AutoRemover: Automatic Object Removal for Autonomous Driving Videos

Rong Zhang,[1*] Wei Li,[2,4†] Peng Wang,[2] Chenye Guan,[2] Jin Fang,[2]
Yuhang Song,[3] Jinhui Yu,[1] Baoquan Chen,[4] Weiwei Xu,[1†] Ruigang Yang[2]
[1]Zhejiang University, [2]Baidu Research, Baidu Inc.
[3]University of Southern California, [4]Peking University
cadzhangrong@zju.edu.cn, {liweimcc, jerryking234}@gmail.com, {guanchenye, fangjin}@baidu.com,
yuhangso@usc.edu, {jhyu, xww}@cad.zju.edu.cn, baoquan@pku.edu.cn, ryang2@outlook.com

Abstract

Motivated by the need for photo-realistic simulation in autonomous driving, in this paper we present a video inpainting algorithm *AutoRemover*, designed specifically for generating street-view videos without any moving objects. In our setup we have two challenges: the first is the shadow, shadows are usually unlabeled but tightly coupled with the moving objects. The second is the large ego-motion in the videos. To deal with shadows, we build up an autonomous driving shadow dataset and design a deep neural network to detect shadows automatically. To deal with large ego-motion, we take advantage of the multi-source data, in particular the 3D data, in autonomous driving. More specifically, the geometric relationship between frames is incorporated into an inpainting deep neural network to produce high-quality structurally consistent video output. Experiments show that our method outperforms other state-of-the-art (SOTA) object removal algorithms, reducing the RMSE by over 19%.

1 Introduction

With the explosive growth of AI robotic techniques, especially the autonomous driving (AD) vehicles, countless images or videos as long as other sensor data are captured daily. To fuel the learning-based AI algorithms (such as perception, scene parsing, planning) in those intelligence systems, a large number of annotated data are still in great demand. Thus, building virtual simulators for saving massive efforts on labeling and processing the captured data are essential to make the data best used for various AD applications (Alhaija et al. 2018; Seif and Hu 2016). One basic procedure in those applications is removing the unwanted or hard-to-annotate parts of the raw data, a.k.a the object removal or image/video inpainting. As shown in Figure 1, with the developed simulation system in (Li et al. 2019), the background image obtained by removing the foreground vehicles can be used to synthesize new traffic images with annotations or reconstruct 3D road models with clean textures, which is one of the desirable ways for data augmentation.

The image inpainting problem has been widely investigated, which also forms the basis of video inpainting.

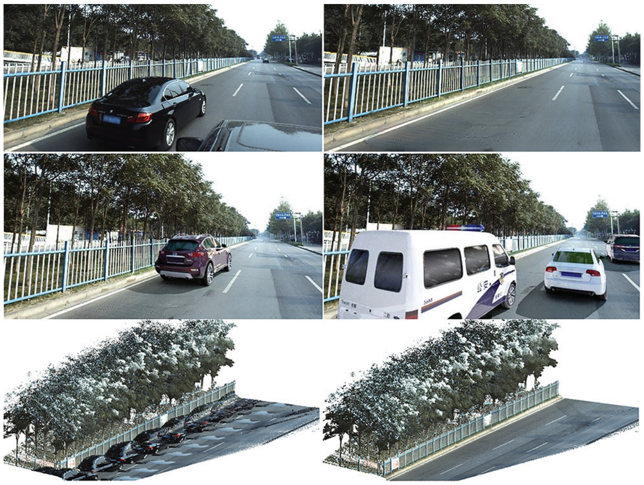


Figure 1: 1st row shows the source image and inpainted one from a video. 2nd row shows the usage of inpainting in data augmentation and simulator. With the inpainted background, the vehicle can be moved or inserted to synthesize new traffic images. 3rd row shows inpainted videos are used to yield 3D model with clean texture.

Technically, image inpainting algorithms either utilize similar patches in the current image to fill the hole by the optimization-based methods or directly hallucinate from training images by the learning-based methods. Recently, the CNNs, especially GANs, hugely advanced the image inpainting technique (Pathak et al. 2016; Iizuka, Simo-Serra, and Ishikawa 2017; Yu et al. 2018a), yielding visually plausible and impressive results. However, directly applying image inpainting techniques to videos suffers from jittering and inconsistency. Thus, different kinds of temporal constraints are introduced in recent video inpainting approaches (Huang et al. 2016; Xu et al. 2019), whose core is jointly estimating optical flow and inpainting color.

Even several video inpainting systems have been proposed in the very close recent, their target scenarios are usually with only small camera ego-motion in the behind of foreground objects movements, where the flow between frames are easy to estimate. Unfortunately, the videos captured by AD vehicles have large camera ego-motion (Fig-

[*] The authors from Zhejiang University are affiliated with the State Key Lab of CAD&CG.

[†] Corresponding authors.

ure 2 shows the statistics comparison of the optical flows). In addition, the camera ego-motion is usually moving along the heading direction of vehicles, which is also close to camera optical direction. The large vehicle movement and camera projection effect lead to large invisible parts in frames by surrounding vehicles. Moreover, the shadows of foreground objects are either ignored or manually labelled in those system, which does not work in AD scenario obviously.

In this paper, we propose a novel CNN-based object removal algorithm to automatically clean AD videos. The key idea is to utilize 3D convolution to extend the 2D contextual attention in (Yu et al. 2018b) to video inpainting. Specifically, we construct the system using three novel modules: temporal warping, 3D feature extractor and 3D feature assembler. The first module takes the advantage of multi-sensor data to help inpainting. While the last two modules are used to utilize temporal as well as contextual attention (CA) information for inpainting. Technically, naively combining temporal information and CA module is impractical due to large GPU memory footprint. We solve this problem by decomposing and simplifying the CA procedure.

With regarding to the shadow problem, which is always overlooked in previous inpainting literature, we propose a learning-based module along with an annotated shadow dataset. Our dataset has 5293 frames, which exceeds the SBU (Vicente et al. 2016), UCF (Vicente, Hoai, and Samaras 2015) and ISTD dataset (Wang, Li, and Yang 2018) w.r.t the size. Furthermore, ours advance those datasets in terms of our temporal consistent shadow annotation. Thus, our dataset could be beneficial to more vision tasks compared with typical shadow datasets, e.g. object tracking refinement, illumination estimation.

