Distributed Radiance Fields for Edge Video Compression and Metaverse Integration in Autonomous Driving

Eugen Šlapak*, Matúš Dopiriak*, Mohammad Abdullah Al Faruque†, Juraj Gazda*, Marco Levorato†
*Department of Computers and Informatics, Technical University of Košice, Slovakia
Email: {eugen.slapak, matus.dopiriak, juraj.gazda}@tuke.sk
†Department of Electrical Engineering and Computer Science, University of California, Irvine, United States
Email: {alfaruqu, levorato}@uci.edu

2024 IEEE International Conference on Smart Computing (SMARTCOMP) | 979-8-3503-4994-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/SMARTCOMP61445.2024.00031

***Abstract*—The metaverse is a virtual space that combines physical and digital elements, creating immersive and connected digital worlds. For autonomous mobility, it enables new possibilities with edge computing and digital twins (DTs) that offer virtual prototyping, prediction, and more. DTs can be created with 3D scene reconstruction methods that capture the real world's geometry, appearance, and dynamics. However, sending data for real-time DT updates in the metaverse, such as camera images and videos from connected autonomous vehicles (CAVs) to edge servers, can increase network congestion, costs, and latency, affecting metaverse services. Herein, a new method is proposed based on distributed radiance fields (RFs), multi-access edge computing (MEC) network for video compression and metaverse DT updates. RF-based encoder and decoder are used to create and restore representations of camera images. The method is evaluated on a dataset of camera images from the CARLA simulator. Data savings of up to 80% were achieved for H.264 *I-frame - P-frame* pairs by using RFs instead of *I-frames*, while maintaining high peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM) qualitative metrics for the reconstructed images. Possible uses and challenges for the metaverse and autonomous mobility are also discussed.**

***Index Terms*—autonomous driving, digital twin, edge computing, metaverse, radiance fields, video compression.**

I. Introduction

The advent of connected autonomous vehicles (CAVs) with their supporting technologies enables interactions between vehicles, servers, and peripheral devices, but at the cost of increased data transmission demands. The hierarchical structure of multi-access edge computing (MEC) network, with distributed local offloading of data and computations of spatially close CAVs helps to mitigate excessive data transfer, minimizing the necessity for large-scale data transmission to centralized servers. However, the application of advanced data compression techniques is still needed to further decrease the transmission latency. According to the research of Hirlay Alves et al. [1], enabling the digital twin (DT) in smart city necessitates a network latency in the 5 to 10 ms range, and a reliability of $1 - 10^{-5}\%$.

Extending the DT concept, the metaverse enables multiple virtual models to coexist and interact, creating a dynamic platform for autonomous mobility applications [2].

Seamless deployment of sophisticated machine learning models trained in simulations to real-world scenarios is hindered by the graphical fidelity gap between them. Therefore, the metaverse aims to bridge this gap by constructing photorealistic digital environments, including dynamic elements, for more effective training and deployment. Recent research [1]–[6] also underscores the importance of implementing metaverse and DTs for reconstructing alternative digital environments, which do not necessitate advanced sensor data, e.g., LiDARs and their subsequent transmission over the network.

In this work, we address the challenge of efficient real-time metaverse updates for CAVs by introducing a novel video compression method based on distributed radiance fields (RFs) within the context of an MEC network. This approach not only achieves significant data reduction but also contributes to a high-fidelity digital twin representation within the metaverse.

Our solution will use visual data compression employing advances in implicit 3D scene reconstruction, specifically RFs. RF is a 3D scene representation that can be created just using a sparse set of 2D images. Following the training phase, RF is able to reconstruct any camera view of the scene, including views that are not present in the training set. Ideally, transmitting only the sender's camera pose data would enable the receiver, equipped with the RF representation, to reconstruct the scene. However, practical limitations like imperfections in RF reconstruction, dynamic objects, lighting changes, and inaccurate pose estimations necessitate additional data transmission.

For these reasons, our method equips both sender and the receiver with RF, so that sender can use standard video encoder to encode any differences between the frame rendered from RF and real 3D scene frame. At the receiver end, these differential transformations are reapplied to the rendered RF frame. While standard video encoding requires periodic sending of non-differential frames containing the whole view information, our approach can completely omit such frames, and in ideal case decrease the differences between real and virtual 3D scene that need to be encoded. This brings large throughput savings.

