the development of self-driving vehicles. Any autonomous system must be able to anticipate changes in their surroundings. Humans implicitly and intuitively predict the motion of objects in their environment, based on past trajectories and past experience. The aim of this work is to predict changes in an assumed world state, which leads to changes in observable parameters. For a system to be able to model the evolution of these hidden world states, it needs to incorporate parameters that model the creation of observable parameters from the hidden world state, as well as a way to infer the hidden state from observations.

In this article, we propose a method that combines the encoding of a continuous latent space provided by a variational autoencoder [10], [11] and the temporal persistence of information, carried by Long Short Term Memory cells [7].

The performance of the proposed system is shown in the task of predicting future frames of video streams, recorded by a car-mounted camera.

II. RELATED WORK

Hidden Markov Models (HMMs) can be used to determine the most likely sequence of state transitions. However, the world states are not observed directly, but they are inferred based on measurements of observable variables [17].

HMMs are directed graphs, which represent hidden world states, as well as observable variables as nodes. The graph's edges describe state transition probabilities - and respectively - emission probabilities of the observables, dependent on the current underlying world state [17].

Kalman filters are another common approach for state estimation purposes. The underlying assumption is that hidden state and the emitted observables are Gaussian-distributed. The hidden state is estimated with a joint probability distribution of each observed variable.

Another way to model sequential processes implicitly are Long Short Term Memory Cells (LSTMs). LSTMs are recurrent artificial neural networks with parameters that can be trained to preserve or omit information from previous time steps based on their benefit for the task at hand [7].

Ondruska and Posner propose a method, which can estimate changes over time in an underlying world state, derived from simulated and real-world Lidar measurements. To reach this result, they use a convolutional neural network, which is rolled out in a recurrent manner. To train the model, a sequence of occupancy-grids is fed to the network. The training loss is evaluated on the network's ability to predict future occupancy grids. To minimize the prediction error for future time steps, the model implicitly learns an estimation of the current world state, including positions and motion parameters of occluded objects at every timestep [3], [15]. However, there is no explicit data association, which is suggested by the term tracking.

Sadeghian et al. developed a tracking system, which uses multiple cues to track objects. The system uses LSTMs to estimate and compare appearance, motion and interaction of object candidates [19].

A related approach was proposed by Hsieh et al. who first decompose and disentangle objects in video frames and predict their future behaviour with autoencoders [9].

To predict sequences of text, based on preceding sequences

Alex Graves and Sutskever et al. proposed the usage of LSTMs, consisting of densely connected layers [6], [21]. The result of such a work is summarized impressively in a blog post by Andrej Karpathy¹.

Variational Autoencoders (VAEs) were first developed by Kingma et al. and Rezende et al. independently from each other [10], [11]. The neural networks estimate the lower variational bound for directed graphical models with continuous latent variables. The latent variable model is trained on an auto-encoding (compression/decompression) problem [12].

VAEs are used as a base for the system presented in this article. In future work, the properties of semi-supervised VAEs are supposed to be incorporated. [12]

Bhattacharyya et al. contributed a best of many sample objective, which leads to less blurry output images and still maximize the likelihood of the emission process of the predicted frames [1].

Cox et al. developed a system to predict future video frames with a system, based on VAEs, they call Prednet. Prednet is finetuned on the following frame, whereas this work predicts multiple following states. The model consists of recurrently connected convolutional neural networks, trained as a Generative Adversarial Network (GAN) [13].

Another example of an approach that uses a GAN to enhance the visual appeal of the created images is Wang et al.'s Video-to-Video Synthesis [22].

Using a GAN instead of a VAE interferes with the strong probabilistic properties of the underlying latent variable model. This is why the model in this work was strictly trained with the proposed VAE loss and the re-parameterization trick [10], [11].

An approach which tries to predict multiple likely future frames from just one input image instead of an input video is shown in [23]. Prediction of future states of traffic scenes from single dynamic occupancy grid maps, which feature information about the dynamics of the modeled system is shown in [8].

