2024 IEEE International Symposium on Biomedical Imaging (ISBI) | 979-8-3503-1333-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISBI56570.2024.10635490

DIVIDE AND CONQUER: VIDEO INPAINTING FOR DIMINISHED REALITY IN LOW-RESOURCE SETTINGS

Zhou Ji Wu* Ryosuke Seshimo* Hideo Saito Mariko Isogawa

Keio University

ABSTRACT

Although recent deep learning-based inpainting techniques achieve excellent restoration quality, their stringent requirement for computational resources such as GPU/VRAM render them difficult to use in settings where high-end hardware is not available. We address this gap between research and real-world applications in our submission to the DREAMING challenge, which explores such a use case – diminished reality – where run time and GPU hardware are limited. Specifically, this paper proposes a method that divides a video optimally into sections of variable lengths for the downstream inpainting model. By maximizing the number of frames, i.e., spatio-temporal information, in each subvideo, inference using a SOTA model can often be performed without significant degradation to output quality. In addition, to expedite inference, we examine techniques that intelligently reduce the amount of input pixels, e.g., downsampling and cropping, while maintaining decent inpainting quality.

***Index Terms*—** Video inpainting, diminished reality, object removal, oral and maxillofacial surgery, deep learning

1. INTRODUCTION

Recently, diminished reality (DR), the technique for removing regions from a scene and replacing them with their background, has been gaining attention for its enormous potential in a variety of fields, including medicine [1, 2], robotics [3], and automobile safety [4]. In a scene enhanced by DR, the virtual content replacing a removed object is assumed to be a reconstruction of the real environment in the background, which often has been visible, or otherwise known, to observers at an earlier time or from another angle. Hence, in general, observers expect a seamless transition and overall plausibility in the diminished visual field.

These characteristics of DR make it useful in medical applications such as anatomy education [5]. For instance, when a surgeon's view of a patient is obstructed by medical instruments, DR can be used to generate an uninterrupted view of the operation site [6]. As one of the applications that can leverage the benefits of DR, the DREAMING challenge focuses on removing unwanted regions applied to oral and maxillofacial surgery where regions of interest such as faces are concealed by medical instruments and hands holding them. The challenge dataset consists of videos from the viewpoint of surgeons in an operating room setting. Such videos, when enhanced with DR, are valuable sources of educational material and could serve as records to facilitate exchange of information between healthcare professionals in after-action reviews.

{jameswu, seshimo.ryosuke, hs, mariko.isogawa}@keio.jp

*Equal contribution

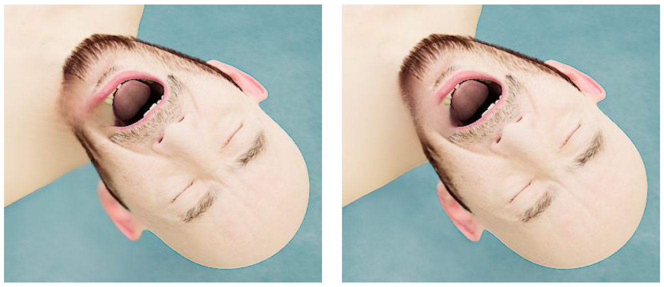
Fig. 1. Left: baseline method; right: our method

Since the challenge requires an algorithm to generate diminished views from photos taken with a single stationary camera, we chose to build our solution using video inpainting techniques; we explain our choice in Sec. 2. Recent state-of-the-art (SOTA) inpainting algorithms include ProPainter [7], DeepDR [1], DynaFill [8], DeepFillv2 [9], and latent diffusion [10]. While they produce excellent results, many of them rely on neural models that require powerful GPUs to run within a reasonable amount of time, as they are designed to process a large number of frames at once to achieve high-quality inpainting. A common shortcoming of these algorithms is that the level of hardware they require, e.g., VRAM capacity, is often not available in a real-world setting. We saw a need for research to bridge this gap to enable better trade-off between restoration quality and computational resources. Our solution to the DREAMING challenge consists of methods that allow the aforementioned models to be used effectively in an environment with limited hardware and run time.

