2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Evaluation of Predictive Display for Teleoperated Driving Using CARLA Simulator

Fatima Kashwani[1,2], Bilal Hassan[1,2], Peng-Yong Kong[2], Majid Khonji[1,2], and Jorge Dias[1,2]

Abstract— **Before the world-wide deployment of autonomous vehicles, it is essential to implement intermediate solutions with partial autonomy. One such solution is the use of vehicle teleoperation, the act of controlling a vehicle from a distance. In real time applications of teleoperation, it is often pertinent to use augmented reality components within the teleoperator view, which are referred to as a predictive display. In this work, we evaluate our predictive display method, which is a guiding path based on the free space in the environment. The path is generated based on our Dual Transformer Network (DTNet), which uses both object detection and lane semantic segmentation to define the free space in the environment. While the model has previously performed well on image data, it is necessary to observe its accuracy in the presence of time delay and packet loss, to assess its performance in a real-time setting. Thus, in this work, we use CARLA simulator to compare the detected free space on the teleoperator side to the true free space on the vehicle side across different values of time delay and packet loss. Under optimal network conditions, our model yielded a remarkable 87.9% DSC score and 81.3% IoU score. Defining our minimum performance threshold as 80% DSC and 70% IoU, we conclude that our model can effectively mitigate the challenges of time delay below 100ms and packet loss below 1%, both of which represent substantial tolerances.**

I. Introduction

In recent years, many applications have risen for teleoperation, the control of robots remotely. In autonomous vehicles, teleoperation has emerged as a stepping stone towards full autonomy. Namely, it satisfies the availability of a human operator on standby without requiring their presence inside the vehicle. However, a major concern in teleoperation systems is the presence of time delay, meaning the video feed received by the teleoperator does not accurately represent the state of the environment at the present moment. To mitigate this, we propose the use of a predictive display, which superimposes guiding visuals on the delayed video feed. Our Dual Transformer Network (DTNet) is composed of an object detection module and a lane semantic segmentation module, the results of which are combined to define the free space in the environment. However, due to the dynamic nature of the road environment, the free space detected in the teleoperator-side video may not be consistent with the actual free space on the vehicle side. In this work, we use CARLA simulator to compare the free space on either side across different values of time delay and packet loss.

*This work is supported by KUCARS.

[1]These authors are affiliated with the Center for Autonomous Robotic Systems (KUCARS), Abu Dhabi, 127788, UAE.

[2]These authors are affiliated with the Department of Electrical Engineering and Computer Science, Khalifa University, Abu Dhabi, 127788, UAE. {100045434, bilal.hassan, pengyong.kong, majid.khonji, jorge.dias}@ku.ac.ae

II. State of the Art

There are two main approaches for mitigating time delay in teleoperation: supervisory control, where the aim is to introduce some level of autonomy so that the robot can correct the inaccurate commands of the teleoperator, and direct teleoperation, where the aim is to provide enhanced information to the teleoperator to improve driving accuracy. Supervisory control allows the vehicle to operate semi-autonomously while still bound by the teleoperator's high-level instructions. For instance, Schitz et al. proposed the use of a "corridor" method, whereby the vehicle traverses semi-autonomously within the guidelines of a corridor defined by the teleoperator [1]. Several other applications utilize path planning as a basis for supervisory control [2], [3]. Another application involves modifying teleoperator commands to mitigate the time delay. For instance, Prakash et al. opted for sending reference poses to the vehicle rather than individual steering commands, so as to more efficiently convey commands across the delay [4].

Direct teleoperation, in contrast, mitigates time delay by providing enhancements to the information provided to the teleoperator. This can come in many forms; for instance, Chen et al. employed the use of force feedback as a guide to the teleoperator, to provide real-time force perception to the teleoperator [5]. Additionally, a popular approach is the use of visual guides to augment the delayed video feed. For instance, Liu et al. used virtual fixtures as a guiding display for space applications, which highlighted the obstacle-free regions in the teleoperator's view [6]. Another approach by Graf et al. involved the use of steering and curvature to calculate and display the accurate position of the vehicle superimposed on the video feed [7]. To account for the dynamic environment, Schitz et al. proposed a path-planning algorithm composed of two components: a global algorithm for detecting all feasible paths, and a local algorithm for optimizing whichever path the teleoperator chooses to pursue [2].

