

# Deep Motion Blind Video Stabilization

## Supplementary Material

Muhammad Kashif Ali\*

kashifali@hanyang.ac.kr

Sangjoon Yu\*

kiddy1991@gmail.com

Tae Hyun Kim†

taehyunkim@hanyang.ac.kr

Department of Computer Science

Hanyang University

Seoul, South Korea

## 1 Overview

Due to the space limitation in the main manuscript, additional details of some of the modules are presented in this supplementary material. Specifically, we introduce the details of Dataset Generation Pipeline (DGP) such as training strategy of the refinement network. Moreover, we present details of the proposed stabilization network; training losses, run-time, ablation and user study in this supplementary material. Additionally, the expanded view of the qualitative results is also included in this supplemental.

## 2 Additional Details of DGP

In this section, we discuss additional details of our DGP. Generating a high-quality dataset for video stabilization through iterative frame interpolation schema needs additional refinement steps due to the artifacts produced by the iterative frame interpolation, as discussed in the main manuscript. These artifacts are shown in Fig. 1. After extensive experimentation and observation, we concluded that the introduction of a refinement step in every  $k$  iterations through the frame interpolation models can remove the produced artifacts.

The refinement network presented in our work takes advantage of the image quality improvement techniques presented in [2, 8, 10, 11, 12]. For our refinement, we use a modified version of the ResNet-based network proposed by Mehdi et al. [12] with an integrated channel attention module from [2] to focus on the high-level features along with the spatial relations in the consecutive frames. The model architecture and the integrated attention module is illustrated in Fig. 2. The refinement network takes in a single degraded frame  $[I_t]$  (acquired through iterative frame interpolation) along with four original high-quality neighboring frames  $[U_{t-2}, U_{t-1}, U_{t+1}, U_{t+2}]$ , and produces a refined version  $[I'_t]$  of the degraded frame without modifying the spatial integrity of the degraded frame. Note that,  $[U_{t-2}, U_{t-1}, U_{t+1}, U_{t+2}]$  are high-quality frames taken from the input unstable video frames.

\*Equal contribution.

†Corresponding author.

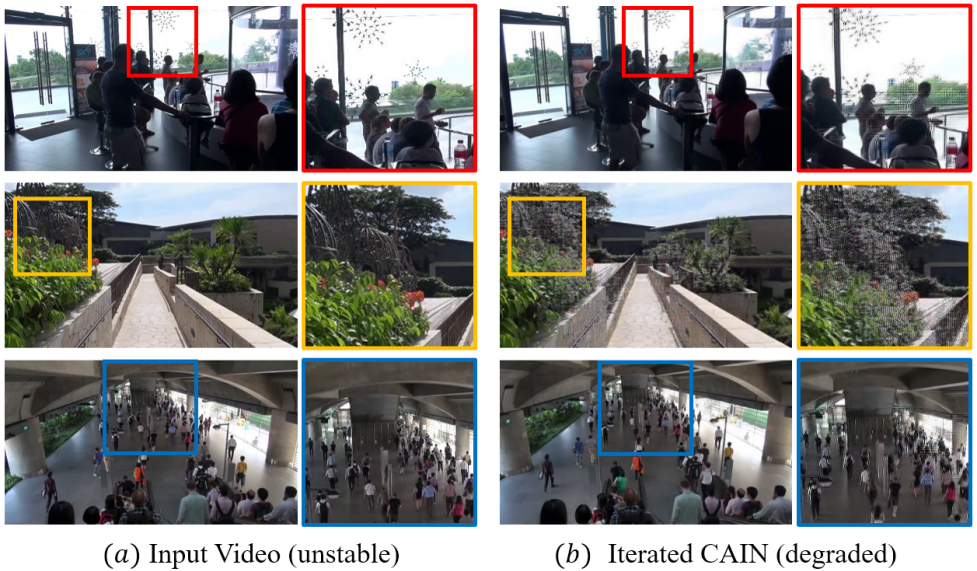


Figure 1: (a) Input unstable video frames. (b) Produced artifacts after 20 iterations with CAIN [2].

## 2.1 Refinement Details

### • Refinement network training

As shown in Fig. 3, we need synthetic video frames  $[C_1, C_2, \dots, C_n]$  to train the refinement network. This synthetic dataset is generated with the help of a physics-based sequential cropping mechanism that moves a cropping window over high-quality static frames to simulate the linear motion present in the videos acquired through hand-held cameras (Fig. 3 (Step 1)). Then, alternate frames  $[C_1, C_3, C_5, \dots]$  from this synthetic video are passed through the iterative frame interpolation configuration (Fig. 3 (Step 2)) for an odd number of iterations to generate degraded frames  $[i_2, i_4, i_6, \dots]$ . This strategy of utilizing alternate frames exploits the idea that the frame interpolation modules are trained to generate intermediary frames of the given frames. By generating middle frames between alternate frames, we get the modified versions of skipped frames. The original skipped frames  $[C_2, C_4, C_6, \dots]$  of the synthetic video can now be treated as the high-quality targets of the degraded frames.