Our main contributions are as follows: 1) given some hardware constraints such as VRAM capacity, we propose a memory estimation technique that allows a video to be divided into sections of variable, optimal lengths, where the number of frames that can be processed simultaneously is maximized – thereby providing more spatio-temporal information to facilitate inpainting; and 2) we examine strategies that speed up inference by reducing the amount of input pixels via downsampling and cropping, while maintaining decent output quality.

2. RELATED WORK

Traditionally, the hidden view generation task in DR is implemented through novel-view synthesis from images taken at different angles capturing the hidden area [2, 4]. When a multi-view camera system or an RGB-D camera is available, 3D reconstruction techniques can be leveraged to extract 3D geometry and depth maps [2, 11, 12, 8]. When only one camera is available and the hidden background is unobservable, inpainting algorithms, which can fill a hole using only neighboring pixels, are preferable [2, 13].

One of the earliest papers that laid the foundation for deep learning-based methods is Context Encoders [14], which demonstrated strong inpainting performance using an encoder-decoder architecture with adversarial loss. DeepFillv2 [9] is another seminal paper, introducing techniques that allow free-form masks and a patch-based GAN loss which greatly enhanced performance. RGB-D inpainting is an important area of research for DR because the latter requires coherent geometric structure (e.g., depth). DeepDR [1] is SOTA in this regard, enabling inpainting not only with coherent 3D structure and minimal temporal artifacts but also at real-time frame rates.

Current mainstream techniques used in video inpainting include video transformers [15, 16, 17] and flow-guided propagation [18, 19, 16]. ProPainter [7] combines and improves upon these ideas – particularly in efficiency – to attain the latest SOTA performance. However, despite such gains, using ProPainter under the DREAMING challenge's hardware constraints still causes out-of-memory errors, unless the input is shortened to only a handful of frames – which produces not only weak inpainting results but also fails to complete the task within the time limit. Our methods aim to alleviate this issue, as we describe in the next section.

3. METHODOLOGY

Given a video sequence $X = \{x_t \in \mathbb{R}^{W*H*3}\}_{t=1}^{T}$ to be inpainted and a corresponding mask sequence $M = \{m_t \in \mathbb{R}^{W*H*1}\}_{t=1}^{T}$ with video length T, our goal is to generate a sequence of inpainted images $Y = \{y_t \in \mathbb{R}^{W*H*3}\}_{t=1}^{T}$.

Intuitively, processing more frames at once is expected to produce better inpainted result since frames at different time intervals likely contain new information that helps to fill a hole. To enhance information aggregation across the temporal dimension, we developed a method to estimate VRAM usage and dynamically divide a video into sections of variable lengths (Sec. 3.1). The main idea is to take a mask sequence as input and determine optimal points of division such that the number of frames in each section is maximized. Our implementation uses ProPainter [7] as the backbone of our inpainting pipeline. We modified its code base to optimize VRAM usage, thereby allowing more frames to be processed simultaneously (Sec. 3.4). Our pipeline is shown in Fig. 2.

To expedite inference, we incorporated resizing algorithms as well as cropping algorithms (Sec. 3.2) that intelligently reduce the size of the input to the downstream inpainting model. Finally, we fine-tuned ProPainter on the challenge dataset (Sec. 3.3).

Our code is available at `github.com/wujameszj/divide_and_conquer`.

3.1. Dynamic video division based on mask region

In this competition, an input video contains more than 700 frames at 1280×720 resolution, while VRAM usage is limited to 16 GiB. As mentioned above, video inpainting consumes a lot of memory, so the entire video cannot be processed at once. This problem can be addressed by extracting only a certain number of frames from the video and processing them separately. In this case, processing many frames at once allows high-quality restoration because it includes more time-series information. Therefore, the goal is to maximize the number of frames processed at a time while limiting the amount of VRAM used.

