Removal of Image Obstacles for Vehicle-mounted Surrounding Monitoring Cameras by Real-time Video Inpainting

Yoshihiro Hirohashi[1] Kenichi Narioka[1] Masanori Suganuma[2,3]
Xing Liu[2] Yukimasa Tamatsu[1] Takayuki Okatani[2,3]
[1] DENSO CORPORATION, Japan
[2] Graduate School of Information Sciences, Tohoku University, Japan
[3] Center for Advanced Intelligence Project, RIKEN, Japan

Abstract

One of the practical problems with surrounding view cameras (SMCs) of a vehicle is the degradation of image quality due to obstacles by substances adherent to their lens surface, such as raindrops and mud. Such image degradation could be improved by image restoration techniques that have been studied in the field of computer vision. However, to assist the driver, real time processing and fidelity of the recovered image are essential, which disqualifies most of the existing methods. In this study, we propose to adopt a recently developed video-inpainting method that can restore high-fidelity images in real time. It estimates optical flows using a CNN and use them to match occluded regions in the current frame to unoccluded regions in previous frames, restoring the former. Although the direct application does not lead to satisfactory results due to the peculiarities of the SMC videos, we show that two improvements make it possible to obtain good results that are useful in practice. One is to use a model-based flow estimation method to obtain target flows for training the CNN, and the other is to improve how the estimated flows are used to match the current and previous frames. We conducted experiments using real images mainly of parking spaces in urban areas. The results, including subjective evaluation, show the effectiveness of our approach.

1. Introduction

Image sensors are used in a variety of ways to support the driving of automotive vehicles, from the visualization of blind spots around vehicles to more advanced applications such as autonomous driving and advanced driver assistance systems (AD/ADAS). In these applications, surrounding monitoring cameras (SMCs) are playing an important role.

Raindrops, dust and dirt on the surface of the SMC lens obstruct the driver's view, as shown in Figure 1. Since raindrops are usually distributed sparsely on the lens, pedestrians and objects on the road are intermittently occluded, as the car and/or the objects move. This will be an issue even if the occlusion is only intermittent, since the driver cannot stare at the monitor all the time.

This problem can be solved if clean images free of these obstacles can be restored and displayed on the monitor. There are two requirements for this image restoration, i.e., real time computation and fidelity of the restored images. These two are vital to support the drivers from a safety point of view.

There have been many studies of image restoration to improve the quality of images degraded due to various factors; [10, 25, 8, 13, 7, 30] to name a few. These include several studies for removing raindrops from images [21, 20, 26, 12]. In particular, recent applications of CNNs have greatly accelerated the overall research on image restoration. However, these existing methods do not meet the above two requirements and thus cannot be used for our purpose.

First, most of the methods do not meet the fidelity requirements of the restored images. This is arguably obvious for the recent methods that use only a single image. If there are large occluded areas in the image due to raindrops or dust, it is in principle impossible to reconstruct the scene behind them from that image alone. Not only the studies of raindrop removal but a broader range of image restoration studies (e.g., GAN-based super-resolution and single-image inpainting) have tended to aim at producing images that appear more natural to human eyes, even if they are fake, rather than pursuing the fidelity.

In this study, we propose a method based on the principle of video inpainting to remove raindrops, dust, dirt etc. on SMC images. It finds the pixel values of the scene occluded by these obstacles in another frame and then copies their pixel values to the pixels in the current frame. Thus, the fidelity of the restored images can be maximally en-

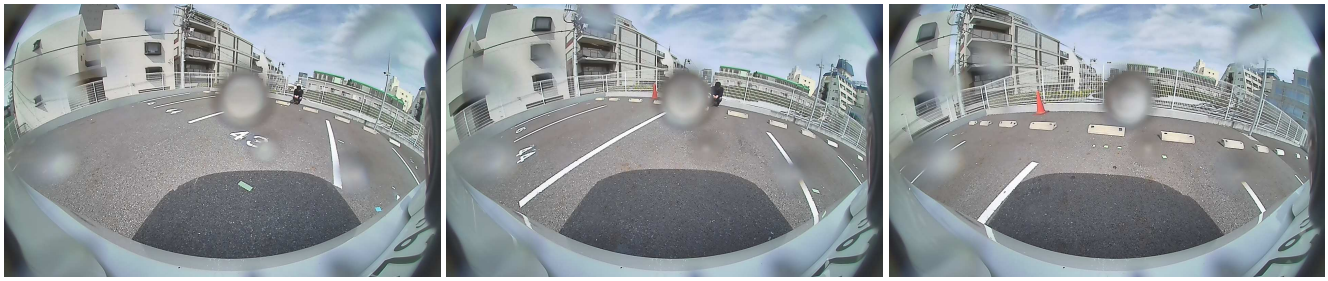

Figure 1. An example of a video sequence captured by the surrounding monitoring camera (SMC) used in our experiments. One of the raindrops adherent to the lens surface occludes a person sitting on the ground after the frame (107th) shown in the middle.

sured. While the above single-image restoration methods could synthesize fake images, this method can recover the correct image as long as the occluded scene parts are visible in any of the previous frames. We assume in this paper that these obstacles can be accurately detected from each video image, which provides the image regions to be restored, called masks, in video inpainting.

To satisfy another requirement of real-time processing, we employ the recently proposed method for video inpainting [17]. While conventional video inpainting methods cannot operate in real time, their method can, with fairly good image quality. This is made possible by using a CNN to estimate optical flow fields, making high speed and accuracy compatible.