### • Refinement network configurations

During the training phase of our refinement network, neighboring frames from the set of alternate frames  $([C_{t-2}, C_{t-1}, C_{t+1}, C_{t+2}])$  of the synthetic video are used along with a degraded frame  $i_t$ . Whereas, during the finalized DGP (as shown in Fig. 4), the actual unstable neighboring frames  $([U_{t-2}, U_{t-1}, U_{t+1}, U_{t+2}])$  are used to refine the generated stable frames. In our arrangement,  $k$  (number of iterations before the refinement step) was empirically evaluated to be 4 for the frame interpolator CAIN [2]. We experimented with various loss functions and architectures before selecting the optimum configuration for effectively removing the generated artifacts. A visual comparison of the effects of tested configurations on the generated frames and edge profiles is provided in Fig. 5. Almost all of the investigated configurations managed to get rid of the produced artifacts but compromised the sharpness of

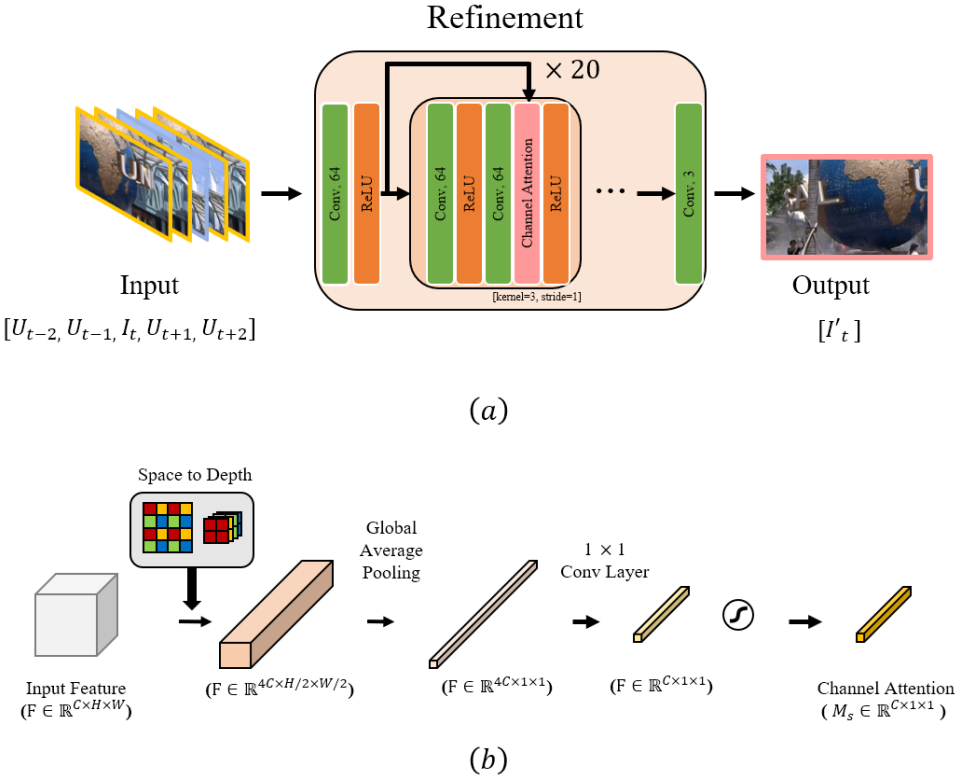


Figure 2: (a) Refinement network architecture. (b) A slightly modified version of channel attention module as used by [2].

the resulting frames. Therefore, our finalized refinement network is trained with a modified RGB-based gradient-map loss (described in Sec. 2.2) inspired from the gradient guidance presented in [14] along with a conventional  $\mathcal{L}_1$  loss, as shown in Fig. 3 (Step 3). From Fig. 5, it can be deduced that, the model with attention module along with the gradient-map loss produces the sharpest results. The quality of the frames produced by the refinement network as compared to the original input frames and degraded frames (generated by 20 iterations of CAIN [2]) can be assessed through Fig. 10.

## 2.2 Gradient-map Loss

The gradient-map loss used in our application utilizes the conventional horizontal and vertical edge detection kernels in all RGB channels of the generated frames and the target frames to acquire the edge profiles. These edge profiles are concatenated in channel dimension to assist the model in effectively learning the underlying reasoning to generate artifact free and sharp images. This channel-wise convolution and concatenation operation is denoted by  $M(*)$  in Fig. 3 (Step 3). The obtained RGB edge profiles are compared with the help of a conventional  $\mathcal{L}_1$  loss.