This approach dynamically determines the number of frames to be processed at a time based on the size of the mask region, and aims to achieve high-quality restoration without exceeding the VRAM limit. The Mask-Guided Sparse Video Transformer (MSVT) in ProPainter [7] uses mask images to reduce the transformer's queries, thus reducing computational complexity. Specifically, the mask image is first downsampled and summed with the downsampled masks of neighbouring frames. Then, only the window of the feature map at the same location as the window where the value is greater than 0 is used as the transformer's query. In other words, a filter is created to determine whether the frame of interest and its neighbouring frames have ever contained a mask region and whether it should be used as a query. This filter is called sparse mask.

Since the sparse mask area is calculated by summing with neighboring frames, multiple sparse mask areas are calculated within the number of frames processed at a time. Under the condition that the number of frames is constant, there is a linear relationship between the maximum sparse mask area and the maximum VRAM usage. Therefore, setting an upper limit for the sparse mask area S^{limit} and ensuring $S^{max} < S^{limit}$, where S^{max} is the maximum sparse mask area of a given set of frames, prevents the VRAM usage limit from being exceeded. As the number of frames processed simultaneously

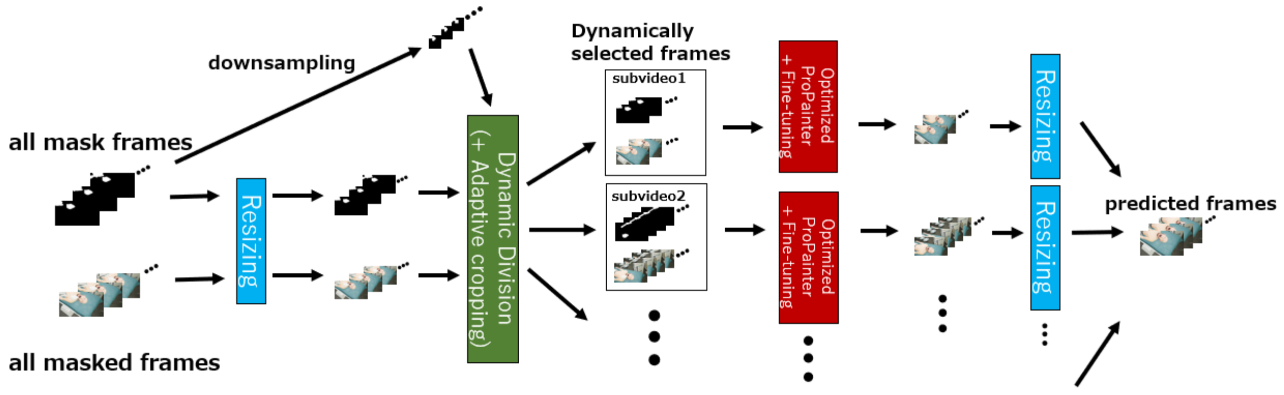

Fig. 2. Flow chart of our proposed approach

increases, the VRAM usage also increases, so the upper limit S_i^{limit} for processing i frames at a time is greater than the upper limit S_j^{limit} for processing j frames, where $i < j$.

The method for determining the number of frames to use is as follows. First, start with a sufficiently small number of frames x and check that the largest area of the sparse mask at that time S^{max} is $S^{max} < S_x^{limit}$. Then, increase the number of frames by one and calculate the largest sparse mask, and verify that $S_{max} < S_{x+1}^{limit}$. Continue the same process until the inequality no longer holds, and then subtract 1 from the number of frame at that time to determine the maximum number of frames that can be used. S^{limit} used in this method is obtained by experiment and is described in detail in Sec. 4.1.

3.2. Run time reduction

Video inpainting algorithms using deep learning tend to have high computational cost. However, many use cases, like the one simulated by the DREAMING challenge, have constraints on run time and hardware. Hence, we prepare two options to speed up inference: resizing and adaptive cropping.