However, we found a few issues when applying their method directly to raindrop removal with our SMC images, although their method achieves good inpainting accuracy on the standard benchmark datasets. There are two major problems. One is that i) CNNs trained with the standard dataset for optical flow estimation, which is commonly used in related studies (and therefore employed in [17]), only achieve lower-than-expected estimation accuracy for the SMC videos. This is attributable to the discrepancy between the training data and the SMC videos, which causes the domain shift. The other is that ii) the restored images for typical SMC videos tend to be generally blurry, leading to insufficient image quality.

Both of (i) and (ii) can be attributed to the following characteristics of the SMC videos, which differentiate them from the above standard datasets. Firstly, the motion of the SMC induced by that of the car is mostly parallel to its optical axis, making the dominant image motion zooming. As a result, a scene point tends to be occluded for a longer time period. Moreover, the scenes of urban parking spaces tend to have fewer textures. Finally, the SMC has a wide angle of view, inducing larger optical distortion.

For (i), we create training data using the SMC videos themselves for the optical flow estimation from them. To obtain the ground-truth flows that are necessary for the training, we employ a flow estimation method based on a flow model and optimization [19], which does not rely on machine learning and thus is free from the domain shift, and treat the estimated flows as 'ground truths' for training the CNN. Although they are not error-free, we show through experiments that using them does contribute to improvement of restoration accuracy. For (ii), we first analyze why the restored images by the method [17] tend to be blurry, and then present a solution that solves the issue at the expense of a certain level of increase in memory usage. Our experimental results show that it can greatly improve image quality while meeting the requirement of real time computation.

2. Related work

2.1. Single-image raindrop removal

A number of methods have been proposed to remove raindrops adherent on the glass surface in front of the camera from a single image and then restore a clean image free of the raindrops. In early days, a method based on an optical model of raindrops was explored [9]. Recently, the application of CNN has accelerated the research, resulting in the proposal of many methods that can restore high-quality images [21, 26, 12, 20, 16]. However, even with the latest methods, if the raindrops are large and the background scene is largely invisible, the recovered image will have to be fake, since the single image lacks enough information. In such cases, similar results will be obtained by the general-purpose inpainting method [5] applied with raindrops specified as the region to be inpainted; the above limitation holds, too.

2.2. Multi-image raindrop removal

Several methods have been proposed to remove raindrops from video images [30, 18, 28, 22, 29]. As with the video inpainting methods described below, they match image points or patches across occluded and unoccluded regions in the of spatio-temporal video volume to remove raindrops. The problem with these methods is that the quality of the restored images is not enough, and neither is the

computational speed. In [15], a method for detecting the position of raindrops in real time is presented.

The problem of raindrop removal reduces to that of video inpainting, provided that the positions of raindrops are identified. Many general-purpose methods have been proposed so far, all of which can potentially be employed. They can be classified into several categories depending on how to establish correspondences in spatio-temporal video volumes. Early methods assume offline computation and need long computational time. In recent years, the application of CNNs has enabled us to improve both the quality of restored images as well as the computational speed. Our study relies on [17], which focuses on good quality video inpainting in real time.

3. Method

As mentioned earlier, the proposed method extends the baseline method of Murase et al. [17] in several aspects. We first briefly summarize the baseline method in Section 3.1 and then explain our extensions in Section 3.2 and 3.3. Figure 2 shows the baseline and the proposed method.

Notation

We follow the standard formulation of video inpainting, where, given N pairs of an input image and an associated mask specifying the regions to be restored, we want to provide N restored images. In this paper, we denote the input pair by $(\mathcal{I}_k, \mathcal{H}_k)$, and the resultant image by $\hat{\mathcal{I}}_k$. The input and resultant images are color images of size $H \times W \times 3$, where H and W are their height and width, respectively. The masks $\mathcal{H}_k$'s specifying the image regions to be restored are binary images of the same size, i.e., $\{0,1\}^{H\times W}$; the pixels with the value one should be restored and those with zero be left untouched. Following [17], we denote the inverted (i.e., $0 \leftrightarrow 1$) version of $\mathcal{H}$ by $\bar{\mathcal{H}}$; for a given image $\mathcal{I}$, we represent the warping operation with an optical flow $\mathcal{W}$ by $\mathcal{W} \odot \mathcal{I}$ and an element-wise masking operation with a mask $\mathcal{H}$ by $\mathcal{H} \cdot \mathcal{I}$.

3.1. The baseline method

As shown in Figure 2(A), the baseline method [17] restores the image regions specified by the mask $\mathcal{H}_k$ of the image I_k, where k is the latest frame. It is the first method that can perform video inpainting in real time by leveraging recently developed CNN-based optical flow estimators, which is also the reason that we chose it. The pseudo code of processing an input sequence is given in Algorithm 1. The method consists of two components, the optical-flow estimator and the warp and copy-paste module.

The key problem of video inpainting is to match image pixels from masked to unmasked regions in the spatio-temporal video volume. Suppose that a scene point is unobserved in the latest frame and thus specified with a mask. If we can find a previous frame in which the same scene point is observed, then we can inpaint the masked pixel by copy-pasting the color of the observed pixel. To perform this spatio-temporal matching in the video volume, different video-inpainting methods employ different approaches.