Further, our model defines the free space in the environment using lane semantic segmentation and object detection, both of which utilize a swin transformer backbone [8]. Semantic segmentation of road parts, including lanes and road lines, is a prevalent area for autonomous vehicles [9], [10]. Some applications focus on reducing computational complexity for real-time deployment, such as by processing rows rather than individual pixels [11]. On the other hand, whereas object detection is less computationally expensive, it requires increasing the precision of bounding boxes to

yield reliable results. As compared to convolutional neural networks (CNNs), transformers have shown significant improvements in recent years [8], [12]. Specifically, swin transformers have shown a higher precision than YOLOv4 and YOLOx [13].

Our proposed predictive display leverages a collision-free path as a visual guide to the teleoperator. The model contains an object detection module and lane semantic segmentation module, the results of which are combined to form the free space mask by which the path is generated. In previous work, the proposed model was shown to surpass state-of-the art methods in both regards [14]. In this work, we expand upon this by testing the model in a simulated teleoperation environment under the effects of time delay and packet loss.

III. Methodology

Our predictive display is composed of a collision-free path used as a visual guide. We aim to define the range of network conditions within which our predictive display functions robustly and effectively. The upcoming sections discuss our methodology in further detail.

A. Experimental Setup

In order to observe the performance of our model across a variety of network conditions, we use CARLA simulator, an open-source framework for autonomous vehicle research [15]. Further, we utilize the teleoperation extension for CARLA, TELECARLA [16].

Figure 1 summarizes the components provided by CARLA simulator, those added by TELECARLA, and our proposed components. TELECARLA utilizes the client-server architecture provided by CARLA to simulate teleoperation, namely concerning the driving controls and the video feed. The CARLA-ROS bridge is used to structure these components as ROS nodes communicating with each other, as though across a network. For instance, the video feed is streamed from a ROS node on the server side to another node on the client side, via GStreamer software. The GStreamer parameters are configured and adjusted according to the requirements of the experiment. Likewise, a pair of nodes utilizing RPC protocol are used to communicate the teleoperator controls.

Our proposed additions to the TELECARLA architecture are shown in purple in figure 1. On the client side, we add the collision-free path node. The collision-free path is generated based on the free space in the environment, which is based on the received frame on the GStreamer client. We implement our model on the teleoperator end. This aids in identifying the extent of the processing delay more definitively. We also visualize the original feed captured by the vehicle camera on the server end; the frames captured by this display are used as ground-truth for the evaluation.

To evaluate the predictive display, we use the tools provided by linux traffic control network emulator, tc-netem. Namely, we evaluate across different values of both communication delay and packet loss. Across different delay values, we observe how accurate the predicted mask is to the vehicle-side mask. With regards to packet loss, we observe the range of packet loss added to the image within which the model is successfully able to recognize the road.

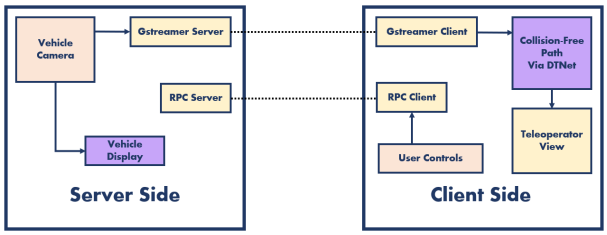

Fig. 1. Experimental setup for the simulation environment using CARLA simulator. The simulator architecture is based on a client-server model. Components shown in orange are provided by CARLA simulator. Components shown in yellow are provided by the TELECARLA extension. Finally, we propose the components shown in purple to further enhance the teleoperation system.