III. ARCHITECTURE

The concept of the presented state estimation system is based on Hidden Markov Models (HMM). A first-order HMM models a stochastic process, which consists of state transitions. The actual states of the system cannot be observed and are therefore called hidden states $h \in H$. The hidden states emit observables $o \in O$, which can be observed, recorded or measured. These observables can be e.g. object detections, trajectories or images. This article focuses on images as observables o , which are emitted by hidden world states h [17].

While HMMs usually model transitions of discrete states, the approach presented in this article is based on the assumption that the considered state space is continuous.

In contrast to first-order HMMs, we assume that multiple preceding states determine the following states, which are

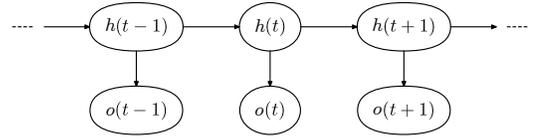


Fig. 1. Three states of a Hidden Markov Model

supposed to be estimated.

To model the emission process, Variational Autoencoders (VAE), developed by Kingma, et. al. [11] and Rezende, et. al. [10] are used. Each VAE is trained to be a generative model, which maximizes the emission probability $p_{\theta}(h)p_{\theta}(\hat{o}|h)$, as well as to approximate the intractable posterior $p_{\theta}(h|o)$. Because of the intractable character of $p_{\theta}(h|o)$, the variational approximation $q_{\phi}(h|o)$ is introduced [11]. The variational parameters ϕ , as well as the generative parameters θ are learned during the supervised training process. The generative subnetwork produces an approximation of the observed input image o , which is denoted by \hat{o} . To learn a continuous distribution in the latent space, Gaussian noise, denoted by ε in eq. (1) is added to the encoded latent variables. To create a compact, symmetric distribution around the point of origin in the latent space, the Kullback-Leibler-Divergence D_{KL} to a standard normal distribution is added as a penalty to the reconstruction loss [11]. The loss function for the VAEs can be seen in eq. (1).

$$L_{VAE} = \lambda D_{KL}(q_{\phi}(h|o)||p_{\theta}(h + \varepsilon)) + \nu \sum(o - \hat{o}) \quad (1)$$

To enable a weighting of the two terms contributing to the loss, the factors λ and ν are introduced, so that the overall loss is a linear combination of the reconstruction error and the KL penalty D_{KL} .

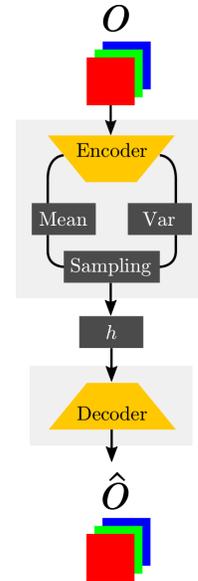


Fig. 2. One VAE cell that approximates the emission of an observable (generative part/decoder) and the variational approximation (encoder)

¹<https://karpathy.github.io/2015/05/21/rnn-effectiveness/>

The state estimation network consists of n full VAEs and m sequential decoders, which predict the m following frames. The LSTM creates the latent variables h_0, \dots, h_{n+m-1} , which can be decoded by the subnetworks, that share their weights with the decoders of the preceding VAE cells. The overall structure of the state estimation network is shown in figure 3.

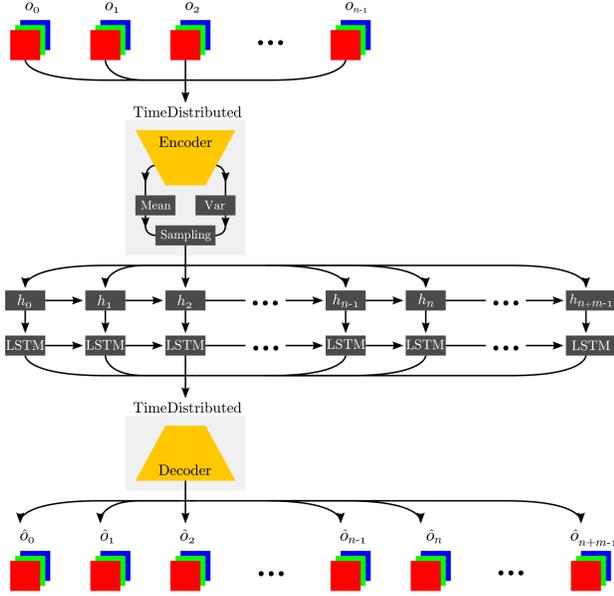


Fig. 3. The overall architecture, which predicts hidden states, as well as emitted observables