The baseline method employs optical-flow fields for this purpose, to fulfill the requirements of causality and computational speed. A problem is that an optical-flow field, which is well-defined for a consecutive video frames, is not well-defined for the image regions that are occluded and thus specified with masks. To cope with this, the baseline method extends an existing CNN-based flow estimator, such as FlowNet2 [14] and PWC-Net [24], to be able to simultaneously estimate the optical flows of unmasked, observed regions as well as masked, unobserved regions. This is enabled by inputting the masks $\mathcal{H}_k$ and $\mathcal{H}_{k-1}$ through additional channels along with the original inputs $\mathcal{I}_k$ and $\mathcal{I}_{k-1}$ to an optical-flow estimator, and training it to predict the optical flows of unmasked as well as masked regions. The data for this training are generated synthetically by randomly creating occluding objects with corresponding masks and pasting them into the images of *FlyingChairs* dataset [6]. The FlyingChairs dataset is a standard dataset for optical flow estimation, which is created by superimposing synthetic 3D-chair models [1] into natural images retrieved from Flicker, providing accurate and reliable ground truths of optical flow fields. In summary, the extended optical-flow estimator estimates the optical flow field over the whole image including masked regions using two consecutive image frames along with their masks. In Algorithm 1, we denote it by FlowEstimator and its output by $\hat{\mathcal{W}}^{(k)\to(k-1)} \in \mathbb{R}^{H\times W\times 2}$.

The second component, the warp and copy-paste module, first warps the restored image $\hat{\mathcal{I}}_{k-1}$ at the last frame with the estimated optical-flow field $\hat{\mathcal{W}}^{(k)\to(k-1)}$. This operation yields the warped image $\hat{\mathcal{W}}^{(k)\to(k-1)} \odot \hat{\mathcal{I}}_{k-1}$ by propagating the pixels of $\hat{\mathcal{I}}_{k-1}$ to geometrically align them with the current image $\hat{\mathcal{I}}_k$. As the optical flows $\hat{\mathcal{W}}^{(k)\to(k-1)}$ are estimated with subpixel precision, bilinear interpolation is employed for this pixel propagation. Then, as in line 4 of Algorithm 1, the masked regions of the warped image and the unmasked regions of the current image are merged to construct the restored image $\hat{\mathcal{I}}_k$.

3.2. Fine-tuning to mitigate domain shift

An important key to the success of the above approach is accurate estimation of optical flows. The CNN-based flow estimators trained on the FlyingChair dataset perform fairly well on the standard benchmark datasets, which are widely used in the studies of optical flow estimation. However, we found through experiments that they, even without the extension enabling flow estimation behind masks, do not perform well on the data dealt with in this study, which are the

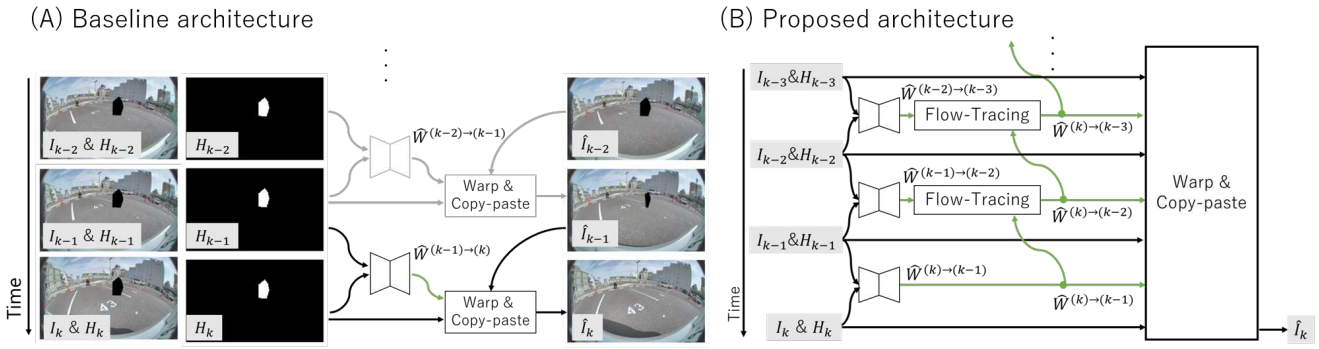

Figure 2. Overview of the baseline and the proposed method. The core components for the both methods are the extended flow estimator, which estimates the flow of unoccluded (unmasked) as well as occluded (masked) regions using a CNN, and the subsequent operation of copying pixels from previous frames to compute a restored image $\hat{\mathcal{I}}_k$. (A) The architecture of the baseline method. To compute the restored image $\hat{\mathcal{I}}_k$ from the current input $(\mathcal{I}_k, \mathcal{H}_k)$, it re-uses the the restored image $\hat{\mathcal{I}}_{k-1}$ of the last frame, making the restored images blurry. (B) That of the proposed approach. Instead of re-using the previously restored image(s) $\hat{\mathcal{I}}_{k-1}$, it uses previous inputs $(\mathcal{I}_k, \mathcal{H}_k), \ldots, (\mathcal{I}_{k-1}, \mathcal{H}_{k-1})$ by back-tracing optical flows as explained in Section 3.3 to geneerate the restored image $\hat{\mathcal{I}}_k$.