In summary, our contributions are as follows:

- We introduce an end-to-end inpainting network which consists of temporal warping, 3D feature extractor and assembler modules to utilize not only temporal but also contextual attention (CA) information for video inpainting, which is experimentally proven to be efficient to the results.
- We design an indispensable branch to deal with the shadows in AD videos. Our experiments show that it is a must-have module for high-quality results, and flexible to transfer to other SOTA algorithms.
- We announce a dataset of shadow annotated video AD videos. To the best of our knowledge, our dataset is the first temporal oriented (video) dataset with the largest number of annotated frames.

2 Related Work

2.1 Image Inpainting

Single image inpainting aims to reconstruct the lost parts in images. Patch-based inpainting methods are developed to better reconstruct the contextual structures in images (Barnes et al. 2009; Telea 2004; Sun et al. 2005; Hays and Efros 2007; Huang et al. 2014). These methods aimed at finding the best-matching patches with structural similarity to fill the missing regions.

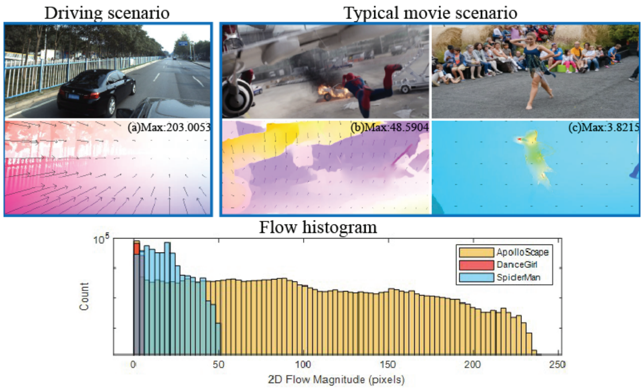


Figure 2: Visualized flows comparison. The first two rows show RGB images and visualized flow maps on which the arrows represent the flow directions and magnitudes from different videos. (a) is from our dataset while (b)(c) are from other papers (Xu et al. 2019; Huang et al. 2016). The last column is the flow histogram. It can be seen that the camera motion of our data is quite large.

The emergence of deep learning inspires recent works to investigate various deep architectures for image inpainting. Learning-based image inpainting directly learns a mapping to predict the missing information (Xie, Xu, and Chen 2012; Köhler et al. 2014; Ren et al. 2015). By interpreting images as samples from a high-dimensional probability distribution, image inpainting can be realized by generative adversarial networks (Goodfellow et al. 2014; Radford, Metz, and Chintala 2015; Mao et al. 2016; Mroueh and Sercu 2017; Arjovsky, Chintala, and Bottou 2017; Pathak et al. 2016; Iizuka, Simo-Serra, and Ishikawa 2017).

Most recently, Yu *et al.* (Yu et al. 2018b) presented a contextual attention mechanism in a generative inpainting framework, which improved the inpainting quality. They further extended it to free-form masks inpainting with gated convolution and SN-PatchGAN (Yu et al. 2018a).

These methods achieve excellent image inpainting results. Extending them directly to videos is, however, challenging due to the lack of temporal constraints modeling.

2.2 Video Inpainting

Video inpainting is generally viewed as an extension of the image inpainting task with larger search space and temporally consistent constraints. Extended from patch-based image inpainting, video inpainting algorithms (Wexler, Shechtman, and Irani 2007; Granados et al. 2012; Newson et al. 2014; Ebdelli, Le Meur, and Guillemot 2015) recovered masked regions by pasting the most similar patches somewhere in the video. By estimating the optical flow and color jointly, Huang *et al.* (Huang et al. 2016) formulated video inpainting problem as a non-parametric patch-based optimization in a temporally coherent fashion. However, the computation time of these methods is still long. In addition, patch-based models still lack modeling distribution of real images, so they fail to recover unseen parts in the video.

Recently, learning-based video inpainting also gains dra-

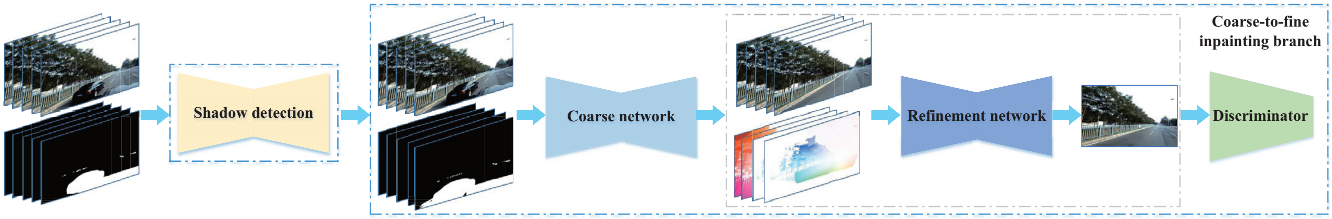


Figure 3: The pipeline of our approach, which consists of a shadow detection branch and a coarse-to-fine inpainting branch. The shadow detection extends the input object masks to cover their shadows. The inpainting branch inpaints the extended masks in a coarse-to-fine fashion, where the coarse network provides blurry predicts and the refinement network to detailed inpaint the target frame with assembled multi-frames information under the flow guidance.

matic improvements. Wang *et al.* (Wang et al. 2018) introduced the first learning-based video inpainting framework to predict 3D temporal structure with 3D convolutions implicitly and infer 2D details combined with 3D information. Ding *et al.* (Ding et al. 2019) and Chang *et al.* (Chang, Liu, and Hsu 2019) focused on exploring the spatial and temporal information with convLSTM layers. Kim *et al.* (Kim et al. 2019) enforced the outputs to be temporally consistent by a recurrent feedback. (Xu et al. 2019) inpainted all the incomplete optical flows of the video and propagated valid regions to hole regions iteratively.

3 Approach

Video inpainting aims to remove the foreground objects and synthesize plausible hole-free background. We propose an end-to-end pipeline with a shadow detection branch to remove the objects more thoroughly (Section 3.2), and a coarse-to-fine inpainting branch (Section 3.2) to synthesize the background from the information of multi-frames. Figure 3 shows the pipeline of our approach.