B. Dual Transformer Network (DTNet)

The free space mask used to generate the collision-free path is obtained through the two modules of DTNet, which is shown in figure 2. The refined free space is the area which is part of the segmented ego lane and does not intersect with any obstacles. Using center point extraction, the collision-free path is defined within this free space.

Both detection and segmentation modules leverage the swin transformer, a hierarchical vision model based on the vision transformer (ViT) [17]. The swin transformer improves upon ViT by using a shifted-window multi-head self-attention (SW-MSA), which yields the aggregated attended features in each patch by computing the attention scores of each shifted window. The swin transformer also contains an optional token merging step after a certain number of blocks, which enhances global context understanding. The next sections describe the model components in more detail.

1) Token and Positional Embeddings: The input image is embedded by the model as both token and positional embeddings. The image is first divided into patches, each of which is represented as a token. Each patch is then embedded into d-dimensional embedding space. We also include 2D positional embeddings to preserve relative spatial information. The positional embedding $P(i)$ and the token embedding $T(X_i)$ are combined for each path x_i, to yield the final embedding E_i, as shown:

$$E_i = T(x_i) + P(i), \tag{1}$$

2) Shifted Window Mechanism: By using the shifted window mechanism, the network can combine the local context effectively. This is because the mechanism allows a patch to attend to a certain number of neighbour patches in a window. Equations 2 and 3 show how the learnable projection matrices, W_q, W_k, and W_v, are used to compute the attention.

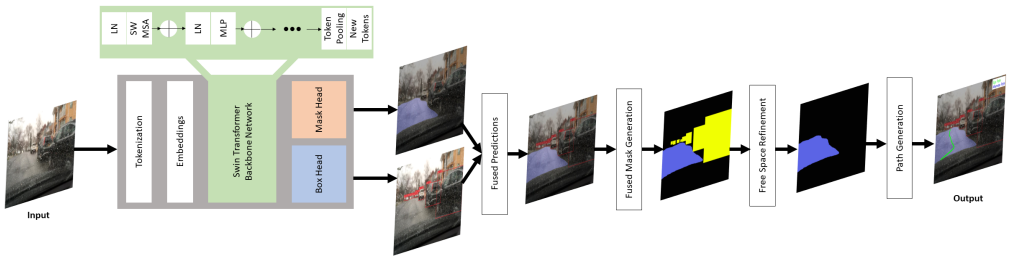

Fig. 2. The Dual Transformer Network, DTNet. The model is composed of a detection module and a segmentation module, each of which utilize a swin-s transformer backbone. The results of these modules are combined to form the free space mask, based on which the collision-free path is generated.

$$MSA(E_i, E_j) = SM((E_i * W_q) * (E_j * W_k)^T / \sqrt{(d_k)}), \tag{2}$$

$$MSA_{output}(E_i) = \Sigma(MSA(E_i, E_j) * (E_j * W_v)), \tag{3}$$

3) Hierarchical Stages: The swin-s backbone, which is utilized by both detection and segmentation modules, is composed of four hierarchical stages. Each hierarchical stage contains a stack of swin blocks 2, 2, 18, and 2. The input patches first undergo layer normalization (LN) to increase stability during training. Next, each block utilizes the shifted window mechanism and the feed-forward network (FFN). The outputs of one stage are used as inputs to the next, enabling the model to effectively capture both local and global features.

4) Segmentation Module: The segmentation module in DTNet utilizes a mask head, containing a convolutional layer and upsampling layer. Within the mask head, each pixel in the image is classified into one of three classes: ego lane, alternate lane, and background.

5) Detection Module: The detection module serves to both localize bounding boxes for the detected objects and to label them accordingly. Thus, the module contains a class head, used to predict labels of the detected objects, as well as a box head to regress the bounding boxes of each object. This module also employs the use of anchor boxes of different aspect ratios, so as to account for the variety in object sizes. Finally, non-maximum suppression (NMS) is used to remove duplicated and low-confidence predictions, ensuring the quality and confidence of the remaining predictions.