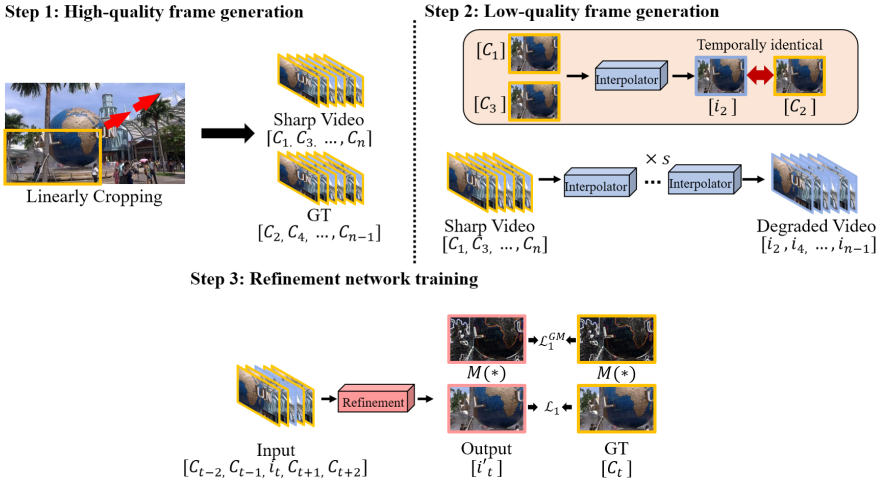


Figure 3: Training pipeline of the refinement network. (Step 1) High-quality frame generation through sequential linear cropping and splitting into two groups namely, sharp videos and GT. (Step 2) Introduction of artifacts to the training data through iterative frame interpolation. (Step 3) The training process of the refinement network with  $\mathcal{L}_1$  and Gradient-map Loss. The subscript  $t$  signifies temporal relation of instantaneous frames and their neighboring frames.

### 3 Details of Losses in Stage 3 (Strengthening)

This section highlights the details of additional losses and hyper parameters used in the stage 3 of training the stabilization network. The final loss for training at this stage is given by the following equation,

$$\mathcal{L} = \lambda_1 \cdot \mathcal{L}_\phi + \lambda_2 \cdot \mathcal{L}_{CX} + \lambda_3 \cdot \mathcal{L}_{td} + \lambda_4 \cdot \mathcal{L}_{id} + \lambda_4 \cdot \mathcal{L}_{cml}, \quad (1)$$

where,  $\mathcal{L}_\phi$ ,  $\mathcal{L}_{CX}$ ,  $\mathcal{L}_{td}$ ,  $\mathcal{L}_{id}$  and  $\mathcal{L}_{cml}$  represent perceptual, contextual, temporal discriminator, image discriminator and contrastive motion loss, respectively. At this stage of the training, the generator has already learnt to produce high-quality stable frames, but the stability and quality of the generated frames can be further enhanced with the introduction of targeted loss functions addressing each aspect of the video generation process. Specifically, we let the generator focus on strengthening the three main aspects of digital stabilization namely, *Stability*, *Naturalness* and *Content Preservation*. We will define each of the aspects and their devised losses in the following subsections.

#### 3.1 Stability

In order to further enhance the learnt stability, we propose a novel contrastive motion loss which utilizes an off-the-shelf video ResNet trained for action recognition [9] along with a triplet loss [10]. This loss does not require any prior or estimate of the motion in input sequence and defines it in abstract embeddings. To formulate this loss, we utilize the stable videos provided in the DeepStab [14] dataset. Notably, we cannot directly use stable videos from DeepStab dataset as these videos often contain a perspective mismatch. To