Resizing. If the resolution of the original image is high, the input size to the deep learning model will be large and, consequently, inference will take a long time. Downsampling the image before sending it to the deep learning model and then restoring the output to the original resolution can reduce the run time. And since VRAM usage for processing the same number of frames will be reduced, this strategy also increases the number of frames that can be processed at once, which means that more temporal information can be utilized.

Adaptive cropping. In this approach, only the cropped section of each frame is sent to the inpainting pipeline. This effectively speeds up inference because there are less pixels to process in total. At the same time, a lower resolution allows more frames to be processed at once, which would improve inpainting results in cases where pixels in distant frames are particularly useful for filling a ROI.

3.3. Fine-tuning

We fine-tuned the pretrained ProPainter model [7] on the challenge dataset, using the Adam optimizer and an initial learning rate of 5e-5. The videos were resized to 728×408 pixels due to hardware limitations.

3.4. VRAM usage optimization

We modified ProPainter's inference pipeline to optimize VRAM usage – doubling the number of frames that can be processed simultaneously. We achieved this improvement by sending only relevant tensors to the GPU – and only during computation – while keeping tensors on CPU/RAM otherwise. In addition, we prevented memory leaks by promptly releasing memory held by tensors after their final usage.

4. EXPERIMENTS AND RESULTS

4.1. Preliminary Experiment: Sparse Mask threshold

As discussed in Sec. 3.1, when the number of frames processed at once is the same, the area of the sparse mask and VRAM usage have a linear relationship. Based on this relationship, the threshold for the upper limit of the sparse mask area S^{limit} can be determined. Specifically, a video is split into parts with a fixed number of frames N (N=80, 100, 120, etc.), and the corresponding area of the sparse mask and VRAM usage are measured. From these data, a linear equation between the area of the sparse mask and VRAM usage was obtained. To ensure that the upper limit of VRAM usage was not exceeded, the linear equation was calculated by 95% quantile regression instead of the least-squares method. The measurements are taken on only the first five scenes in the dataset due to limited computational resources. At the end, we observed that both the coefficient and the intercept of the linear equation had a linear relationship with the number of frames, i.e., subvideo length. This relationship can be used

Table 1. Quantitative comparison and ablation study on 10 test scenes. Dynamic division (D), resizing (R), fine-tuning (F), and memory optimization (M).

	LPIPS ↓	FID ↓	MAE ↓	PSNR ↑
Baseline	0.0643	24.62	0.3490	37.61
w/o D	0.0607	21.22	0.3471	37.72
w/o F	0.0610	21.75	0.3494	37.68
w/o R&M	0.0608	21.96	0.3473	37.70
w/o R	**0.0605**	21.94	0.3474	37.71
Full	**0.0605**	**20.95**	**0.3471**	**37.73**

to determine the thresholds for the upper limit of the sparse mask area for any given set of frames.

4.2. Baseline

For the baseline, we split each scene into groups of 20 temporal frames, which is the maximum amount on which the original ProPainter framework can be consistently run without OOM errors using the GPU on the challenge platform. We ran ProPainter on each group of frames separately, combined the frames back into a full scene, and evaluated the results.

4.3. Quantitative Evaluation

We report metrics on 10 randomly selected scenes. We would have liked to conduct k-fold cross-validation, but due to limited computational resources, we conduct quantitative evaluation only on this set of 10 scenes.

As shown in Tab. 1, our method outperforms the baseline in LPIPS, FID, MAE, and PSNR. Moreover, while the baseline takes 145.3s on average to process 100 frames, our full method takes 106.0s. Therefore, our full method is quantitatively better and faster than the baseline method.

4.4. Qualitative Evaluation

As shown in Fig. 1 and 3, while the baseline has unnatural artifacts, our solution produces clearer edges with fewer artifacts in the facial region.