3.1 Shadow Detection Branch

In autonomous videos, the objects are always under complex illuminations, which cast bonded shadows. Simply removing objects with given masks would cause terrible inpainting result. In previous works, shadows are always overlooked. However, in objects removal, dealing with the side effects of objects onto the environment is necessary. Actually, the un-removed shadows not only remain in the result videos leading to moving ghosts, but also heavily misguide the context inpainting in the holes as the shadows are tended to be selected as the best-matching patches.

One solution to the shadow problem is dilating the mask. However, we experimentally found the increase of hole size would dramatically decrease the inpainting result, since the closer to the original hole, the more content information to guide the inpainting is encoded. In order to automatically generate the inpainting mask with shadow in the as-small-as-possible manner, we propose a shadow detection branch ahead of the practical inpainting blocks.

We construct a classical U-net structure (Ronneberger, Fischer, and Brox 2015) for shadow detection. The branch takes RGB images and foreground masks as input and extends the masks with corresponding shadows. The details of this branch can be found in the supplementary material.

3.2 Coarse-to-fine Inpainting Branch

Regarding the coarse-to-fine inpainting branch, our method is built on the state-of-the-art single image inpainting network (Yu et al. 2018a). Our method adapts the GAN structure from them, and then introduces the flow constraint from the geometry and temporal consistent constraint to assemble multi-frames together. Concretely, we design three modules to deal with the autonomous driving videos. A temporal warping module is used to aligned different frames to same location, which is experimental important especially when the camera ego-motion is large. Besides, a “3D feature extractor” extracts information of multi-frames and enlarges the searching spaces to supply more alternative matching patches. Then a “3D feature assembler” merges multi-frames to inpaint the target frame, which is effective to improve the temporal consistency and reduce the jittering artifacts among different frames.

Given a video, we utilize every F continuous frames with masks M as input sequences to inpaint the incomplete target frame I_{in}^{m}. We use I_{gt} to denote the ground truth of the sequence. The incomplete sequence I_{in} is equal to $I_{gt} * M$, where $*$ is the element-wise multiplying. $U^{m \to i} (i = 1, 2, \ldots, F, i \neq m)$ denotes the flow fields from the target frame to all other ones. I_{in}, M and U are input to our coarse-to-fine inpainting branch with the supervision of I_{gt}.

As shown in Figure 3, the generator G of the inpainting branch is a two-stage coarse-to-fine structure. The coarse network G_c provides preliminary and blurry conjectures of the invisible parts, while the refinement network G_f refine the results and enhance the details. We follow single image inpainting work (Yu et al. 2018b) to define the structure of the coarse network G_c. All input frames are processed independently with shared weights of G_c in our framework.

The outputs of coarse inpainting branch $G_c(I_{in}, M)$ will be feed into the refinement branch. Figure 4 shows the detailed structure of the refinement branch, which consists of a temporal warping module, two 3D feature extractors, a 3D feature assembler and a decoder.

Temporal warping We propose a *temporal warping module* to transform the coarse network outputs $G_c(I_{in}, M)$ to the same camera location using the geometrical guidance. Without this warping module, stacking the frames together

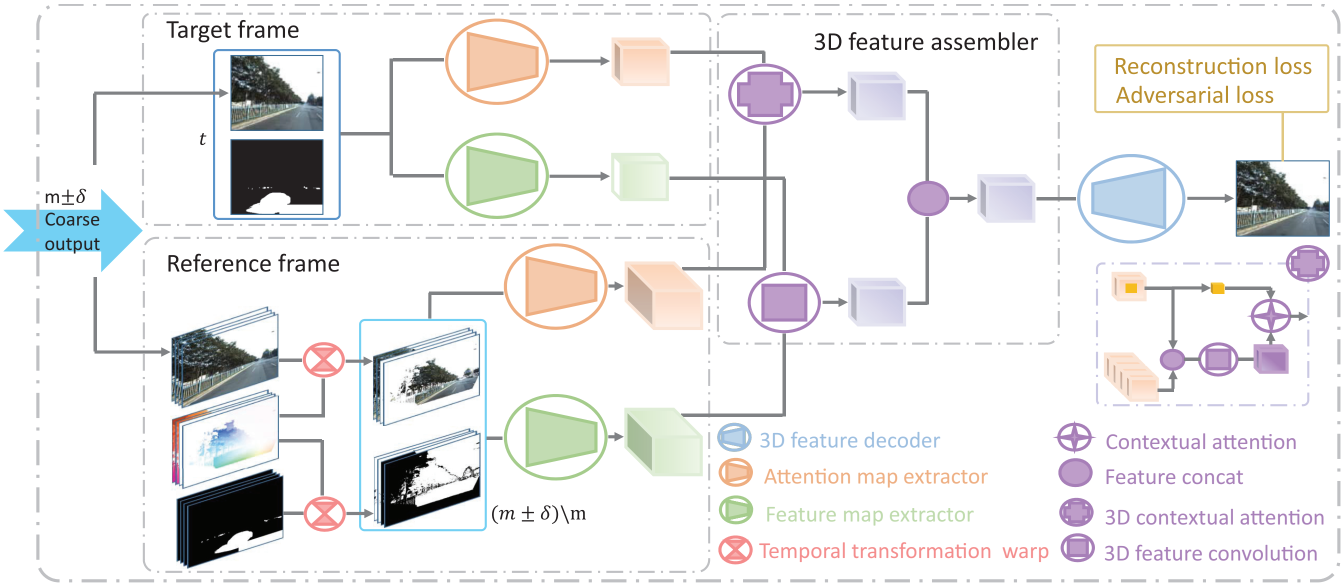


Figure 4: Architecture of the refinement network. δ is the interval of frames. $m \pm \delta$ represents all frames from $m - \delta$ to $m + \delta$. $(m \pm \delta) \setminus m$ represents all frames except m. With the guidance of the flows, we align the coarse outputs of each reference frames to the target frame. Then, we extract local contextual attention features and global features using two branches, where all features are later aggregated using a 3D feature assembler to predict target frame.

directly would lead to the following network blocks to learn the geometry information implicitly. This may work in the typical movie videos in Figure 2. But for the AD scenarios, the camera ego-motion is relative large. In other words, the receptive field of one convolution kernel on the stacking the frames is small w.r.t the flow caused by camera motion. The second column of Figure 5 shows that the result without temporal warping fails to recover the geometry structure of the scene. Thus, we propose a geometric guided temporal warping module to reduce the effects of large camera ego-motion. Specifically, with $p = (i, j)$ represents the 2D spatial location, all pixels of frames G_c^i will be warped to the target frame as follows:

$$G_c^i(I_{in}, M)'(p) = G_c^i(I_{in}, M)(p + U^{m \to i}) \qquad (1)$$

The warping modlue is implemented with differentiable bilinear sampling.