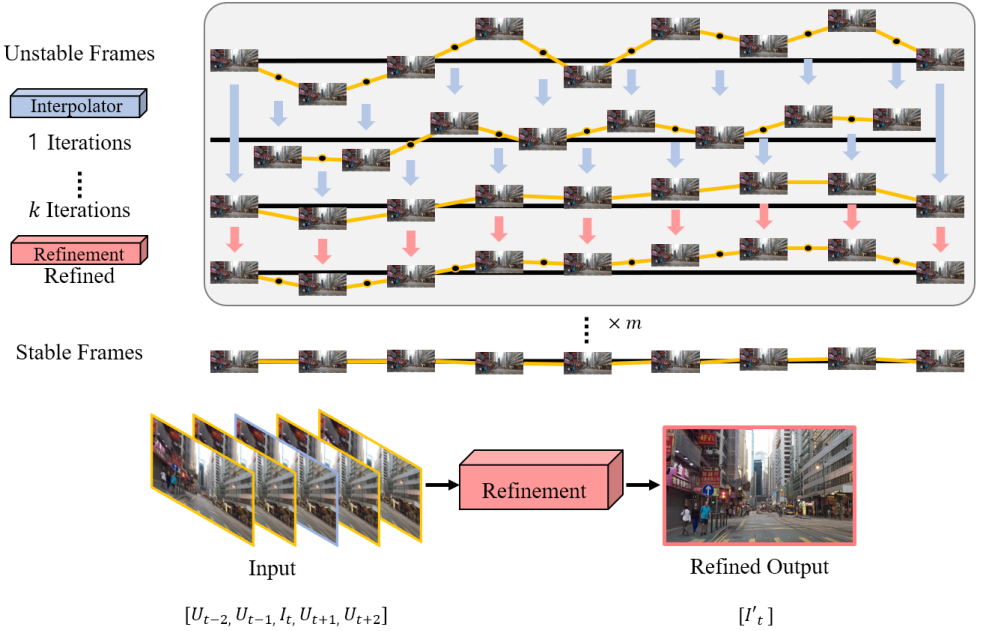


Figure 4: Our finalized dataset generation pipeline (DGP). The refinement module takes in the generated frames  $I_t$  after  $k = 4$  together with the original high-quality unstable frames  $U_{t-2}, \dots, U_{t+2}$  and generates the refined frame  $I'_t$ . This interpolation-refinement cycle is repeated for  $m = 5$  times to obtain temporally stable counterparts of the unstable videos.

overcome this problem, we firstly find the closest corresponding patch of unstable frames in stable video frame (with the help of template matching), and sequentially crop the stable frames to get a sequence of patches that closely correspond to the content of unstable patches. These unstable and stable patches are passed through the video ResNet to acquire negative and anchor embeddings, respectively. Meanwhile, the positive embeddings are acquired by processing the unstable patches through the stabilization network and the video ResNet sequentially. The triplet loss, minimizes the distance between the positive and the anchor embeddings while maximizing the same for anchor and negative embeddings. With the introduction of this loss, an average increase of 2 – 3% is observed in the final stability score. The contrastive motion loss is defined as follows,

$$\mathcal{L}_{\text{cml}} = \max(d(A, P) - d(A, N) + \alpha, 0), \quad (2)$$

$$d(x, y) = \|x - y\|_2, \quad (3)$$

where,  $A$ ,  $P$  and  $N$  denote anchor, positive and negative sequence embeddings, respectively, and the  $\alpha$  represents the margin parameter. In our formulation,  $\alpha$  was equated to 1.0. The weight parameter  $\lambda_4$  for this loss was 0.01, this parameter was evaluated empirically.

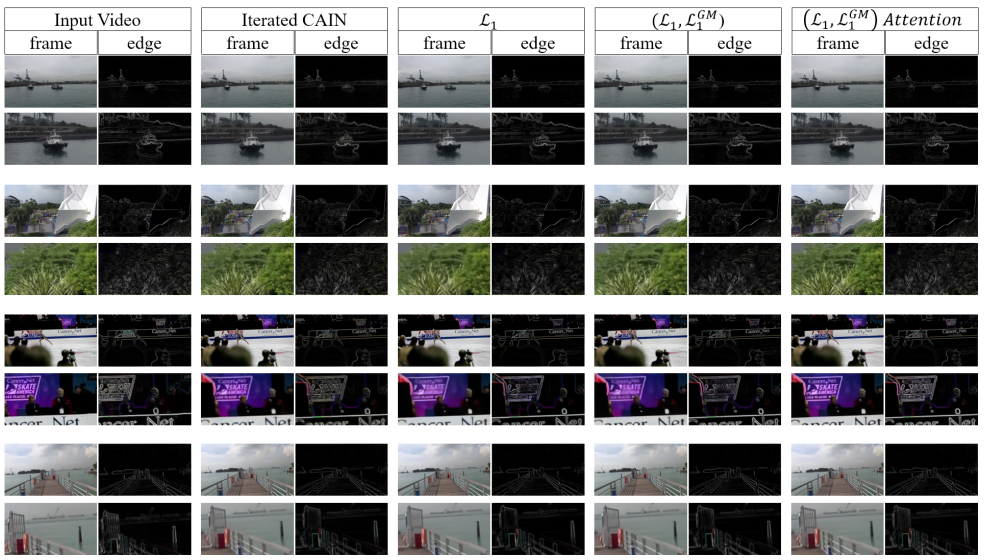


Figure 5: Sharpness comparison of the frames generated by the various configurations of the refinement network.

### 3.2 Naturalness

Up to this point, the model was trained using individual frames rather than sequences of frames. Generally, this frame dependant training coupled with adversarial learning strategies can often introduce undesirable temporal inconsistencies (e.g., wobble effect) in the generated videos as discussed in [15]. To overcome this problem, we propose a temporal discriminator which discriminates between a natural sequence (DeepStab [14] stable) and an artificial (generated) one. The temporal discriminator loss is defined as follows,

$$\mathcal{L}_{td} = E_{F_t'}[\log(D_t(F_t'))] + E_{F_t}[1 - \log(D_t(G(F_t)))], \quad (4)$$

where,  $D_t$  represents the temporal discriminator, whereas,  $F_t$  and  $F_t'$  denote sequences of DeepStab [14] stable and generated stable frames. Through this loss, we can minimize the temporal distortions in the generated videos. The contribution parameter for this loss was empirically equated to be 0.1.