Algorithm 1 Warp & Copy-paste procedure of the baseline method [17]

Input: $\{(\mathcal{I}_1, \mathcal{H}_1), \ldots, (\mathcal{I}_N, \mathcal{H}_N)\}$
Output: $\{\hat{\mathcal{I}}_1, \ldots, \hat{\mathcal{I}}_N\}$

1: $\hat{\mathcal{I}}_1 \leftarrow \bar{\mathcal{H}}_1 \cdot \mathcal{I}_1$
2: **for** $k = 2, \cdots, N$ **do**
3: $\quad \hat{\mathcal{W}}^{(k)\to(k-1)} \leftarrow \text{FlowEstimator}(\mathcal{I}_k, \mathcal{H}_k, \mathcal{I}_{k-1}, \mathcal{H}_{k-1})$
4: $\quad \hat{\mathcal{I}}_k \leftarrow \mathcal{H}_k \cdot \left(\hat{\mathcal{W}}^{(k)\to(k-1)} \odot \hat{\mathcal{I}}_{k-1}\right) + \bar{\mathcal{H}}_k \cdot \mathcal{I}_k$
5: **end for**

video images captured by a rear-view camera of a vehicle while it is backing up in parking spaces. There are the following differences from the above standard datasets. i) The dominant image motion is zooming not translation, which is created by the ego-motion of the camera (vehicle). ii) The scenes are mostly parking spaces that tend to have less textures. iii) Moreover, the camera has a wider field of view, causing lens distortion in the captured images. These differences could have caused domain shift for the CNN-based estimator, arguably leading to worse estimation accuracy.

An obvious solution to resolve the issue is to train the CNN on a training dataset created using the same or similar camera and experimental setup. However, it is generally hard to obtain accurate ground-truth optical flows for the videos acquired by such a real camera system. Thus, we propose to use classical methods of optical flow estimation [4, 3] for obtaining the 'ground-truth' optical flows. To be specific, we apply *pyflow*[1] to the video images captured by our SMC with a vehicle. These methods are model-based and rely on optimization to estimate optical flows. They do not depend on machine learning, making them immune to the aforementioned domain shift issue.

Note that these methods are superior in terms of domain (in)dependency but inferior in terms of computational speed due to iterative nature of the optimization, as compared with the CNN-based methods. Their accuracy tends to be on par with or often lower than the CNN-based methods provided that the CNN-based methods are free from domain shift.

Of course, the flow fields estimated by them are not error-free, but training with the estimated flows do contribute to improvement of estimation accuracy, as will be shown in our experimental results. We will refer to the estimated optical-flow as *pseudo-ground-truth* in what follows. In our experiments, we train the CNN by first training it using FlyingChairs with occluded objects, as mentioned ealier, and then fine-tuning it using the parking videos with the pseudo-ground-truth data.

3.3. Optical flow tracing to maintain resolution

In the second step after the optical flow estimation, we use the calculated optical flow to copy the corresponding pixels from previous frames to each pixel in the masked region of the current frame. Originally, the pixel of the previous frame that is the source of the copy must be located in the unmasked area in that frame, but it is not necessarily the frame right before the current one. In general, it is often a more previous frame, and how far back we need to go depends on a combination of the size of the masked area and the flow length. That is, depending on the position of the masked region (for each pixel), the corresponding frame is generally different.

The baseline method avoids this complexity of matching the current frame and previous frames by propagating pixel values only between the consecutive frames. More specifically, for each pixel in the masked region of $\mathcal{I}_k$, the baseline method always refers to the corresponding point in the last

[1] https://github.com/pathak22/pyflow

Algorithm 2 Proposed method of back-tracing optical flows to restore masked image regions

Input: $\{(\mathcal{I}_1, \mathcal{H}_1), \ldots, (\mathcal{I}_N, \mathcal{H}_N)\}$
Output: $\{\hat{\mathcal{I}}_1, \ldots, \hat{\mathcal{I}}_N\}$

$\hat{\mathcal{I}}_1 \leftarrow \mathcal{H}_1 \cdot \mathcal{I}_1$
for $k = 2, \cdots, N$ **do**
  $\mathcal{H} \leftarrow \mathcal{H}_k$
  $\hat{\mathcal{I}}_k \leftarrow \bar{\mathcal{H}} \cdot \mathcal{I}_k$
  $k' \leftarrow k - 1$
  while $k' \geq \max(1, k - D)$ and $\mathcal{H} \neq \mathbf{0}$ **do**
    Compute $\hat{\mathcal{W}}^{(k)\rightarrow(k')}$
    $\mathcal{R} \leftarrow \mathcal{H} \cdot \left(\hat{\mathcal{W}}^{(k)\rightarrow(k')} \odot \bar{\mathcal{H}}_{k'}\right)$
    $\hat{\mathcal{I}}_k \leftarrow \mathcal{R} \cdot \left(\hat{\mathcal{W}}^{(k)\rightarrow(k')} \odot \mathcal{I}_{k'}\right) + \bar{\mathcal{R}} \cdot \hat{\mathcal{I}}_k$
    $\mathcal{H} \leftarrow \mathcal{H} \cdot \bar{\mathcal{R}}$
    $k' \leftarrow k' - 1$
  end while
end for

restored image $\hat{\mathcal{I}}_{k-1}$ and copies its pixel value, regardless of whether it is inside or outside the mask. Figure 3 illustrates this in one-dimensional case. The repetition of this operation makes it eventually possible to refer to the right point in the right previous frames. While this operation has the advantages of high speed and low memory consumption due to its simplicity, it has a problem with image quality. Since the flow is obtained with sub-pixel accuracy, interpolation is necessary when propagating pixel values between the consecutive frames. This interpolation acts as a kind of low-pass filters and the resulting image tends to be blurry with a strength proportional to the number of propagation counts, as will be shown in Figure 7.