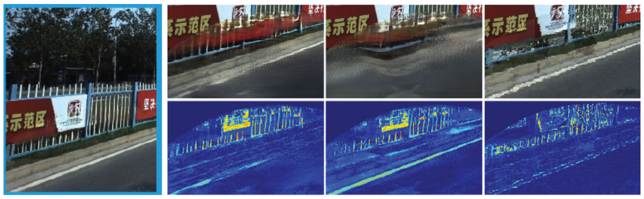


Figure 5: Left to right: our result, result w/o temporal warping, result with predicted flow, result of (Xu et al. 2019) using computed flow but w/o additional blending. The second row shows the visualized difference to our result.

3D feature extractor There are two different branches to extract feature maps: *contextual attention extractor* and *global feature extractor*. The former one is used to prepare features for the 3D contextual attention block. The latter one is utilized to get an overall impression of the scene and guide the network to hallucinate invisible regions. They take every frame of $G_c(I_{in}, M)'$ and warped masks M' as input and extract features with gated convolution layers. All frames except the target one are convoluted with shared weights.

3D feature assembler We design a 3D feature assembler to keep the temporal consistency by assembling the outputs of the two extractors. It consists of a multi-frames-to-one 3D contextual attention block and a 3D feature merging block to aggregate all features.

Although neighboring frames supply more information to inpaint the holes, there still be invalid regions that can not be seen in any frame. These regions can be inpainted with similar patches in the feature spaces. So we construct the multi-frames-to-one *3D contextual attention block* by combining temporal information and the CA module in (Yu et al. 2018a). The key of this module is to slice the background and foreground features into patches, then choosing the best-matched background patches to inpaint the foreground by similarity scores. To extend to our multi-frames scheme, a straightforward way is to stack background patches in all frames for matching. However, it is impractical since full 3D CA requires large memory footprint. For example, 46080 background patches will be extracted for our $96 * 96$ feature map. Therefore, we simplify the full 3D contextual attention block using a two-stage module. The first stage is a 3D convolution layer of one kernel in depth F to assemble the features of all F frames to aggregated features. The second stage is a CA layer that treat the aggregated features as background and target frame features as foreground, which is used to find matching scores between coarse foreground and background. In the first stage, we can also use bidirectional long short-term memory (LSTM) layer instead of 3D convolution layer. However, it achieves similar result but with more training cost.

The *3D feature merging block* is also a 3D convolution layer used for the global guidance feature map extractor. The output of 3D feature merging block and 3D contextual attention block will be concatenated together to maintain the global structure information and local patches for inpainting.

The assembled features are input to the decoder and the incomplete target frame I^m_{gt} are inpainted as $G^m(I_{in}, M, U)$. At last, a spectral-normalized Markovian discriminator (SN-PatchGAN) D as (Yu et al. 2018a) is used to hallucinate the missing regions in all frames. The details can be referenced in the supplementary material.

3.3 Loss Function

The objective function of the video inpainting branch consists of a reconstruction loss and an adversarial loss. The reconstruction loss L_g is an $L1$ loss combined with the coarse network and the refinement network. L_g is defined as:

$$L_g = \alpha||G_c(I_{in}, M) - I_{gt}|| + ||G^m(I_{in}, M, U) - I^m_{gt}|| \quad (2)$$

where α is the balancing parameter and $||\cdot||$ is the l_1 norm. During the training, gradients only back-propagate at non-object regions. The discriminator takes $G^m(I_{in}, M, U)$, I^m_{gt} as input and outputs a feature map with each value represents the corresponding region in the image is fake or not. The adversarial loss is a hinge loss:

$$\begin{aligned} L_D =& \mathbf{E}_{\mathbf{x}\sim\mathbf{P}_{\mathbf{data}}(\mathbf{x})}[max(0, 1 - D(x))] + \\ & \mathbf{E}_{\mathbf{z}\sim\mathbf{P}_{\mathbf{z}}(\mathbf{z})}[max(0, 1 + D(z))] \\ L_G =& -\mathbf{E}_{\mathbf{z}\sim\mathbf{P}_{\mathbf{z}}(\mathbf{z})}[D(z)] \end{aligned} \quad (3)$$

where $x = I^m_{gt}$ and z is the generator output $G^m(I_{in}, M, U)$.

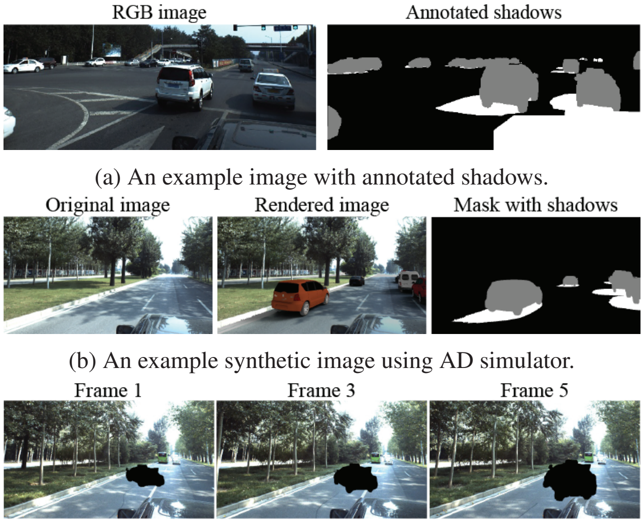
For the shadow detection branch, a weighted binary cross entropy loss same as (Xie and Tu 2015) is adopted. Please refer to the supplementary material for details.

3.4 Data Generation

Based on the AD dataset, we prepare two kinds of data in order to feed our pipeline:

- shadow dataset: a number of images with object shadows are manually annotated for training shadow detection branch.
- inpainting data: the inpainting data are augmented in two ways: 1) synthesized images using AD simulator with realistic objects and shadows, which is used for evaluation and training (both shadow detection branch and inpainting branch), 2) large number of images with generated temporal consistent masks for training inpainting branch.

