Unsupervised Model-based Learning for Simultaneous Video Deflickering and Deblotching

Anuj Fulari, Satish Mulleti, and Ajit Rajwade
IIT Bombay, India
anujfulari@cse.iitb.ac.in,mulleti.satish@gmail.com, ajitvr@cse.iitb.ac.in

Abstract

Vintage videos, as well as modern day videos acquired at high frame rates, suffer from a visually disturbing artifact called flicker, which is the rapid change in average intensity across consecutive frames. Vintage videos also suffer from blotch artifacts, i.e., each video frame contains small regions at random locations with undefined pixel values. We present a model-based learning approach to remove flicker as well as blotches simultaneously. Our work uses a pixel-wise affine intensity model for flicker between neighboring frames, with coefficients that vary smoothly in the spatial sense but randomly across time. Due to smooth spatial variation, the flicker coefficients for any given frame can be modelled as linear combinations of low-frequency discrete cosine transform (DCT) bases. We also model blotches as heavy-tailed but sparse artifacts affecting every frame. We then present a novel framework to restore the video frames by jointly *estimating the blotches as well as the DCT coefficients of the flicker via convex optimization. Given the high computational cost of the optimization-based method for processing an entire video, we use a deep unrolled neural network approach to achieve similar restoration quality at significantly reduced cost. Our approach is completely unsupervised and model-based, and hence simple and interpretable. It produces high-quality reconstructions, in terms of visual appeal as well as numerical metrics, on a variety of vintage videos as well as high-speed videos. It does not suffer from generalization issues unlike some recent state-of-the-art supervised methods which use end-to-end neural networks for restoration.*

1. Introduction

Videos recorded on magnetic tapes, as was the norm from the 1930s till the 1980s, often suffer from several different types of visual artifacts. From the point of view of heritage preservation, it is important to enhance the visual appeal of these videos by removing or mitigating some of these artifacts. Artifact removal also makes these videos better suited for compression via standards such as MPEG. The primary artifacts include flicker and blotches. Flicker involves rapid variation in the average intensity value across video frames, and is caused primarily due to variation in exposure period across frames during video recording, or due to chemical processing, film aging or aliasing. Blotches are caused due to mechanical damages, dust, scratches or cracks on the magnetic tape used for video recording. More information on the causes of these artifacts can be found in [15, 30]. Besides vintage videos, flicker can also adversely affect modern-day videos acquired under artificial illumination at high frame rates [14, 25]. Here, the flicker arises due to the alternating currents (AC) used for illumination, leading to periodic illumination intensity variation from frame to frame.

Manual approaches to restore all such videos are very laborious and error-prone and hence automated techniques are needed. There has been surprisingly limited literature on this problem. Earlier work in [26, 30] proposes a pixel-wise affine intensity model for the flicker, i.e., for intensity variation across consecutive frames. The coefficients of this model vary smoothly in space but randomly in time. Using iterated reweighted least squares (IRLS) to minimize a non-convex ℓ_p-based ($0 < p < 1$) cost function, the frame-to-frame flicker coefficients are obtained and then smoothed across time to obtain the restored video [26]. The IRLS technique is computationally demanding and its convergence rate is not necessarily always bounded. Moreover, this approach does not account for blotches explicitly and instead relies on a pre-processing step via other techniques such as the JoMBaDI (Joint Model-Based Detection and Interpolation) algorithm from [15, Sec. 7.6]. However, JoMBaDI uses Markov Chain Monte Carlo (MCMC) and can be computationally prohibitive. The work in [8] performs flicker removal at the level of local patches by temporal filtering, involving minimization of a weighted distance function between a reference patch and its most similar patches from nearby frames. The patch similarity is computed in a manner invariant to affine intensity changes.

This method side-steps the requirement for accurate motion estimation but ignores blotches and requires expensive patch-matching.