4.5. Ablation study

Our solution combines four techniques: dynamic division (D), resizing (R), fine-tuning (F), and memory optimization (M). We perform an ablation study by removing one component at a time from the full solution, except for M which had to be removed with R together. Without dynamic division, all subvideos are restricted to a length of 80 frames regardless of the sparse mask area, so as not to exceed the VRAM limit. As shown in Tab. 1, the results worsen when any element is removed from our full method. Therefore, all four techniques contribute to improved results.

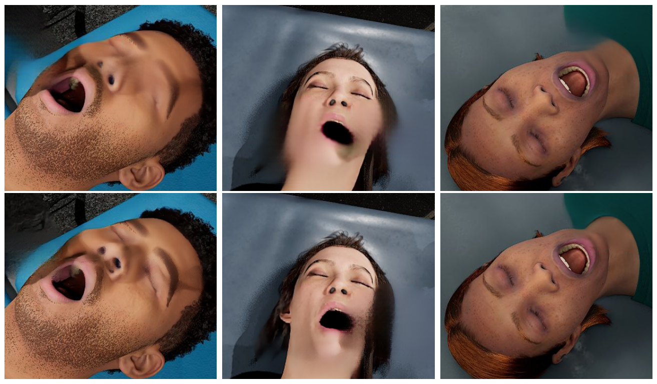

Fig. 3. Qualitative comparison. Top row: baseline method; bottom row: our method

Dynamic division (Sec. 3.1) and memory optimization (Sec. 3.4) lead to better results because they both increase the number of frames that can be processed at once, while fine-tuning (Sec. 3.3) improves results as it allows the model to learn the data distribution of the challenge dataset.

Resizing (Sec. 3.2) shortens the run time drastically, as expected, from 145.0s to 106.0s per 100 frames. It also, surprisingly, improves output quality. We hypothesize that this is because the advantage of being able to process more frames at once – more temporal cues to guide inpainting – overweighs the disadvantage of information loss from downsampling. Finally, we observed that adaptive cropping (Sec. 3.2) did shorten run time considerably but also severely reduced inpainting quality – likely due to overly sacrificing information in the height and width dimensions.

5. CONCLUSION

In this study, we demonstrated both quantitatively and qualitatively that our methods enabled inference using a SOTA model within a reasonable amount of time on GPU hardware with limited VRAM – all while maintaining a high level of inpainting accuracy and consistency in the context of diminished reality. Although we based our solution on ProPainter [7], our idea of estimating memory usage and dynamically determining the lengths of subvideos can be applied to a variety of video processing tasks, not only limited to inpainting, e.g., super resolution, video stabilization.

6. FUTURE WORK

Currently, dynamic division (Sec 3.1) only considers masks of local frames, with no reference to non-local frames. Implementing the latter will likely improve results.

7. COMPLIANCE WITH ETHICAL STANDARDS

We use a synthetic dataset provided by the competition organizers and no ethical approval is required.

8. ACKNOWLEDGMENTS

We thank the organizers of the DREAMING challenge for providing us with an environment to learn and test our ideas. We enjoyed working on the problem and competing with other teams across the world. We hope the findings in this paper, as well as our open-sourced code base, will contribute back to the community by facilitating future research and applications in relevant fields. We also wish to thank the authors of ProPainter for open-sourcing their well-documented code, which made it easy for us to adapt the library for the challenge. Last but not least, we thank our colleagues from the Hyper Vision Research Lab and the Integrated Sensing and Media Computing Lab at Keio University for their support and insightful discussions.

No funding was received for this study. The authors have no relevant financial or non-financial interests to disclose.

9. REFERENCES

[1] Christina Gsaxner, Shohei Mori, Dieter Schmalstieg, Jan Egger, Gerhard Paar, Werner Bailer, and Denis Kalkofen, "Deepdr: Deep structure-aware rgb-d inpainting for diminished reality," arXiv preprint arXiv:2312.00532, 2023.