Shadow Dataset To train our shadow-detection branch, we annotated a shadow detection dataset including 5293 images. Shadow regions of the foreground objects especially the cars are labelled manually. Figure 6(a) shows an example of the dataset. As the shadow areas provide implicit hints for lighting sources, illumination conditions and scene geometry, shadow detection is helpful for scene understanding and geometry perception. Cucchiara *et al.* (Cucchiara et al. 2001)and Mikic *et al.* (Mikic et al. 2000) have shown that some vision tasks, like efficient object detection and tracking (which is the key topic in the autonomous driving), can be beneficial from shadow removal.To best of our known, our annotated shadow dataset is the first video objects shadow dataset and supplies comparable data with the largest shadow dataset. There are three common shadow detections: the largest SBU Shadow Dataset with 4727 images, the UCF Shadow Dataset with 221 and the ISTD dataset with 1870 images while our dataset provides 5293 images. In these datasets, all shadows are marked without caring about foreground objects or background and the images in the dataset are all independent. Different from them, our dataset only annotates the shadows of the foreground objects and supplies the temporal consistent annotations of videos, which can be beneficial to some vision tasks, like efficient object detection and tracking. This dataset will be released to the public with the paper.



(a) An example image with annotated shadows.

(b) An example synthetic image using AD simulator.

(c) A training image sequence with temporal consistent masks.

Figure 6: Data generation

Inpainting Data Synthetic images are generated using the AD simulator AADS (Li et al. 2019). AADS is a close-loop AD simulator which can simulate traffic flow, LiDAR and color images. The reason of choosing AADS is that AADS could augment AD scenes with realistic objects under estimated illuminations. Thus, our synthetic images can be used as the supplementary of annotated shadow dataset. Furthermore, synthetic images are also used for quantitative evaluation, since the ground truth of video inpainting is not easy to obtain. Actually, there is no method who used the clean backgrounds after removing the real objects as the ground truth. Existing methods were only evaluated visually or by user study. Specifically, we run the simulator on ApolloScape images with few vehicles. Benefiting from

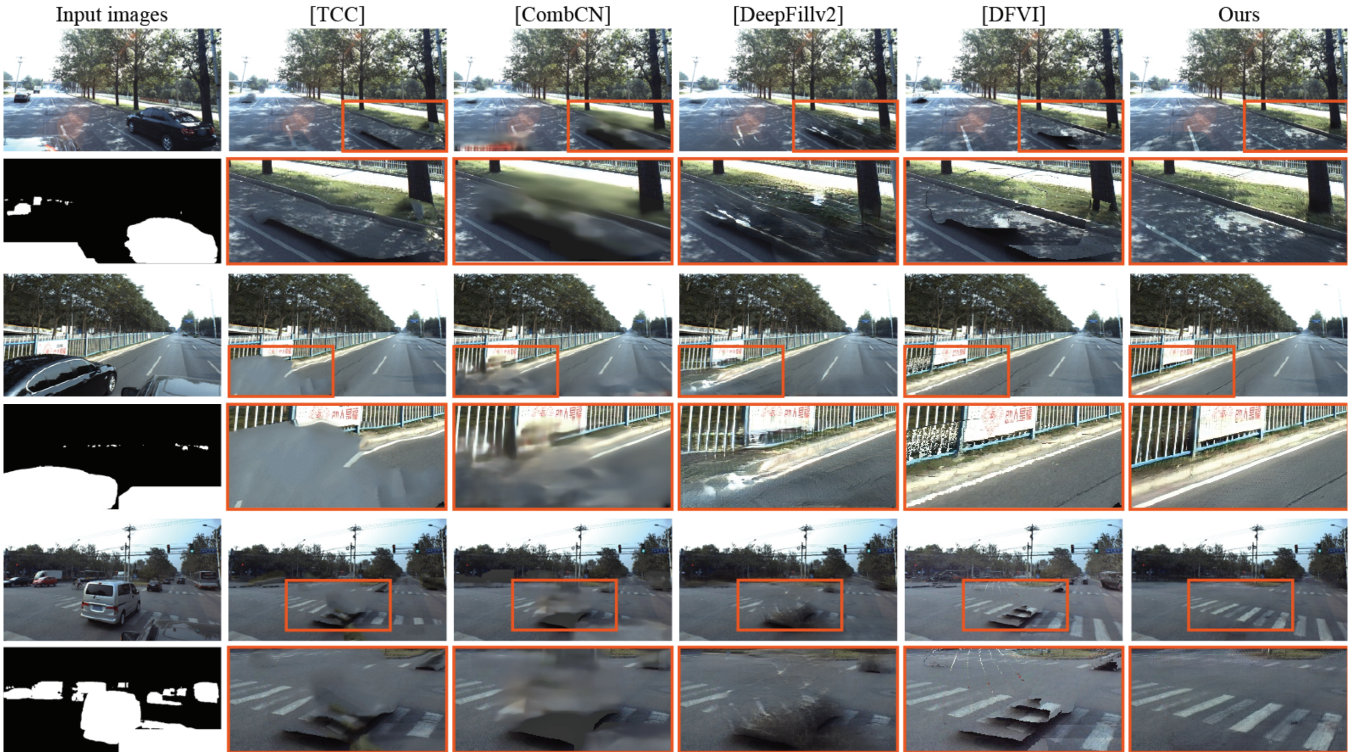


Figure 7: Comparisons with existing methods. The regions of the orange boxes are enlarged for details.

the simulator's lighting estimation module and traffic simulation backend, the synthetic images are with environment consistent shadows and temporal consistent object masks. Figure 6(b) shows an example of our synthetic data.

As extra bundled information, such as HD maps with lanes, is required to run the AD simulators. Synthetic images using simulator are very limited. Thus, for those images without extra information, we generate temporal consistent holes by warping the masks onto different frames and add random displacements to simulate object moving. Such temporal consistent masks are used as the main source of training data, which is shown in Figure 6(c).