There have been recent deep learning-based techniques to tackle the problem of video restoration. The recent work in [13] targets denoising, blotch removal and colorization via an attention-guided temporal convolutional neural network (CNN), but does not explicitly account for flicker. The more recent work in [31] presents an end-to-end approach that trains a recurrent transformer network (RTN) to jointly optimize a combined cost function consisting of an intensity-based ℓ_1-loss, perceptual loss and a GAN-based loss, all computed between the restored result and the ground truth image. The network is trained on large amounts of video data, synthetically degraded by flicker, noise, blotches, blur and blocking artifacts. This approach produces excellent results on a wide variety of videos. However, as is true of supervised approaches, its results do not always generalize to datasets different from what the network was trained on, as we shall later demonstrate. Even more recently, [18] proposes a strategy to produce a learn a mapping network, an atlas network for filtering of flickery videos and a local refinement network. The networks are trained on synthetic data, and produce good deflickering results, but the problem of removal of blotches is not addressed. There are many techniques which perform per-frame processing of temporally consistent videos for applications like dehazing, white balancing, denoising, deblurring, etc. The video outputs, however, often contain flicker artifacts. There exist approaches to improve temporal consistency and stability of such videos using methods such as recurrent networks [16], variants of the Poisson equation [7, 34] and U-net-based priors [19, 20]. But such techniques require a temporally consistent original video to act as a guide for the deflickering algorithm. In applications such as deflickering and deblotching of old movies or high-speed videos, such a guiding video is just not available and hence the applicability of these techniques is quite limited, as will be demonstrated via numerical comparison to a variant of [16]. The techniques in [14, 17] performs filtering of intensity values across optical flow-based trajectories or super-pixel trajectories to achieve deflickering. However, it does not use the compactness of flicker in the spatial frequency domain, unlike the method we propose in this paper. In addition, none of the techniques [7, 16–19, 34] perform deblotching.

In this paper, we adopt an unsupervised and model-based approach for simultaneous flicker and blotch removal. The complete framework of the proposed method is given in Fig. 1. Our method is interpretable and simple and proposes a novel strategy to jointly deal with flicker and blotches via convex optimization. The core engine in our approach is analytical, and it is fine-tuned and made more efficient via an unrolled neural network approach. We produce high-quality results on a wide variety of real videos gathered from diverse sources. Our results are on par or better than those produced by recent state-of-the-art approaches, with efficient computing time.

2. Method

Our goal is to develop a model to eliminate flicker and blotches from a given video without any manual intervention. Let $\{\widetilde{I}_1, \widetilde{I}_2, ..., \widetilde{I}_T\}$ be the frames of the video sequence to be restored, where each $\widetilde{I}_i, i \in \{1, 2, ..., T\}$, contains n pixels.

2.1. Mathematical Models

2.1.1 Flicker

Consider two consecutive frames $\widetilde{I}_1$ and $\widetilde{I}_2$ which are images of the same scene from different viewpoints. Pixels corresponding to the same physical location in $\widetilde{I}_1$ and $\widetilde{I}_2$ would have the same/similar intensity in the absence of flicker. But flicker artifacts arise for various reasons, and can be modeled as multiplicative and/or additive changes in the intensities at corresponding locations across frames. Mathematically, this can be expressed as follows:

$$\widetilde{I}_1^{\,corr}(x, y) = \widetilde{I}_2(x, y)m(x, y) + a(x, y) + \eta(x, y). \quad (1)$$

Here, (x, y) is the pixel location; $\eta(x, y)$ represents noise at (x, y); $m(x, y)$ and $a(x, y)$ are multiplicative and additive changes respectively in the intensity at pixel (x, y) of the frame $\widetilde{I}_2$ which will result in the corresponding pixel value in the frame $\widetilde{I}_1$. Note that $\widetilde{I}_1^{\,corr}$ represents the frame $\widetilde{I}_1$ after it is spatially aligned with $\widetilde{I}_2$ using dense optical flow. To perform the alignment, any suitable illumination-invariant optical flow method can be used. In our work, we have used a state-of-the-art optical flow method called RAFT [29].

From empirical observations, it appears that flicker causes spatially *low-frequency* intensity changes (multiplicative as well as additive), as also mentioned in [26, Ch. 2], though its temporal variation is of a random nature. In order to compactly represent the spatially smooth nature of the field of multiplicative coefficients $\boldsymbol{m} \in \mathbb{R}^n$ and the field of additive coefficients $\boldsymbol{a} \in \mathbb{R}^n$, we express them in some frequency domain such as discrete cosine transform (DCT), in the form $\boldsymbol{m} = \boldsymbol{\Psi\theta_m}, \boldsymbol{a} = \boldsymbol{\Psi\theta_a}$. Here $\boldsymbol{\Psi} \in \mathbb{R}^{n\times K}, K < n$ is the 2D IDCT (Inverse DCT) matrix containing $K < n$ low frequency cosine bases, whereas $\boldsymbol{\theta_m} \in \mathbb{R}^{K\times 1}$ and $\boldsymbol{\theta_a} \in \mathbb{R}^{K\times 1}$ are 2D-DCT (low frequency) coefficients of $\boldsymbol{m}$ and $\boldsymbol{a}$ respectively. Plugging these relations into Eqn. 1, we obtain:

$$\widetilde{I}_1^{\,corr} = \text{diag}(\widetilde{I}_2)\boldsymbol{\Psi\theta_m} + \boldsymbol{\Psi\theta_a} + \boldsymbol{\eta}. \quad (2)$$