These $n + m$ neural networks are placed in a sequence, to capture the capabilities of a hidden Markov model. The latent variables h for all $n + m$ networks are fed into a Long Short Term Memory Network (LSTM) [7]. If the observables consist only of image data, a sparse convolutional LSTM [20] is used, to process the content of the two-dimensional feature maps, extracted by a Convolutional Neural Network (CNN) more efficiently.

The first n full variational autoencoders learn n encoded latent space representations. These representations are compact in the latent space because of our assumption that the data is sequential and the variational autoencoder clusters together similar data. The first n VAEs can compute the encodings in the latent space from the input images. The following m subnetworks must infer the progression that should be taken m time steps into the future from the past evolution of the hidden states. This inference is achieved by using the LSTM. The cells learn, which data from preceding states is relevant to the future predictions. Once the latent space representations for the last m subnetworks have been learned, the decoder subnetwork transforms the encodings into output data, which is supposed to match the according observables.

While in the original article concerning VAEs [11] the intractable posterior $p_\theta(h|o)$ is supposed to be approximated by the variational recognition model $q_\phi(h|o)$, in the recurrent case presented here, the variational approximation needs to be adapted to incorporate the corresponding sequence

$(o_n)_{n \in \mathbb{N}_0}$ of preceding observables. This leads to (2).

$$q_\phi(h = h_j | (o_n)_{n \in \mathbb{N}_0} = o_0, \dots, o_j) \quad (2)$$

The current world state h_j emits the observable \hat{o}_j , such that there is a mapping from each h_j to an \hat{o}_j . \hat{o}_j is supposed to resemble the observed frame o_j as closely as possible. The Loss for each single VAE cell is shown in equation (3).

$$L_{VAE} = \lambda D_{KL}(q_\phi(h_i | o_i) || p_\theta(h_i^*)) + \nu (o - \hat{o})^2 \quad (3)$$

h_i^* is sampled with Gaussian noise around h_i . The loss function of the state estimation network is defined as the mean of the sum of the losses for each of the $m + n$ full and partial variational autoencoders.

$$L_{SEN} = \frac{1}{m+n} \sum_{i=0}^{n+m-1} L_{VAE,i} \quad (4)$$

The training data is split into sequences of $n + m$ succeeding frames. The first n frames are used as the input to the network, while the remaining frames are padded with zeros. To compute the reconstruction loss, the whole set of $n + m$ frames is used as a ground truth.

The optimization algorithm used in this paper is

TABLE I
TRAINING HYPERPARAMETERS

| | |
|-------------------|----------|
| L2-Regularization | 0.001 |
| optimizer | ADADELTA |
| epochs | 800 |
| batch_size | 8 |

ADADELTA [24]. The optimizer has shown promising results in diverse tasks like digit classification and voice datasets [24]. In this particular case this optimizer was chosen, because it has proven experimentally that it provides a stochastic gradient descent, which is more robust against numerical problems than other methods.

To make the model generalize better to unfamiliar data, noise was added to the encoded latent variables and the weights were penalized with an L2-Loss. To speed up the training process, batches of 8 training examples were used, with applied batch normalization. To prevent numerical problems, all input images were normalized. The convolutional layers had ReLU-activation functions, to prevent vanishing gradients. This whole work was implemented in Python, using the Keras framework on top of the Tensorflow backend. The training and inference was run on an Nvidia GTX 1080 Ti.

IV. EXPERIMENTS

To assess the performance of the proposed system it was trained and evaluated on the Caltech pedestrian dataset. The dataset consists of about 10 hours of 640x480 30 fps video sequences taken from a driving vehicle [4], [5]. The annotations provided were not used, because in these experiments the aim was only to predict future frames and not to explicitly track objects visible in these scenes.