4 Experiments and Results

4.1 Implementation Details

We focus on inpainting the street-view videos, especially those coming from AD datasets. In our paper, we use the ApolloScape (Huang et al. 2018) for experiments. The ApolloScape is a large-scale AD dataset which contains rich labeling including per-pixel semantic labelling, instance segmentation, projected depth images and corresponding camera poses. Please note that ApolloScape provides depth maps of static background, so the flows U can be directly computed from depth without flow inpainting algorithm like (Xu et al. 2019). In our implementation, number of frames in one sample sequence is set to 5. The original images in ApolloScape are downsampled to $1/4$. Total 16373 samples of training sequence are generated. During the training, the images are bottom cropped to $562 * 226$ and then randomly cropped to $384 * 192$ around the generated holes. The network is trained with an Adam optimizer for 210k iterations, whose learning rate is 0.0001 and batch size is 8. All the experiments are implemented with Tensorflow & PaddlePaddle and performed on 4 NVIDIA Tesla P40. During the testing, images are center cropped into $560 * 448$.

4.2 Comparisons with existing methods

Baselines We compare our approach with four different state-of-the-art methods, from single image inpainting to classical video inpainting and video inpainting with deep neural networks. The baselines are as following:

- Deepfillv2 (Yu et al. 2018a) is the state-of-art single image inpainting algorithm. We re-implement the method by extending the released code of (Yu et al. 2018b).
- TCC (Huang et al. 2016) is a classical optimization-based video inpainting algorithm, which released MATLAB code for comparison.
- CombCN (Wang et al. 2018) is the first work to utilize deep neural networks for video inpainting. We re-implement the architecture with Tensorflow, and modified the inputs to be consistent with our data.
- DFVI (Xu et al. 2019) is the most recent video inpainting algorithm. It learns to inpaint flow maps as well as holes. We use the release code for comparison.

All neutral network are re-trained with our generated data. For TCC and DFVI, we use our generated flow as their predicted flow for a fair comparison.

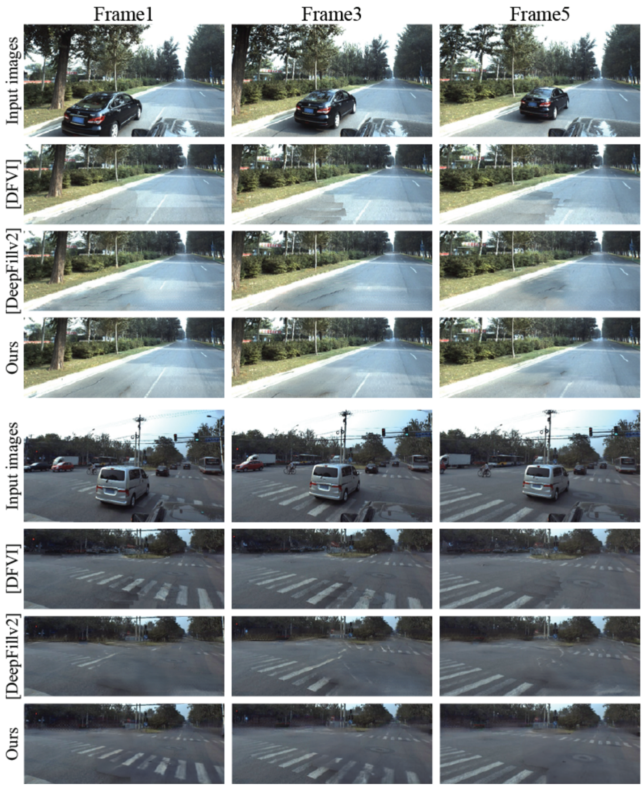


Figure 8: Comparisons with existing methods. To remove the impacts of the shadows and emphasize the temporal consistency, shadow detection is introduced to all methods.

Method	MAE	RMSE	PSNR	SSIM	TME
TCC	22.595	31.322	32.332	0.9657	29.490
CombCN	19.952	28.059	33.332	0.9686	25.848
Deepfillv2	18.725	28.004	33.053	0.9626	30.260
DFVI	23.005	35.110	31.282	0.9674	26.514
Ours	**15.143**	**22.611**	**34.435**	**0.9697**	**24.822**

Table 1: Comparison with different methods. Our method outperforms others on all metrics.

Method	MAE	RMSE
Deepfillv2	19.799	29.544
Deepfillv2 + shadow detection	18.725	28.004
Ours - w/o temporal warping	16.202	24.437
Ours - w/o contextual attention	16.236	24.051
Ours - w/o shadow detection	15.414	23.217
Ours	**15.143**	**22.611**

Table 2: Evaluation metrics of ablation study.

Comparison and Analysis Figure 7 shows visual comparisons on the single frame of inpainted videos between the baseline methods and our method. One significant improvement of our method is the shadows and ghosts eliminating. With shadow-aware branch, our method can remove the moving objects completely and obtain cleaner backgrounds.

Comparing to the baseline methods, our method achieve better results with less artifacts. Deepfillv2 may fail when the holes are large, as it is hard to keep geometric structure for this single image inpainting framework. TCC method follows joint flow and color optimization. Even using our computed flow as reliable initialization, TCC still produces unacceptable results with mismatched boundaries shown in the second column of Figure 7. CombCN relies on typical 3D convolution to implicitly maintain the temporal information which is not enough under large camera ego-motion. Thus the results of CombCN is short of structure information. As the newest SOTA video inpainting method based on neural networks, DFVI performs well on maintain the geometry structure of the scenes. However, the brightness and illumination of frames always change a lot even in one sequence when the camera motion is large. Thus, propagating patches directly, which is used by DFVI, cannot yield smooth blended results. Especially when the flow is not very accurate, the results are noisy. This usually appears on thin objects like the fences in the second rows of Figure 7. Besides, the propagation used by DFVI is very time consuming especially when the hole regions can not be borrowed from other frames directly. It will inpaint the key-frames and propagates them to all frames iteratively. In terms of one sequence of 175 frames in our evalation data, the runtime of DFVI is 20 minutes while our method can inpaint them in 40 seconds.

In our method, with the temporal warping module, the geometry structure can be well maintained even when camera ego-motion is large. With the 3D feature assembler, the features of different frames can be blended smoothly. With the shadow-aware branch, the moving-shadow artifacts can be solved. Thus, our method yield the best visual results comparing to existing methods in AD videos. More results can be found in the supplementary materials.

