Towards zero-latency video transmission through frame extrapolation

Melan Vijayaratnam, Marco Cagnazzo, Giuseppe Valenzise, Anthony Trioux, Michel Kieffer

TOWARDS ZERO-LATENCY VIDEO TRANSMISSION THROUGH FRAME EXTRAPOLATION

Melan Vijayaratnam[1] *Marco Cagnazzo*[1,2] *Giuseppe Valenzise*[3] *Anthony Trioux*[4] *Michel Kieffer*[3]

[1]LTCI, Télécom ParisTech, Institut Polytechnique de Paris, France
[2]University of Padua, Department of Information Engineering, Italy
[3]Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire des Signaux et Systèmes, France
[4]UMR 8520 - IEMN, DOAE, Univ. Polytechnique Hauts-de-France, CNRS, France

ABSTRACT

In the past few years, several efforts have been devoted to reduce individual sources of latency in video delivery, including acquisition, coding and network transmission. The goal is to improve the quality of experience in applications requiring real-time interaction. Nevertheless, these efforts are fundamentally constrained by technological and physical limits. In this paper, we investigate a radically different approach that can arbitrarily reduce the overall latency by means of video extrapolation. We propose two latency compensation schemes where video extrapolation is performed either at the encoder or at the decoder side. Since a loss of fidelity is the price to pay for compensating latency arbitrarily, we study the latency-fidelity compromise using three recent video prediction schemes. Our preliminary results show that by accepting a quality loss, we can compensate a typical latency of 100 ms with a loss of 8 dB in PSNR with the best extrapolator. This approach is promising but also suggests that further work should be done in video prediction to pursue zero-latency video transmission.

Index Terms— Extrapolation, low-latency video delivery, video deep learning

1. INTRODUCTION

Ultra low-latency video delivery is an essential feature in many applications involving interactions among humans (*e.g.*, video conferencing, virtual and augmented reality) or between humans and machines (*e.g.*, teleoperation of unmanned vehicles or robots, remote surgery, *etc*.). In these scenarios, the Glass-to-Glass (G2G) latency, intended as the delay between the acquisition of a video frame by an agent and its display by a second (remote) agent [1], plays a major role in the overall quality of experience perceived by users [2].

G2G latency consists of acquisition, coding, buffering at transmitter, transmission, buffering at receiver, decoding, and display delays. In the past decades, significant efforts have been devoted to optimizing each of these individual sources of delay. Nevertheless, the minimum achievable latency is still constrained by technological and physical limits (the most evident being the speed of light), which represent a hard lower bound beyond which latency cannot be further reduced. With these clear limitations in mind, in this paper, we investigate how we can further *reduce the G2G latency to be arbitrary low*. Given the uncompressible physical delays in video delivery, a possible way to achieve this goal is to *predict* video frames that have not been received yet using those already available at the receiver.

The use of prediction/extrapolation to compensate latency is not new. It has been already employed in a number of applications, including virtual reality [3], tactile interfaces [4], cloud gaming [5], *etc*. Nevertheless, to the best of our knowledge, this simple idea of using extrapolation has not been explored and analyzed yet for the case of latency compensation in video transmission. In this paper, we consider for the first time the application of extrapolation to ultra-low or even *zero* latency video communication. To this end, we analyze two possible schemes to integrate extrapolation together with a conventional video codec, *i.e.*, extrapolating from the original frames at the encoder side before coding and transmission, or at the decoder side based on the available decoded (quantized) frames. In both cases, if the prediction horizon is large enough, all sources of delay (including the time needed for extrapolation) can be compensated and a predicted frame can be displayed at the receiver *while* it is acquired at the transmitter.

Since the extrapolated frames are in general different from the true ones, latency compensation produces a loss of fidelity. In this paper, using different recent learning-based extrapolation algorithms, we characterize the trade-off between latency and fidelity. Our aim is to study the feasibility of latency compensation based on frame extrapolation. Our preliminary results on a driving car dataset indicate that the effectiveness of this approach depends significantly on the content and that there is still large space for improvement in video extrapolation techniques.

2. RELATED WORK

Extrapolation as a means to compensate latency has been applied in many fields. In virtual reality headsets, predicting future head movements provides a low-cost solution to reduce latency, which is one of the main causes of motion sickness [3]. To provide a seamless experience with respect to touchscreen devices, the authors of [4] extrapolate the finger's movement to compensate software latency. In cloud gaming, it has been proposed to render speculative frames of future possible outcomes, delivering them to the client ahead of time [5]. Our work relies on similar ideas, but applies them to a generic video coding architecture.