[2] Shohei Mori, Sei Ikeda, and Hideo Saito, "A survey of diminished reality: Techniques for visually concealing, eliminating, and seeing through real objects," IPSJ Trans. Computer Vision and Applications, vol. 9, pp. 17, 2017.

[3] Kazuya Sugimoto, Hiromitsu Fujii, Atsushi Yamashita, and Hajime Asama, "Half-diminished reality image using three rgb-d sensors for remote control robots," in 2014 IEEE SSRR, 2014, pp. 1–6.

[4] Peter Barnum, Yaser Sheikh, Ankur Datta, and Takeo Kanade, "Dynamic seethroughs: Synthesizing hidden views of moving objects," in 2009 8th IEEE ISMAR, 2009, pp. 111–114.

[5] Naoto Ienaga, Felix Bork, Siim Meerits, Shohei Mori, Pascal Fallavollita, Nassir Navab, and Hideo Saito, "First deployment of diminished reality for anatomy education," in 2016 IEEE ISMAR-Adjunct, 2016, pp. 294–296.

[6] Jan Egger and Xiaojun Chen, Computer-aided oral and maxillofacial surgery: Developments, applications, and future perspectives, Academic Press, 2021.

[7] S. Zhou, C. Li, K. K. Chan, and C. Change, "Propainter: Improving propagation and transformer for video inpainting," in 2023 IEEE/CVF ICCV. 2023, pp. 10443–10452, IEEE Computer Society.

[8] Borna Bešić and Abhinav Valada, "Dynamic object removal and spatio-temporal rgb-d inpainting via geometry-aware adversarial learning," IEEE Transactions on Intelligent Vehicles, vol. 7, no. 2, pp. 170–185, 2022.

[9] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang, "Free-form image inpainting with gated convolution," in Proceedings of the IEEE ICCV, 2019, pp. 4471–4480.

[10] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer, "High-resolution image synthesis with latent diffusion models," 2021.

[11] François Rameau, Hyowon Ha, Kyungdon Joo, Jinsoo Choi, Kibaek Park, and In So Kweon, "A real-time augmented reality system to see-through cars," IEEE TVCG, vol. 22, no. 11, pp. 2395–2404, 2016.

[12] S. Zokai, J. Esteve, Y. Genc, and N. Navab, "Multi-view paraperspective projection model for diminished reality," in The Second IEEE and ACM ISMAR, 2003. Proceedings., 2003, pp. 217–226.

[13] Otto Korkalo, Miika Aittala, and Sanni Siltanen, "Light-weight marker hiding for augmented reality," in 2010 IEEE ISMAR, 2010, pp. 247–248.

[14] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros, "Context encoders: Feature learning by inpainting," in 2016 IEEE Conference on CVPR, 2016, pp. 2536–2544.

[15] R. Liu, H. Deng, Y. Huang, X. Shi, L. Lu, W. Sun, X. Wang, J. Dai, and H. Li, "Fuseformer: Fusing fine-grained information in transformers for video inpainting," in 2021 IEEE ICCV, 2021, pp. 14020–14029.

[16] Zhen Li, Cheng-Ze Lu, Jianhua Qin, Chun-Le Guo, and Ming-Ming Cheng, "Towards an end-to-end framework for flow-guided video inpainting," in 2022 IEEE/CVF Conference on CVPR, 2022, pp. 17541–17550.

[17] Kaidong Zhang, Jingjing Fu, and Dong Liu, "Flow-guided transformer for video inpainting," in ECCV. Springer, 2022, pp. 74–90.

[18] Rui Xu, Xiaoxiao Li, Bolei Zhou, and Chen Change Loy, "Deep flow-guided video inpainting," in 2019 IEEE/CVF CVPR, 2019, pp. 3718–3727.

[19] Chen Gao, Ayush Saraf, Jia-Bin Huang, and Johannes Kopf, "Flow-edge guided video completion," in Computer Vision – ECCV 2020, 2020, pp. 713–729.