The video frames were scaled down to 88x88 pixel images because of hardware limitations. The provided dataset exhibits video sequences, which show scenes from a realistic perspective with respect to the task of assessing the environment of self driving cars. This perspective was chosen to match the context of self driving cars motivated in the introduction of this paper. An example image to illustrate the task the network is trained for is shown in Figure 4. The figure shows a sequence of input data and the number of unknown output frames to be predicted by the network. The reconstruction loss for all 12 frames - 8 reconstructed frames from input data and 4 frames without any input - is included in the calculation of the overall loss for one image sequence.

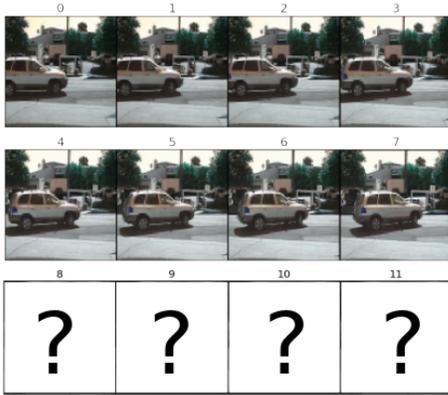


Fig. 4. An example image sequence with unknown following frames to illustrate the underlying problem

The experiments leading to the results of this work were conducted with 10 fps and 30 fps videos. The sequence parameters were chosen, such that the number of input frames is $n = 8$ and the number of predicted frames is $m = 4$ in 10 fps sequences. In 30 fps videos, there are $n = 20$ input frames and $m = 10$ predicted frames, for which no ground truth input to the network is provided. To further increase the number of training examples fed to the network, the input sequences are created with a sliding time window, which uses every frame as the start frame of a sequence, until the end of the original video sequence is reached.

In Figure 5, the short video sequence that was already used as the introductory example is depicted. The first two rows represent the input images, the third row consists of the future frames, predicted by the network. The last row shows the corresponding ground truth images, which can be compared to the predicted frames.

In this sequence, the car moves from the left border of the frame to the right and turns to the left, by rotating counterclockwise around its vertical axis. This motion of the car is visible in the predicted frames as well. The rear wheel moves further to the right in every image, while the side of the car gets shorter, which is caused by the rotation. Additionally to this, the blurred part of the image, which resembles the rear of the car increases in size, because more of it becomes visible to the camera during the turn to the left.

The parts of the image that were not visible to the camera

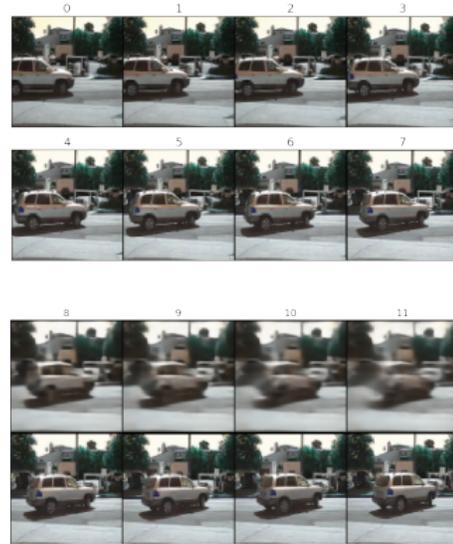


Fig. 5. Video sequence with 4 predicted frames. The car moves from left to right and turns to the left.

during the input frames are especially blurry. In addition to that, every area of the image that is in motion gets blurred further with every additional predicted frame, because of the increased uncertainty. Regularization and the reparameterization trick [10], [11], [13] also cause the frames to gain more blur with each timestep.

The second example, which shows the ability of the proposed system is shown in Figure 6. Pedestrians move to the right and leave the camera’s field of view. The background of this scene stays unchanged. By comparing the predicted frames with the groundtruth, it is obvious that the system estimated the speed of the pedestrian wearing the backpack wrongly. In the groundtruth frames, the pedestrian has already left the scene in the last frame, while in the corresponding estimated image the backpack is still visible.