From a video coding point of view, all recent video coding standards from ITU-T ISO/IEC (*e.g.*, HEVC, SHVC, or VVC) have low-latency profiles that optimize the predictive coding structure. The MPEG-I VVC [6] has added some features enabling low latency at a signaling level such as the Gradual Decoding Refresh (GDR). GDR redistributes the bitrate of the intra-frames over many frames, smoothing the bandwidth consumption, and thus the transmission latency, over time. To reduce the transmission delay, the 5G ultra-reliable low latency communications (URLLC) mode targets radio access delays of less than one millisecond [7]. Despite these de-

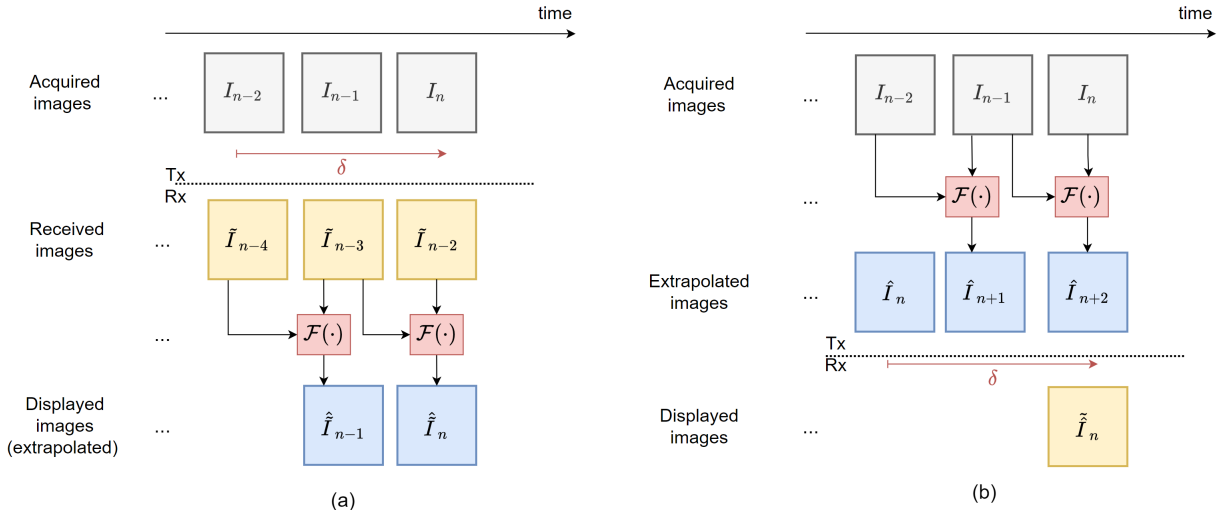

Fig. 1. An example of latency compensation with (a) extrapolation at the decoder and (b) extrapolation at the encoder. $\tilde{I}$ and $\hat{I}$ indicate decoded (quantized) and predicted (extrapolated) frames, respectively. $\mathcal{F}$ is the extrapolation function.

velopments, as pointed out in [8], current video systems are still subject to G2G delays from 50 to 400 ms. In order to satisfy the requirements of latency-sensitive applications, Bachhuber *et al.* [1] analyzed all delay components in a video communication system. Frame skipping and frame preemption mechanisms are proposed to further reduce the G2G latency. Neglecting the buffering and network transmission delays, direct communication with a latency of about 20 ms is achieved.

These works optimize specific sources of delay in the processing pipeline, which intrinsically limits the maximum latency reduction that can be achieved. Instead, in this paper we consider the global G2G latency, with the goal to arbitrarily reduce it.

3. LATENCY COMPENSATION VIA EXTRAPOLATION

We propose to compensate latency using frame extrapolation. This extrapolation can be applied at either the encoder or at the decoder side, as illustrated in Fig. 1. For the sake of simplicity, we assume here a low-delay encoder configuration, where the encoding order of frames is identical to the display order. Nevertheless, the proposed scheme generalizes to any GOP structure. In the following, we describe the two compensation schemes in detail.