6) Free Space Refinement: To obtain the refined free space mask, we combine the output of both detection and segmentation modules. Any overlapping areas between the ego lane mask and the detected objects will be eliminated, resulting in a refined free space mask of the drivable, obstacle-free road space.

C. Collision-Free Path

1) Center Line Extraction and Smoothing: The collision-free path is generated based on the center points of the road mask. The y-axis center points are extracted in a row-wise manner, as shown in equation 4, where CP_y is the y-coordinate of the center point of the given row, N is the number of columns in the mask, y_i is the y-coordinate of the i-th pixel in the given row, and $Mask_i$ denotes whether i-th pixel belongs to the free space.

$$CP_y = \frac{\Sigma_{i=1}^{N} y_i * Mask_i}{\Sigma_{i=1}^{N} Mask_i}, \tag{4}$$

It is necessary to distinguish that all center points lie within the free space mask, even in the presence of disjoint free space segments. This is due to the nature of defining the middle pixel of a row exclusively within the free space columns of that row, as shown in figure 3.

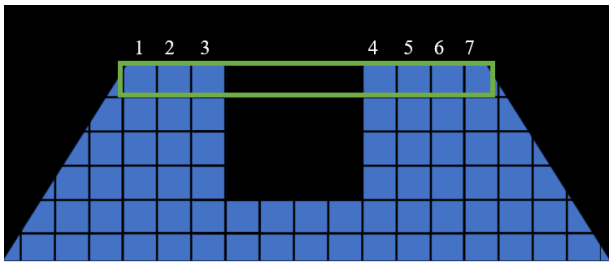

Fig. 3. Visualization of center point extraction. Note that in the top row (highlighted in green), the only columns considered are those belonging to the segmentation mask. As such, the algorithm accounts for non-continuous segments. in this example, the pixel in column 4 is the center point of the row.

Next, we reduce noise by eliminating outlier points, which we define as points exceeding a certain Euclidean distance threshold from adjacent points. Mathematically, the Euclidean distance (E_d) between two points (x_i, y_i) and (x_j, y_j) is as follows:

$$E_d(i, j) = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}, \tag{5}$$

The remaining points are then connected. To address jitters and irregularities along the connected path, we employ smoothing via the moving average. given P as the set of connected center points, the smoothed path P_{MA} is defined by:

$$P_{MA}(i) = \frac{1}{N} \Sigma_{j=i-W}^{i+W} P(j), \tag{6}$$

Wherein $P_{MA}(i)$ is the i-th point in the path P,W is the size of the smoothing window, and N is the number of points in the smoothing window. Additionally, for video and real-time applications, we include averaging over previous frames to further enhance the smoothing, which created an elegant and robust display for the teleoperator.

Finally, to provide a more intuitive user experience, we incorporate curvature adjustment, which adjusts the center points based on the current angle of the steering wheel. Note that, since each point corresponds to a row, the y-values remain unchanged. The x-values for each point are updated as shown in equation 7, where x_i' is the updated x-value, x_i is the original x-value, y_i is the y-value of the point, y_0 is the y-value of the lowermost point in the image, θ is the steering angle, and s is a constant denoting sensitivity. Note that the use of $y_i - y_0$ ensures that the curvature is increasing as the distance from the vehicle increases.

$$x_i' = x_i + s \cdot (y_i - y_0) \cdot sin(\theta) \tag{7}$$

2) TELECARLA Implementation: To evaluate the collision-free path under different network conditions, we observe its accuracy in the TELECARLA environment. The path is generated based on the free space in the environment. The accuracy of the free space relies on the similarity between the predicted mask and ground truth mask. However, with the presence of time delay, the accuracy also depends on the similarity between the server-end frame and client-end frame; if delay is high, there is a notable difference even between the ground truth masks, as shown in figure 4.