Because there is no motion in the background of this scene, the background of the estimated images stays sharp, except for the area behind the bright car in the back, to the right of the scene. In the input images provided to the network, this area is occluded by the pedestrians. So there is no information to the system, how long the car in the background is and where it ends. So the pixels in this area get estimated to match the color of the car. Thus the car is shown elongated and blurred in the predicted frames.

This behaviour leads to the conclusion that the network has not learned a general shape of cars to aid the task of predicting frames, but rather estimates consecutive areas in the background to be continued with areas of a common texture and color.

Figure 7 shows an example, in which the vehicle taking the pictures itself is in motion. This leads to a blurring of the background in the predicted frames. The uncertainty of the estimation affects the whole field of view.



Fig. 6. Video sequence with pedestrians walking through the scene. In the predicted frames, the car in the back gets elongated.

The size of the car on the right increases, because the recording vehicle approaches it. The area, which was occluded by that car before is now blurred between the fence in the background and the car itself, so that it gets difficult to determine, which pixels belong to the car and which are part of the background. Even the relative motion of the lane markings is predicted correctly, as they move closer to the camera and further to the left boundary of the image. However, the lane markings are blurred, due to the uncertainty inherent to the system.

Although in all examples presented up to this point the system



Fig. 7. Video sequence taken from moving vehicle.

performed considerably well, we will now present an image sequence, which shows the shortcomings of the current state of this work. In this example, a white pickup truck moves from left to right through the apart from that motionless scene. In

the first estimated frame, the truck can still be recognized, but considering the estimated positions of the wheels, it is too far to the left of the frame. Therefore, the speed was underestimated. In the following three frames, the truck is not visible any longer, but the background of the scene, in the area that was previously occluded by the truck is blurred. The ground truth images of the predicted frames show that the background did not change in comparison to the first eight input frames. But because the system did not keep track of the background state, it estimates this area of the scene as uncertain and thus blurry. The upper third of the image is still sharp in the predicted frames, because there was neither motion nor occlusion in this area.

In the second set of experiments, where 20 input frames were



Fig. 8. 4 frames with blurry predictions.

fed to the system, with the original frame rate of 30 fps the results show similar properties. In Figure 9 the image sequence closely resembles the scene from Figure 5. In these examples with a higher number of predicted frames the increasing uncertainty throughout the predicted sequence, represented by increasing blur is more obvious than in the examples with only four predicted images. Again, the translation and the rotation around the vertical axis can be seen in the estimated frames.

In the second 30 fps example, shown in Figure 10, a truck crosses the scene from left to right, while occluding the background in the process. There is no information about the length of this vehicle available to the network in the first 20 input frames. Thus the area behind the truck stays blurred, even when in the ground truth images this space is not occluded anymore. The darker grey blob, which resembles the truck's front bumper in the predicted frames gets more elongated at each subsequent frame. With the elongation of the bumper, a larger area above it gets blurred as well. The expanding of the blurred region can be explained by the uncertainty of the trucks velocity, which leads to an uncertainty of the trucks position in the scene.