### 3.3 Content Preservation

When using the proposed temporal discriminator and contrastive motion losses, the models can often learn to predict solid colored frames as they are stabler and temporally consistent. In order to overcome this problem we employ certain reconstruction losses [16, 17] along with an image discriminator to preserve the input image statistics. The Contextual loss [18] is a loss function that is used for various image tasks with unaligned targets. We define our contextual similarity in terms of VGG-19 features and use the unstable (unaligned) images as contextual similarity targets for the generated frames based on the fact that the stabilized videos should contain the content of input videos at different spatial locations. This loss is defined as follows,

$$\mathcal{L}_{CX} = -\log(CX(\phi^l(f_t'), \phi^l(f_t))), \quad (5)$$

where,  $f_i$  denotes the corresponding input frame of the stabilized frame  $f'_i$  and  $\phi^l$  represents features extracted through layer  $l$  (*relu\_3\_3*) of a pre-trained VGG-19 network. In addition to the Contextual loss  $CX(\cdot)$ , we also employ a VGG-based perceptual loss and an adversarial loss for generating high-quality frames with similar content to the input frames. The adversarial loss used is defined as follows;

$$\mathcal{L}_{id} = E_{f'_i}[\log(D_i(f'_i))] + E_{f_i}[1 - \log(D_i(G(f_i)))], \quad (6)$$

where,  $D_i$  and  $G$  represent Image discriminator and the stabilization network, respectively. The contribution parameters of both the perceptual and the image discriminator loss were set to 0.1 throughout the third stage of training.

## 4 Implementation Details

The proposed stabilization network is implemented using PyTorch on a system with two 2080Ti GPUs. In each stage, various augmentation techniques like random flipping both horizontally and vertically, resizing and reversing the frame order along with the conventional image processing augments like random brightness, hue, gamma, and contrast adjustment were employed to increase and modify the visual information present in the training samples. All the stabilization stages were optimized using ADAM optimizer, and the learning rates for each stage was 0.0001, 0.00005 and 0.00001, respectively. The learning rate was lowered in each successive stage in order to both retain and learn the different aspects of video stabilization. The complete training process (all three stages) takes around five days with roughly 70,000 training iterations in each step.

## 5 Run-time

Table 1. Time comparison	
DIFRINT [10]	528ms
Robust L1 [11]	600ms
CAIN (20-iterations) [12]	1802ms
DGP ( $k=4, m=5$ )	2300ms
Ours	210ms

Table 1: Runtime comparison table.

The average per frame ( $640 \times 360$  pixels) generation time comparison are provided in Tab. 1. Through Tab. 1 and the qualitative results presented in this supplemental, it can be deduced that our proposed model produces comparative results in a significantly lesser time. All the experiments are conducted on the same hardware and software environment as described in Sec. 4.

## 6 Stability Metric

This metric defines the stability in terms of frequency component analysis. To calculate this metric, the feature trajectories are analyzed in the frequency domain as follows

$$f_x = FFT(V_x), \quad (7)$$

where  $f_x$  is frequency representation of the translational and rotational camera trajectories  $V_x$ .  $f_x$  is acquired through the discrete 1D Fourier Transform of  $V_x$  after subtracting the DC component. Here,  $x$  represents both translational and rotational trajectories. The stability score is calculated according to the description provided by [12] as follows:

$$S_x = \sum_{k=2}^6 f_x(k) / \sum_{k=2}^n f_x(k), \quad (8)$$

where  $S_x$  represent the stability score for both the translational ( $S_t$ ) and rotational motions ( $S_\theta$ ), and  $n$  is the number of total frequency components present in the signal. The final stability score from both the calculated scores is acquired by taking the minimum as:

$$S_{\text{final}} = \min(S_t, S_\theta). \quad (9)$$

## 7 More Qualitative Results

Due to the space limitation, the qualitative results presented in the main manuscript were down-scaled. This supplemental also provides expanded view of the qualitative results presented in the manuscript, and highlights the common artifacts produced by conventional methodologies in figures 11–14.

## 8 Ablation Study

### 8.1 DGP hyper-parameter selection

Fig. 6 highlights the effects of iterative stabilization and refinement. Through this figure, it can be deduced that in the results acquired after 20 iterations ( $m \geq 5$ ), the stabilization effect becomes saturated and excessive use of the refinement network also introduces a fading effect in the produced dataset. Based on these observations, we select  $k$  and  $m$  to be 4 and 5 respectively throughout the process of dataset generation in our application.

