10-1-2023

Structure-aware image translation-based long future prediction for enhancement of ground robotic vehicle teleoperation

Md Moniruzzaman
Edith Cowan University

Alexander Rassau
Edith Cowan University

Douglas Chai
Edith Cowan University

Syed M. S. Islam
Edith Cowan University

10.1002/aisy.202200439

Moniruzzaman, M., Rassau, A., Chai, D., & Islam, S. M. S. (2023). Structure-aware image translation-based long future prediction for enhancement of ground robotic vehicle teleoperation. Advanced Intelligent Systems, 5(10), article 2200439. https://doi.org/10.1002/aisy.202200439

Structure-Aware Image Translation-Based Long Future Prediction for Enhancement of Ground Robotic Vehicle Teleoperation

Md Moniruzzaman, Alexander Rassau, Douglas Chai, and Syed Mohammed Shamsul Islam*

Predicting future frames through image-to-image translation and using these synthetically generated frames for high-speed ground vehicle teleoperation is a new concept to address latency and enhance operational performance. In the immediate previous work, the image quality of the predicted frames was low and a lot of scene detail was lost. To preserve the structural details of objects and improve overall image quality in the predicted frames, several novel ideas are proposed herein. A filter has been designed to remove noise from dense optical flow components resulting from frame rate inconsistencies. The Pix2Pix base network has been modified and a structure-aware SSIM-based perpetual loss function has been implemented. A new dataset of 20 000 training input images and 2000 test input images with a 500 ms delay between the target and input frames has been created. Without any additional video transformation steps, the proposed improved model achieved PSNR of 23.1; SSIM of 0.65; and MS-SSIM of 0.80, a substantial improvement over our previous work. A Fleiss' kappa score of >0.40 (0.48 for the modified network and 0.46 for the perpetual loss function) proves the reliability of the model.

1. Introduction

Predicting and reasoning about future events is the core of the decision-making process and the essence of intelligence.[1] Anticipating, predicting, and synthesizing an image frame or frames from either a single input or a sequence of images is referred to as future frame or future video prediction. Future frame prediction frameworks have been presented as representation learning.[2,3] The future frame or video prediction is conditioned upon previously learnt feature representations from a set of previous frames, unlike conventional video generation problems which are predominantly unconditional.[4] In future frame prediction problems, the target image frame works as a label. Therefore, future video prediction needs to be dealt with through a supervised learning approach. The label information need not to be provided as an additional channel, and no external supervision is needed, as it is already available in the input frame sequence. Therefore, in practical terms, conventional frame prediction is a self-supervised task. The current approaches try to fill the gap between supervised and unsupervised learning techniques. Although future frame or video prediction is a relatively new research domain, a significant amount of work has been done to attempt to predict future frames. The aim of this article is not to provide a detailed literature review on future video prediction techniques. There are a number of recent survey works that have been published for that purpose such as.[4–6] Instead of categorizing all of the video prediction techniques into different classes, they can more easily be represented under the set of distinct paths as illustrated in **Figure 1**.

Future frame prediction is critical for a number of application areas including video understanding,[7] video interpolation,[8] video captioning,[9] anticipating pedestrians' intentions,[10] action recognition,[11] driverless car technology,[12] predicting events and activities,[13,14] and anomaly detection[15] among others. However, the first indication of the possible use of future frame prediction for robotic teleoperation enhancement was suggested in.[16] In our later work[17] we demonstrated that the notion of future frame prediction and future video generation can directly enhance control over ground robot teleoperation. This article provides further support for this notion and improvement of the state-of-the-art techniques to generate future frames.

The primary challenge with long-distance robotic teleoperation is the impact of latency in the communication channel. For a robotic teleoperation system that uses 2D camera-based

M. Moniruzzaman, A. Rassau, D. Chai
School of Engineering
Edith Cowan University
Joondalup, WA 6027 Perth, Australia
E-mail: m.moniruzzaman@ecu.edu.au

S. M. S. Islam
School of Science
Edith Cowan University
Joondalup, WA 6027 Perth, Australia

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/aisy.202200439.

DOI: 10.1002/aisy.202200439

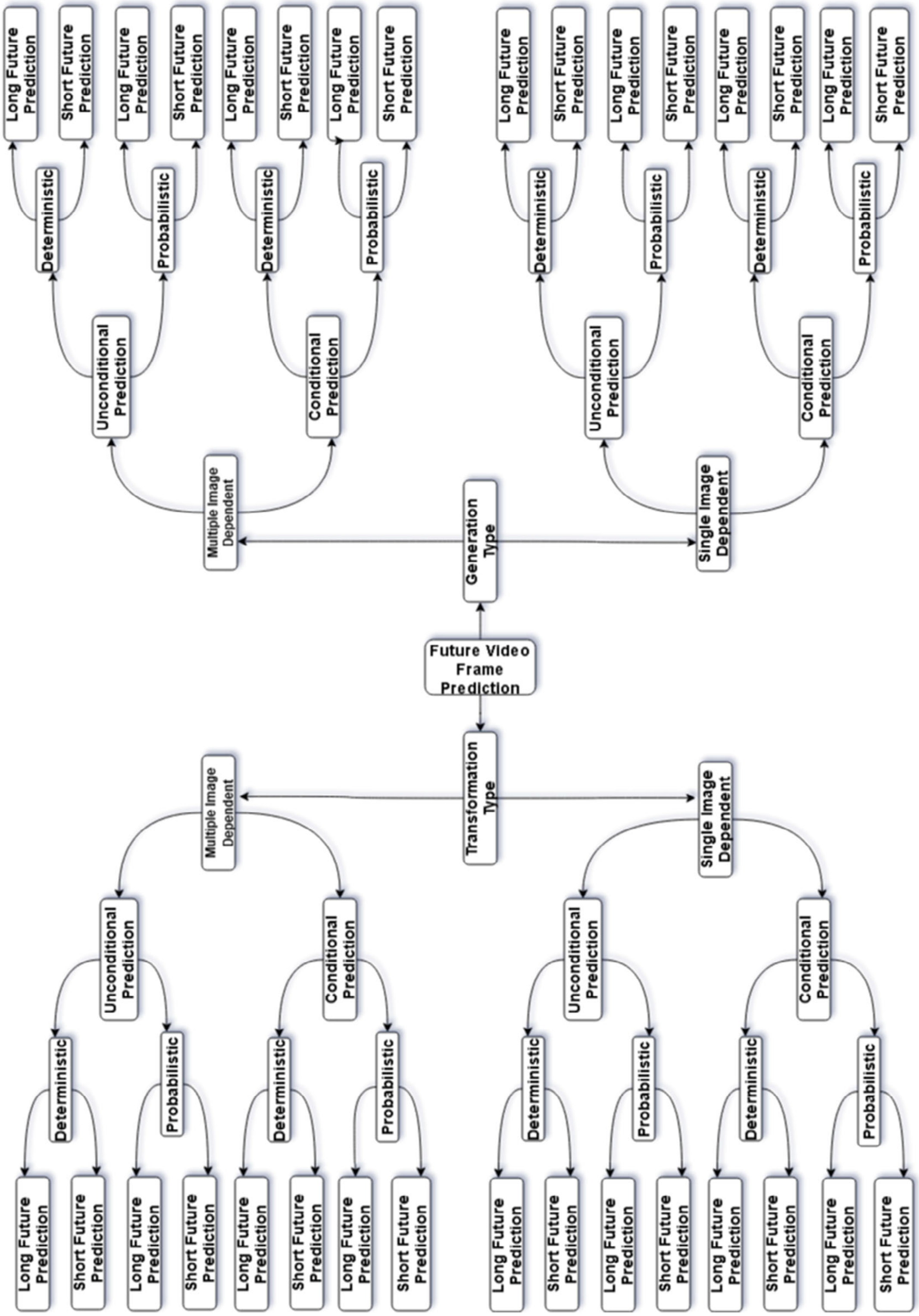


Figure 1. Set of future frame prediction techniques.

visual feedback for situational awareness, whether the delay is in the video feed transfer channel, in the control signal transmission channel, or, as is generally the case, distributed across both, the impact of latency to a teleoperator is the same.[16] Teleoperators can notice latency even if it is as low as 20 ms. MacKenzie and Ware[18] reported that for latencies of only 225 ms, operator action time increased by 64% and the error rate increased by more than double. Some studies (such as refs. [19,20]) claim a teleoperator's ability to track and manipulate objects that can be severely compromised by latency if it reaches 300–320 ms.[21] While driving at high speeds up to 90 km h^{-1}, vehicle control has been shown to degrade significantly once latencies exceed 170 ms.[22] Almost all of the ground vehicle robotic teleoperation tasks reported in the literature are for slow-moving (<10 km h^{-1}) unmanned ground vehicles (UGVs). To more accurately judge impacts on UGV teleoperation in a wider range of use cases, in our previous work we experimented with higher speed (10–25 km h^{-1})-simulated UGVs and found that at these ground speeds, task completion time increased up to 200% with a latency of 900 ms. Once latency reached 1200 ms, we found that overcorrection-related oscillations made teleoperation effectively impossible even at these relatively modest speeds, demonstrating a need for effective latency compensation methods.

Attempting to predict future state and navigation path,[23] field of view change,[24] and possible collision avoidance[25] has been used by the robotic research community to aid the operators affected by latency. However, these first-order predictive technologies are not capable of overcoming the challenges of long-distance and high-latency ground robotic vehicle teleoperation at reasonable ground speeds. Therefore, in our previous work,[17] we hypothesized that instead of discrete first-order state prediction, a continuous prediction and generation of long future video from delayed past frames could provide the operators with a constant flow of information about the remote environment that closely mimics the present frames captured by the robot-mounted camera sensor provided the state of the vehicle and operator control signals are incorporated in the prediction. As a proof of concept, we demonstrated the generation of future frames from frames delayed by 500 ms and achieved promising results. Conventional long future frame prediction techniques use multiple past frames to generate multiple future frames. For a live teleoperation control loop, this technique would require a buffer state where the required number of past frames is held and then fed into the deep neural network for prediction. This would add further delay to the communication loop, compounding the problem instead of enhancing teleoperation. Therefore, our approach is to use image-to-image translation to continuously predict single frames deep into the future based on single past frames. To improve the ability of the network to predict the matching predicted frames, we also incorporate optical flow information to a conditional generative adversarial network (cGAN). This optical flow information provides an indication of the vehicle state and any changes in this state based on operator control inputs. This previous work provided a positive indication that the proposed image-to-image translation-based long future frame prediction approach has merit as a possible solution to the challenges of high-latency long-distance teleoperation. However, the final predicted video missed a lot of structural details of the remote scene that would be important for teleoperating a UGV at reasonable ground speeds.

This article further develops this future frame prediction approach and improves the image quality of the predicted frames by maintaining better scene structural integrity and reducing loss without affecting the performance and prediction speed compared to the base cGAN network. The primary enhancement methods were the application of a filter to reduce input data noise, changing the loss function to a structural-aware perpetual loss function and finally modifying the base cGAN network to preserve more structural information to the decoder end of the network. We have created a new dataset for the research described in this article using the simulator described in refs.[16,17]. A number of training and frame generation sessions have been performed. For image quality measurement and comparison, we have used well-recognized image comparison metrics including the peak signal-to-noise ratio (PSNR), the structural similarity index measure (SSIM), and the multiscale structural similarity index measure (MS-SSIM). To compare the reliability of the different combinations of the losses and the network architectures, we have calculated the Fleiss' kappa, P values, standard deviations, skewness, and kurtosis. Our enhanced techniques significantly improve the predicted future video quality and are fast enough to be implemented in a real-time teleoperation control system. The main contributions of this article are as follows. 1) Development of a filter to remove frame rate disparity-induced noise and generation of a new noise-free virtual ground vehicle teleoperation dataset using the simulator from ref. [16]. 2) Formulation of a unique SSIM-based perpetual loss function for the generator training that provides significant benefits in maintaining the structural integrity of the predicted future frames. 3) Modification of the base Pix2Pix cGAN network architecture to preserve the supporting optical flow information and prevent it from suffering loss during the down-sampling and up-sampling operations of the generator and offering better structural integrity preservation to the prediction task. 4) In-depth evaluation and discussion of the outcomes of our proposed system and their implications for future teleoperation enhancement research.