The proposed RF-based compression method reduces data transmission while enabling seamless metaverse integration.

2693-8340/24/$31.00 ©2024 IEEE
DOI 10.1109/SMARTCOMP61445.2024.00031

Trained on diverse camera images, the RFs serve as the foundation for a high-fidelity digital twin, continuously updated with compressed deltas from CAVs. This ensures real-time synchronization between the physical world and its metaverse counterpart, allowing users and applications to interact with a dynamic and visually accurate digital environment.

II. Related Work

A. Deep Learning-Based Video Compression

The method presented in [7] uses implicit neural representation with a separate neural network for each frame, however, individual video frames are represented implicitly instead of the whole 3D scene. The models used in this work are designed to be relatively simple, with their size further decreasing via quantization and use of the similarities between neighboring frames. Additionally, no pretrained network is used on the receiver side.

The study in [8] introduces two compression schemes, motion residual compression (MRC) and disparity residual compression (DRC), exploiting redundancies in binocular automotive videos. These methods leverage geometric and temporal correlations to compress motion and disparity offsets, using deformable convolution for warping.

Reviews of a wide range of deep learning techniques used for video compression are available in [9] and [10]. In [10], the main methods proposed in existing research are categorized as end-to-end schemes, next video frame prediction, generative models and autoencoder schemes. To the best of our knowledge, none of the existing approaches uses encoder and decoder structures based on RFs.

B. RFs as DTs in Autonomous Mobility

Neural radiance field (NeRF) refers to approximation of 3D scene radiance, capable of reconstructing views on the scene from arbitrary location and angle. Block-NeRF [11] demonstrates how neighbourhood-scale NeRF representation can be built from a set of individually trained NeRFs from visually very diverse data collected over timespan of three months. This work provides an important proof of practical feasibility of building the distributed set of standalone NeRF neural networks seamlessly modelling large area, that far exceeds volumes manageable by a single NeRF.

Distributed visual data collection for creation of NeRF-based DT for autonomous mobility was examined in [12]. The work has tested the speed of real-world to DT updates when considering different DT quality and network conditions. While this work tests many of the assumptions crucial for our work, it misses the key idea of use of NeRFs for rapid compression to further improve the latency.

NeRF-based simulator for autonomous driving was developed in [13]. This approach models background environment and foreground objects separately and allows multiple different NeRF backbones and sampling strategies. Modelling the dynamic foreground objects, like vehicles, using NeRFs increases utility of such environment for training of other machine learning models. Limited modification of captured dynamic foreground objects is demonstrated, like addition, deletion, rotation and translation.

Robustness of training other algorithms for downstream tasks with NeRF-based simulators is shown in [14]. Here, robot is trained in NeRF with environment collisions determined by NeRF volume density and fully synthetic dynamic object, as opposed to NeRF rendered ones, with its physics approximating the real one. Policy learned in NeRF-based simulation was successfully transferred into the real world.

These works show both the proof of potential of large-scale RF-based metaverse with real-time transfer of state from real world into the metaverse, but also the knowledge gap our work tries to close.

III. Video Compression Using Distributed RFs

A. Neural Radiance Fields

NeRFs [15] utilize multi-layer perceptrons (MLPs) to encode 3D scenes into a neural network, parametrizing images with camera poses and optimizing a volumetric scene function approximated by MLP F_Θ:

$$F_\Theta : (\boldsymbol{x}, \boldsymbol{d}) \rightarrow (\hat{\boldsymbol{c}}, \sigma), \tag{1}$$

where $\boldsymbol{x}$ is the location of the point in 3D space, $\boldsymbol{d}$ is the viewing direction, $\hat{\boldsymbol{c}}$ is the emitted radiance, and σ is the volume density.