### 8.2 Inter-stage ablation

After the first stage of training, the results produced by the proposed stabilization network are stable but quite blurry. By overlaying the generated frames with the target frames (acquired through DGP), it becomes evident that the model has learned the necessary high-level reasoning to generate stable frames, as highlighted in Fig. 7 (a)-(b), where the red marks indicate the positions of prominent corners in our target frames and the generated frames after *Stage 1*. Furthermore, Fig. 7 (c) presents the frames produced after the *Stage 2* of training. It is evident from this comparison that the results produced by the second stage retain the learned spatial relations during *Stage 1* and contain much sharper edges as well. Whereas,



Figure 6: A visual ablation for highlighting the effects of the iterative stabilization and refinement in various settings.

*Stage 3* increases the overall stability by approximately 2-3%, and reduces the temporal distortions significantly. The quality of the results produced after the *Stage 3* of training can be assessed through the accompanied videos.

### 8.3 Ablation of Losses

In order to properly evaluate the contribution of the specialized losses used in the training of our stabilization network we conducted a thorough ablation study and report our findings with the help of the table provided below along with visual samples obtained from inferring the mentioned models on one of the videos from the NUS dataset [9]. The table presents the stability, cropping, and distortion scores of models trained with various losses. In the table presented below,  $CML$ ,  $D_v$ ,  $D_i$  and  $C_x$  denote Contrastive Motion, video discriminator, image discriminator, and Contextual losses respectively. Whereas, "Stage - 1" and "Stage - 2" refer to the models obtained from the first two stages of training respectively.

Name	Stability	Cropping	Distortion
Stage - 1	0.9016	0.9998	0.9947
Stage - 2	0.8918	0.9999	0.9963
CML only	0.9060	0.9928	0.9928
CML + $D_v$	0.9059	0.9999	0.9925
CML + $D_v$ + ( $D_i$ + $C_x$ )	0.9057	0.9999	0.9975

Table 2: Quantitative comparison of the models trained with different loss functions.

Through Tab. 2 and Fig. 8, it can be seen that the highest and the best results (in terms of stability and visual quality) are acquired through the final combination of losses. The results produced through the models trained with only MSE loss (Stage - 1) produce quite blurry results but their stability scores are higher than the model trained with perceptual and adversarial loss of the second stage, this discrepancy occurs because the stability metric perceives blurry videos as smoother than the sharper videos. After the second stage of the

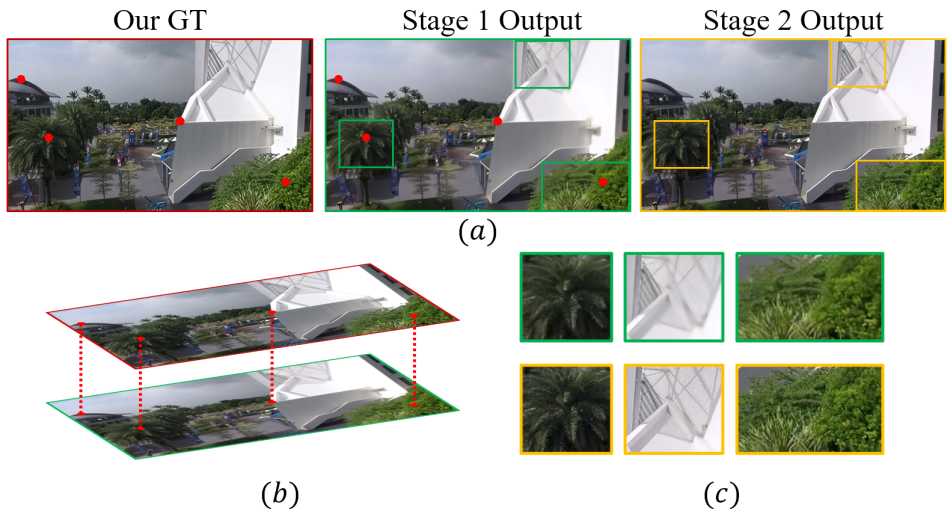


Figure 7: The results generated by our network after the both training stages. (a) The red indicators mark the same positions on Our GT and the Stage 1 results. (b) Highlights the alignment by overlaying the generated frames with the frames acquired by the Our GT after the first training stage. (c) Quality improvement after the second stage.

training, if the model is trained with only the contrastive motion loss, the model learns to compensate the jerky motion and a diminishing hue around the fast-moving objects can be observed, this is taken care of by introducing a temporal discriminator loss ( $D_v$ ) which helps to restore the natural temporal semantics of the generated videos but does not fully restore the integrity of the content present in the video streams. To tackle this issue, we introduce an image discriminator along with a contextual loss. Through the introduction of these losses, a striking restoration of original content integrity can be observed.

Furthermore, during our experimentation phase, an instance of our model was also introduced to a single-stage training with all the finalized losses, but it was unable to achieve the same level of stability as the models trained in multiple stages with different objectives in each stage.