The rest of the article is structured as follows: Section 2 offers background about the current state-of-the-art image-to-image translation techniques and their prospects for use in future frame prediction-based teleoperation enhancement. Section 3 discusses the research methodology with detailed information about the data collection and filtering with our custom filter, the new perpetual loss function, the modified cGAN network, and the evaluation methods of the new techniques. Section 4 presents the results and findings of the experiments and Section 5 discusses these findings. Section 6 concludes the article and provides recommendations for future research directions.

2. Background

Future video prediction techniques can be utilized to generate a continuously predicted future video feed from the incoming delayed frames sent by the remote robot/vehicle camera. However, achieving high-quality synthetic video feeds will require extended research in the domain. One of our previous works[6] lists some of the recent and promising future video prediction approaches that can be used as a baseline for creating future technologies for robotic teleoperation enhancement.

State-of-the-art and contemporary future image frame prediction models require multiple input frames (past frames) to predict one or more future frames. For these required multiple past frames, these techniques require a buffering time, and thus, the future frames need to be predicted even further into the future to offset this additional latency, greatly increasing the challenge. Further, these models are not suitable for a system where a constant influx of past frames is expected, and a continuous stream of predicted future frames is required. A high-latency, but real-time ground vehicle teleoperation system provides a delayed but constant influx of frames that could be low in quality and the frame rates may vary over time. We assume that the bulk of the information required to generate a future frame within a reasonable time horizon is present in the past frames. Therefore, we hypothesize that if a series or influx of past frames can be translated to image frames that are close enough to the original ground truth (present time frames), this would solve the issues caused by latency in the communication loop. For a visual feed-based human in the loop ground vehicle teleportation scenario, which is similar to the simulator and experimental setup designed by us, the aim is to design a future frame prediction model that can translate a delayed frame to a new one that is reasonably close to the present frame or ground truth. Therefore, in contrast to conventional future frames prediction problem, for latency reduction and teleoperation enhancement, we believe the problem is better handled as of an image-to-image translation problem.

2.1. Image-To-image Translation Networks

The application of GANs for image-to-image translation has had some admirable success. Unconditional GANs use techniques such as L2 regression to force translate the input to match a target image. Some of the examples of unconditional GANs for image-to-image translations are: for future state prediction,[26] style transfer,[27] inpainting,[28] super-resolution,[29] and SAR image translation[30] among others. However, GANs in a conditional setting are more suited to the problem domain of data or image translation[31] because a conditional GAN can condition an input image to generate a corresponding target image. There are several attempts that have used conditional GANs to predict images from a provided map,[32] generate images from sparse annotations,[33,34] predict future frames,[35] generate synthetic product photographs,[36] and perform simple but versatile image-to-image translation.[31]

For integrating conventional GAN-based future video prediction networks to a real-time ground vehicle teleoperation system, an intermediate buffer image frame loader is required, where the required set of frames will arrive and be held before being passed to the neural network for future frame prediction. This intermediate buffer stage will add more latency to the system. Moreover, when multiple future frames are being generated by a network, the image quality of the frames in a single batch varies significantly and degrades for later frames on a single batch. This creates an uneven image quality event in the predicted video stream. To avoid these issues the neural network should be able to translate a single incoming image frame to a single future frame deep into the future with an acceptable level of image quality (in terms of both pixel and structural similarity) in a first-in-first-out (FIFO) fashion. This network also needs to be fast enough so that a relatively smooth video output can be produced and provided to the teleoperators. This video stream will be the primary source of situational awareness for the operators. In real-life situations, remote robotic environments are mostly dynamic and have varying scene conditions, so the chosen network needs to be robust enough to translate varying scene conditions with a higher degree of uncertainty.

Considering the constraints and limitations of the conventional future frame prediction networks, an image-to-image translation network is more suited to fulfill the requirements of a visual feed-based telerobotic system and compensate for the latency in the control loop. However, finding a suitable image-to-image network requires careful consideration. **Figure 2** provides a taxonomy of the state-of-the-art image-to-image translation networks. All of the contemporary image-to-image translation networks can be segregated as either supervised or unsupervised image translation networks.[37]

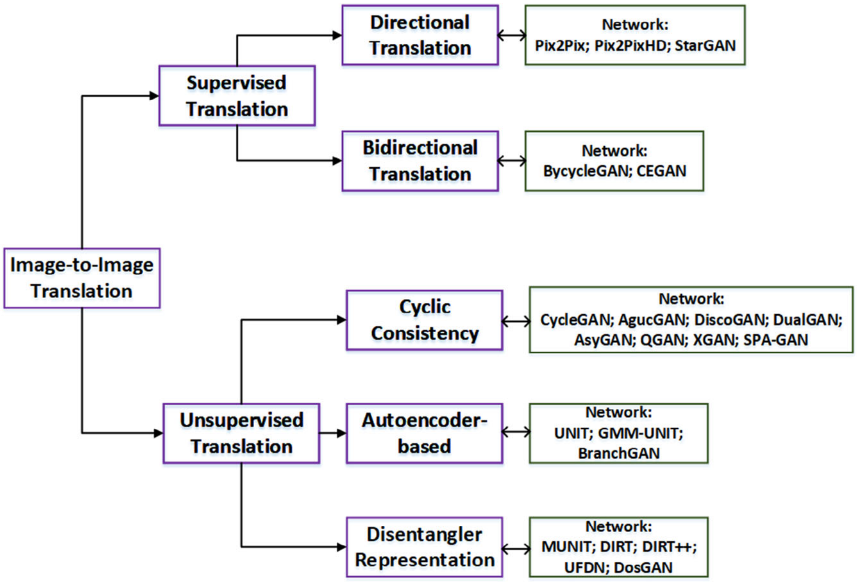


Image-to-Image Translation
Supervised Translation
Directional Translation
Network: Pix2Pix; Pix2PixHD; StarGAN
Bidirectional Translation
Network: BycycleGAN; CEGAN
Unsupervised Translation
Cyclic Consistency
Network: CycleGAN; AgucGAN; DiscoGAN; DualGAN; AsyGAN; QGAN; XGAN; SPA-GAN
Autoencoder-based
Network: UNIT; GMM-UNIT; BranchGAN
Disentangler Representation
Network: MUNIT; DIRT; DIRT++; UFDN; DosGAN

Figure 2. Taxonomy of image-to-image translation techniques (adapted from ref. [37]).

The CycleGAN[38,39] (cyclic consistency-based), DosGAN[40] (disentangler representation-based), and UNIT[41] (autoencoder-based) are unsupervised image translation networks. Unsupervised image translation networks are suited for unpaired translation between two or more domains and show a high level of uncertainty and thus are not suitable for the use case described in this article where input (delayed) and ground truth (real-time) image pairs are accessible for supervised learning.

The supervised image-to-image translation networks are categorized into directional and bidirectional image translation networks.[37] BicycleGAN[42] and CEGAN[43] are the bidirectional networks. These networks produce diverse outputs by multimodal cross-domain translation. For our high-latency teleoperation enhancement use case, we need a single future frame as an output, dictated and constrained by a real-time ground truth target image. Therefore, we are left with only the directional supervised translation networks. Currently there are three directional image-to-image translation networks: StarGAN,[44] Pix2Pix,[31] and Pix2PixHD.[45] The StarGAN[44] network was designed to mimic facial expressions such as happy, angry, and fearful on CelebA data. This network performs multidomain image translation. Unlike the remote environment of teleoperated ground vehicles, the CelebA dataset does not have the kinds of dynamic scene changes and the StarGan is not robust enough to handle the uncertainty of objects and their movements in consecutive image frames. Further, the multidomain image output does not match our requirements. The Pix2PixHD[45] network is designed to generate HD images (2048 × 1024) from semantic label maps, which is also not in line with our use case due to the bandwidth constraints of remote teleoperation. Therefore, the only network that is potentially suitable for experimentation with as a starting point to translate dynamic delayed images into future frames is the Pix2Pix cGAN.

Considering the factors discussed above, we have experimented with the Pix2Pix[31] as a base network for our future frames prediction task in ref. [17]. We have customized the network to accept the angle and magnitude components of dense optical flow as additional channels with the optical flow represented in the form of Red, Green, and Blue (RGB) channels. If a single image is fed to this network, it is capable of learning to translate the input to an output that is conditioned to a predefined target image. The base network is a cGAN that has a U-Net generator and a feed-forward discriminator. A down-sampler (encoder) and an up-sampler (decoder) with skip connections make up the U-Net architecture of the generator. The encoder and decoder structure creates a large bottleneck. However, the use of skip connections helps to reduce low-level information loss during the down-sampling and up-sampling operations. If the total number of layers is n for the generator, the skip connections are between layer i and layer $n - i$. The network is fast enough to translate and generate future frames such that a video stream can be produced in real time from the output frames.

The discriminator of the base network is a feed-forward patchGAN. This discriminator penalizes the generator output at the scale of patches in the image by classifying whether the generated image patches are real or fake. To provide an output the discriminator operates conventionally across the entire predicted image. As the patch sizes are small and have fewer parameters due to the reduced size, the discriminator runs fast. The discriminator and the generator of the base network have been adapted from ref. [46]. For both the generator and discriminator the layer formation is convolution-batch normalization-ReLu. Figure A1 and A2 in the Appendix section provide a schematic diagram of the generator and the discriminator respectively. We also have made changes to the generator architecture; these architecture changes are discussed in Section 3.2.2.

3. Methodology

Our previous work[17] has provided initial proof of concept for our idea that the cGAN-generated synthetic predicted frames have the potential to compensate for latency in the ground vehicle teleoperation loop. However, UGVs operating at reasonable ground speeds need better video feeds than were achieved in this initial work for effective and smooth teleoperation. This article intends to improve the quality of the predicted video feed with better preservation of the structural integrity of objects in the scene based on changes to the network architecture and a new perpetual loss function. The research presented in this article can be divided into four stages: a new dataset creation, implementation of the revised perpetual loss function, modification to the network architecture, and evaluation of the new techniques. All of these stages are described in detail in the section below. **Figure 3** provides an illustration of all of the stages of the research using a flow diagram.

3.1. Dataset Creation

3.1.1. Why We Needed a New Dataset?

In our previous work described in ref. [17], we have created three different datasets (Forza_GT + UE + Std_D, Forza_GT + UE + VT_DL, and Forza_GT + UE + Forza_DL) and tested different aspects of the proposed future video prediction-based teleoperation enhancement techniques. Therefore, a valid question is why a new dataset is needed for this current research where we intend to improve the image quality of the predicted frames. All of the above-mentioned datasets created for ref. [17] were recorded

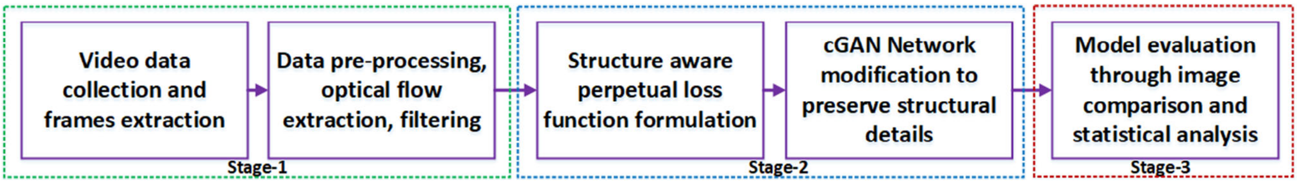

Video data collection and frames extraction
Data pre-processing, optical flow extraction, filtering
Stage-1
Structure aware perpetual loss function formulation
cGAN Network modification to preserve structural details
Stage-2
Model evaluation through image comparison and statistical analysis
Stage-3

Figure 3. Flow diagram of the proposed methodology.