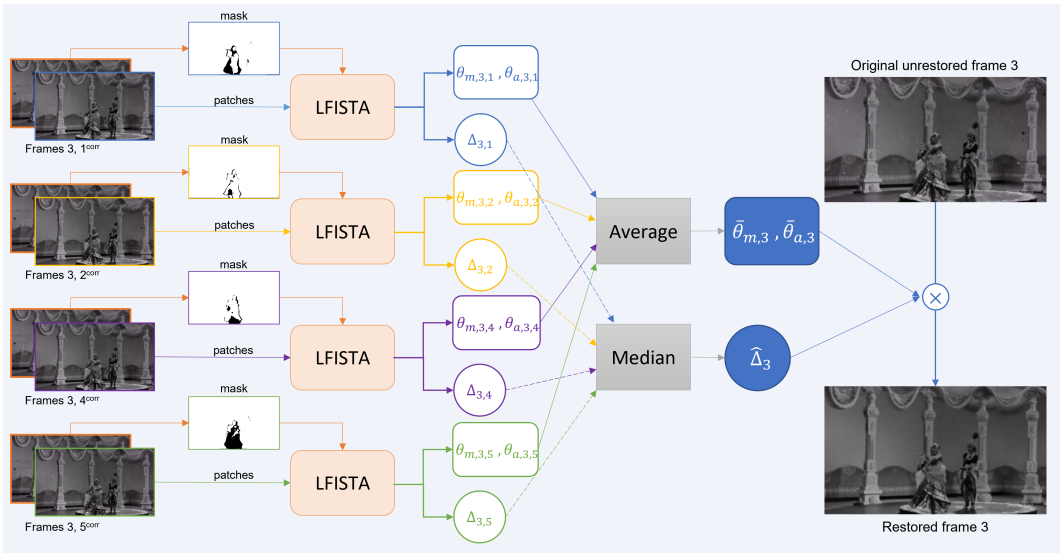

Figure 1. Overview of the LFISTA Algorithm: restoration of frame 3 using its neighboring frames $\mathcal{T}(3) := \{1, 2, 4, 5\}$. For more details including definitions of flicker coefficients $\bar{\boldsymbol{\theta}}_{\boldsymbol{m,3}}, \bar{\boldsymbol{\theta}}_{\boldsymbol{a,3}}, \boldsymbol{\theta}_{\boldsymbol{m,i,j}}, \boldsymbol{\theta}_{\boldsymbol{a,i,j}}$ and blotch vectors $\hat{\boldsymbol{\Delta}}_{\boldsymbol{3}}, \boldsymbol{\Delta}_{\boldsymbol{i,j}}$, refer to Sec. 2.1.2, 2.2.

Here $\widetilde{I}_1^{\,corr}, \widetilde{I}_2$ are expressed as $n \times 1$ vectors (just like $\boldsymbol{m}, \boldsymbol{a}$), and $\boldsymbol{\eta}$ is the additive noise vector. Moreover, we model $\boldsymbol{\theta_m}, \boldsymbol{\theta_a}$ to be sparse (see Eqn. 7), which enables a conservatively large choice of K, whose precise value would be unknown in practice. This model is similar to that presented in [26, Sec. 4.2], but as we will see in the next section, the novelty of our technique emerges from its combination with blotches, and a solution to *jointly* remove flicker and blotch artifacts via convex optimization.

2.1.2 Flicker and Blotches

Eqn. 2 represents flicker but does not model blotches. Generally, blotches occur at random locations within a frame, and their locations change completely across frames. Let I_1 and I_2 be the clean frames without any blotch artifacts corresponding to the observed (to be restored) frames $\widetilde{I}_1$ and $\widetilde{I}_2$ respectively, as shown in Eqn. 3 below:

$$I_1 = \widetilde{I}_1 + \boldsymbol{\Delta_1} \implies I_1^{corr} = \widetilde{I}_1^{corr} + \boldsymbol{\Delta_1^{corr}}, I_2 = \widetilde{I}_2 + \boldsymbol{\Delta_2}. \tag{3}$$