Rendering from RF involves calculating the expected color $\hat{C}(\boldsymbol{r})$ of a ray $\boldsymbol{r}(t) = \boldsymbol{o} + t\boldsymbol{d}$, defined by its origin $\boldsymbol{o}$ and direction $\boldsymbol{d}$, within near and far bounds t_n and t_f, by integrating transmittance $T(t)$:

$$\hat{C}(\boldsymbol{r}) = \int_{t_n}^{t_f} T(t)\sigma(\boldsymbol{r}(t))\hat{\boldsymbol{c}}(\boldsymbol{r}(t), \boldsymbol{d})dt. \tag{2}$$

The loss $\mathcal{L}_{NeRF}$ is the squared error between rendered $\hat{C}(\boldsymbol{r})$ and ground truth (GT) $C(\boldsymbol{r})$ colors:

$$\mathcal{L}_{NeRF} = \sum_{\boldsymbol{r}\in\mathcal{R}} [\|\hat{C}(\boldsymbol{r}) - C(\boldsymbol{r})\|_2^2], \tag{3}$$

where $\mathcal{R}$ is the set of rays in each batch.

Instant neural graphics primitives (INGP) [16] tackles the issue of NeRFs in excessive training and rendering times using neural graphics primitives and multiresolution hash encoding. The model contains trainable weight parameters ϕ and encoding parameters θ structured into $L \in \mathbb{N}$ levels, each holding up to T feature vectors of dimension F. Each level $l \in L$ operates independently, storing feature vectors at grid vertices. The grid resolution at each level l follows a geometric progression from the coarsest N_{min} to the finest N_{max} resolution by formulas:

$$N_l := N_{\min} \cdot b^l, \tag{4}$$

where N_l is the resolution at level l, and b is the growth factor:

$$b := \exp\left(\frac{\ln N_{\max} - \ln N_{\min}}{L-1}\right). \tag{5}$$

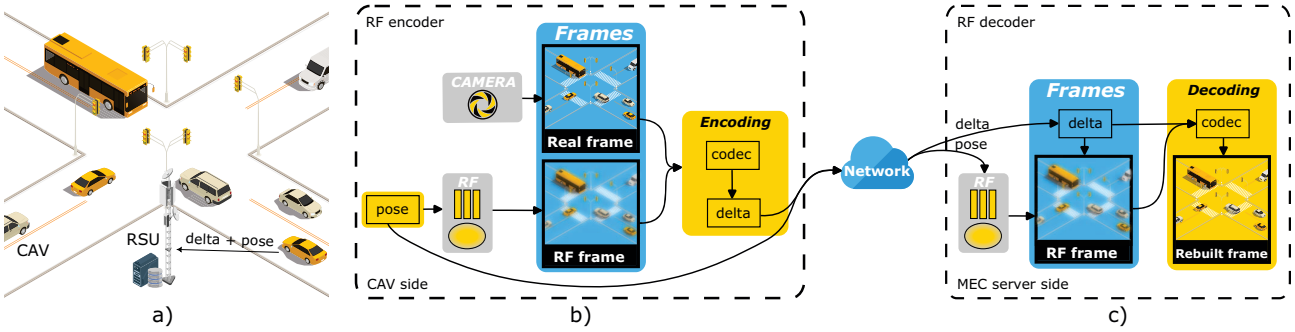

Fig. 1. Proposed novel video compression using distributed RFs in MEC network. The diagram shows a) the real-world scene with CAVs and MEC infrastructure b) RF encoder preparing the delta with differences between the real and RF frame c) RF decoder reapplying the delta to RF frame rendered by local copy of the RF.

B. 3D Gaussian Splatting for RFs

The 3D Gaussian Splatting (3DGS) [17] uses differentiable 3D Gaussians to model scenes without the use of neural components. 3DGS constructs 3D Gaussians $G(x)$ represented by point (mean) μ, covariance matrix Σ, and opacity α:

$$G(x) = exp(-\frac{1}{2}(x)^T\Sigma^{-1}(x)). \tag{6}$$

These Gaussians are projected to 2D using a transformation W and Jacobian J, resulting in a camera-space covariance matrix Σ':

$$\Sigma' = JW\Sigma W^T J^T. \tag{7}$$

The covariance matrix Σ defines scaling S and rotation R:

$$\Sigma = RSS^T R^T. \tag{8}$$

Stochastic gradient descent (SGD) optimizes Gaussian parameters p, α, Σ, and spherical harmonics (SH) representing color c of each Gaussian. The loss function $\mathcal{L}_{3DGS}$ combines mean absolute error $\mathcal{L}_1$ and a differentiable structural similarity index measure (SSIM) $\mathcal{L}_{D-SSIM}$ with a balance hyperparameter λ:

$$\mathcal{L}_{3DGS} = (1-\lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{D-SSIM}. \tag{9}$$