using the Open Broadcaster Software (OBS) with a frame rate of 30 frames per second (fps). However, the frame rates of the individual feeds (the ground truth and the delayed feed) are different than that used for OBS and also different from each other due to performance limitations in the Simulink implementation. In the teleoperation simulator, the delayed frame rate varies within a range of 16–20. Additionally, due to variations in computational overhead, the frame rate of the UE cosimulation also varies from 14 to 18 fps. In a real-life video feed-based teleoperation session, frame rate variations and disparity could also be expected and would be considered normal for high latency and low-bandwidth communication channels. This does, however, create a significant issue for neural network-based future frame prediction methods. When the synchronized recorded video of these different frame rate feeds is converted into individual frames, the transition between sequential frames is not consistent for the ground truth relative to the delayed frames. The relative changes for the ground truth frames are smoother and more uniform due to the ground truth frame rate being 60 fps, double the OBS recording frame rate. However, for the delayed frames, the changes are irregular, which introduces noise when extracting the optical flow from consecutive frames for the cGAN training task. **Figure 4**a shows that the frame rate disparity creates irregular changes from one frame to the next, and thus the extracted dense optical flow exhibits significant noise. Reducing the frame rate of the OBS recording to 10 fps ensures that the recorded video contains fewer transition states of the delayed video frames, resulting in less noise in the frames converted from the recorded video and, subsequently, in the extracted optical flow components. It should be noted that simply extracting every third frame from the previous data would not solve the noise issue since noisy frames and the resulting optical flow components are already present in the dataset due to the frame rate disparities during capture. In Section 3.1.3, we have presented a filtering technique that can eliminate this type of noise in the data. However, when we applied this technique to the existing data, we found that it significantly shrank the dataset size and created an irregular scene-shifting effect in the cleaned dataset. As a result, a network trained on such a dataset would produce irregular future frames, thereby impacting the quality of the predictions. To overcome these problems, a new dataset was needed where the amount of noise can be substantially reduced, and the transitions between the frames were kept regular for both the delayed and ground truth feeds. To achieve this we have reduced the frame rate of the video data recording for our new dataset as well as applying the developed filter to eliminate any frames containing excessive noise.

3.1.2. Video Data Collection

In our initial attempt of using future video prediction to compensate for latency in ground vehicle teleoperation, reported in ref. [17], we have experimented with conventional three-channel RGB image frames and optical flow incorporated via five-channel image data that included optical flow from delayed frames, transformed frames (transformed to offer a predicted point of view dictated by the live operator control input), real-time frames, and unreal engine-generated synthetic optical flow directly reflecting the operator control inputs in real time. Therefore, for this research, our primary goal is not to find a better source of optical flow, but rather to improve the predicted video quality

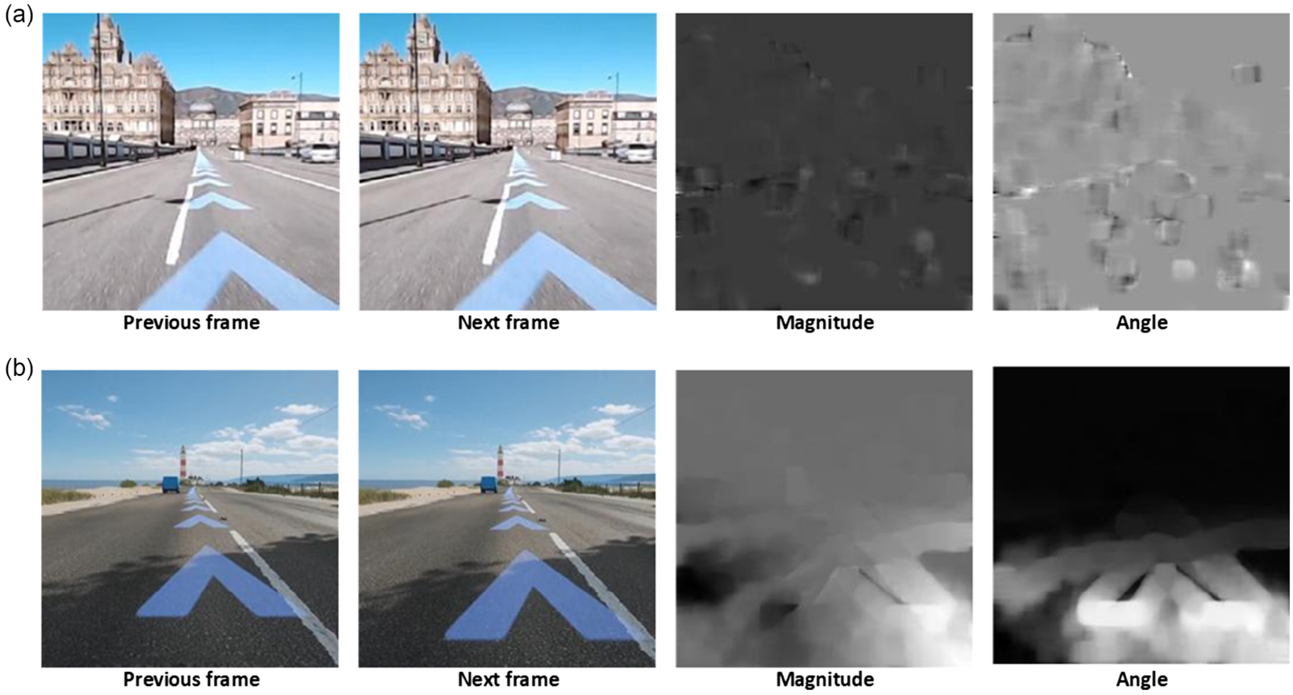


Figure 4. a) The noise in the angle and magnitude due to the frame rate issue in the teleoperated delayed frames. b) A pair of consecutive frames with proper magnitude and angle component after dense optical flow extraction. Our filter (Algorithm 3.1.2) keeps the frames like (b) and removes the frames like (a) to provide a noise-free input stream to the future prediction model.

for a system that does not require an additional video transformation unit to support the future prediction; however, it still performs better with improved image quality. If this improvement can be achieved, then the neural network-based future prediction model can be integrated into any long-distance and high-latency systems without needing to rely upon additional and computationally bulky supportive systems.

For the data collected for the research described in this article, we have used the same teleoperation simulator described in ref. [17]. However, instead of three different settings, we have only used one where the ground truth is the Forza game screen, and the delayed feed is the nontransformed delayed feed where the raw delay of 500 ms is added to the original feed. No data was recorded from the transformed video feed. A detailed description of the simulator and the human teleoperator involved data collection process is described in ref. [17]. Similar to the previously collected data, we have used the OBS software for the synchronized video data recording process. We have capped the data recording frequency at 10 fps. A number of video recordings have been collected and converted into individual frames.

3.1.3. Filter Design and Pre-Processing

Once the video recording and frame extractions are done, the individual frames are cropped into corresponding segments (the ground truth frames and the delayed frames). These separated frames are then resized to 256×256 pixels to reduce the computation time during the neural network training. For this dataset, we have created 20 000 images as the ground truth, which are the segments from the 4 K Forza real-time gaming screen, and the corresponding 20 000 delayed image frames. These delayed frames (500 ms delayed) have a low frame rate and low pixel quality to mimic real teleoperation through low-bandwidth and high-latency communication channels. These 20 000 image sets are used as the training set, where the ground truth images have been used as the target and the delayed frames have been used as the input to the neural network. The test set consists of 2000 images of the ground truth and the delayed versions. We have named this latest data as " Forza_GT + Std_DL."

Moniruzzaman et al.[17] provide good evidence that instead of only using the three-channel RGB input, adding optical flow components (angle and magnitude) as two additional channels to the input offers much better future prediction. Therefore, similar to ref. [17], we have extracted optical flow for the delayed frames of both the training and testing tests. We have extracted the dense or per-pixel optical flow from two consecutive frames through the Gunner–Farneback[47] method. However, due to the difference in the frame rates among the ground truth and delayed feed, while extracting the optical flow, the synchronized recorded segments generate noise similar to the examples shown in Figure 4a. To eliminate this noise from the dataset, we have used a MS-SSIM comparison-based filter. Algorithm 3.1.2 provides the steps and the details of the filter. As the OBS recording frame rate and the delayed frame rate are not the same, there are instances where the OBS records two consecutive frames in a stage where a previous frame (frame n) has not transitioned to the next frame (frame $n+1$). Therefore, the next frame is almost identical to the previous frame. While extracting the angle and magnitude components of the per-pixel dense optical flow, the optical flow extractor algorithm thus generates noise. Our filter (**Algorithm** 1) compares the previous and next frames and if the similarity is more than a threshold ($\alpha = 0.95$), the filter removes the previous frames from the delayed, ground truth, the angle, and the magnitude components set. This leaves a set of frames that has a sequence of image frames that have a sufficient per pixel information gap so that the optical flow generator can generate meaningful dense optical flow. This filter works on a sequential influx of frames, which makes it capable of being used in a live system. The reduction of frames from the feed also reduces the computational burden in the later training and prediction stages. For our use case, this filter was able to reduce more than 95% of the noisy angle and magnitude components. By changing the threshold (α) value, this filter can also be used to eliminate intermediate frames to reduce the computational burden further without sacrificing the image quality of the predicted future frames.

3.2. Structure-Aware Training

3.2.1. Perpetual Loss Integration

In conditional GAN training, the loss function penalizes and adjusts the learning of the network for a possible output that is different than the target. In the GAN, the loss is calculated for both the discriminator and the generator. For our network, the discriminator loss consists of the real_loss and the generated_loss. The sigmoid cross entropy of the target image and

Algorithm 1. Algorithm to Filter the Forza GT + Std DL Data Set to Avoid Frame Rate Disparity Related Noise Issue.

```
Require: Filtered dataset = f (Ground truth frame, Delayed frame, Optical flow
                              components)

Ensure: Ground truth = Real-time frames from Forza game screen;
Ensure: Delayed frames = Delayed frames by the simulator;
Ensure: Optical flow components = Magnitude and angle components of the dense
        optical flow extracted from the delayed frames;
while the delayed frames are passed through the filter do
    $Previous_{frame}$ = rgb2gray (read the frame $n$);
    $Next_{frame}$ = rgb2gray (read the frame $n+1$);
    Similarity = MS-SSIM (Previous$_{frame}$, Next$_{frame}$);
    $\alpha = 0.95$;
    if then $Similarity \geq \alpha$
        Remove $n$th ground truth frame;
        Remove $n$th delayed frame;
        Remove $n$th Angle component;
        Remove $n$th magnitude component;
    end
    Filtered dataset = Sort & rename (Filtered ground truth, Filtered delayed
        frame, Filtered Optical flow);
    Result: Dataset that filters out more than 95% irregular frame rate issue
        induced irregular changes in frame and noise in optical flow components
```

an array of ones provide the real_loss. The generated_loss comes from the sigmoid cross entropy of the generator output (predicted synthetic image) and an array of zeros. The discriminator loss of the network can be represented as

$$\text{discriminator\_loss} = \text{real\_loss} + \text{generated\_loss} \tag{1}$$

During training, the discriminator ignores the generator loss and classifies the generator predicted synthetic images based on the discriminator loss. For predicting and generating synthetic future frames, the generator loss plays a more significant role and modifications to the generator loss function can have a direct impact on the quality of the synthetic image. For our previous article,[17] the generator loss function was

$$\text{generator\_loss} = \text{gan\_loss} + (\lambda \times \text{L1}) \tag{2}$$

where the gan_loss is the sigmoid cross entropy of the generated synthetic image and an array of ones. The value of the λ was kept at 100. The L1 loss is the mean absolute error (MAE) between the target image and the generator output. This loss function does not account for the structural integrity of the predicted future frames. To improve the image quality and create better structural similarity we have introduced an SSIM-based perpetual loss function. This loss function can be expressed as

$$\text{generator\_loss\_perpetual} = \text{gan\_loss} + (\lambda \times (1 - \text{SSIM})) \tag{3}$$

Here, SSIM is the structural similarity index of the target and the predicted images. We have also experimented with a combination of MAE and perpetual loss. This combined loss function can be expressed as

$$\begin{aligned}\text{generator\_loss\_mixed} = \text{gan\_loss} + (\lambda \times ((1 - \beta) \times \text{L1} \\ + \beta \times (1 - \text{SSIM})))\end{aligned} \tag{4}$$

where β is an adjustable weight parameter. For our experiment, we kept the value to 0.50.