For evaluation, we obtain the road mask on the vehicle end. To extract this mask, we utilize the CARLA semantic segmentation camera, which segments the view into several classes, including road and road line. When packet loss is added, the use of GStreamer to simulate video streaming introduces some pixelation and distortion to the frame on the teleoperator side, the extent of which relies on the GStreamer network configuration and added packet loss values. We thus evaluate the accuracy of the teleoperator-side road mask across a variety of network conditions, using the server-end road mask as ground truth.

Further, the semantic segmentation camera provides the full road mask, whereas our model segments the ego and alternate lanes separately. To address this, we use a combined free space mask for evaluation, containing both ego and alternate lanes. An example of ground-truth and predicted masks is shown in figure 4.

We evaluate our model across different values of added time delay, namely 50ms, 100ms, and 150ms. To account for the delay's effect on controls and, consequently the vehicle speed, we eliminate the use of throttle input. Instead, we manually set the acceleration value such that the speed of the vehicle is capped at 30 m/s and increasing otherwise, meaning the vehicle is at an approximate speed of 30 m/s during the experiment. For packet loss, we test values between 1-5%, observing the accuracy of the model in the presence of pixelation and distortion. However, we maintain the use of manual steering. We evaluate in both the absence and presence of traffic. For the latter, we deploy 93 moving vehicles and pedestrians across the map of Town 1, utilizing the"generate traffic" python API provided by CARLA.

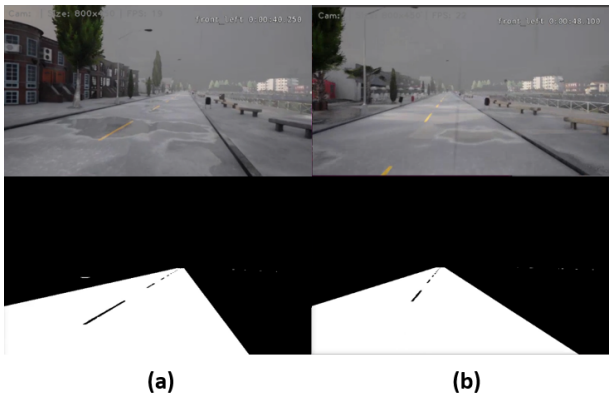

Fig. 4. Camera view and its corresponding ground truth road mask at two different points in time. With a high enough delay and vehicle speed, the teleoperator view may display frame (a) while the actual vehicle position is at frame (b). Regardless of accuracy to the ground truth of frame (a), the generated road mask will not be fully accurate to the mask at frame (b).

IV. Results

In this section, we observe the quantitative and qualitative results of implementing DTNet on the TELECARLA framework. We begin by observing the model's qualitative performance under different delay and packet loss values where the vehicle is moving at approximately 30 m/s. Next, we proceed to the quantitative evaluation, first discussing the general performance with no added communication challenges, in comparison to the model's previous performance on image datasets, followed by introducing time delay and packet loss in both the absence and presence of traffic.

A. Qualitative Results

In this section, we discuss the visual results of implementing DTNet on TELECARLA, as well as the qualitative implications of added time delay and packet loss. Using TELECARLA, we implement DTNet on the teleoperator-end RGB image, and define the ground truth as the vehicle-end mask extracted from the semantic segmentation image. We combine the ego lane and alternate lane masks provided by the DTNet free-space mask to yield the whole road mask, which is shown in the sections below.

1) Addition of Time Delay: We observe the results of adding 50ms, 100ms, and 150ms of time delay using linux traffic control (tc-netem). Figure 5 shows two examples of 150ms added time delay. As is evident in example (I), even for a high delay value as 150ms, there is not much discrepancy between the two masks when the vehicle is travelling along a straight road, as consecutive frames are very similar. On the other hand, as shown in example (II), rapid changes in consecutive frames affect the predicted mask when delay is present, such as when an intersection

appears in the ground-truth mask but does not yet appear in the delayed predicted mask.