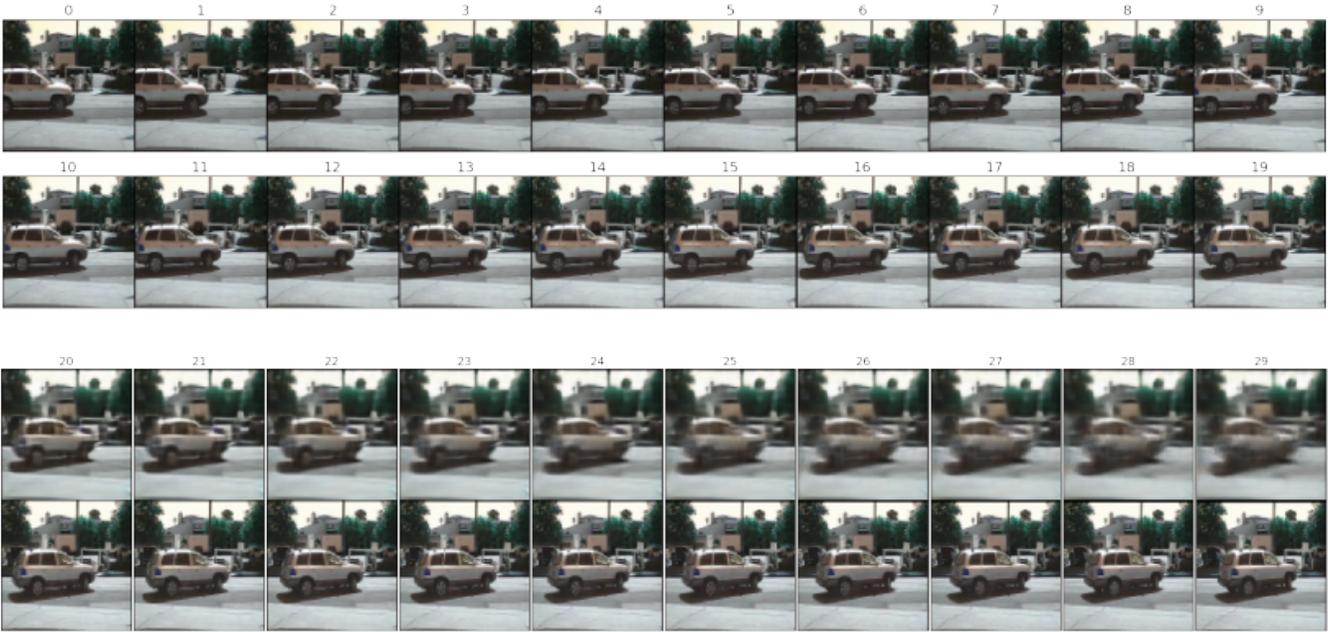


Fig. 9. Video sequence with 10 predicted frames.

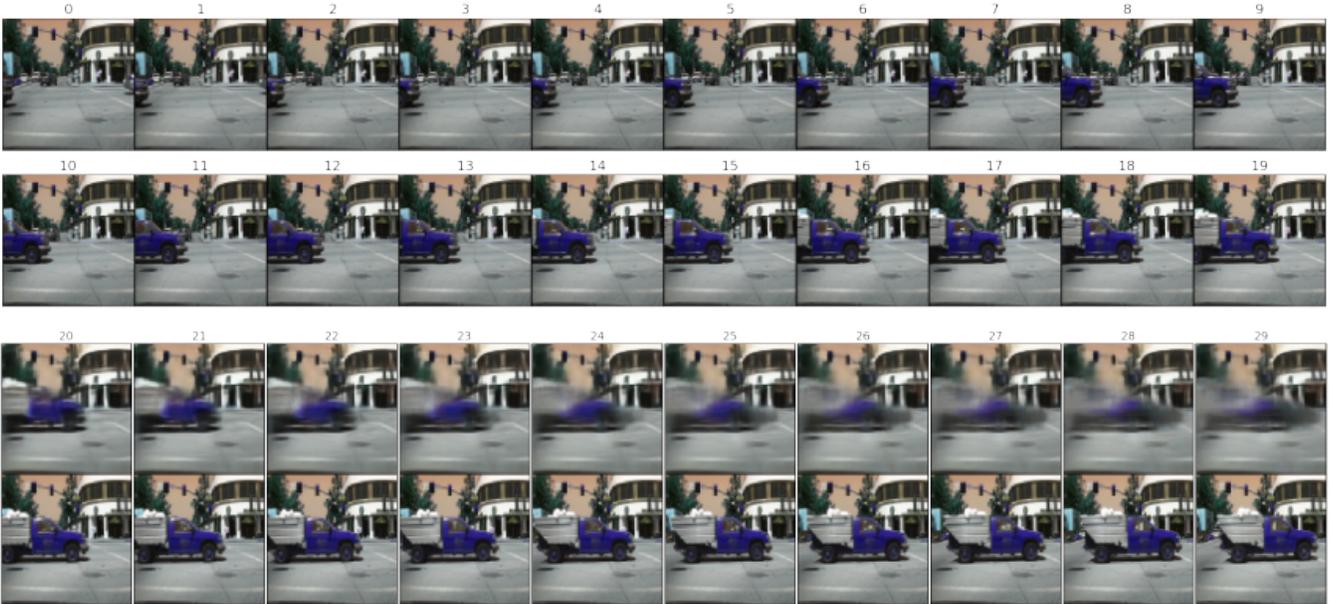


Fig. 10. Video sequence with a truck driving from left to right. The position of the truck gets blurred.