Here $\boldsymbol{\Delta_1}$ and $\boldsymbol{\Delta_2}$ are sparse vectors which contain non-zero values at the location of blotches in frames I_1 and I_2 respectively and are zero-valued elsewhere. These vectors are sparse because blotches generally occupy a very small area in actual vintage videos. Modifying Eqn. 2 using the blotch terms from Eqn. 3, we obtain:

$$\begin{aligned} \widetilde{I}_1^{\,corr} + \boldsymbol{\Delta_1^{corr}} &= \text{diag}(\widetilde{I}_2 + \boldsymbol{\Delta_2})\boldsymbol{\Psi\theta_m} + \boldsymbol{\Psi\theta_a} + \boldsymbol{\eta}, \\ \widetilde{I}_1^{\,corr} &= \text{diag}(\widetilde{I}_2)\boldsymbol{\Psi\theta_m} + \boldsymbol{\Psi\theta_a} + \boldsymbol{\Delta_{2,1}} + \boldsymbol{\eta}, \end{aligned} \tag{4}$$

where $\boldsymbol{\Delta_{2,1}} := \text{diag}(\boldsymbol{\Delta_2})\boldsymbol{\Psi\theta_m} - \boldsymbol{\Delta_1^{corr}}$. We note that the vector $\boldsymbol{\Delta_{2,1}}$ emerges from blotch artifacts and is a vector with sparse support, since both $\boldsymbol{\Delta_1}$ and $\boldsymbol{\Delta_2}$ are sparse.

2.2. Joint Deflickering and Deblotching

Consider a frame $\widetilde{I}_k$ and a temporal neighborhood of frames with radius W around it. Given a neighboring frame $\widetilde{I}_j$ with $j \in \mathcal{T}(k) := \{k - W, ..., k + W\}$, we have:

$$\widetilde{I}_j^{\,corr} = \text{diag}(\widetilde{I}_k)\boldsymbol{\Psi\theta_{m,k,j}} + \boldsymbol{\Psi\theta_{a,k,j}} + \boldsymbol{\Delta_{k,j}} + \boldsymbol{\eta}, \tag{5}$$

where $\boldsymbol{\theta_{m,k,j}}, \boldsymbol{\theta_{a,k,j}}$ are coefficients for multiplicative and additive intensity changes to make $\widetilde{I}_k$ similar to $\widetilde{I}_j^{\,corr}$, and $\boldsymbol{\Delta_{k,j}}$ is defined as follows (similar to the definition of $\boldsymbol{\Delta_{1,2}}$ in Eqn. 4):

$$\boldsymbol{\Delta_{k,j}} := \text{diag}(\boldsymbol{\Delta_k})\boldsymbol{\Psi\theta_{m,k,j}} - \boldsymbol{\Delta_j^{corr}}. \tag{6}$$

Given this relationship, for each $j \in \mathcal{T}(k)$ we can determine $\boldsymbol{\theta_{m,k,j}}, \boldsymbol{\theta_{a,k,j}}, \boldsymbol{\Delta_{k,j}}$ by setting them to be respectively equal to the minimizers of the following convex optimization function where $p \geq 1$:

$$\begin{aligned} &J_1(\boldsymbol{\theta_{m,k,j}}, \boldsymbol{\theta_{a,k,j}}, \boldsymbol{\Delta_{k,j}}) := \\ &\|\widetilde{I}_j^{\,corr} - \text{diag}(\widetilde{I}_k)\boldsymbol{\Psi\theta_{m,k,j}} - \boldsymbol{\Psi\theta_{a,k,j}} - \boldsymbol{\Delta_{k,j}}\|_p^p + \\ &\lambda\|\boldsymbol{\theta_{m,k,j}}\|_1 + \lambda\|\boldsymbol{\theta_{a,k,j}}\|_1 + \lambda\|\boldsymbol{\Delta_{k,j}}\|_1, \end{aligned} \tag{7}$$

where λ is a regularization parameter that can be tuned via cross-validation [35]. Note that the above formulation exploits the sparsity of $\boldsymbol{\Delta_{k,j}}$, as well as that of $\boldsymbol{\theta_{m,k,j}}, \boldsymbol{\theta_{a,k,j}}$. If $p = 2$, the above problem is a robust version [23] of the well-known LASSO problem in statistics [12]. It can be