Considering our application, we employ a method that maximizes image quality while maintaining the constraints of real time execution. Specifically, we compute the optical flow between any distant frames k and k' by integrating the flows between adjacent frames (e.g., k and $k-1$), as $\hat{\mathcal{W}}^{(k)\rightarrow(k')} = \hat{\mathcal{W}}^{(k)\rightarrow(k'+1)} \odot \hat{\mathcal{W}}^{(k'+1)\rightarrow(k')} = \hat{\mathcal{W}}^{(k)\rightarrow(k-1)} \odot \cdots \odot \hat{\mathcal{W}}^{(k'+1)\rightarrow(k')}$. We then select the optimal frame for each pixel in the masked region and copy the pixel in the unmasked region of that frame. The pseudo code is shown in Algorithm 2. Note that the binary variable $\mathcal{R}$ in Algorithm 2 represents the region of k-th frame restored by the k'-th frame, and D is a parameter of integer that specifies the maximum depth of tracing. In this method, it is necessary to save the estimated optical flow at each frame in the memory. We assume here that sufficient memory is available.

3.4. Detecting image obstacles

We have assumed so far that the masks are given that specify the image regions to be restored. These masks are created by detecting the image obstacles we want to remove. There are several substances adherent to lens surface that are the target of this study, such as raindrops, dusts, dirt, etc. They are relatively easy to detect, by using standard CNN-based detectors trained with a realistic amount of training data. We can also employ existing methods for detecting raindrops etc. [27, 23, 11], some of which are model-based and do not need training data. As they work well on a single image, we can obtain a mask from each frame independently. We do this right after the latest video frame arrive before the start of the video inpainting pipeline. Its computational cost is much smaller than that of video inpainting, and thus the overall procedure can be performed in real time.

4. Experimental results

4.1. Dataset

We created a dataset for the evaluation of the proposed method as well as for its training. We captured videos of multiple scenes using a fish-eye rear-view camera mounted on the rear-end, above the license plate, of a vehicle; examples are shown in Figure 1. They are captured with the resolution of 1280×800 at the frame rate of 30 fps. We drove the vehicle on the street and parking spaces in urban areas to capture videos of 73 scenes. The total number of video frames is 191k.

We split these 73 scene videos into evaluation and training sets. The evaluation set consists of 22 scenes, which are the videos of different parking spaces containing 13k images in total. The training set consists of the remaining 51 scenes. We add masks to each of the video images in the evaluation set. The masks are created such that pedestrians including sitting persons, paintings, and other objects are occluded to assess the quality of the restored images in the most effective way. Figure 4 shows examples of images of a scene with the created masks.

4.2. Details of fine-tuning of the flow estimator

As explained above, we fine-tune the flow estimator on the video images obtained as above. For the flow estimator, we selected PWC-Net [24] and extended it to estimate the flows behind occluding objects by modifying its input layer. We first pre-train the extended model on the FlyingChairs dataset with synthetic occluding objects and their masks, following the procedure in [17].

We then fine-tune the pre-trained model. For this purpose, we choose 3,324 pairs of consecutive frames randomly from the videos of the training split. We also choose 360 pairs from the evaluation split for the evaluation of the optical flow estimation, which will be shown in Section 4.3.1. For each pair, we apply the flow estimation method [19] to obtain the pseudo ground-truth flows, as ex-

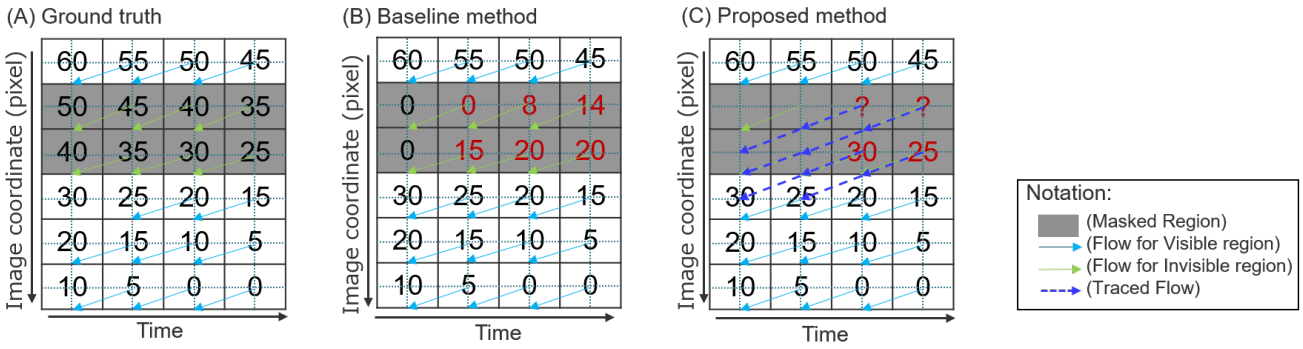

Figure 3. One-dimensional illustration of how each masked pixel in the current frame is matched to a unmasked pixel in a previous frame in the baseline and the proposed methods. (A) Ground-truth pixel values and flow field. (B) The strategy of reusing the restored image pixels in [17]. It needs the repetition of interpolation of pixel values. (C) The proposed strategy that back-traces the flows in previous frames. It needs the interpolation of flow vectors but does not need interpolation of pixel values.

Figure 4. An example of a video sequence of a parking space used for evaluation. The region highlighted in red color indicates the mask we specify. The curb is occluded by the mask in the last frame.

plained in Section 3.2.