V. SUMMARY OF RESULTS

To summarize the results of the aforementioned experiments, the performance of the proposed system is presented as the per pixel mean squared error of the predicted frames. The 10 fps task was evaluated on 1800 short sequences. The 30 fps task was evaluated on 2168 sequences. The mean squared error between the ground truth and a repetition of a copy of the last provided input image is shown as a baseline to evaluate the results against. The prediction of future frames leads to a blurring of the images, due to the increasing uncertainty

inherent to the problem. Thus, we also consider copies of the last seen frame, which are convolved with a Gaussian filter with an increasing variance.

Figure 11 shows the mean squared error of the experiments with a frame rate of 10 fps. The repetition of the last seen input frame leads to an error, which increases. The increase gets slower from frame to frame. The error of the predicted frames increases more slowly than the error of the copied frames. The progress of the error function through the subsequent frames is approximately linear. The blurred images predict

the future development of the scene much better than just the copied frames. The respective error curve increases in an almost linear way, similar to the error of the predicted frames. The errorbars, representing the standard deviation of the mean squared error are only plotted for the predicted frames, for better visibility. The standard deviation of the other shown methods is higher than that of the proposed method in every frame. In the experiments with 10 fps videos the proposed system's error is lower than the presented baselines at every considered frame.

In the 30 fps experiments - shown in Figure 12 - the increase

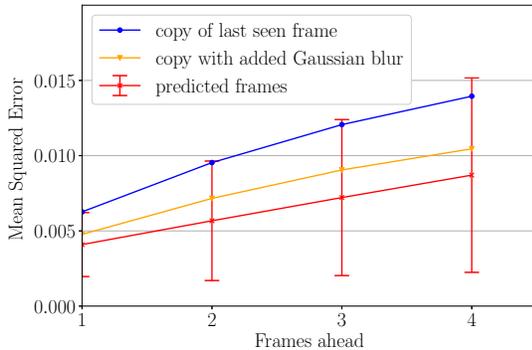


Fig. 11. Mean squared error of predicted frames - 10 fps

of the error of the copied frames slows down over time, until the error increases almost linearly. In this experiment, the course of the error curve of the blurred images is very similar to the mean squared error or the copied frames. Again, the frames convolved with the Gaussian filter show a lower error than frames that are copied without any additional processing. The error of the predicted frames increases slower and approximately linearly, just as in the previously shown results. Because the difference between subsequent frames increases slower than in the 10 fps case, the error of the baseline starts at a lower value. Because of the increased frame rate the baseline's error is even lower than the one of the first predicted frame. From the second timestep on the proposed system's error is lower than the considered baselines.

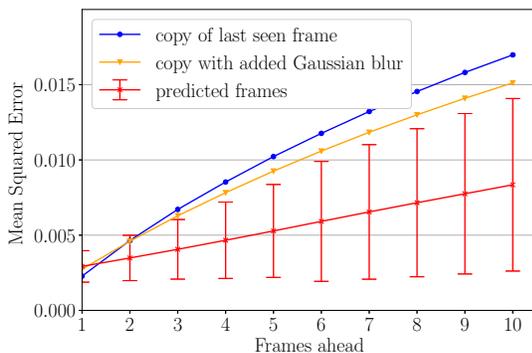


Fig. 12. Mean squared error of predicted frames - 30 fps

VI. DISCUSSION AND FUTURE WORK

The ability to predict future video frames only from the information encoded in the latent variables of the proposed system shows that these compressed representations incorporate information about the state of the scene, which generates the subsequent frames. The presented examples show that the motion of objects is learned in terms of translation, scaling and rotation. The inherent uncertainties lead to blurring of the generated images. While there seems to be a differentiation into moving objects and unchanged background, the appearance of the background that gets occluded is not learned as is shown in the experiments.