To quantitative compare our method with other methods, we utilize five metrics for the evaluations: mean absolute error(MAE), root mean squared error(RMSE), peak signal to noise ratio(PSNR), structural similarity index(SSIM) and temporal warping root mean squared error(TWE). We calculate the TWE by warping one inpainted frame to next frame and computing the RMSE on the valid regions. The TWE is applied on different frames to evaluate the temporal consistency. Note that those metrics are evaluated only on the inpainted hole regions. Table 1 shows the evaluation results of the baseline methods and our method. Note that our method outperforms others on all the metrics.

To remove the effects of shadows and compare the video inpainting branch only, we also add the shadow detection branch to the baseline methods for comparison. Figure 8 shows some frames of the inpainted videos. As this figure shows, our method yields better results with temporal consistency, which benefits from the guidance of the geometric information from multi-frames features. Please refer to the supplementary video for a better view and more results.

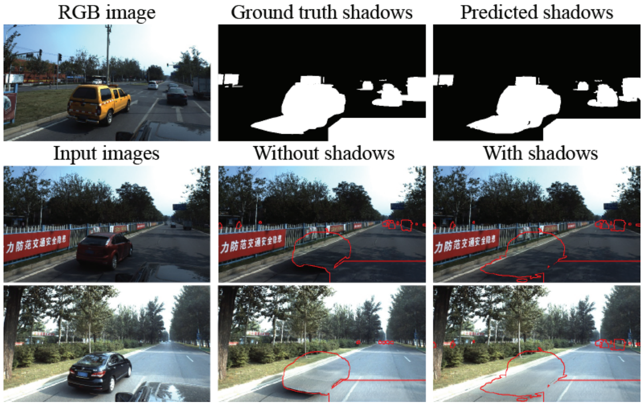


Figure 9: The first row shows the results of shadow detection network. The second and third rows show inpainting results with/without shadow detection. The removal regions are marked with red boundaries.

4.3 Ablation Study

To explore the effects of different parts of our algorithm, we conduct several ablation experiments about the multi-frames-to-one scheme, the temporal warping module and the contextual attention in 3D feature assembler and shadow detection branch. Table 2 shows the evaluation metrics of the ablation study. Adding the multi-frames-to-one scheme without temporal warping module reduces the MAE from 18.725 to 16.202 as the network could find matching patches from more frames. Besides, adding the temporal warping module can also gain further promotion of the metrics. It can utilize the geometric information to guide the inpainting process explicitly. Features of similar objects are aligned together, making it easier to find the matching patches. Please refer to Figure 5 for the image results. Removing the contextual attention in 3D feature assembler leads to larger errors. The ablation studies of the temporal warping module and the contextual attention in Table 2 show that objects removal can be beneficial from the combination of temporal information and the contextual attention.

The shadow detection branch is one of the most effective parts to get the clean inpainted background. The first row of Figure 9 shows some examples of our shadow detection network's predicts. Removing shadows along with cars has a shortcoming that the holes are enlarged. It is acknowledged that larger holes are harder to inpaint. However, it also can reduce the difficulty to find the matching patches and remove the moving shadow ghosts in videos. With the predicted shadow maps, the influence of wrong matching patches with different brightness can be reduced and the ghosts of the shadows will be removed. From the Table 2, adding shadow detection branch improves the performance.

Besides the reducing of the inpainting errors, the most obvious improvement is the visual effects. As the bottom two rows of Figure 9 shows, with the shadow detection branch, the foreground objects could be removed more thoroughly. Shadow detection branch is a unified block that could be added to any AD video inpainting algorithm as a pre-processing operation.

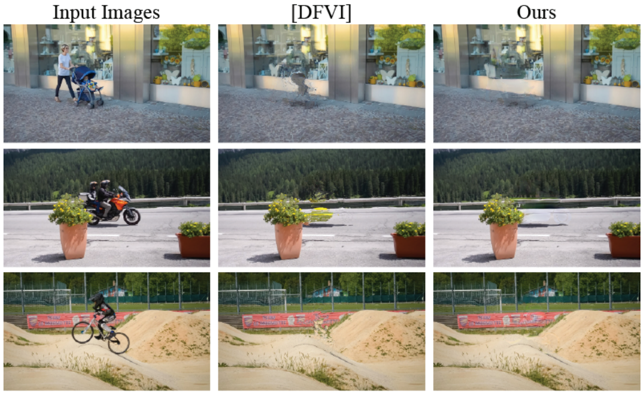


Figure 10: Comparison with DFVI on the DAVIS dataset.

4.4 The Generalization Ability

To evaluate the generalization ability of our method, we train our model on the DAVIS dataset (Perazzi et al. 2016). The results are shown in Figure 10. The inaccurate predicted flows misguide the inpainting in DFVI while our results can preserve the geometry structures of the backgrounds.

5 Conclusion

In this paper, we present a shadow-aware video inpainting pipeline for street-view foreground objects removal in AD with large camera motion. We use a multi-frames-to-one scheme with the geometry guidance to aggregate information of multi-frames and keep the temporal consistency. A unified shadow detection branch is adopted for removing the shadow ghosts and reducing the impacts of redundant patches. The first foreground objects shadow detection dataset focusing on AD will be open source. In the experiments, we propose a new evaluation method for objects removal when there is no ground truths. The experiments demonstrates that our methods could reduce the artifacts and reconstruct clean background images. In the future, we will investigate the method to reduce the running time for real-time application and improve the performances of regions can not be borrowed from other frames.

6 Acknowledgements

Weiwei Xu is partially supported by NSFC (No. 61732016) and the fundamental research fund for the central universities. Jinhui Yu is partially supported by NSFC (No. 61772463). We also thank the reviewers and all the people who offered help.

References

Alhaija, H. A.; Mustikovela, S. K.; Mescheder, L.; Geiger, A.; and Rother, C. 2018. Augmented reality meets computer vision: Efficient data generation for urban driving scenes. *International Journal of Computer Vision* 126(9):961–972.

Arjovsky, M.; Chintala, S.; and Bottou, L. 2017. Wasserstein gan. *CoRR* abs/1701.07875.

Barnes, C.; Shechtman, E.; Finkelstein, A.; and Goldman, D. B. 2009. Patchmatch: A randomized correspondence algorithm for structural image editing. In *ACM Transactions on Graphics (ToG)*, volume 28, 24. ACM.