3.2.2. Network Architecture Modification

In our previous work[17] on future frame prediction, we have used the Pix2Pix[31] architecture as the base network and customized it to accept our five-channel data input. The generator of this network is a U-net[48] with skip connections (Figure A2). This generator consists of an encoder (down-sampler) and a decoder (up-sampler). For the n layers in the network, the skip connection joins the i and $n - i$ layers to reduce low-level information loss between the encoder and the decoder. Along with the three-channel delayed RGB frames, our future frames prediction also incorporates the angle and magnitude optical flow components. This additional optical flow information is intended to afford better prediction by providing the network with information on temporal changes in the scene; however, due to the nature of the encoder and decoder-based neural network architecture, this additional single-channel information can easily get lost in down-sampling and up-sampling operations. Therefore, to preserve the structural integrity of the angle and magnitude information, and thus maintain better structural integrity between the ground truth and the predicted synthetic future frame, we have modified the architecture to incorporate these two additional pieces of information without any distortion immediately prior to the prediction layer of the network. **Figure 5**a shows the old architecture and (b) shows the updated architecture. We have added two new Conv2D layers (Sequential D8 and Sequential D9) after the sequence of decoder layer Sequential D7 (Upsampler layer 7) (Figure 5b). Before passing the output of block D8 to D9, we have separated the angle and magnitude components from the input and bypassed them with a skip connection to the concatenation layer C8 (Figure 5b), so that the optical flow information, which provides a representation of the control input from the operator, does not get lost through the bottleneck of the encoder and decoder and can have more influence on the prediction of the future frame. The complete architecture of the modified generator has been provided in Figure A3.

The discriminator of the complete network is a patchGAN that works as a classifier to distinguish between the predicted synthetic frames and the target. It penalizes the predicted output at the scale of patches. The sizes of the patches are small, meaning that the discriminator network has to deal with a small number of parameters and thus runs faster. The layer formations of the blocks of the discriminator are convolution-batch Normalization-ReLu. Figure A2 provides a schematic diagram of the discriminator used.

3.2.3. Training Specifications

To maintain fairness in the comparison of the recent work with our previous works, we have kept the training specifications similar. We have used two desktop computers for all the training sessions for our dataset: a 48 GB NVIDIA RTX A6000 GPU machine and a 24 GB NVIDIA GeForce TITAN RTX GPU machine. The training data have 20 000 (20 K) delayed input RGB images, 20 K dense optical flow components, and 20 K target ground truth images. We have set the training buffer size to 20 k so that for every step of training, the whole dataset gets shuffled. The batch size has been kept at 1. For every training session, we have trained the network for 2 000 000 steps or 100 epochs. Checkpoints are saved for every 20 000 training steps. For optimizing the training, we have used the Adam optimizer.[49] To find a suitable combination of the network and the loss function for better structural integrity preservation for the predicted future frames, we have experimented with an L1 loss function (2), an SSIM-based perpetual loss function (Equation (3)), and a combination of both (Equation (4)) through a total of seven training sessions.

In a GAN, during training, the generator works to enhance its samples by minimizing the discrepancy between the fake samples it creates and the real samples in the dataset. Meanwhile, the discriminator tries to maximize the difference between the real and fake samples by learning to correctly classify them. This leads to a scenario where the discriminator loss decreases as it becomes more proficient at distinguishing real and fake samples, while the generator loss increases as it endeavors to generate better samples that can deceive the discriminator. The ultimate objective of GAN training is to reach a point where

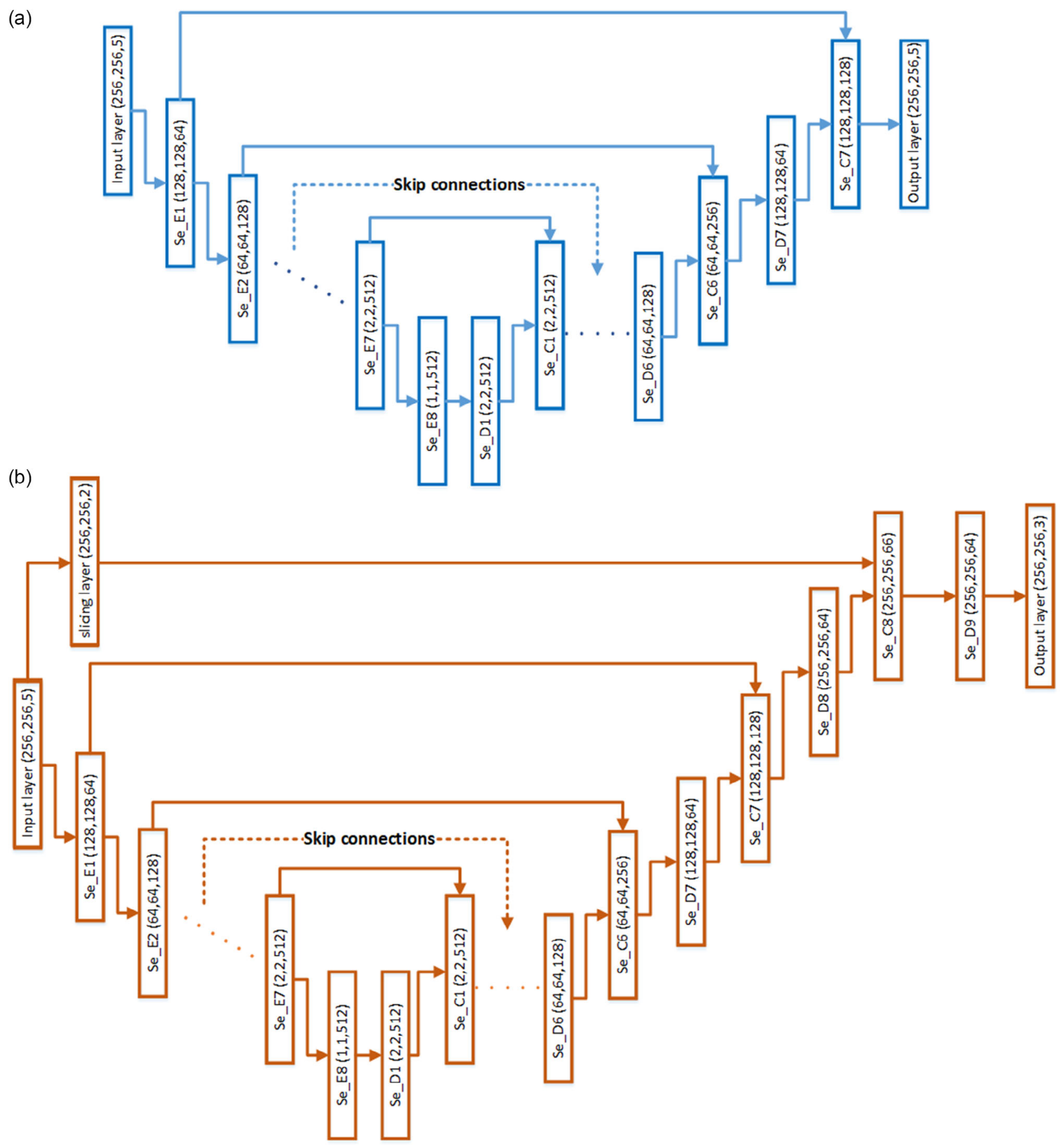


Figure 5. a) The generator U-Net architecture used in ref. [17] for predicted future frames generation. b) The generator network used in this article after modification to preserve structural details while generating future frames.

the generator produces samples that are indistinguishable from real samples, and the discriminator is unable to differentiate between real and fake samples. When this is achieved, the GAN is considered to have converged, and the loss values for both the generator and discriminator have stabilized. In Figure A4 we have presented the generator and discriminator loss graphs for three of our models: 1) modified network with perpetual and L1 losses, 2) modified network with perpetual loss function only, and 3) modified network with L1 loss function only. For all three of these training sessions we can see that the discriminator loss values steadily decrease and the generator loss values increase, with both values stabilizing toward the end of training, indicating that successful training outcomes were achieved.

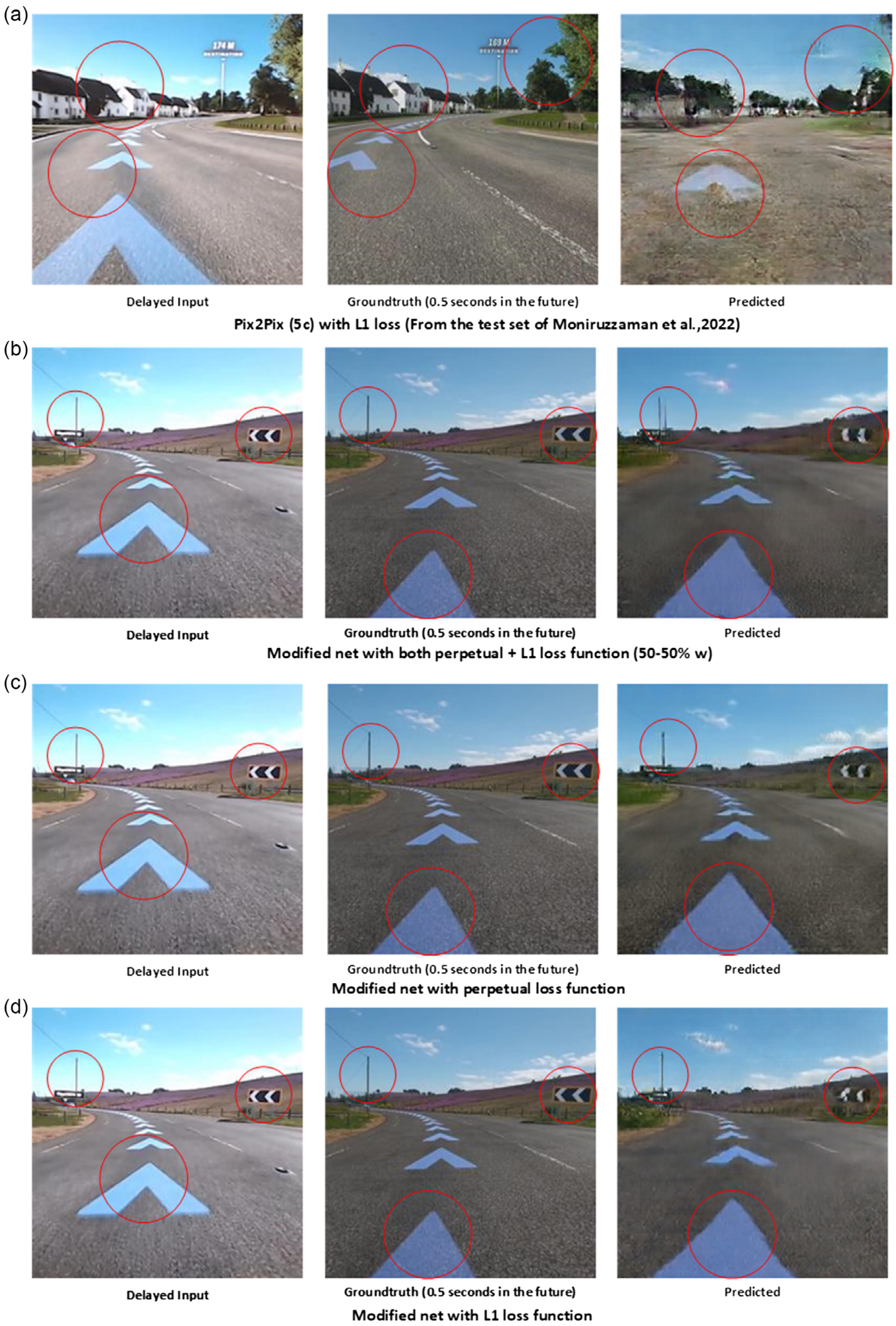

(a)
174 M DESTINATION
169 M DESTINATION
Delayed Input
Groundtruth (0.5 seconds in the future)
Predicted
Pix2Pix (5c) with L1 loss (From the test set of Moniruzzaman et al.,2022)
(b)
Delayed Input
Groundtruth (0.5 seconds in the future)
Predicted
Modified net with both perpetual + L1 loss function (50-50% w)
(c)
Delayed Input
Groundtruth (0.5 seconds in the future)
Predicted
Modified net with perpetual loss function
(d)
Delayed Input
Groundtruth (0.5 seconds in the future)
Predicted
Modified net with L1 loss function

Figure 6. Comparisons of the delayed, predicted, and ground truth frames for different trained models: a) the model presented in ref. [17] trained with Forza_GT + UE + Std_DL data, b) our modified network with SSIM perpetual + L1 loss (50% weightage each), c) modified network with SSIM perpetual loss function, and d) modified network with L1 loss function.