For the training of the flow estimator, we synthesize occluding objects on the images of each pair for training. The experimental results in [17] show that the size and shape of the occluding objects synthetically generated for training do not have a large impact on the estimation accuracy. Considering also the shape similarity with raindrops into account, we choose a circular disk having the radius of 100 pixels for the occluding objects. Following [17], we fill them with black pixels. We randomly generate five disks per image and also create the masks having the same position and shape. Note that the pseudo ground-truth flows are estimated by the method of [19] from the pairs of original images without a synthetic occluding object. Using the Adam optimizer for the training with standard hyperparameters, we train the flow estimator for 500 epochs.

4.3. Results

4.3.1 Accuracy of optical flows

We first evaluate the effectiveness of the fine-tuning of the flow estimator. For this purpose, we compare two models, the pretrained and the fine-tuned models, in terms of accuracy of their estimated flows. It should be noted that for the 'true' flows to measure errors, we used the flows estimated by pyflow, as in the fine-tuning. Although it does not represent the true accuracy, we believe that this evaluation is useful, since we confirmed that the flows by pyflow is quite accurate, as is observed from the restored images using them; see the 'PsuedoGT' column in Figure 7. For the evaluation, we use 360 image pairs chosen as above, and create circular disks on each image pair in the same way as the training data except their sizes and positions. To be specific, we consider disks with three different radi, i.e., 10, 50, and 100 pixels; we then place a disk on three positions (i.e., left, center, and right) along the horizontal line located in the middle of the image. We measure the accuracy of the estimated flows over the whole image and also inside the occluding disks using *root-mean-square-error (RMSE)*.

Figure 5 shows the results, the RMSEs of the two models (pretrained denoted by 'FlyingChairs' vs. fine-tuned denoted by 'FineTuning') over the whole image and the mask region for three disk sizes. It is clearly seen that the fine-tuning on the additional dataset decreases the RMSEs; the improvements tend to be larger for smaller occluding disks. There is little difference due to the positions of the disks, and thus the results are omitted here.

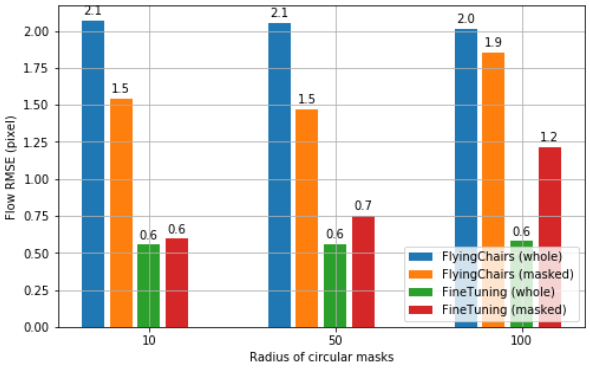

Figure 5. Accuracy of the estimated optical flows by the pretrained ('FlyingChairs') and the fine-tuned ('FineTuning') models. RMSE is used. The bars with '(whole)' show errors over the whole image and those with '(masked)' show errors of only the masked regions.

Figure 6. Results of the subjective evaluation experiment. Six subjects were asked to sort the restored images by the four methods plus the ground truth image according to their naturalness. The number in each cell indicates the frequency of the rank (column) of the method (row). The red dots show the average ranking scores for the five images.

4.3.2 Quality of restored images

Figure 7 shows examples of the restored images. To evaluate the effectiveness of the proposed two extensions, we apply four methods, i.e., the baseline method [17], that with fine-tuning (Section 3.2), that with flow tracing (Section 3.3), and that with both of them, to images with manually specified masks, which are shown in the second column of Figure 7. For the flow tracing, the parameter D in Algorithm 2 was set as $D = \infty$ to obtain maximum restoration quality. It is seen that the four methods restore the occluded scenes with different accuracy. Comparing the results obtained by those with fine-tuning and without it, we can see that the former yields results with smaller geometric distortion, confirming that the fine-tuning contributes to obtain more accurate flows. By comparing those with flow tracing and without it, it is observed that the former yields less blurry images, validating the proposed method of flow tracing. Thus, we conclude that the method with the proposed two extensions yields the best results.

For each result shown in Figure 7, we also show quantitative differences between the ground-truth image and the restored image, which are measured by the two standard metrics of image restoration, *peak signal-to-noise ratio (PSNR)*, and *structural similarity (SSIM)*, and shown below the title of each restored image. It is seen from them that the proposed method does *not* necessarily achieve the best scores. This is also confirmed from the average errors (PSNR/SSIM) over the 22 scenes for the four methods: 17.653/0.539 (baseline), 17.790/0.543 (fine-tuning), 16.493/0.508 (flow-tracing), and 16.711/0.524 (both). These are evaluated for the last frame of each video using its cropped regions associated with the masks, as shown in Figure 7. These results show that the method with only fine-tuning achieves the best score in the both metrics.

However, we do *not* believe that this result is inconsistent with the above observation. As has been recognized in recent image reconstruction studies (e.g., [2]), the error measures based on differences in pixel values, such as PSNR and SSIM, generally tend to show better scores for more blurry images. In particular, in our problem, there has to be some amount of alignment errors between the restored and the ground truth images, due to inevitable estimation errors of optical flows. In this case, blurring the restored images tends to make the difference in pixel values smaller. In short, these metrics are not suitable for evaluating the quality of the restored images.