3.1. Extrapolation at the decoder side

In this scheme, no modification of the transmitter is required with respect to a standard transmission pipeline: the acquired frames are compressed and transmitted to the receiver, where they arrive with some latency. Figure 1(a) illustrates the extrapolation at the decoder side with an example. In this figure, I_n represents the n-th frame of the video. We assume in this example that the G2G latency δ is equal to two frames (corresponding to a time of $2/f$ seconds, where f is the frame rate). Therefore the decoder receives $\tilde{I}_{n-2}$ (the compressed version of I_{n-2}) when the encoder is already acquiring I_n. In order to compensate this latency, the decoder runs a frame extrapolation algorithm $\mathcal{F}$ which takes as input a given number k of available decoded frames, and produces a prediction $\hat{I}_n$ of frame n as:

$$\hat{\tilde{I}}_n = \mathcal{F}(\{\tilde{I}_{n-h}, \tilde{I}_{n-h-1}, \ldots, \tilde{I}_{n-h-k+1}\}; h). \quad (1)$$

The k input frames are also referred to as *context* frames. For example, in Figure 1, $k = 2$ context frames are considered. The parameter h is the *temporal horizon* of the extrapolation mechanism. It determines the amount of latency we want to compensate for. The larger is h, the more difficult it is to get a reliable prediction, as we show in the experimental results. The parameter h determines the trade-off between latency compensation and quality degradation. In the example we have $h = 2$, meaning that we want to fully compensate for the latency δ. While the context frames k are dependent of the extrapolators used, the temporal horizon h is chosen depending on the application, there is no relationships between these two variables. With these settings, the extrapolation algorithm produces:

$$\hat{\tilde{I}}_n = \mathcal{F}(\{\tilde{I}_{n-2}, \tilde{I}_{n-3}\}; 2). \quad (2)$$

Therefore, the estimation $\hat{\tilde{I}}_n$ of frame I_n, extrapolated from compressed frames, can be displayed at the decoder *while* frame I_n is acquired by the encoder.

3.2. Extrapolation at the encoder side

The extrapolation at the encoder sides is illustrated in Figure 1(b), following up the example used in the previous section. The extrapolation method $\mathcal{F}(\cdot; h)$, is used this time with the acquired frames: I_n, I_{n-1}, …, I_{n-k+1} used as context frames. Likewise, the prediction depends on the temporal horizon h, *i.e.*, we compute

$$\hat{I}(n+h) = \mathcal{F}(\{I_n, I_{n-1}, \ldots, I_{n-k}\}; h). \quad (3)$$

As for the previous example, we have k=2 and h=2 here. Then, the extrapolated frames are compressed, transmitted, decoded, and displayed. In this case, only the transmitter has to be adapted and a plain, standard receiver is employed. The extrapolation performed at the encoder provides a preemptive transmission latency compensation. In the considered example, the compressed, extrapolated frame $\hat{\tilde{I}}_n$ is displayed *while* the frame I_n is acquired by the transmitter.

	context k	horizon h
Copylast	1	1
FlowNet2 + warping	2	1
MCNet	5	5
SDCNet	2	1

Table 1. Frame extrapolation methods and their number of context and default prediction horizon.

With either the decoder-side or encoder-side extrapolation schemes, it is possible to compensate the same amount of latency. This potentially includes also the time to perform extrapolation, which might be not negligible, depending on the adopted method. In a multicast scenario, the extrapolation at the encoder has an advantage compared to the extrapolation at the decoder, as extrapolation is run only once, and more complex techniques with possibly more powerful hardware may be employed. Nevertheless, this also means that the latency compensation is the same for all the receivers. This may be suboptimal when the G2G latency differs among receivers. In such case, the extrapolation at the decoder gives more flexibility. Furthermore, one could also consider a variant of the proposed approaches where extrapolation is performed both at the encoder and the decoder side. In this case, extrapolation at encoder is used to compensate for some minimum, common latency; then, each receiver can further compensate latency according to its needs. Finally, latency compensation comes with a cost in terms of fidelity loss. In the following experiments, we study this latency-distortion trade-off using some off-the-shelf video extrapolators.

4. EXPERIMENTS AND DISCUSSION

This section evaluates the feasibility of frame extrapolation as an effective latency compensation tool. Specifically, we characterize the loss in quality for different amounts of compensated latency, using different off-the-shelf extrapolators to implement $\mathcal{F}$. All the experiments are conducted using the HEVC HM codec implementation (16.24) [9] using the encoder_intra_main.cfg configuration.