3.3. Evaluation Methods

3.3.1. Image Analysis

We have compared and evaluated the predicted frames using the same three evaluation metrics used previously in ref. [17] so that a fair comparison can be done and the improvement can be evaluated. For the per-pixel comparison, we have used the PSNR[50] technique. For the per-pixel comparison, the PSNR is calculated using the following equation.

$$\mathrm{PSNR} = (20 \times \log(\mathrm{MAX})) - (10 \times \log(\mathrm{MSE})) \tag{5}$$

Here, MAX is the maximum value of the pixels of a frame and MSE is the mean squared error. If an image is closer to the ground truth than another, the PSNR value would be higher. We have compared the PSNR values of the predicted synthetic future frames with the ground truth target frames. These results are reported and discussed in Section 4 and 5.

To check the structural integrity of the objects in a scene and the similarity between the predicted synthetic future frames and the ground truth, we have used the SSIM and MS-SSIM metrics. If two image frames f_1 and f_2 have the same size of $X{*}X$, the estimation of the SSIM can be done using Equation (6).[51]

$$\mathrm{SSIM}(f_1,f_2) = \frac{(2\mu_{f_1}\mu_{f_2} + c_1)(2\sigma_{f_1 f_2} + c_2)}{(\mu_{f_1}^2 + \mu_{f_2}^2 + c_1)(\sigma_{f_1}^2 + \sigma_{f_2}^2 + c_2)} \tag{6}$$

Here, μ_{f_1} is the average off_1, μ_{f_2} is the average off_2, $\sigma_{f_1}^2$ is the variance off_1, $\sigma_{f_2}^2$ is the variance off_2, $\sigma_{f_1 f_2}$ is the covariance of f_1 and f_2, and c_1–c_2 are the variables to stabilize the division. These variance and covariance elements of Equation (6) account for the objects' structural integrity and change between two images.

In addition to PSNR and SSIM, we also evaluated the performance of the Multi-Scale SSIM (MS-SSIM) method. According to several studies,[51–54] MS-SSIM is a more robust method than SSIM and has been shown to perform better for both images and video data. The MS-SSIM algorithm utilizes the SSIM technique, but it operates over multiple scales using a series of subsampling stages. During the MS-SSIM process, the images are first downsampled by a factor of 2 and then passed through a low-pass filter. The estimation of the MS-SSIM is defined as 7.[55]

$$\mathrm{MS} - \mathrm{SSIM}(x,y) = \left[\prod_{i=1}^{n} l(x,i)(y,i)\right]^{\gamma} \cdot \prod_{i=1}^{n} c(x,i)(y,i)^{\beta} \cdot s(x,y) \tag{7}$$

The MS-SSIM algorithm computes the similarity between a reference image and a distorted image by considering multiple scales of the images. In this process, the reference and distorted images are denoted as x and y, respectively, and n is the number of scales used. At each scale i, the luminance of the images is represented by $l(x,i)$ and $l(y,i)$, and the contrast is represented by $c(x,i)$ and $c(y,i)$. The structural similarity index at the lowest scale is denoted as $s(x,y)$. The luminance and contrast terms are controlled by the parameters γ and β, respectively.

3.3.2. Statistical Analysis

Our test dataset contains 2000 input images and the corresponding optical flow components, and the image comparison-based evaluation offers the image quality analysis based on the mean values of the PSNR, SSIM, and MS-SSIM metrics. For a better understanding of the quality of the individual predicted frames, the overall data distribution of the predicted frames, the presence of outliers, the reliability of a specific combination of the loss function and network architecture, the robustness of specific techniques, and to obtain the inter-rater agreements of the evaluation metrics, we have performed statistical analysis of the PSNR, SSIM, and MS-SSIM values of the individual predicted image frames. We have calculated the mean, median, standard deviation, skewness, and Kurtosis for the entire test datasets for all the combinations of datasets, network architectures, and loss functions. For inter-rater agreement and reliability of all the combinations, we have calculated Fleiss's kappa and p values.

4. Results

This article aims to significantly improve the image quality and structural integrity of objects in a scene for predicted synthetically generated future frames for a teleoperation video feed. To keep the results easily comparable to the result provided in ref. [17], we have carried out image comparison-based analyses using the three metrics (PSNR, SSIM, MS-SSIM). For reliability comparison, of the different combinations of the architecture and used loss function, we have also performed statistical analysis of the image comparison analysis. Both of these results are presented in this section.

4.1. Image Comparison-Based Results

The image comparison (relative to the ground truth)-based results are presented in **Table 1**. The first three rows of the table provide the results presented in ref. [17] for the delayed frames (that did not go through any video transformation)-based future prediction. From Table 1 we can see that the base Pix2Pix network with L1 generator loss function and three-channel RGB input achieved 16.27, 0.40, and 0.47 for PSNR, SSIM, and MS-SSIM evaluation metrics consecutively. For the delayed and unreal engine optical flow, the value sets increased to 16.42-0.40-0.48, and 16.41-0.38-0.48. These values indicated that the image-to-image translation techniques have the potential to generate future frames that can be used to mitigate the latency in a long-distance teleoperation scenario. However, the predicted image quality is low and misses a lot of structural details of objects in the scene.

For effective teleoperation enhancement, we need to significantly improve the image quality and maintain the structural integrity of the objects when predicting and generating synthetic future frames. To move closer to this goal, we have implemented a filter for reducing noise in the input data, changed the architecture of the network, and implemented a modified perpetual loss function in the training. Table 1 presents the PSNR, SSIM, and MS-SSIM values for the different combinations of the loss function and the network. When the input data is filtered

Table 1. Comparison of predicted future frames' similarities with ground truth (image quality) for the different trained models described in this article. The comparable models with similar training parameters and dataset described in ref. [17]. The comparison and evaluation were performed using mean PSNR, SSIM, and MS-SSIM values.

Network	Loss function	Input data spec.	Dataset	PSNR	SSIM	MS-SSIM
Pix2Pix	L1 (MAE)	RGB (3 Channel)	Forza_GT + UE + Std_DL	16.27	0.40	0.47
Pix2Pix (5C)	L1 (MAE)	RGB & DL Opt.		16.42	0.40	0.48
Pix2Pix (5C)	L1 (MAE)	RGB & UE Opt.		16.41	0.38	0.48
Pix2Pix (5C)	L1 (MAE)	RGB & DL Opt.	Forza_GT + Std_DL	20.8	0.51	0.70
Pix2Pix (5C)	L1 (MAE)	RGB & UE Opt.		20.9	0.53	0.73
Pix2Pix (5C)	Perpetual SSIM	RGB & DL Opt.		22.7	0.65	0.80
Modified Net	L1 (MAE)	RGB & DL Opt.		22.9	0.64	0.78
Modified Net	Perpetual SSIM	RGB & DL Opt.		23.1	0.65	0.80
Pix2Pix (5C)	L1 + SSIM	RGB & DL Opt.		23.1	0.66	0.80
Modified Net	L1 + SSIM	RGB & DL Opt.		22.9	0.64	0.80

with our custom filter, the predicted image quality significantly improves. For the filtered delayed optical flow-based five-channel input data, the PSNR-SSIM-MS_SSIM values increased to 20.8-0.51-0.70. For the Unreal engine developed optical flow, the values are 20.9-0.53-0.73. This shows that for both types of additional optical flow information, the filter has reduced the noise from the input incoming data and contributed to improving the image quality.

When the SSIM-based perpetual loss function was used for training in place of the Ll MAE loss function, the PSNR-SSIM-MS_SSIM values increased further to 22.7-0.65-0.80. With our modified network architecture, we found similar improvements and the values are 22.9-0.64-0.78. While applying both the modified network and the perpetual loss function, the image quality was further improved and increased to 23.10.65-0.80. Our modified network with a loss function that has 50% weight on the absolute error and 50% on perpetual loss also performed very well and the metrics are 22.9-0.64-0.80.

4.2. Statistical Representation of the Image Analysis

Table 2 and 3 present a statistical analysis of our research design and evaluate the performance of different combinations of the loss function and model architecture. This statistical analysis validates the result shown in Table 1. The statistical analyses were performed on the PSNR, SSIM, and MS-SSIM values of all the test-set output frames. In Table 2 we can see the mean values for all the combinations are almost the same as those of the median values. This indicates that with reference to the image quality for all of our improvements to the future prediction technique described in ref. [17], the predicted frames generated by our different models follow a normal and almost symmetrical distribution. We can see that the standard deviations for all evaluation metrics and combinations are very low. For PSNR evaluation, the standard deviations are 1.22–1.48; and for the SSIM and MS-SSIM the standard deviations are only 0.04–0.05. The Skewness values are very low as well. For the PSNR evaluation, the skewness ranges from -0.86 to -0.24; for SSIM evaluation the skewness ranges from -0.32 to 0.002; and for the MS-SSIM, the range is -1.17 to -0.88. Kurtosis has been calculated to check the presence of outliers in the data distribution. The Kurtosis for the PSNR evaluation ranges from 0.24 to 1.24; for SSIM evaluation, it ranges from -0.27 to 0.18; and for MS-SSIM from 1.14 to 1.99.

As previously discussed, for image quality and structural integrity measurement, we have used three evaluation parameters: PSNR, SSIM, and MS-SSIM. The agreement rates between these three evaluation metrics provide an idea about the usability and reliability of a specific trained model. We have calculated the Fleiss' kappa to measure the inter-rater agreements. From Table 3, the kappa for the five-channel filtered data only, without changing the loss function or architecture, is 0.32 (for UE optical flow) and 0.34 (for delayed optical flow). While using the perpetual loss function without changing the network, the kappa value increases to 0.46. For the modified network the kappa value is 0.48 and for combining the modified network and the perpetual loss in the training the kappa is 0.44. It appears that the inclusion of L1 loss (50% w) with the perpetual loss reduces the kappa a little (0.37). For the modified network trained with a generator loss function based on both the L1 and perpetual loss elements, the kappa is 0.43. For all of our trained models the p values are less than 0.05.

5. Discussion

The primary goal of this research is to improve the image quality of the predicted synthetic future frames by preserving the structural details of objects in a frame so that they can be used to generate a good quality visual feed to a human teleoperator. This research tries to address the limitations identified in our previous work.[17] To predict future frames that account for the teleoperators' control inputs, we have used dense optical flow as an additional channel along with the RGB channels of the delayed input frames. We have shown that without the optical flow information, the predicted frames' image quality is not adequate enough to be used for teleoperation enhancement.[17] However, the frame rate of the delayed visual feed is often not uniform and low. These irregularities of frame rates create noise in the angle and magnitude components when generating dense optical flow from consecutive frames. Our filter, presented in Algorithm 3.1.2, removes the noisy outputs from the optical flow generation

Table 2. Statistical analysis of the predicted future frames' PSNR, SSIM, and MS-SSIM values for all the trained models.