Aiming at performing better evaluation of image quality, we also conduct an experiment of subjective evaluation. In the experiment, we show the restored images along with the ground-truth image for each scene to six subjects whose ages ranged from 28 to 56. For each of 22 scenes, we show five images (four restored images and the ground-truth) to each subject and ask her/him to sort them according to the naturalness of the restored images. The results are shown in Figure 6, which shows the distribution of the ranks of the five images given by the subjects. As there are 22 scenes, five images per scene, and six subjects, we collected $6 \times 22 \times 5 = 660$ answers in total. Each of the five rows of the matrix in Figure 6 shows the rank distribution of $132 (= 6 \times 22)$ answers. It is seen that the results by the method with both fine-tuning and flow tracing are given higher ranks than others but the ground-truth, and the baseline method are given the lowest rank. This validates well the effectiveness of the proposed method.

5. Conclusion

In this paper, we have shown a method for removing obstacles from the video of a SMC mounted on a vehicle for

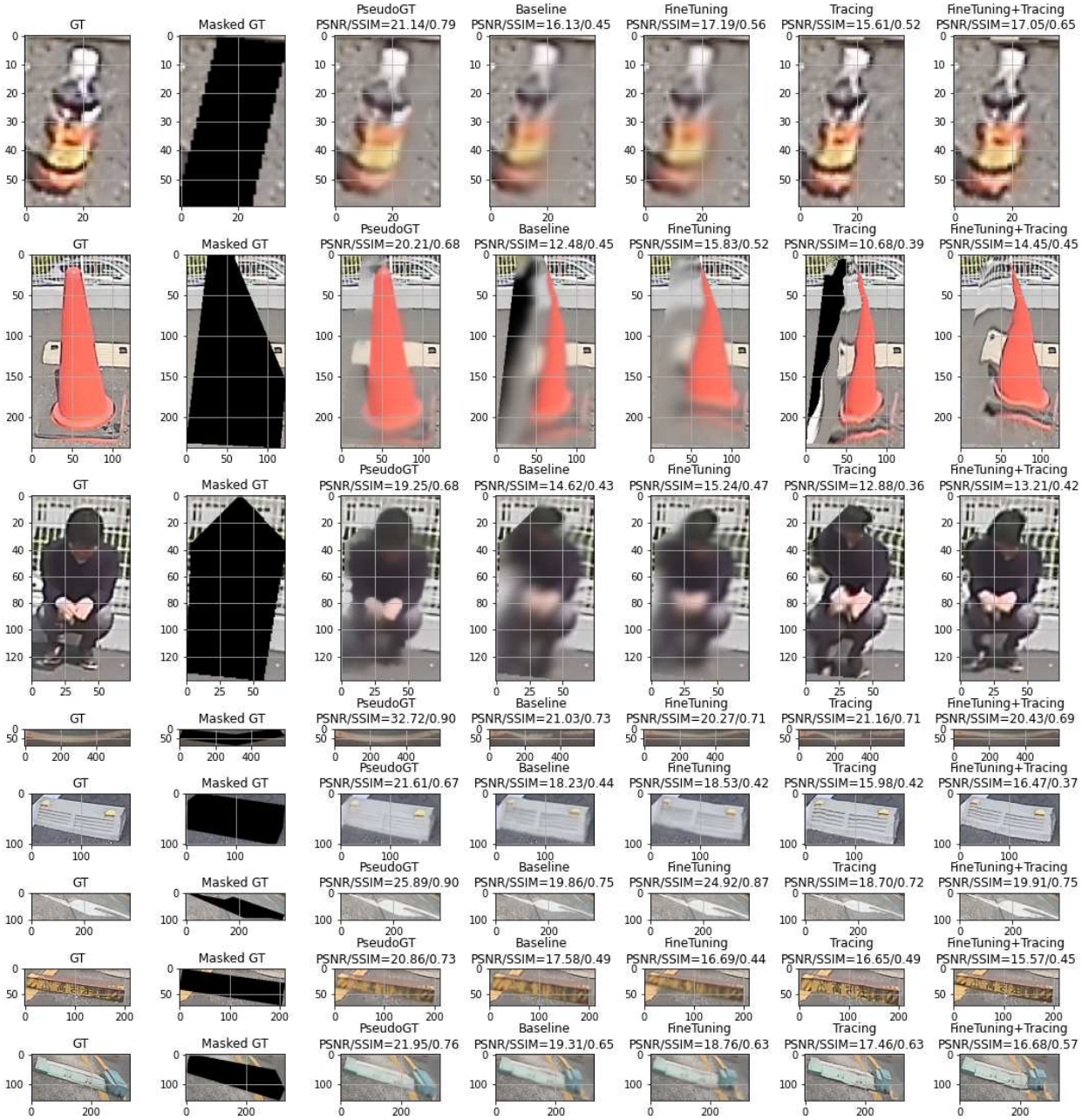

Figure 7. Examples of the image restoration by the baseline method with the pseudo ground-truth (pyflow) flows, the baseline method [17], with fine-tuning, with flow tracing, and with fine-tuning + flow tracing (the proposed method). Each image is cropped from its original version based on the specified mask (shown in the second column).

the purpose of assisting the driver. Considering the requirements for real-time processing and high fidelity of restored images, we adopt a recently proposed method for real time video inpainting, which estimates optical flows by a CNN and use them to match occluded and unoccluded image regions to restore the former. However, we found that the direct application does not lead to satisfactory results due to the peculiarities of SMC images. Specifically, the estimation accuracy of the flow is insufficient and the restored images tend to be blurry. To solve the former, we used a model-based optical flow estimation method, which is unaffected by domain shift associated with the training data, to obtain target flows and train the CNN to predict them. To solve the latter problem, we improved how the estimated flows are used to match occluded and unoccluded image regions. Experiments with real images confirm that these improvements make it possible to remove obstacles from SMC camera images for the purpose of supporting the driver.