Because the proposed system is strictly based on a variational autoencoder without any additional modifications to enhance the visual quality of the output images like generative adversarial networks in Prednet [13] or shortcuts in fully convolutional networks like U-Net [18], the probabilistic model of the variational autoencoder is preserved. The proposed system is a latent variable model and thus the distribution of predicted frames is generated by the continuous normally distributed latent variables. It estimates both - the hidden states and the expression of observable parameters.

In future work, the approximated latent variables are to be examined further to infer state representations, which can be used to aid in object tracking. To determine the dimensions represented by the axes of the latent variables, the approach of semi-supervised variational autoencoders shall be used [12]. In contrast to other related systems the focus of future work will not be on enhancing the visual quality of predicted frames, but on leveraging the learned state representation for object tracking purposes.

ACKNOWLEDGMENT

This work was supported by the DAAD Rise Germany internship, Mitacs Globalink Research Awards, and Antje Graupe Pryor International Awards.

REFERENCES

- [1] Apratim Bhattacharyya, Bernt Schiele, and Mario Fritz. Accurate and diverse sampling of sequences based on a "best of many" sample objective. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 8485–8493, 2018.
- [2] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2980–2988, 2015.
- [3] Julie Dequaire, Peter Ondruška, Dushyant Rao, Dominic Wang, and Ingmar Posner. Deep tracking in the wild: End-to-end tracking using recurrent neural networks. *The International Journal of Robotics Research*, 2017.
- [4] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *CVPR*, June 2009.
- [5] Piotr Dollár, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: An evaluation of the state of the art. *PAMI*, 34, 2012.
- [6] Alex Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.
- [7] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

- [8] Stefan Hoermann, Martin Bach, and Klaus Dietmayer. Dynamic occupancy grid prediction for urban autonomous driving: A deep learning approach with fully automatic labeling. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pages 2056–2063, 2018.
- [9] Jun-Ting Hsieh, Bingbin Liu, De-An Huang, Li F Fei-Fei, and Juan Carlos Nieves. Learning to decompose and disentangle representations for video prediction. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 517–526. Curran Associates, Inc., 2018.
- [10] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *31st International Conference on Machine Learning, ICML 2014*, 4, 12 2013.
- [11] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2013.
- [12] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3581–3589. Curran Associates, Inc., 2014.
- [13] William Lotter, Gabriel Kreiman, and David D. Cox. Deep predictive coding networks for video prediction and unsupervised learning. *CoRR*, abs/1605.08104, 2016.
- [14] Peter Ondruska, Julie Dequaire, Dominic Zeng Wang, and Ingmar Posner. End-to-End Tracking and Semantic Segmentation Using Recurrent Neural Networks. In *Robotics: Science and Systems, Workshop on Limits and Potentials of Deep Learning in Robotics*, 2016.
- [15] Peter Ondruska and Ingmar Posner. Deep tracking: Seeing beyond seeing using recurrent neural networks. *CoRR*, abs/1602.00991, 2016.
- [16] M. Panzner and P. Cimiano, “Comparing Hidden Markov Models and Long Short Term Memory Neural Networks for Learning Action Representations”, Machine Learning, Optimization, and Big Data : Second International Workshop, MOD 2016, Volterra, Italy, August 26-29, 2016. Revised Selected Papers, P.M. Pardalos, et al., eds., Lecture Notes in Computer Science, vol. 10122, Cham: Springer International Publishing, 2016, pp.94-105.
- [17] L. R. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” in *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257-286, Feb 1989.
- [18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.
- [19] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *The IEEE International Conference on Computer Vision (ICCV)*, 10 2017.
- [20] Xingjian SHI, Zhouong Chen, Hao Wang, Dit-Yan Yeung, Wai-kin Wong, and Wang-chun WOO. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 802–810. Curran Associates, Inc., 2015.
- [21] Ilya Sutskever, James Martens, and Geoffrey E. Hinton. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024, 01 2011.
- [22] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. Video-to-video synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.
- [23] Tianfan Xue, Jiajun Wu, Katherine L Bouman, and William T Freeman. Visual dynamics: Stochastic future generation via layered cross convolutional networks. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2018.
- [24] Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.