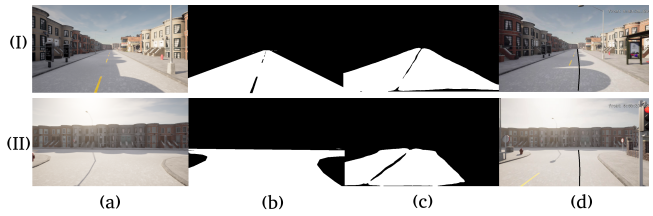

Fig. 5. Two examples of 150ms time delay, where (a) shows the vehicle-end RGB camera view, (b) shows the corresponding ground-truth segmentation, (c) shows the free space mask predicted by DTNet on the teleoperator end, and (d) shows the resultant path. Example (I) shows the model's performance on a straight road, while example (II) shows an intersection.

2) Addition of Packet Loss: Similarly, packet loss between 1% - 5% was added using linux traffic control. As shown in figure 6, the addition of packet loss introduced some distortion and pixelation in the teleoperator-side image, which introduced some difficulty in detecting the free space depending on the extent of the distortion. Further, packet loss caused some frames to be lost entirely, resulting in similar implications as those of time delay. As shown in the figure, our model was able to detect the road free space in the image relatively well in the presence of visual distortions. When

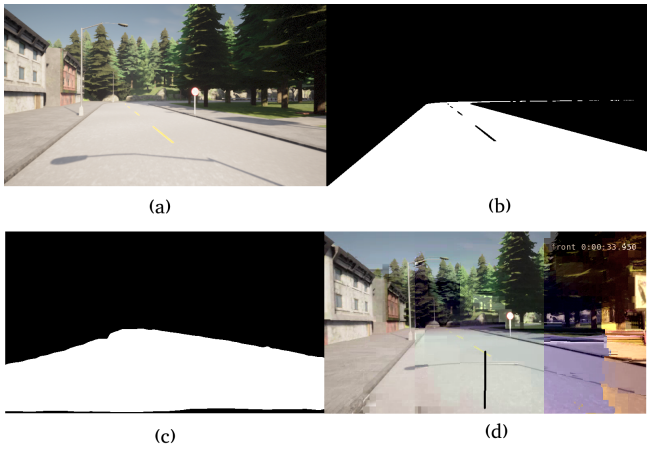

Fig. 6. Two examples of 5% packet loss, where (a) shows the vehicle-end RGB camera view, (b) shows the corresponding ground-truth segmentation, (c) shows the free space mask predicted by DTNet on the teleoperator end, and (d) shows the resultant path.

driving along a straight road, free space masks can appear very similar across time. Due to this, our model can yield high accuracy values, even when the discrepancy between driver-side and vehicle-side frames is large. To address this, we introduce traffic to the simulation. Figure 7 shows an example of how packet loss can cause erroneous perception in the presence of traffic.

B. Quantitative Results

In this section, we observe the accuracy of DTNet in TELECARLA by computing DSC and IoU. We begin by comparing the accuracy to that of image datasets, then

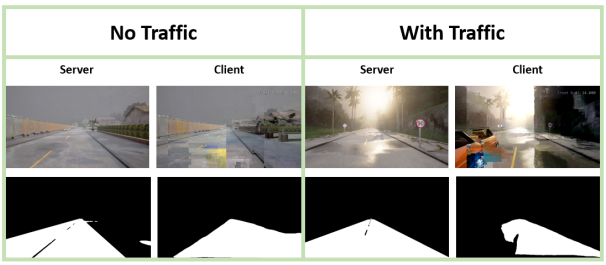

Fig. 7. Examples comparing 5% packet loss in the absence and presence of traffic. The server-side masks are ground truth, while the client-side masks are generated by DTNet. As shown, the presence of traffic causes a much more dynamic environment, increasing the likelihood of inaccuracies in detecting the free space correctly. This example shows our model detecting a vehicle and eliminating it from the free space mask, whereas the server side shows that the vehicle is no longer in view.

proceed to evaluating accuracy across different values of time delay and packet loss.