C. Video Compression Using RFs

In stark contrast to widely used compression algorithms, our approach uses models with much higher informational content about 3D scenes, surpassing traditional compression by accessing even pixel information not available in previously encoded 2D frames. This capability is particularly crucial for capturing details about unseen pixels beyond the camera field of view boundaries or occluded object parts that become visible during movement.

While our approach relies on RFs, real-world camera view will still have multiple differences when compared to RF views, introduced by mobile objects, lighting changes and noise, among other factors. For this reason, we use H.264 compression algorithm to bridge the RF-to-real gap. CAVs capture and encode images continuously during their operation, transmitting the compressed data to the MEC server for decoding and metaverse integration. H.264 compressed video streams consist of *I-frames* with complete image information, *P-frames* that encode only the differences from the preceding *I-frames*, and *B-frames* capable of encoding differences from temporally bidirectional frames. From now we will refer to the differences encoded by *P-frame* as *P-frame delta*, or just *delta*. Conventionally, every n-th frame in a compressed video is an *I-frame*, with parameter n being set prior to video compression, our approach is able to omit such frames.

Our proposed approach consists of RF encoder, the encoding data scheme combining camera pose with H.264 encoded difference between real and RF frame and RF decoder as depicted in Fig. 1. RF encoder uses camera pose at CAV to obtain a view rendered by the local copy of the RF and then encodes the difference between real frame and RF rendered one, into *P-frame* delta using H.264 encoder. Camera pose and *P-frame* delta are sent through the wireless network to roadside unit (RSU) that contains local MEC server performing the decoding. RF decoder decodes the original image from camera pose, used to render the view with RF, and *P-frame* delta, which encodes the differences between the real world and stored RF. The RF models are pre-trained and distributed through wireless and wired links of MEC network to both CAVs and MEC servers, ensuring consistent rendering and decoding of the visual information.

The whole process of RF-based encoding, transmission of encoded data, and RF-based decoding is described in pseudocode Alg. 1, using the three corresponding procedures.

An experimental setup was established in a virtual urban environment using the CARLA simulator to validate this approach.

D. Technical Challenges

The proposed RF-based video compression method, faces several technical challenges, that can however be resolved :

- **Dynamic Object Representation:** Integrating dynamic objects directly into RFs for metaverse applications beyond video transmission is challenging. Potential solu-

Algorithm 1 RF-based Encoder, Network Transmission, and RF-based Decoder

```
procedure RFENCODER(CAV_Images, RF)
    Encoded_Frames ← empty list
    for each image ∈ CAV_Images do
        pose ← ExtractPose(image)
        RF_frame ← RenderViewFromRF(RF, pose)
        delta ← EncodeDifference(image, RF_frame)
        Add(Encoded_Frames, (pose, delta))
    end for
    return Encoded_Frames
end procedure

procedure NETWORKTX(Encoded_Frames, Throughput)
    Transmitted_Frames ← empty list
    for each (pose, delta) ∈ Encoded_Frames do
        frame_size ← GetSize(pose, delta)
        τ ← frame_size/Throughput
        SendFrameOverNetwork(pose, delta, τ)
        Add(Transmitted_Frames, (pose, delta, τ))
    end for
    return Transmitted_Frames
end procedure

procedure RFDECODER(Transmitted_Frames, RF)
    Decoded_Images ← empty list
    for each (pose, delta, τ) ∈ Transmitted_Frames do
        WaitForTransmission(τ)
        RF_frame ← RenderViewFromRF(RF, pose)
        image ← DecodeDifference(RF_frame, delta)
        Add(Decoded_Images, image)
    end for
    return Decoded_Images
end procedure
```

tions include combining RFs with polygonal 3D models or developing RFs specifically for dynamic scenes.