## 8.4 Patch size

During our experiments with the stabilization network, we experimented with different patch sizes and deduced that the network trained with bigger patch sizes obtains similar stability score with the networks trained with a relatively smaller patch size. Therefore, to speed up the training process, we opt for a patch size of  $220 \times 220$  for our first 2 stages of training and further reduce it to  $160 \times 160$  for the third stage, to compensate for the calculation overhead added by the 16 sequential frames used in that training stage.

## 9 User Study

As stated in the main manuscript, the metrics used for this task do not take quality of the produced videos into account; therefore, a user study was conducted to properly evaluate

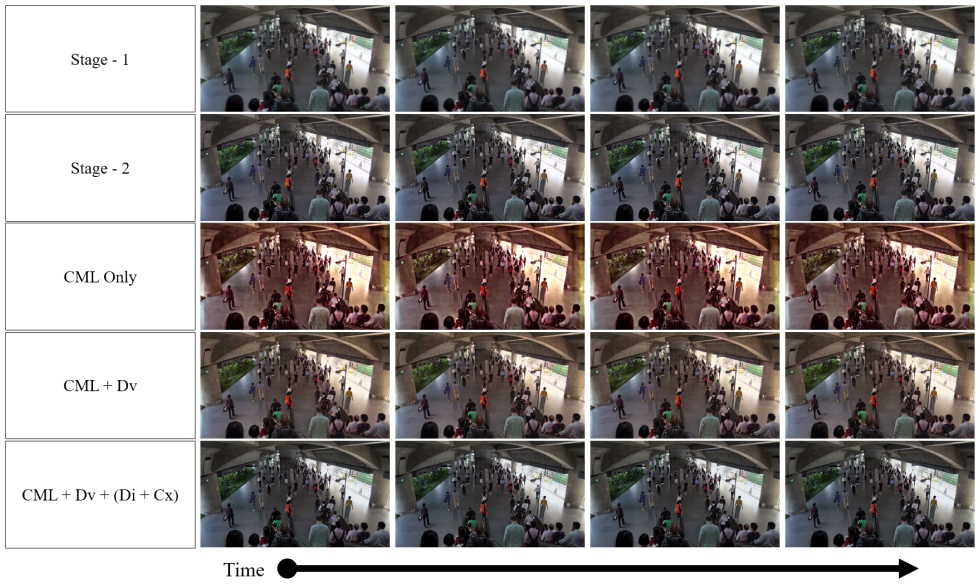


Figure 8: Qualitative comparison of the models trained with different loss functions. (best viewed on a computer screen)

the visual quality of the results generated by the proposed stabilization network. In order to conduct the user study, we created an application that plays four videos generated by Robust L1 [8], Adobe Premiere Pro 2018 CC, DIFRINT [14] and our stabilization network, simultaneously. The placement of videos generated through different methods is randomized for fair evaluation. The application selects the videos from an address pool containing paths to 144 stabilized videos (from NUS Dataset [8]) through each method. Each participant evaluates five randomly selected videos. The interface of the user study application allows the users to pause and replay videos. We asked the participants to judge the videos based on the three aspects, stability, quality and cropping. The user study was conducted with 31 participants, and the results of the user study are presented in Fig. 9. The repository containing the generated dataset, code, pre-trained models, metrics and user study application will be available on [https://github.com/MKashifAli/Motion\\_Blind\\_Video\\_Stabilization](https://github.com/MKashifAli/Motion_Blind_Video_Stabilization).

## 10 Video

Please refer to the provided supplementary videos for the comparative results of our stabilization network.

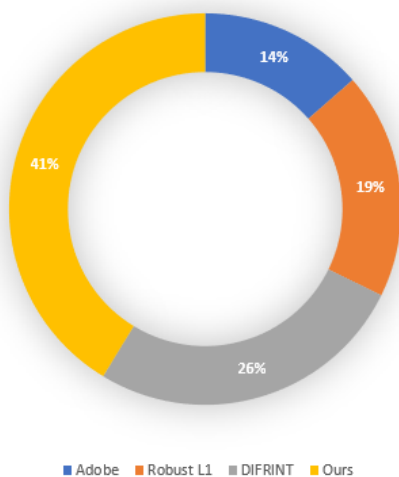


Figure 9: User study results of videos stabilized through Adobe Premiere Pro 2018 CC, Robust L1 [8], DIFRINT [10] and the proposed motion blind stabilization network.