4.1. Choice of the extrapolators

As discussed in [10], video prediction methods can be motion-based, pixel-based, or fusion-based. Motion-based methods seek to compute the motion of image pixel intensities which corresponds to the motion of objects in a scene. Deep learning methods in this category compute optical flow with neural networks. Combined with a warping operation, next frames are inferred from previous images. Pixel-based methods seek to generate each pixel from scratch, *i.e.*, they do not use explicit motion representations such as an optical flow or motion vectors. Fusion-based methods combine both motion and pixel-based methods. In this work, we select one method from each category. **FlowNet-2** [11] is a motion-based technique combined with a warping operation, which can effectively predict the next frame. **MCNet** [12] is a pixel-based technique built upon the Encoder-Decoder Convolutional Neural Network and Convolutional LSTM for pixel-level prediction, which independently capture the spatial layout of an image and the corresponding temporal dynamics. Finally, **SDCNet** [13] is a fusion-based approach that models moving appearances with both convolutional kernels and vectors as optical flow. Some characteristics of the considered extrapolation methods are reported in Table 1. Each extrapolator is designed to work with a certain number k of context frames, and a certain prediction horizon h. The latter might be different (smaller) than the desired latency to compensate. In this case, we simply apply the extrapolation recursively, *i.e.*, we progressively add extrapolated frames to the context.

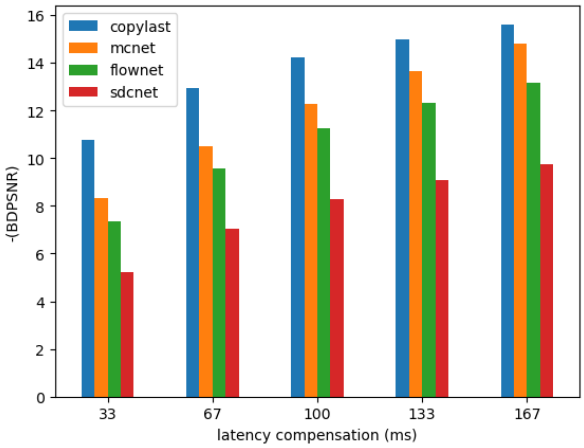

Fig. 2. Evolution of the latency compensation-distortion trade-off in terms of PSNR.

We also include a simple frame-copy extrapolation, dubbed **Copylast**. This method just copies the last available frame at the encoder/decoder, and is reported as a very low complexity baseline.

4.2. Datasets

The considered learning-based extrapolation methods require data to be trained on. In our experiments, we have trained the unsupervised methods SDCNet and MCNet on the Caltech Pedestrian dataset [14], which contains 128419 images from 65 different video sequences at 30 fps with the resolution 640×480. All the frames are center cropped into 640×448. This dataset provides us with both quantity and variety and reflects a real-case scenario. For the supervised method FlowNet2 [11], we need ground-truth optical flow, which is not available on the previous dataset. Therefore, we use the weights pre-trained on the MPI Sintel dataset [15]. Sintel is an action movie and as such contains many fast movements that are difficult for traditional non learning-based methods. The test experiments (along with rate-distortion analysis shown in next section) are done on the DriveSeg [16] Manual scene. Similar to the training set, the dataset has been captured from a moving vehicle during continuous daylight driving through a crowded city street. It comprises 5000 frames taken at 30 fps. We center crop the frames of the sequence into 640×448.

4.3. Study of the distortion-fidelity trade-off

To evaluate the scheme where extrapolation is performed at the decoder, the DriveSeg test set at 30 fps is compressed using HEVC with different quantization parameters $QP \in \{22, 27, 32, 37\}$ for all the selected methods. Rate-Distortion (RD) points are computed considering an extrapolation horizon $h \in \{1, \ldots, 5\}$ to compensate a latency $\delta \in \{33\text{ ms}, 67\text{ ms}, 100\text{ ms}, 133\text{ ms}, 167\text{ ms}\}$. For the case of extrapolation at the encoder, the same test set is considered, with the same an horizon $h \in \{1, \ldots, 5\}$ and the same QPs for compressing extrapolated frames with HEVC.

latency compensation (ms)	−BDPSNR ↓ 33	100	167	−BDSSIM ↓ 33	100	167	−BDVMAF ↓ 33	100	167
Copylast	10.78 (+0.02)	14.21 (+0.02)	15.61 (+0.01)	0.19 (+0.00)	0.35 (+0.00)	0.43 (+0.00)	46.52 (-0.14)	65.44 (-0.09)	71.65 (-0.07)
MCNet	8.34 (-0.15)	12.26 (-0.28)	14.79 (-0.29)	0.06 (+0.00)	0.17 (-0.01)	0.28 (-0.02)	30.28 (-0.94)	50.86 (-2.49)	64.54 (-3.08)
FlowNet-2 + warp	7.35 (+0.03)	11.23 (-0.03)	13.17 (-0.04)	0.05 (+0.00)	0.15 (+0.00)	0.24 (+0.00)	25.42 (-0.20)	51.15 (-0.03)	65.29 (+0.23)
SDCNet	**5.20 (+0.14)**	**8.28 (+0.10)**	**9.74 (+0.09)**	**0.05 (-0.02)**	**0.15 (-0.07)**	**0.24 (-0.13)**	**17.27 (-0.13)**	**36.00 (-0.13)**	**46.63 (-0.09)**