- **Real-time Synchronization and Latency:** Efficient data transmission and low latency are crucial for real-time DT updates in the metaverse. RF-based compression must minimize data size while ensuring fast transfer to maintain synchronization. Network conditions can impact latency and system performance.
- **Camera Pose Estimation:** Accurate camera pose estimation is vital for RF rendering. Inaccurate pose information can distort reconstructed images, affecting visual quality and compression. Robust pose estimation is essential, especially in dynamic real-world scenarios.

E. Qualitative Metrics

To evaluate both the possible quality degradation introduced by RFs and differences between the RFs and changes in the environment (different lighting, presence of vehicles at new locations, etc.) we have used multiple metrics widely applied for image quality comparisons. These include peak signal-to-noise ratio (PSNR), SSIM, and learned perceptual image patch similarity (LPIPS). These metrics measure the difference between a reference image and a distorted image in terms of pixel values, structural features, and perceptual features, respectively.

IV. EXPERIMENTAL RESULTS

An experimental framework was established to evaluate the quality of the 3D scene reconstructed by RFs and compression savings of a newly RF-based compression algorithm within an urban 3D environment, as illustrated in Fig. 2.

A. Dataset

A training dataset for the RFs was meticulously compiled by rendering the specified urban 3D scene utilizing the CARLA simulation software using 18 cameras attached to the car driving in both directions of the street. Fig. 2 depicts images from CARLA simulator as GT for two scenarios, empty road and parked vehicles on the sides of the road. The compression efficiency of the proposed approach was systematically evaluated by plotting the compression gains across 144 frames captured from multiple cameras positioned along the vehicle's trajectory.

B. H.264 Encoding

Fig. 3 illustrates the percentages of compression savings achieved at varying resolutions using INGP in the scene without vehicles. The efficiency of our proposed method is quantified in terms of compression savings, which are determined in relation to a baseline established by frame pairs (*I-frame* and *P-frame*) compressed using the H.264 codec. This assessment encompassed a range of frame resolutions, extending from 300 $\times$ 168 to 1920 $\times$ 1080 (full HD). Images within the dataset were encoded utilizing configuration settings, specifically presets including 'veryslow', 'medium', and 'veryfast', coupled with constant rate factors (CRFs) of 18, 23, and 28. Subsequently, the values obtained for each encoded image pair were averaged to derive the final results. Notably, in our approach, the *I-frame* is not transmitted over the network. Instead, it is generated using RFs. The formula employed for the calculation of compression savings thus calculates percentual data size decrease resulting from *I-frame* omission as follows $100 * I_{size}/(I_{size} + P_{size})$, where I_{size} and P_{size} denote the file sizes of the *I-frame* and *P-frame*, respectively.

C. Evaluation

Table I presents an evaluation of the RF models utilizing PSNR, SSIM, and LPIPS metrics to reveal differences between GT and RF-generated image. In the scenario devoid of external elements not present in RF, the INGP model exhibited notably inferior performance, primarily attributable to its less precise reconstruction of street lamp structures compared to the 3DGS model, as illustrated in Fig. 2. This disparity in model efficacy is further reflected in the context of video compression savings. Fig. 3 and Fig. 4 demonstrate that compression savings of the

Fig. 2. GT images from CARLA simulator with matched INGP and 3DGS images from the same camera pose. Note the varying degree of blur and missing details in INGP and 3DGS rendered frames, like distortion of the letters in the "MUSEUM" sign and missing parts of the lamp structures.

TABLE I
AVERAGE MEAN VALUES OF RF MODELS

Metrics	**PSNR ↑**		**SSIM ↑**		**LPIPS ↓**	
Scenario	INGP	3DGS	INGP	3DGS	INGP	3DGS
Empty	26.33	29.41	0.75	0.85	0.38	0.24
Vehicles	21.33	21.78	0.68	0.71	0.44	0.30

3DGS model are ranging from 45% to 80%, whereas the INGP model achieved a lower range of 30% to 68%.

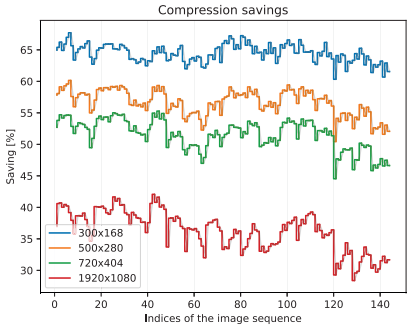

Fig. 3. Encoder setting-averaged compression savings relative to H.264 achieved for individual images in empty scene scenario using INGP.