References

[1] Mathieu Aubry, Daniel Maturana, Alexei A Efros, Bryan C Russell, and Josef Sivic. Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3762–3769, 2014. 3

[2] Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6228–6237, 2018. 7

[3] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. High accuracy optical flow estimation based on a theory for warping. In *European conference on computer vision*, pages 25–36. Springer, 2004. 4

[4] Andrés Bruhn, Joachim Weickert, and Christoph Schnörr. Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *International journal of computer vision*, 61(3):211–231, 2005. 4

[5] Joachim Dahl, Per Christian Hansen, Søren Holdt Jensen, and Tobias Lindstrøm Jensen. Algorithms and software for total variation image reconstruction via first-order methods. *Numerical Algorithms*, 53(1):67, 2010. 2

[6] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015. 3

[7] David Eigen, Dilip Krishnan, and Rob Fergus. Restoring an image taken through a window covered with dirt or rain. In *Proceedings of the IEEE international conference on computer vision*, pages 633–640, 2013. 1

[8] Raanan Fattal. Single image dehazing. *ACM transactions on graphics (TOG)*, 27(3):1–9, 2008. 1

[9] Kshitiz Garg and Shree K Nayar. Photometric model of a rain drop. In *CMU Technical Report*. 2003. 2

[10] Jinwei Gu, Ravi Ramamoorthi, Peter Belhumeur, and Shree Nayar. Removing image artifacts due to dirty camera lenses and thin occluders. *ACM Transactions on Graphics (TOG)*, 28(5):1–10, 2009. 1

[11] T. Guo, S. Akcay, P. A. Adey, and T. P. Breckon. On the impact of varying region proposal strategies for raindrop detection and classification using convolutional neural networks. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3413–3417, 2018. 5

[12] Z. Hao, S. You, Y. Li, K. Li, and F. Lu. Learning from synthetic photorealistic raindrop for single image raindrop removal. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 4340–4349, 2019. 1, 2

[13] Kaiming He, Jian Sun, and Xiaoou Tang. Single image haze removal using dark channel prior. *IEEE transactions on pattern analysis and machine intelligence*, 33(12):2341–2353, 2010. 1

[14] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017. 3

[15] Hiroyuki Kurihata, Tomokazu Takahashi, Ichiro Ide, Yoshito Mekada, Hiroshi Murase, Yukimasa Tamatsu, and Takayuki Miyahara. Rainy weather recognition from in-vehicle camera images for driver assistance. In *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.*, pages 205–210. IEEE, 2005. 3

[16] Xing Liu, Masanori Suganuma, Zhun Sun, and Takayuki Okatani. Dual residual networks leveraging the potential of paired operations for image restoration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7007–7016, 2019. 2

[17] Rito Murase, Yan Zhang, and Takayuki Okatani. Video-rate video inpainting. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1553–1561. IEEE, 2019. 2, 3, 4, 5, 6, 7, 8

[18] Fawzi Nashashibi, Raoul de Charrette, and Alexandre Lia. Detection of unfocused raindrops on a windscreen using low level image processing. In *2010 11th International Conference on Control Automation Robotics & Vision*, pages 1410–1415. IEEE, 2010. 2

[19] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2701–2710, 2017. 2, 5, 6

[20] Horia Porav, Tom Bruls, and Paul Newman. I can see clearly now: Image restoration via de-raining. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7087–7093. IEEE, 2019. 1, 2

[21] Rui Qian, Robby T Tan, Wenhan Yang, Jiajun Su, and Jiaying Liu. Attentive generative adversarial network for raindrop removal from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2482–2491, 2018. 1, 2

[22] Martin Roser and Andreas Geiger. Video-based raindrop detection for improved image registration. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 570–577. IEEE, 2009. 2

[23] Martin Roser, Julian Kurz, and Andreas Geiger. Realistic modeling of water droplets for monocular adherent raindrop recognition using bezier curves. In *Asian Conference on Computer Vision*, pages 235–244. Springer, 2010. 5

[24] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8934–8943, 2018. 3, 5

[25] R. T. Tan. Visibility in bad weather from a single image. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. 1

[26] Guoqing Wang, Changming Sun, and Arcot Sowmya. Erl-net: Entangled representation learning for single image de-raining. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 1, 2

[27] Reg G Willson, Mark W Maimone, Andrew E Johnson, and Larry M Scherr. An optical model for image artifacts produced by dust particles on lenses. 2005. 5

[28] A. Yamashita, I. Fukuchi, and T. Kaneko. Noises removal from image sequences acquired with moving camera by estimating camera motion from spatio-temporal information. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3794–3801, 2009. 2

[29] A. Yamashita, Y. Tanaka, and T. Kaneko. Removal of adherent waterdrops from images acquired with stereo camera. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 400–405, 2005. 2

[30] Shaodi You, Robby T Tan, Rei Kawakami, Yasuhiro Mukaigawa, and Katsushi Ikeuchi. Adherent raindrop modeling, detectionand removal in video. *IEEE transactions on pattern analysis and machine intelligence*, 38(9):1721–1733, 2015. 1, 2