Model specification	Mean	Median	Std. Deviation	Skewness	Kurtosis
PSNR Evaluator					
Filtered data with DI optical flow	20.80	21.1	1.22	−0.81	0.85
Filtered data with UE optical flow	20.95	21.05	1.30	−0.55	0.66
Perpetual loss (SSIM)	22.72	22.88	1.34	−0.86	1.24
Modified network with L1 loss	22.90	23.0	1.43	−0.52	0.53
Modified network with perpetual loss	23.10	23.26	1.47	−0.77	1.04
Perpetual loss & L1 loss (50% w)	23.10	23.35	1.48	−0.74	0.38
Modified net with Perp. & L1 loss	22.80	22.81	1.45	−0.24	0.24
SSIM Evaluator					
Filtered data with DI optical flow	0.54	0.54	0.040	−0.18	0.14
Filtered data with UE optical flow	0.56	0.56	0.04	.002	−0.27
Perpetual loss (SSIM)	0.65	0.65	0.04	−0.32	−0.08
Modified network with L1 loss	0.61	0.61	0.04	−0.09	−0.22
Modified network with perpetual loss	0.61	0.60	0.05	−0.13	−0.08
Perpetual loss & L1 loss (50% w)	0.65	0.65	0.04	−0.27	0.18
Modified net with Perp. & L1 loss	0.64	0.64	0.04	−0.09	−0.25
MS−SSIM Evaluator					
Filtered data with DI optical flow	0.63	0.64	0.05	−1.06	1.86
Filtered data with UE optical flow	0.65	0.65	0.04	−1.17	1.99
Perpetual loss (SSIM)	0.78	0.79	0.04	−1.17	1.99
Modified network with L1 loss	0.77	0.78	0.05	−0.97	1.47
Modified network with perpetual loss	0.80	0.79	0.047	−1.063	1.77
Perpetual loss & L1 loss (50% w)	0.80	0.80	0.04	−1.16	1.84
Modified net with Perp. & L1 loss	0.80	0.80	0.04	−0.88	1.14

Table 3. Inter-Rater agreement and reliability comparison of trained models based on Fleiss' Kappa and *P* values.

Optical flow presence	Kappa	*p* value	Lower bound (95% CI)	Upper bound (95% CI)
Filtered data with DI optical flow	0.34	<0.05	0.32	0.36
Filtered data with UE optical flow	0.32	<0.05	0.30	0.34
Perpetual loss (SSIM)	0.46	<0.05	0.44	0.47
Modified network with L1 loss	0.48	<0.05	0.46	0.50
Modified network with perpetual loss	0.44	<0.05	0.42	0.46
Perpetual loss & L1 loss (50% w)	0.37	<0.05	0.35	0.39
Modified net with Perp. & L1 loss	0.43	<0.05	0.42	0.45

algorithm and provides the future prediction deep network better input data. This results in a substantial improvement in the quality of the predicted frames (Table 1).

From **Figure 6a** it is obvious that with the Pix2Pix conventional network with the MAE L1 loss function, the generated frames miss a lot of structural details of the scene including the road marks and the surrounding environment for a 500 ms future frame prediction task. From an image transformation perspective, the future is a transformed version of the past. The more information that is retained during the encoding and decoding actions of a U-Net-based future prediction network, the more structural integrity of the scene would be preserved. Therefore, we have modified the network we used in ref. [17] (see Figure 5). Along with feeding the encoder with five-channel input information, the angle and magnitude components reflecting the structural changes, as well as the operator control intention information, have been included toward the end of the decoder. An additional convolution layer has been added before the output layer so that the network can utilize this nondeformed information to predict a more structurally stable future. Further, we have altered the learning process of the cGAN network as well by changing the loss function. The L1 loss only accounts for the per-pixel change and the generator is penalized for the pixel deformation, no structural change is accounted for. To resolve this issue, we have experimented with the SSIM-based perpetual loss function (Equation (3) and (4)). The variance and covariance components of this perpetual loss (Equation (6)) account for the structural changes in frames and penalize the generator accordingly. With a combination of the modified network and the loss function, we have observed significant improvement in the image quality of the predicted frames (Figure 6c). We have experimented with 50% of the L1% and

50% of the perpetual losses for training and found similar promising outcomes (Figure 6b).

The statistical analysis presented in Table 2 offers insight into the validity of the new techniques described in this article. The fact that the mean and medians are very similar supports the claim that the new filter, the changed network architecture, and the use of the perpetual generator loss function have significantly improved the image quality of the predicted and synthetically generated future frames. The standard deviations are also very low and lower than those reported in ref. [17]. This indicates that the image quality of all the test outputs is clustered around the mean value and uniform. Therefore, the video quality remains uniform and does not change drastically frame by frame. The networks and the loss functions can handle the complexity amid having a high level of uncertainty in the incoming input feed for a 0.5 s delay. The skewness values are also very low. More importantly, the predicted frames' image quality values' distribution is slightly negatively skewed, which means that for most of the predicted frames the image quality is better than the mean values. The kurtosis values measure the shape of a data distribution relative to the normal distribution and can be used to indicate the presence of outliers in the data. The reliability of the kurtosis values depends on the sample size. In our experiment, the sample size is sufficiently large to rely on the values. As shown in Table 2, all of our new settings have kurtosis values of less than 3. For the PSNR and MS-SSIM evaluators, the kurtosis values are slightly positive but close to zero, while for the SSIM evaluator, the values are slightly negative and also close to zero. This indicates that the PSNR, SSIM, and MS-SSIM values are almost normally distributed, and there are very few outliers in the data.

Inter-rater agreement represents the reliability of the measured variables and parameters that are collected during an experiment. For comparing the images and measuring the quality we have applied three well-recognized image comparison matrices or raters. To measure the inter-rater agreement, Feliss's kappa is a well-recognized method. The better the value of kappa for a model (for our use case, network, and loss function combination), the more reliable the model is. Fleiss kappa value is considered fair for the range of 0.21–0.40 and moderate for the range of 0.41–0.60.[56,57] For both our modified network and the perpetual loss the kappa values are high. Therefore, it can be claimed that our modifications have produced more reliable future frames that are supported by all three evaluation metrics.

In addition to the image quality considerations, inference time is crucial for our use case, where the model needs to be capable of predicting future frames in real time to provide a continuous video feed to teleoperators during actual operations. The inference time of a GAN model depends on various factors, including the model's complexity, input data size, hardware used for inference, and the model's specific implementation. We found that our models' inference times range from 0.07 to 0.08 s (**Table 4**) running on an NVIDIA Geforce Titan RTX system with a 3.5 GHz Intel Core i9 CPU, which is fast enough to generate frames at a speed of 13–14 fps based on our computation facility, data, and training specifications. This inference speed is particularly encouraging given our specific use case, where we expect delayed frames at a rate of 13–15 fps due to the challenging communication medium.

Table 4. Inference times for the trained models.

Models	Inference time [s]
Filtered data with DI optical flow	0.07
Filtered data with UE optical flow	0.07
Perpetual loss (SSIM)	0.07
Modified network with L1 loss	0.08
Modified network with perpetual loss	0.08
Perpetual loss & L1 loss (50% w)	0.07
Modified net with Perp. & L1 loss	0.07

6. Conclusion

This article presents a novel technique for the generation of high-quality structure-aware future frames from a stream of past frames delayed by ≈500 ms seconds for ground robotic vehicle teleoperation enhancement. Our previous work[17] proposed the initial idea that the synthetically generated future frames can be an effective method to mitigate the latency problem in visual feed-based ground vehicle teleoperation operated at reasonably high speeds.[17] This work introduced the idea of using five-channel input data and U-net-structured cGAN for future frame prediction. However, the image quality compared to the ground truth future was relatively low and the predicted frames were missing significant structural detail of the objects in the scene. To resolve the frame rate disparity-related issue identified in that work, we have designed a filter to remove noise from the dense optical flow components and input data. To preserve the structural integrity of objects, we have modified the U-Net generator architecture of the cGAN used. We have also used a perpetual loss function for training that accounts for the structural changes of objects in frames. We have found that our modified network along with the SSIM perpetual loss function-based model achieves the highest accuracy with image quality metrics of 23.1-0.65-0.80 (PSNR, SSIM-MS_SSIM). A 50–50 percentage weight for L1 and the perpetual loss function with the modified network also achieved similar performance (22.9-0.65-0.80). Fleiss' kappa values for these models have also confirmed high reliability values of 0.44 and 0.43 respectively. The statistical analysis of the 2000 test frames demonstrates that the use of the filter, the newly updated architecture, and the inclusion of the SSIM-based perpetual loss function have generated synthetic future frames that have higher accuracy and more uniform quality and fewer outliers in the output image frame distribution (based on image quality metrics). We are confident that the improved predicted image frames can be used in a real-time robotic teleoperation control task in order to significantly reduce the impact of latency. As the next step for this research, we plan to implement our trained image-to-image long future prediction model into a real-world robotic system, in order to assess its effectiveness and applicability in practical scenarios. We will also continue to investigate methods to improve the prediction model, both to enable prediction over longer time scales and to improve the quality of the generated frames.