Conversely, in the scenario incorporating vehicles into the 3D scene, while using RFs in which they were absent, both RF models experienced a decline in metric performance due to the introduction of these additional objects. The differences in performance between the models in this scenario were more nuanced. As per Table I, the LPIPS metric was consistent with the empty scenario, showing similar variance. However, Fig. 5 and Fig. 6 suggest more subtle distinctions between the

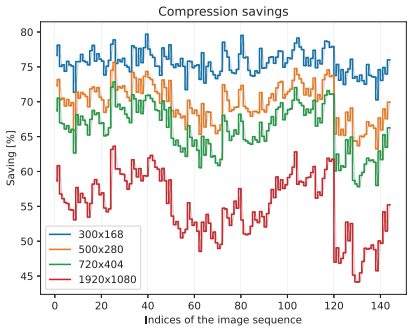

Fig. 4. Encoder setting-averaged compression savings relative to H.264 achieved for individual images in empty scene scenario using 3DGS.

models in this scenario compared to the former. In this case, the INGP model exhibited compression savings between 28% and 68%, while the 3DGS model ranged from 30% to 75%. This suggests that increased variability in the real scene leads to more homogeneous evaluation results of the RF-generated images, indicating a correlation between scene complexity and model performance consistency.

Additional outcome is the observed inverse relationship between resolution and compression savings, partly due to the imperfections in images generated by RF models. In the downscaling process, high-frequency signal components are lost in both original and RF-generated images, acting as a form of lossy compression. This loss brings RF-generated images closer to the GT by removing high-frequency artifacts, highlighting the complex interplay between resolution and compression effectiveness.

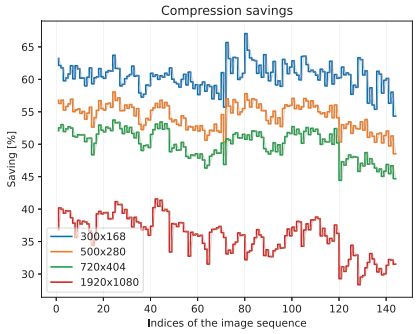

Fig. 5. Encoder setting-averaged compression savings relative to H.264 achieved for individual images in vehicles in the scene scenario using INGP.

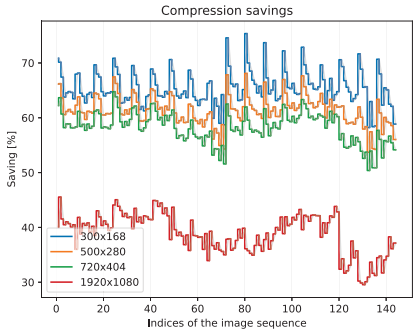

Fig. 6. Encoder setting-averaged compression savings relative to H.264 achieved for individual images in vehicles in the scene scenario using 3DGS.

V. Conclusion

This work introduces a novel compression approach utilizing RF-based encoder and decoder, simultaneously serving as a distributed photorealistic metaverse backbone, featuring low real-to-sim synchronization latency. By transmitting only compressed deltas, the method minimizes network congestion and facilitates rapid metaverse updates. This enables other metaverse users and applications to experience a dynamic, visually accurate, and responsive digital twin of the physical environment.

We have first evaluated feasibility of such approach in review of related work, and then experimentally on a dataset of camera images and videos captured by simulated CAVs in CARLA simulator. Our results show that our approach can achieve significant data compression and high reconstruction quality.

Future research could focus on incorporating dynamic object representations into the RFs for enhanced metaverse realism and enabling interactions with moving objects. Furthermore, validating the method's performance with real-world data, considering factors like sensor noise, lighting variations and complex dynamic interactions, will provide a comprehensive understanding of its real-world applicability.

Acknowledgment

This work was supported by The Slovak Research and Development Agency project no. APVV SK-CZ-RD-21-0028 and the Slovak Academy of Sciences project no. VEGA 1/0685/23.