1) Free Space Refinement: Our model, DTNet, was trained on the BDD100k dataset. Table I showcases the intersection over union (IoU) semantic segmentation results for testing on the BDD100k dataset, the unseen KITTI dataset, and within the TELECARLA simulation environment. We consider the mean IoU, which is across both ego and alternate lanes. Remarkably, the performance on TELECARLA exceeds that on the unseen dataset by a large margin, demonstrating our model's efficacy in detecting free space within the simulation environment.

TABLE I
COMPARISON OF IOU VALUES WHEN TESTING DTNET.

Data	IoU
BDD100k Dataset [18]	83.9%
KITTI Dataset [19]	54.9%
TELECARLA	81.3%

2) Addition of Time Delay: The quantitative results for added time delay of 50ms, 100ms, and 150ms are shown in table II, namely the Dice Coefficient (DSC) and intersection over union (IoU). As shown, the values are similar for all the presented delay values in the absence of traffic; since experiments were run by manual steering, the majority of data involved similar masks of straight roads. On the other hand, in the presence of traffic, a drop in both DSC and IoU is evident for higher values of time delay. In order to maintain DSC and IoU values of 80% and 70%, respectively, the time delay of the system must be under 100ms.

3) Addition of Packet Loss: Finally, we present the quantitative results for the addition of packet loss, which are shown in table III. Similarly to the delay results, the added packet loss causes little change in DSC and IoU in the absence of traffic. This showcases our model's ability to accurately identify the road free space when visual distortions are present. However, as mentioned previously, packet loss can involve loss of entire frames, causing a "lagging" appearance

TABLE II

COMPARISON OF DSC AND IOU ACROSS DIFFERENT DELAY VALUES WITH AND WITHOUT TRAFFIC.

Delay	No Traffic		With Traffic	
	DSC	IoU	DSC	IoU
No Added Delay	87.9%	81.3%	87.0%	78.2%
50ms	85.9%	77.7%	81.7%	73.2%
100ms	85.5%	77.8%	74.8%	68.3%
150ms	86.0%	78.3%	71.9%	64.1%

to the video feed. In this regard, we can see the rapid decline of accuracy in the presence of traffic as packet loss increases. Maintaining our earlier permissible thresholds of 80% DSC and 70% IoU, we can state that pixel loss under 1% is permissible.

TABLE III

COMPARISON OF DSC AND IOU ACROSS DIFFERENT PACKET LOSS VALUES WITH AND WITHOUT TRAFFIC.

Packet Loss	No Traffic		With Traffic	
	DSC	IoU	DSC	IoU
No Packet Loss	87.9%	81.3%	87.0%	78.2%
0.5%	85.9%	77.4%	81.9%	73.0%
1%	84.6%	74.7%	78.3%	67.1%
2%	84.3%	73.9%	78.3%	67.0%
3%	86.0%	76.5%	77.8%	66.8%
5%	85.3%	75.4%	74.5%	66.3%

V. CONCLUSION

A major challenge in real-time teleoperated driving is the presence of time delay, meaning the teleoperator view is not up-to-date. One approach to mitigate this is the use of a predictive display, which provides guiding visuals to provide a more accurate view of the environment. In this work, we test our predictive display, the collision-free path, using CARLA simulator. We observe the accuracy of our predicted free space with respect to different values of added delay and packet loss. When no traffic is present, our model shows similar results across different network conditions. On the other hand, introducing traffic to the environment causes a decrease in model accuracy as we increase the time delay or packet loss. Thus, we claim that the relevance of studying these conditions is in the dynamic road conditions. In other words, even when the network inconsistencies are small, moving obstacles in the environment can cause significant discrepancies. We define our threshold as follows: communication delay must be below 100ms, and packet loss must be below 1%. Both thresholds are reasonable for vehicle teleoperation, as advancements in streaming and network capabilities are able to support these requirements. While this work examined how network conditions can objectively affect the video feed, future work will assess usability with a teleoperator in real time.