Table 2. Quantitative results on DriveSeg scene: results for the extrapolation at the decoder side and gain/loss (in parenthesis) obtained with the extrapolation scheme at encoder side.

Fig. 3. Qualitative results for a latency compensation of 100 ms. The extrapolation at decoder side scheme is used. The HEVC codec uses a quantization parameter $QP = 32$. Images taken from the Kitti [17] dataset.

The distortion between displayed and original frames is evaluated using three objective metrics: PSNR in the YCbCr color space [18], SSIM [19], and VMAF [20]. This yields a rate-distortion curve for each possible extrapolation horizon (latency compensation) h. We compare all curves to the case $h = 0$, *i.e.*, without any extrapolation. The average quality losses over different rates, expressed by the negative Bjøntegaard delta metric (BDPSNR, BDSSIM and BD-VMAF), are reported in Table 2, where the first numbers are the BD metric values for the extrapolation at the decoder side, and the numbers in parentheses are the differences with respect to this scheme when using extrapolation at the encoder. We can observe that extrapolation at the encoder/decoder produces essentially the same quality losses for a given amount of compensated latency.

Extrapolation at encoder is performed on original frames, and may therefore be more efficient. Nevertheless, for the considered values of QP, extrapolation at the decoder is still efficient, even using decoded context frames. In both cases, the fidelity loss introduced by the extrapolation dominates that induced by compression.

Figure 2 illustrates visually the latency-distortion trade-off in terms of BDPSNR. As expected, the larger the amount of latency we compensate, the higher the quality loss we incur. For typical G2G latency values of 100 ms observed in remote control applications [21, 22], on the test dataset used in this work, the PSNR loss ranges between 12.3 and 8.3 dB with the considered extrapolation techniques.

4.4. Discussion

We have verified quantitatively the expected trade-off between fidelity and latency compensation for a specific dataset. This aspect is key for the feasibility of latency compensation, as distortion is the unavoidable consequence of arbitrarily reducing latency. The effectiveness of latency compensation depends significantly on the interpolation operator $\mathcal{F}$. The proposed scheme employing recent frame prediction techniques entails PSNR drops in excess of 8 dB for 100 ms reduction. This motivates the study of better future frame prediction algorithms in the future.

The fidelity metrics alone might not be sufficient to explain the feasibility of latency compensation through extrapolation. Figure 3 shows the results of compensating 100 ms on a frame of the Kitti dataset [17], which similar to the training set. The interpolation of SDCNet has clear visual deformations compared to the original frame. Nevertheless, we can observe that the position of the bike is approximately aligned with the original frame. On the other hand, Copylast (which only uses the last available frame and corresponds to not compensating any latency) produces a lag between the actual position of the bike and the displayed one. We can imagine that, depending on the application (*e.g.*, teleoperation), the SDCNet prediction brings valuable information (the bike position), even though such quality metrics as PSNR are not able to catch this aspect. This example highlights the essential objective of latency reduction, *i.e.*, to gain more knowledge about the future representation of the dynamic scene by anticipating next images. This question is not only essential at the human perceptual level, but also at the machine level. We can imagine this future understanding as key components for the machine to anticipate behaviors.

5. CONCLUSION

This paper introduces a tool that allows us to compensate latency in a video transmission scheme at the price of additional complexity but also of degradation of the frame fidelity. Depending on the application, many configurations are possible extrapolation at the encoder, extrapolation at the decoder, or both at the encoder and decoder. The degradation is essentially caused by the extrapolation approaches. The goal of this paper is not to propose a better extrapolation approach to reduce this loss, but rather demonstrating the applicability of the latency compensation relying on such tool. Future works may concern the improvement of existing extrapolating methods for such task.

6. ACKNOWLEDGMENTS

This work was funded by the ANR AAPG2020 national fund (ANR-20-CE25-0014).