References

[1] H. Alves, G. D. Jo, J. Shin, C. Yeh, N. H. Mahmood, C. H. M. de Lima, C. Yoon, G. Park, N. Rahatheva, O.-S. Park, and et al., "Beyond 5G urllc evolution: New service modes and practical considerations," *ITU Journal on Future and Evolving Technologies*, vol. 3, no. 3, p. 545–554, 2022.

[2] M. Xu, W. C. Ng, W. Y. B. Lim, J. Kang, Z. Xiong, D. Niyato, Q. Yang, X. Shen, and C. Miao, "A Full Dive Into Realizing the Edge-Enabled Metaverse: Visions, Enabling Technologies, and Challenges," *IEEE Communications Surveys and Tutorials*, vol. 25, no. 1, pp. 656–700, 2023.

[3] S. Mihai, M. Yaqoob, D. V. Hung, W. Davis, P. Towakel, M. Raza, M. Karamanoglu, B. Barn, D. Shetve, R. V. Prasad, H. Venkataraman, R. Trestian, and H. X. Nguyen, "Digital Twins: A Survey on Enabling Technologies, Challenges, Trends and Future Prospects," *IEEE Communications Surveys and Tutorials*, vol. 24, no. 4, pp. 2255–2291, 2022.

[4] Y. Ren, R. Xie, F. R. Yu, T. Huang, and Y. Liu, "Quantum Collective Learning and Many-to-Many Matching Game in the Metaverse for Connected and Autonomous Vehicles," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 11, pp. 12 128–12 139, 2022.

[5] M. Xu, D. Niyato, H. Zhang, J. Kang, Z. Xiong, S. Mao, and Z. Han, "Generative AI-empowered Effective Physical-Virtual Synchronization in the Vehicular Metaverse," 2023.

[6] P. Zhou, J. Zhu, Y. Wang, Y. Lu, Z. Wei, H. Shi, Y. Ding, Y. Gao, Q. Huang, Y. Shi, A. Alhilal, L.-H. Lee, T. Braud, P. Hui, and L. Wang, "Vetaverse: A Survey on the Intersection of Metaverse, Vehicles, and Transportation Systems," 2023.

[7] Y. Zhang, T. van Rozendaal, J. Brehmer, M. Nagel, and T. Cohen, "Implicit neural video compression," *arXiv preprint arXiv:2112.11312*, 2021.

[8] Z. Chen, G. Lu, Z. Hu, S. Liu, W. Jiang, and D. Xu, "LSVC: A learning-based stereo video compression framework," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6073–6082.

[9] D. Ding, Z. Ma, D. Chen, Q. Chen, Z. Liu, and F. Zhu, "Advances in video compression system using deep neural network: A review and case studies," *Proceedings of the IEEE*, vol. 109, no. 9, pp. 1494–1520, 2021.

[10] R. Birman, Y. Segal, and O. Hadar, "Overview of research in the field of video compression using deep neural networks," *Multimedia Tools and Applications*, vol. 79, pp. 11 699–11 722, 2020.

[11] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar, "Block-NeRF: Scalable large scene neural view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8248–8258.

[12] Y. Liu, X. Tu, D. Chen, K. Han, O. Altintas, H. Wang, and J. Xie, "Visualization of Mobility Digital Twin: Framework Design, Case Study, and Future Challenges," in *2023 IEEE 20th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*. IEEE, 2023, pp. 170–177.

[13] Z. Wu, T. Liu, L. Luo, Z. Zhong, J. Chen, H. Xiao, C. Hou, H. Lou, Y. Chen, R. Yang *et al.*, "Mars: An instance-aware, modular and realistic simulator for autonomous driving," *arXiv preprint arXiv:2307.15058*, 2023.

[14] A. Byravan, J. Humplik, L. Hasenclever, A. Brussee, F. Nori, T. Haarnoja, B. Moran, S. Bohez, F. Sadeghi, B. Vujatovic *et al.*, "Nerf2real: Sim2real transfer of vision-guided bipedal motion skills using neural radiance fields," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9362–9369.

[15] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[16] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Transactions on Graphics (ToG)*, vol. 41, no. 4, pp. 1–15, 2022.

[17] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D Gaussian Splatting for Real-Time Radiance Field Rendering," 2023.