Appendix

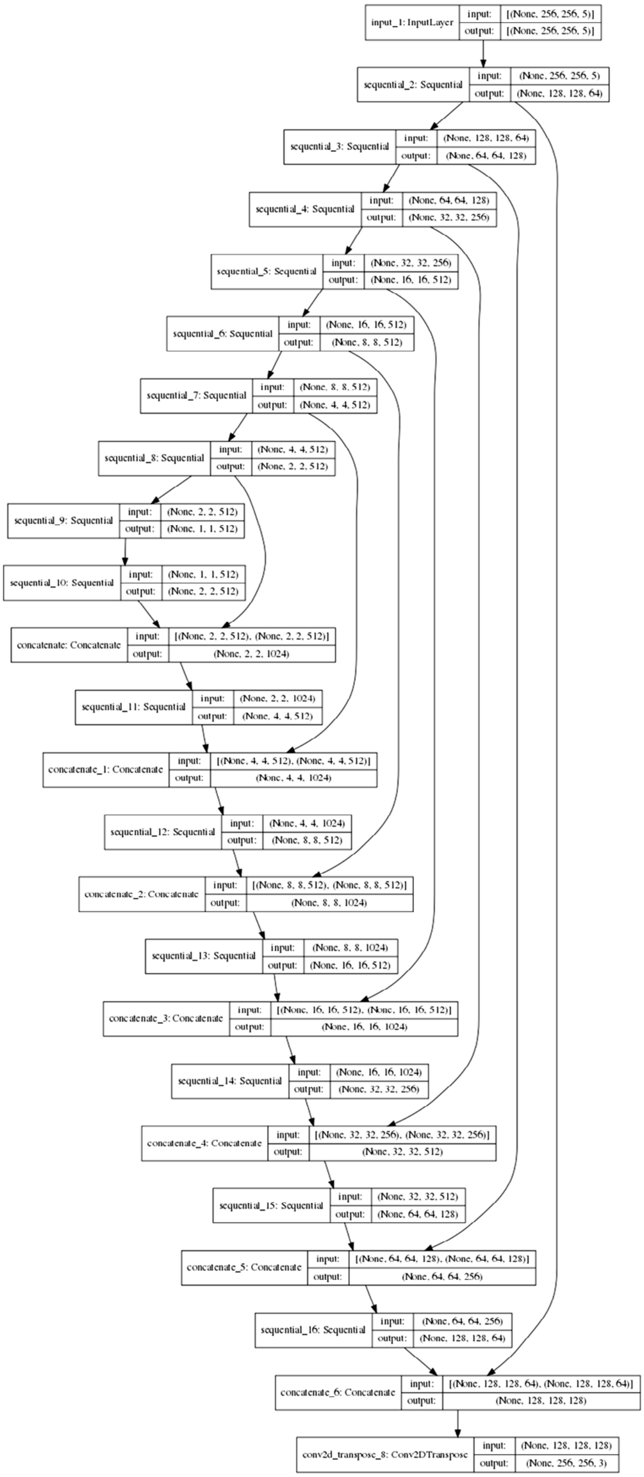

input_1: InputLayer | input: | [(None, 256, 256, 5)] | output: | [(None, 256, 256, 5)]
sequential_2: Sequential | input: | (None, 256, 256, 5) | output: | (None, 128, 128, 64)
sequential_3: Sequential | input: | (None, 128, 128, 64) | output: | (None, 64, 64, 128)
sequential_4: Sequential | input: | (None, 64, 64, 128) | output: | (None, 32, 32, 256)
sequential_5: Sequential | input: | (None, 32, 32, 256) | output: | (None, 16, 16, 512)
sequential_6: Sequential | input: | (None, 16, 16, 512) | output: | (None, 8, 8, 512)
sequential_7: Sequential | input: | (None, 8, 8, 512) | output: | (None, 4, 4, 512)
sequential_8: Sequential | input: | (None, 4, 4, 512) | output: | (None, 2, 2, 512)
sequential_9: Sequential | input: | (None, 2, 2, 512) | output: | (None, 1, 1, 512)
sequential_10: Sequential | input: | (None, 1, 1, 512) | output: | (None, 2, 2, 512)
concatenate: Concatenate | input: | [(None, 2, 2, 512), (None, 2, 2, 512)] | output: | (None, 2, 2, 1024)
sequential_11: Sequential | input: | (None, 2, 2, 1024) | output: | (None, 4, 4, 512)
concatenate_1: Concatenate | input: | [(None, 4, 4, 512), (None, 4, 4, 512)] | output: | (None, 4, 4, 1024)
sequential_12: Sequential | input: | (None, 4, 4, 1024) | output: | (None, 8, 8, 512)
concatenate_2: Concatenate | input: | [(None, 8, 8, 512), (None, 8, 8, 512)] | output: | (None, 8, 8, 1024)
sequential_13: Sequential | input: | (None, 8, 8, 1024) | output: | (None, 16, 16, 512)
concatenate_3: Concatenate | input: | [(None, 16, 16, 512), (None, 16, 16, 512)] | output: | (None, 16, 16, 1024)
sequential_14: Sequential | input: | (None, 16, 16, 1024) | output: | (None, 32, 32, 256)
concatenate_4: Concatenate | input: | [(None, 32, 32, 256), (None, 32, 32, 256)] | output: | (None, 32, 32, 512)
sequential_15: Sequential | input: | (None, 32, 32, 512) | output: | (None, 64, 64, 128)
concatenate_5: Concatenate | input: | [(None, 64, 64, 128), (None, 64, 64, 128)] | output: | (None, 64, 64, 256)
sequential_16: Sequential | input: | (None, 64, 64, 256) | output: | (None, 128, 128, 64)
concatenate_6: Concatenate | input: | [(None, 128, 128, 64), (None, 128, 128, 64)] | output: | (None, 128, 128, 128)
conv2d_transpose_8: Conv2DTranspose | input: | (None, 128, 128, 128) | output: | (None, 256, 256, 3)

Figure A1. Generator architecture from ref. [17].

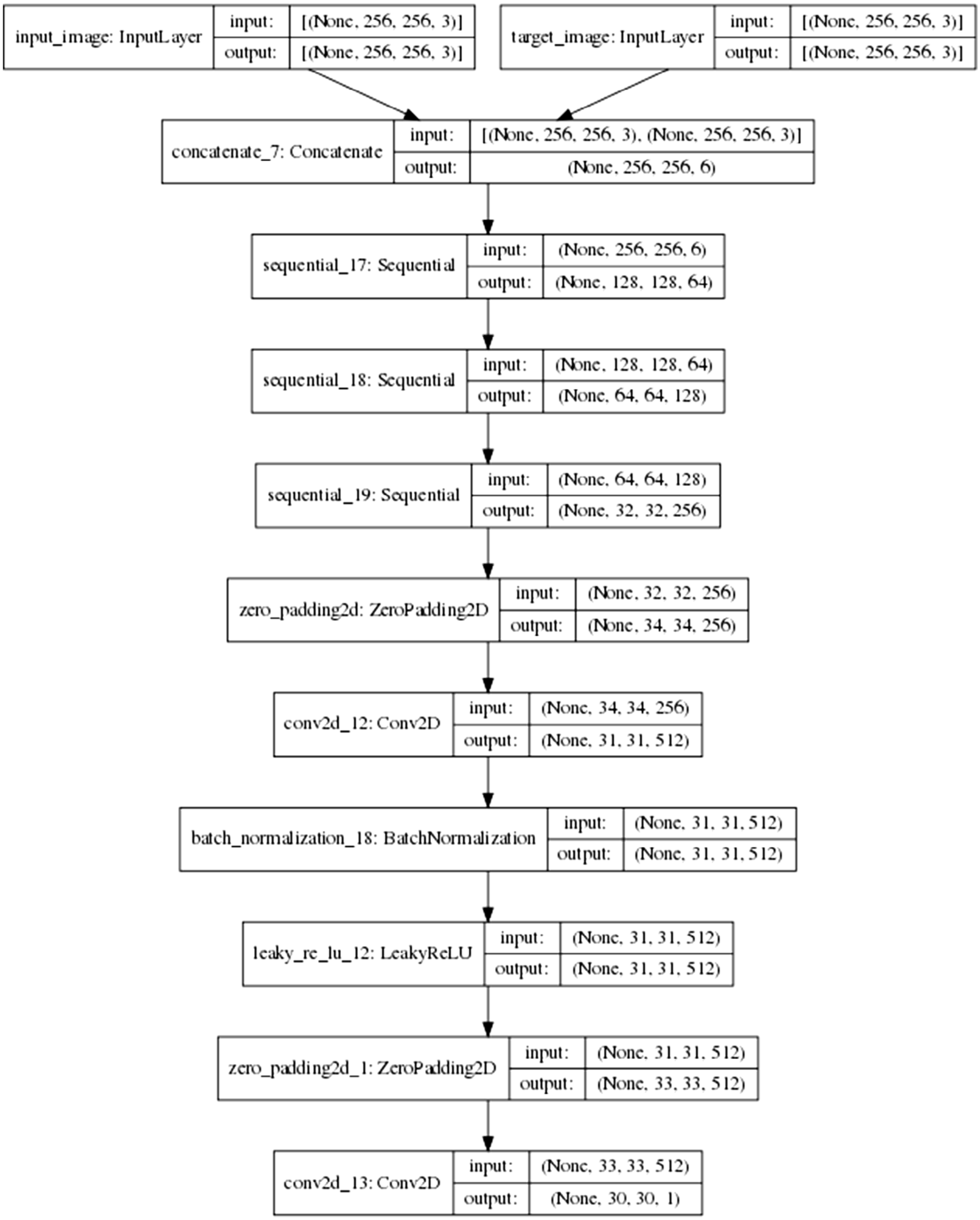

input_image: InputLayer
input: [(None, 256, 256, 3)]
output: [(None, 256, 256, 3)]
target_image: InputLayer
input: [(None, 256, 256, 3)]
output: [(None, 256, 256, 3)]
concatenate_7: Concatenate
input: [(None, 256, 256, 3), (None, 256, 256, 3)]
output: (None, 256, 256, 6)
sequential_17: Sequential
input: (None, 256, 256, 6)
output: (None, 128, 128, 64)
sequential_18: Sequential
input: (None, 128, 128, 64)
output: (None, 64, 64, 128)
sequential_19: Sequential
input: (None, 64, 64, 128)
output: (None, 32, 32, 256)
zero_padding2d: ZeroPadding2D
input: (None, 32, 32, 256)
output: (None, 34, 34, 256)
conv2d_12: Conv2D
input: (None, 34, 34, 256)
output: (None, 31, 31, 512)
batch_normalization_18: BatchNormalization
input: (None, 31, 31, 512)
output: (None, 31, 31, 512)
leaky_re_lu_12: LeakyReLU
input: (None, 31, 31, 512)
output: (None, 31, 31, 512)
zero_padding2d_1: ZeroPadding2D
input: (None, 31, 31, 512)
output: (None, 33, 33, 512)
conv2d_13: Conv2D
input: (None, 33, 33, 512)
output: (None, 30, 30, 1)

Figure A2. Discriminator architecture.

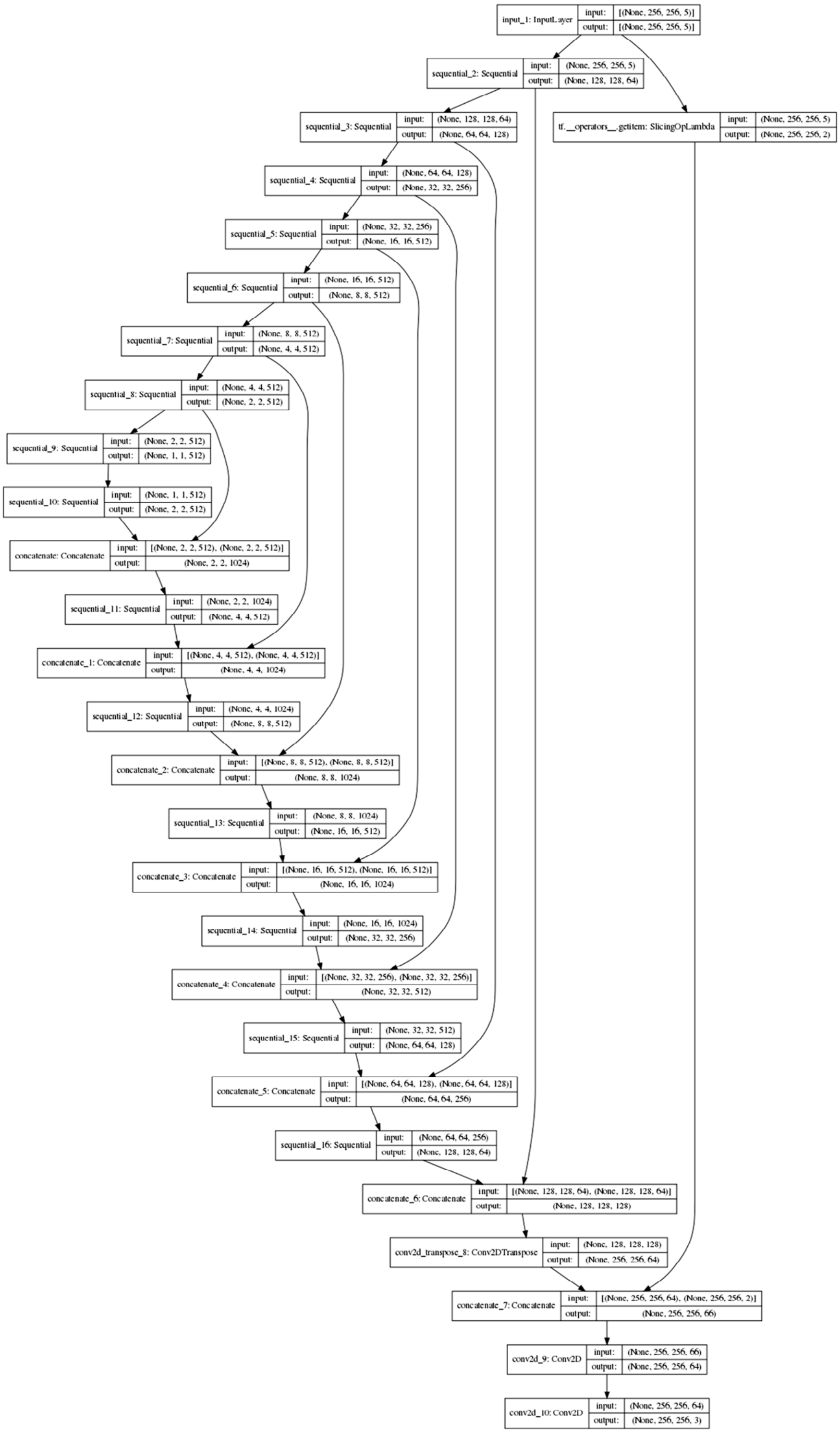
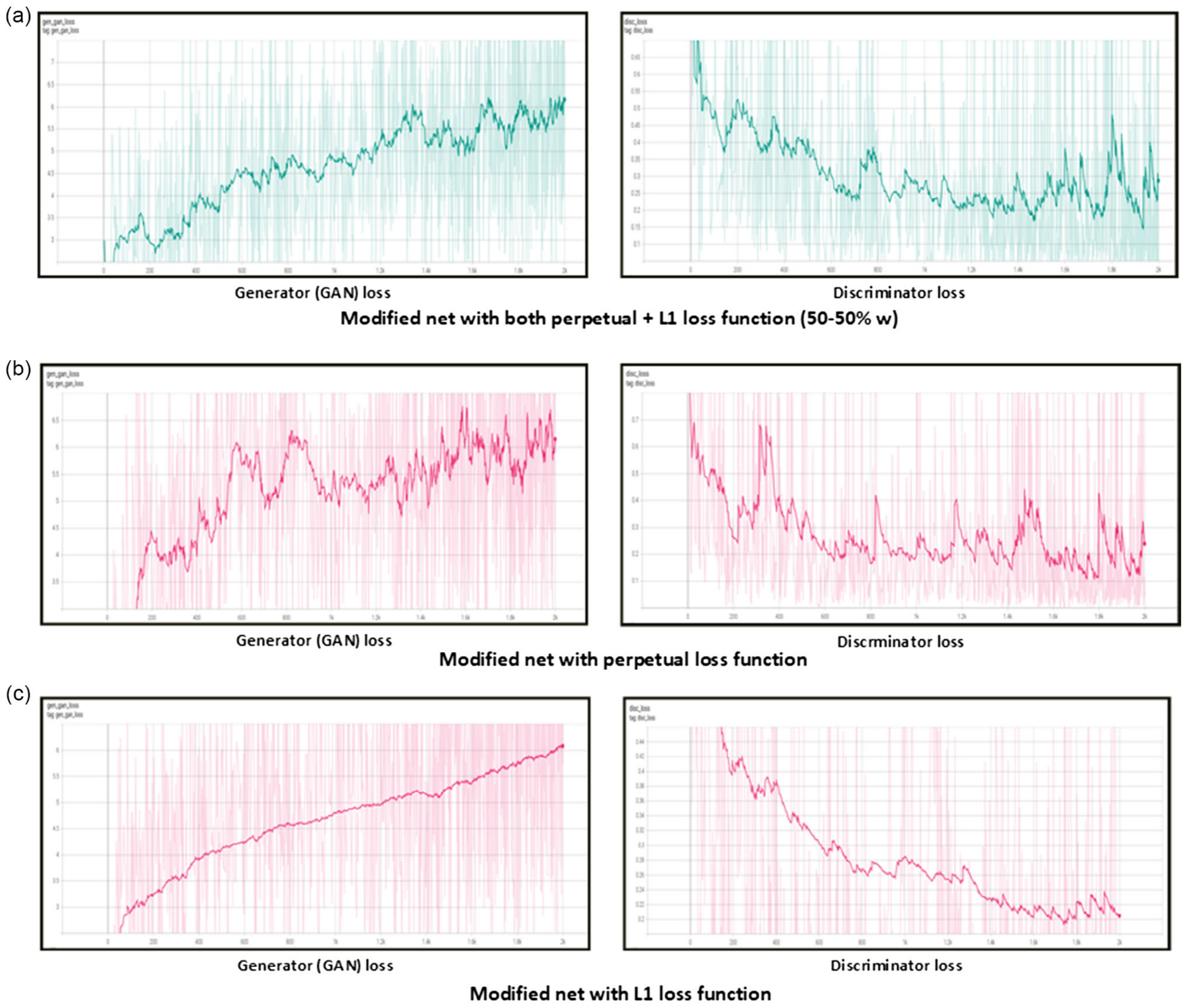