7. REFERENCES

[1] C. Bachhuber, E. Steinbach, M. Freundl, and M. Reisslein, "On the Minimization of Glass-to-Glass and Glass-to-Algorithm Delay in Video Communication," *IEEE Trans. on Multimedia*, vol. 20, no. 1, pp. 238–252, Jan. 2018.

[2] K. Brunnström, E. Dima, T. Qureshi, M. Johanson, M. Andersson, and M. Sjöström, "Latency impact on quality of experience in a virtual reality simulator for remote control of machines," *Signal Processing: Image Communication*, vol. 89, pp. 116005, 2020.

[3] A. Garcia-Agundez, A. Westmeier, P. Caserman, R. Konrad, and S. Göbel, "An evaluation of extrapolation and filtering techniques in head tracking for virtual environments to reduce cybersickness," in *Joint International Conference on Serious Games*. Springer, 2017, pp. 203–211.

[4] N. Henze, M. Funk, and A. S. Shirazi, "Software-reduced touchscreen latency," in *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services*, 2016, pp. 434–441.

[5] K. Lee, D. Chu, E. Cuervo, J. Kopf, Y. Degtyarev, S. Grizan, A. Wolman, and J. Flinn, "Outatime: Using speculation to enable low-latency continuous interaction for mobile cloud gaming," in *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, 2015, pp. 151–165.

[6] B. Bross, K. Andersson, M. Bläser, et al., "General video coding technology in responses to the joint call for proposals on video compression with capability beyond HEVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 5, pp. 1226–1240, 2019.

[7] R. Ali, Y. B. Zikria, A. K. Bashir, et al., "URLLC for 5G and Beyond: Requirements, Enabling Incumbent Technologies and Network Intelligence," *IEEE Access*, vol. 9, pp. 67064–67095, 2021.

[8] S. K. Sharma, I. Woungang, A. Anpalagan, and S. Chatzinotas, "Toward Tactile Internet in Beyond 5G Era: Recent Advances, Current Issues, and Future Directions," *IEEE Access*, vol. 8, pp. 56948–56991, 2020.

[9] C. Rosewarne, K. Sharman, R. Sjöberg, and G. J. Sullivan, "High Efficiency Video Coding (HEVC) Test Model 16 (HM 16) Encoder Description Update 13 | MPEG," in *38th JCT-VC Meeting*, Brussels, Jan. 2020.

[10] H. Gao, H. Xu, Q.-Z. Cai, R. Wang, F. Yu, and T. Darrell, "Disentangling propagation and generation for video prediction," in *IEEE International Conf. on Computer Vision (ICCV)*, 2019, pp. 9006–9015.

[11] E. Ilg, N. Mayer, T. Saikia, et al., "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2462–2470.

[12] R. Villegas, J. Yang, S. Hong, et al., "Decomposing motion and content for natural video sequence prediction," *arXiv preprint arXiv:1706.08033*, 2017.

[13] F. A. Reda, G. Liu, K. J. Shih, et al., "SDCNet: Video prediction using spatially-displaced convolution," 2021.

[14] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," in *2009 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 304–311.

[15] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A Naturalistic Open Source Movie for Optical Flow Evaluation," in *Computer Vision – ECCV 2012*, Berlin, Heidelberg, 2012, Lecture Notes in Computer Science, pp. 611–625, Springer.

[16] L. Ding, J. Terwilliger, R. Sherony, et al., "MIT driveseg (manual) dataset for dynamic driving scene segmentation," Tech. Rep., Technical report, Massachusetts Institute of Technology, 2020.

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

[18] G. Sullivan and K. Minoo, "Objective quality metric and alternative methods for measuring coding efficiency," in *document JCTVC-H0012, ITU-T/ISO/IEC Joint Collaborative Team on Video Coding (JCT-VC), 8th Meeting: San Jose, CA, USA*, 2012, pp. 1–10.

[19] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[20] C. G. Bampis, Z. Li, and A. C. Bovik, "Spatiotemporal Feature Integration and Model Fusion for Full Reference Video Quality Assessment," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 8, pp. 2256–2270, Aug. 2019.

[21] O. E. Marai and T. Taleb, "Smooth and Low Latency Video Streaming for Autonomous Cars During Handover," *IEEE Network*, vol. 34, no. 6, pp. 302–309, Nov. 2020.

[22] P. Sharma, D. Awasare, B. Jaiswal, et al., "On the Latency in Vehicular Control using Video Streaming over Wi-Fi," in *IEEE National Conf. on Communications (NCC)*, Feb. 2020, pp. 1–6.