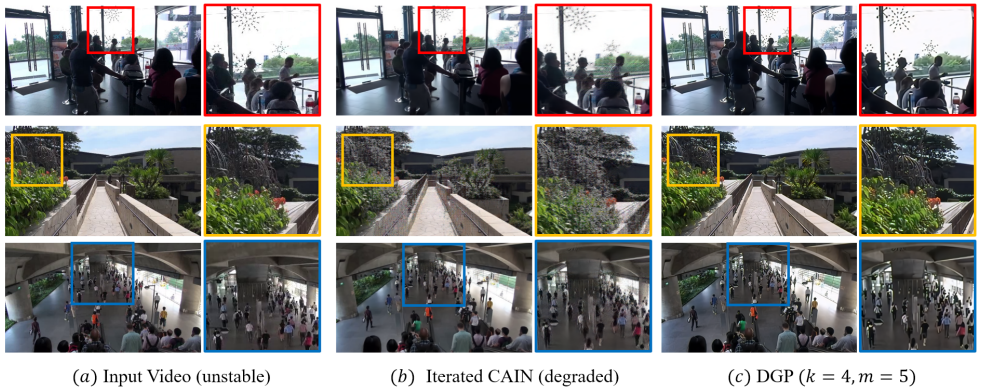


Figure 10: Quality comparison of the input (a), CAIN [2] 20 iterations (b) and the generated frames through the DGP ( $k=4, m=5$ ) (c).



Figure 11: Expanded view of the Fig. 7, Example 1 presented in the main paper for better qualitative comparison.



Figure 12: Expanded view of the Fig. 7, Example 2 presented in the main paper for better qualitative comparison.



Figure 13: Expanded view of the Fig. 7, Example 3 presented in the main paper for better qualitative comparison.

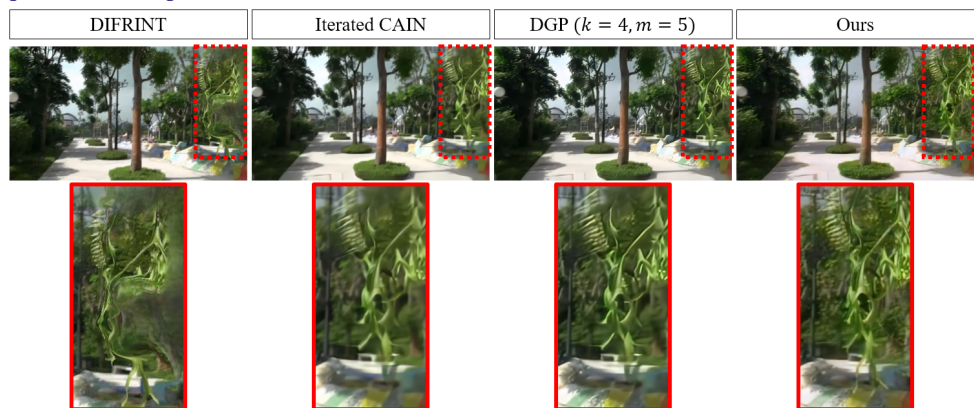


Figure 14: Expanded view of the Fig. 7, Example 4 presented in the main paper for better qualitative comparison.

## References

- [1] Jinsoo Choi and In So Kweon. Deep iterative frame interpolation for full-frame video stabilization. *ACM Transactions on Graphics (TOG)*, 39(1):1–9, 2020.
- [2] Myungsub Choi, Heewon Kim, Bohyung Han, Ning Xu, and Kyoung Mu Lee. Channel attention is all you need for video frame interpolation. In *AAAI*, pages 10663–10671, 2020.
- [3] Mengyu Chu, You Xie, Laura Leal-Taixé, and Nils Thuerey. Temporally coherent gans for video super-resolution (tecogan). *arXiv preprint arXiv:1811.09393*, 1(2):3, 2018.
- [4] Xingping Dong and Jianbing Shen. Triplet loss in siamese network for object tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 459–474, 2018.
- [5] Matthias Grundmann, Vivek Kwatra, and Irfan Essa. Auto-directed video stabilization with robust 11 optimal camera paths. In *CVPR 2011*, pages 225–232. IEEE, 2011.
- [6] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Learning spatio-temporal features with 3d residual networks for action recognition. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 3154–3160, 2017.
- [7] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.
- [8] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 136–144, 2017.
- [9] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Bundled camera paths for video stabilization. *ACM Transactions on Graphics (TOG)*, 32(4):1–10, 2013.
- [10] Cheng Ma, Yongming Rao, Yean Cheng, Ce Chen, Jiwen Lu, and Jie Zhou. Structure-preserving super resolution with gradient guidance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.
- [11] Roey Mechrez, Itamar Talmi, and Lihi Zelnik-Manor. The contextual loss for image transformation with non-aligned data. In *Proceedings of the European conference on computer vision (ECCV)*, pages 768–783, 2018.
- [12] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3883–3891, 2017.
- [13] Mehdi SM Sajjadi, Bernhard Scholkopf, and Michael Hirsch. Enhancenet: Single image super-resolution through automated texture synthesis. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4491–4500, 2017.
- [14] Miao Wang, Guo-Ye Yang, Jin-Kun Lin, Song-Hai Zhang, Ariel Shamir, Shao-Ping Lu, and Shi-Min Hu. Deep online video stabilization with multi-grid warping transformation learning. *IEEE Transactions on Image Processing*, 28(5):2283–2292, 2018.