REFERENCES

[1] D. Schitz, G. Graf, D. Rieth, and H. Aschemann, "Corridor-based shared autonomy for teleoperated driving," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15368–15373, 2020.

[2] D. Schitz, S. Bao, D. Rieth, and H. Aschemann, "Shared autonomy for teleoperated driving: A real-time interactive path planning approach," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 999–1004, IEEE, 2021.

[3] M. E. Walker, H. Hedayati, and D. Szafir, "Robot teleoperation with augmented reality virtual surrogates," in *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 202–210, IEEE, 2019.

[4] J. Prakash, M. Vignati, E. Sabbioni, and F. Cheli, "Vehicle teleoperation: Successive reference-pose tracking to improve path tracking and to reduce time-delay induced instability," in *2022 IEEE Vehicle Power and Propulsion Conference (VPPC)*, pp. 1–8, IEEE, 2022.

[5] H. Chen, P. Huang, and Z. Liu, "Modeling and forecasting of time delay about the space robot teleoperation system," in *2019 IEEE 4th International Conference on Advanced Robotics and Mechatronics (ICARM)*, pp. 529–534, IEEE, 2019.

[6] Z. Liu, Z. Lu, Y. Yang, and P. Huang, "Teleoperation for space manipulator based on complex virtual fixtures," *Robotics and Autonomous Systems*, vol. 121, p. 103268, 2019.

[7] G. Graf, H. Xu, D. Schitz, and X. Xu, "Improving the prediction accuracy of predictive displays for teleoperated autonomous vehicles," in *2020 6th International Conference on Control, Automation and Robotics (ICCAR)*, pp. 440–445, IEEE, 2020.

[8] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 10012–10022, 2021.

[9] F. Pizzati and F. García, "Enhanced free space detection in multiple lanes based on single cnn with scene identification," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 2536–2541, IEEE, 2019.

[10] D. Qiao and F. Zulkernine, "Drivable area detection using deep learning models for autonomous driving," in *2021 IEEE International Conference on Big Data (Big Data)*, pp. 5233–5238, IEEE, 2021.

[11] H. Asgarian, A. Amirkhani, and S. B. Shokouhi, "Fast drivable area detection for autonomous driving with deep learning," in *2021 5th International Conference on Pattern Recognition and Image Analysis (IPRIA)*, pp. 1–6, IEEE, 2021.

[12] Y. Bie and H. Tan, "Image recognition in autonomous driving based on improved swin transformer," in *2022 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, pp. 124–129, IEEE, 2022.

[13] N. Mehajabin, Z. Ma, Y. Wang, H. R. Tohidypour, and P. Nasiopoulos, "Real-time deep learning based road deterioration detection for smart cities," in *2022 18th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pp. 321–326, IEEE, 2022.

[14] F. Kashwani, B. Hassan, M. Khonji, and J. Dias, "Collision-free path generation for teleoperation of unmanned vehicles," in *2023 21st International Conference on Advanced Robotics (ICAR)*, pp. 21–27, 2023.

[15] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 1–16, 2017.

[16] M. Hofbauer, C. B. Kuhn, G. Petrovic, and E. Steinbach, "Telecarla: An open source extension of the carla simulator for teleoperated driving research using off-the-shelf components," in *31st IEEE Intelligent Vehicles Symposium 2020 (IV)*, (Las Vegas, NV, USA), pp. 1–6, IEEE, Oct 2020.

[17] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[18] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2636–2645, 2020.

[19] J. Fritsch, T. Kuehnl, and A. Geiger, "A new performance measure and evaluation benchmark for road detection algorithms," in *International Conference on Intelligent Transportation Systems (ITSC)*, 2013.