Chang, Y.-L.; Liu, Z. Y.; and Hsu, W. 2019. Free-form video inpainting with 3d gated convolution and temporal patchgan. *arXiv preprint arXiv:1904.10247*.

Cucchiara, R.; Grana, C.; Piccardi, M.; Prati, A.; and Sirotti, S. 2001. Improving shadow suppression in moving object detection with hsv color information. In *Intelligent Transportation Systems, 2001. Proceedings. 2001 IEEE*.

Ding, Y.; Wang, C.; Huang, H.; Liu, J.; Wang, J.; and Wang, L. 2019. Frame-recurrent video inpainting by robust optical flow inference. *arXiv preprint arXiv:1905.02882*.

Ebdelli, M.; Le Meur, O.; and Guillemot, C. 2015. Video inpainting with short-term windows: application to object removal and error concealment. *IEEE Transactions on Image Processing* 24(10):3034–3047.

Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A. C.; and Bengio, Y. 2014. Generative adversarial nets. In *NIPS*, 2672–2680.

Granados, M.; Kim, K.; Tompkin, J.; Kautz, J.; and Theobalt, C. 2012. Background inpainting for videos with dynamic objects and a free-moving camera. In *Computer Vision – ECCV 2012*, 682–695.

Hays, J., and Efros, A. A. 2007. Scene completion using millions of photographs. *ACM Transactions on Graphics (TOG)* 26(3):4.

Huang, J.-B.; Kang, S. B.; Ahuja, N.; and Kopf, J. 2014. Image completion using planar structure guidance. *ACM Transactions on graphics (TOG)* 33(4):129.

Huang, J.-B.; Kang, S. B.; Ahuja, N.; and Kopf, J. 2016. Temporally coherent completion of dynamic video. *ACM Transactions on Graphics (TOG)* 35(6):196.

Huang, X.; Cheng, X.; Geng, Q.; Cao, B.; Zhou, D.; Wang, P.; Lin, Y.; and Yang, R. 2018. The apolloscape dataset for autonomous driving. *arXiv preprint arXiv:1803.06184*.

Iizuka, S.; Simo-Serra, E.; and Ishikawa, H. 2017. Globally and locally consistent image completion. *ACM Trans. Graph.* 36(4):107:1–107:14.

Kim, D.; Woo, S.; Lee, J.-Y.; and Kweon, I. S. 2019. Deep video inpainting. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Köhler, R.; Schuler, C. J.; Schölkopf, B.; and Harmeling, S. 2014. Mask-specific inpainting with deep neural networks. In *GCPR*, volume 8753 of *Lecture Notes in Computer Science*, 523–534. Springer.

Li, W.; Pan, C.; Zhang, R.; Ren, J.; Ma, Y.; Fang, J.; Yan, F.; Geng, Q.; Huang, X.; Gong, H.; et al. 2019. Aads: Augmented autonomous driving simulation using data-driven algorithms. *Science Robotics* 4(28):eaaw0863.

Mao, X.; Li, Q.; Xie, H.; Lau, R. Y. K.; and Wang, Z. 2016. Least squares generative adversarial networks.

Mikic, I.; Cosman, P. C.; Kogut, G. T.; and Trivedi, M. M. 2000. Moving shadow and object detection in traffic scenes. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 1, 321–324 vol.1.

Mroueh, Y., and Sercu, T. 2017. Fisher gan.

Newson, A.; Almansa, A.; Fradet, M.; Gousseau, Y.; and Pérez, P. 2014. Video Inpainting of Complex Scenes. *SIAM Journal on Imaging Sciences* 7(4):1993–2019.

Pathak, D.; Krahenbuhl, P.; Donahue, J.; Darrell, T.; and Efros, A. A. 2016. Context encoders: Feature learning by inpainting. *CoRR* abs/1604.07379.

Perazzi, F.; Pont-Tuset, J.; McWilliams, B.; Van Gool, L.; Gross, M.; and Sorkine-Hornung, A. 2016. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*.

Radford, A.; Metz, L.; and Chintala, S. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR* abs/1511.06434.

Ren, J. S. J.; Xu, L.; Yan, Q.; and Sun, W. 2015. Shepard convolutional neural networks. In *NIPS*, 901–909.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 234–241. Springer.

Seif, H. G., and Hu, X. 2016. Autonomous driving in the icity–hd maps as a key challenge of the automotive industry. *Engineering* 2(2):159–162.

Sun, J.; Yuan, L.; Jia, J.; and Shum, H.-Y. 2005. Image completion with structure propagation. In *ACM Transactions on Graphics (ToG)*, volume 24, 861–868. ACM.

Telea, A. 2004. An image inpainting technique based on the fast marching method. *Journal of graphics tools* 9(1):23–34.

Vicente, T. F. Y.; Hou, L.; Yu, C.-P.; Hoai, M.; and Samaras, D. 2016. Large-scale training of shadow detectors with noisily-annotated shadow examples. In *Proceedings of European Conference on Computer Vision*.

Vicente, T. F. Y.; Hoai, M.; and Samaras, D. 2015. Leave-one-out kernel optimization for shadow detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Wang, C.; Huang, H.; Han, X.; and Wang, J. 2018. Video inpainting by jointly learning temporal structure and spatial details. *arXiv preprint arXiv:1806.08482*.

Wang, J.; Li, X.; and Yang, J. 2018. Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wexler, Y.; Shechtman, E.; and Irani, M. 2007. Space-time completion of video. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 29(3):463–476.

Xie, S., and Tu, Z. 2015. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, 1395–1403.

Xie, J.; Xu, L.; and Chen, E. 2012. Image denoising and inpainting with deep neural networks. In *NIPS*, 350–358.

Xu, R.; Li, X.; Zhou, B.; and Loy, C. C. 2019. Deep flow-guided video inpainting. *arXiv preprint*.

Yu, J.; Lin, Z.; Yang, J.; Shen, X.; Lu, X.; and Huang, T. S. 2018a. Free-form image inpainting with gated convolution. *arXiv preprint arXiv:1806.03589*.

Yu, J.; Lin, Z.; Yang, J.; Shen, X.; Lu, X.; and Huang, T. S. 2018b. Generative image inpainting with contextual attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5505–5514.