Figure A3. Modified generator architecture.



Generator (GAN) loss

Discriminator loss

Modified net with both perpetual + L1 loss function (50-50% w)

Generator (GAN) loss

Discrminator loss

Modified net with perpetual loss function

Generator (GAN) loss

Discriminator loss

Modified net with L1 loss function

Figure A4. Generator and discriminator loss graphs for models: a) modified network with both perpetual and L1 loss function, b) modified network with perpetual loss function only, and c) modified network with L1 loss function only.

Acknowledgements

This research was funded by the ECU-DSTG Industry PhD scholarship. Appropriate approval has been granted by the Human Ethics Committee (HREC) of Edith Cowan University (ECU) prior to any operator data collection. Approval number 2020-01643.

Conflict of Interest

The authors declare no conflict of interest.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Keywords

conditional generative adversarial networks, future frame predictions, perpetual losses, robotic vehicles, teleoperation

Received: December 14, 2022
Revised: June 9, 2023
Published online: July 20, 2023

[1] J. Hawkins, S. Blakeslee, S. Russell, P. Norvig, *IEEE Trans. Softw. Eng.* **2002**, *28*, 721.

[2] Y. Bengio, A. Courville, P. Vincent, *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1798.

[3] X. Wang, A. Gupta, in *Proc. of the IEEE Int. Conf. on Computer Vision*, IEEE, Piscataway, NJ **2015** pp. 2794–2802.

[4] S. Oprea, P. Martinez-Gonzalez, A. Garcia-Garcia, J. A. Castro-Vargas, S. Orts-Escolano, J. Garcia-Rodriguez, A. Argyros, *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**. *44*, 2809.

[5] J. S. Castelló, Master Thesis, Universitat Politècnica de Catalunya **2018**.
[6] M. Moniruzzaman, A. Rassau, D. Chai, S. M. S. Islam, *Rob. Auton. Syst.* **2021**, *150*, 103973.
[7] Z. Wu, Y. Fu, Y.-G. Jiang, L. Sigal, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2016**, pp. 3112–3121.
[8] Z. Liu, R. A. Yeh, X. Tang, Y. Liu, A. Agarwala, in *Proc. of the IEEE Int. Conf. on Computer Vision*, IEEE, Piscataway, NJ **2017**, pp. 4463–4471.
[9] H. Yu, J. Wang, Z. Huang, Y. Yang, W. Xu, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2016**, pp. 4584–4593.
[10] M. Chaabane, A. Trabelsi, N. Blanchard, R. Beveridge, in *Proc. of the IEEE/CVF Winter Conf. on Applications of Computer Vision*, IEEE, Piscataway, NJ **2020**, pp. 2297–2306.
[11] K. Simonyan, A. Zisserman, in *Advances In Neural Information Processing Systems*, The MIT Press, Cambridge, Massachusetts, **2014**.
[12] Y.-H. Kwon, M.-G. Park, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2019**, pp. 1811–1820.
[13] C. Vondrick, H. Pirsiavash, A. Torralba, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2016**, pp. 98–106.
[14] K.-H. Zeng, W. B. Shen, D.-A. Huang, M. Sun, J. Carlos Niebles, in *Proc. of the IEEE Int. Conf. on Computer Vision*, IEEE, Piscataway, NJ **2017**, pp. 2999–3008.
[15] W. Liu, W. Luo, D. Lian, S. Gao, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2018**, pp. 6536–6545.
[16] M. Moniruzzaman, A. Rassau, D. Chai, S. M. S. Islam, *J. Intell. Robot. Syst.* **2022**, *106*, 2.
[17] M. Moniruzzaman, A. Rassau, D. Chai, S. M. S. Islam, *J. Field Robot.* **2022**, *40*, 393.
[18] I. S. MacKenzie, C. Ware, in *Proc. of the INTERACT'93 and CHI'93 Conf. on Human Factors in Computing Systems*, Association for Computing Machinery, New York **1993**, pp. 488–493.
[19] J. C. Lane, C. R. Carignan, B. R. Sullivan, D. L. Akin, T. Hunt, R. Cohen, in *IEEE Int. Conf. on Robotics and Automation,*Vol. *3*, IEEE, Piscataway, NJ **2002**, pp. 2874–2879.
[20] R. Held, A. Efstathiou, M. Greene, *J. Exp. Psychol.* **1966**, *72*, 887.
[21] S. Neumeier, E. A. Walelgne, V. Bajpai, J. Ott, C. Facchi, in *Network Traffic Measurement and Analysis Conf. (TMA)*, IEEE, Paris **2019**, pp. 113–120.
[22] L. H. Frank, J. G. Casali, W. W. Wierwille, *Hum. Factors* **1988**, *30*, 201.
[23] M. Wilde, M. Chan, B. Kish, in *IEEE Aerospace Conf.* IEEE, Piscataway, NJ, **2020**, pp. 1–14.
[24] A. Matheson, B. Donmez, F. Rehmatullah, P. Jasiobedzki, H.-K. Ng, V. Panwar, M. Li, in *Proc. of the Human Factors and Ergonomics Society Annual Meeting*, SAGE Publications, London **2013**, pp. 21–25.
[25] C. Ha, J. Yoon, C. Kim, Y. Lee, S. Kwon, D. Lee, *Auton. Robots* **2018**, *42*, 1819.
[26] Y. Zhou, T. L. Berg, in *European Conf. on Computer Vision,* **2016** pp. 262–277.
[27] C. Li, M. Wand, in *European Conf. on Computer Vision*, Springer, Dordrecht **2016**, pp. 702–716.
[28] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, A. A. Efros, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, Computer Vision Foundation, Las Vegas **2016**, pp. 2536–2544.
[29] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al., in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ, **2017**, pp. 4681–4690.
[30] X. Li, Z. Du, Y. Huang, Z. Tan, *ISPRS J. Photogramm. Remote Sens.* **2021**, *179*, 14.
[31] P. Isola, J.-Y. Zhu, T. Zhou, A. A. Efros, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ, **2017**, pp. 1125–1134.
[32] X. Wang, A. Gupta, in *European Conf. on Computer Vision*, **2016**, pp. 318–335.
[33] S. E. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, H. Lee, *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 217.
[34] L. Karacan, Z. Akata, A. Erdem, E. Erdem, arXiv:1612.00215, **2016**.
[35] M. Mathieu, C. Couprie, Y. LeCun, arXiv:1511.05440, **2015**.
[36] D. Yoo, N. Kim, S. Park, A. S. Paek, I. S. Kweon, in *European Conf. on Computer Vision*, **2016**, pp. 517–532.
[37] A. Alotaibi, *Symmetry*, MDPI, Basel, Switzerland **2020**, *12*, 1705.
[38] J.-Y. Zhu, T. Park, P. Isola, A. A. Efros, in *Proc. of the IEEE Int. Conf. on Computer Vision*, IEEE, Piscataway, NJ **2017**, pp. 2223–2232.
[39] Y. Li, S. Tang, R. Zhang, Y. Zhang, J. Li, S. Yan, *IEEE Trans. Image Process.* **2019**, *28*, 5881.
[40] J. Lin, Z. Chen, Y. Xia, S. Liu, T. Qin, J. Luo, *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *43*, 1254.
[41] M.-Y. Liu, T. Breuel, J. Kautz, *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 700.
[42] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, E. Shechtman, *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 465.
[43] F. Xiong, Q. Wang, Q. Gao, *IEEE Access* **2019**, *7*, 126651.
[44] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, J. Choo, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2018**, pp. 8789–8797.
[45] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, B. Catanzaro, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, IEEE, Piscataway, NJ **2018**, pp. 8798–8807.
[46] A. Radford, L. Metz, S. Chintala, arXiv:1511.06434, **2015**.
[47] G. Farnebäxck, in *Scandinavian Conf. on Image Analysis*, Springer, Halmstad, Sweden **2003**, pp. 363–370.
[48] O. Ronneberger, P. Fischer, T. Brox, in *Int. Conf. on Medical Image Computing and Computer-assisted Intervention*, Springer, Munich, Germany **2015**, pp. 234–241.
[49] D. P. Kingma, J. Ba, arXiv:1412.6980, **2014**.
[50] D. H. Johnson, *Scholarpedia* **2006**, *1*, 2088.
[51] Z. Wang, E. P. Simoncelli, A. C. Bovik, in *Thrity-Seventh Asilomar Conf. on Signals, Systems & Computers*, IEEE, CA, USA **2003**, pp. 1398–1402.
[52] D. M. Rouse, S. S. Hemami, in *Human Vision And Electronic Imaging XIII*, SPIE, CA, USA **2008**, p. 680615.
[53] J. Søgaard, L. Krasula, M. Shahid, D. Temel, K. Brunnström, M. Razaak, *Electron. Imag.* **2016**, *13*, 1.
[54] R. Dosselmann, X. D. Yang, *Signal, Image Video Process.* **2011**, *5*, 81.
[55] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, *IEEE Trans. Image Process.* **2004**, *13*, 600.
[56] J. M. Bland, D. G. Altman, *Stat. Methods Med. Res.* **1999**, *8*, 135.
[57] J. R. Landis, G. G. Koch, *Biometrics* **1977**, *33*, 363.