solved by efficient algorithms such as FISTA (fast iterative shrinkage and thresholding algorithm) [6]. By obtaining $\{(\boldsymbol{\theta_{m,k,j}}, \boldsymbol{\theta_{a,k,j}})\}_{j\in\mathcal{T}(k)}$ and computing the average coefficients $\bar{\boldsymbol{\theta}}_{\boldsymbol{m,k}} := \frac{1}{|\mathcal{T}(k)|}\sum_{j\in\mathcal{T}(k)} \boldsymbol{\theta_{m,k,j}}$ as well as $\bar{\boldsymbol{\theta}}_{\boldsymbol{a,k}} := \frac{1}{|\mathcal{T}(k)|}\sum_{j\in\mathcal{T}(k)} \boldsymbol{\theta_{a,k,j}}$, we can perform deflickering in the *absence* of blotches by rendering a restored frame in the following manner: $I_{k,restored} = \text{diag}(\widetilde{I}_k)\boldsymbol{\Psi}\bar{\boldsymbol{\theta}}_{\boldsymbol{m,k}} + \boldsymbol{\Psi}\bar{\boldsymbol{\theta}}_{\boldsymbol{a,k}}$.

However additional work needs to be done if the video frames contain blotches. In particular, we cannot assume that any single frame would be completely free from blotches. For deblotching, we need to determine $\boldsymbol{\Delta_k}$, for which we consider the relation for $\boldsymbol{\Delta_{k,j}}$ in Eqn. 6 for every $j \in \mathcal{T}(k)$. In Eqn. 6, we note that both $\boldsymbol{\Delta_k}$ as well as $\boldsymbol{\Delta_j}$ are unknown, whereas $\boldsymbol{\Delta_{k,j}}, \boldsymbol{\theta_{m,k,j}}$ are obtained via FISTA. Moreover, we note that $\boldsymbol{\Delta_k}$ is common in Eqn. 6 across all $j \in \mathcal{T}(k)$. Also, $\boldsymbol{\Delta_j}$ can be modelled as noise with sparse support but heavy tails. Taking all this into account, we can determine $\boldsymbol{\Delta_k}$ by minimizing the following robust convex cost function:

$$J_2(\boldsymbol{\Delta_k}) := \sum_{j\in\mathcal{T}(k)} \|\boldsymbol{\Delta_{k,j}} - \text{diag}(\boldsymbol{\Delta_k})\boldsymbol{\Psi}\boldsymbol{\theta_{m,k,j}}\|_1, \quad (8)$$

which has a closed form solution given by

$$\hat{\Delta}_k(l) = \text{median}_{j\in\mathcal{T}(k)}\Delta_{k,j}(l)/\boldsymbol{\Psi^l}\boldsymbol{\theta_{m,k,j}}, \quad (9)$$

where $\hat{\Delta}_k(l), \Delta_{k,j}(l)$ are used to denote the lth element of the vectors $\hat{\boldsymbol{\Delta}}_{\boldsymbol{k}}, \boldsymbol{\Delta_{k,j}}$ respectively (and $\hat{\boldsymbol{\Delta}}_{\boldsymbol{k}}$ is an estimate of $\boldsymbol{\Delta_k}$), $\boldsymbol{\Psi^l}$ represents the lth row of $\boldsymbol{\Psi}$ and $l \in \{1, 2, ..., n\}$. In the above equation, note that for every l, the median is computed over the ratios $\Delta_{k,j}(l)/\boldsymbol{\Psi^l}\boldsymbol{\theta_{m,k,j}}$ over all $j \in \mathcal{T}(k)$. Given this estimate of $\boldsymbol{\Delta_k}$, the final restored image is given by:

$$I_{k,restored} = \text{diag}(\widetilde{I}_k + \hat{\boldsymbol{\Delta}}_{\boldsymbol{k}})\boldsymbol{\Psi}\bar{\boldsymbol{\theta}}_{\boldsymbol{m,k}} + \boldsymbol{\Psi}\bar{\boldsymbol{\theta}}_{\boldsymbol{a,k}}. \quad (10)$$

In order to restore the video, this procedure is repeated independently for every video frame.

2.3. Handling Regions of Occlusion/Optical Flow Errors

The aforementioned procedures assume that the optical flow computed between $\widetilde{I}_k$ and $\widetilde{I}_j$ is accurate. However, this is not always true in practice, especially at boundaries of fast moving objects where occlusions are present. In such cases, our experiments reveal that there is undesirable conflation between regions of occlusions due to moving objects or change of field of view or optical flow errors (referred to henceforth as R_o) and the blotch regions determined by $\boldsymbol{\Delta_k}$ (referred to henceforth as R_b). Note that we wish to perform intensity-based inpainting in R_b but not in R_o (implicitly via Eqn. 10), and hence any conflation between R_b and R_o will lead to sub-optimal restoration results. To resolve this issue, we create a **mask**, which will allow us to ignore pixels in R_o while estimating $\boldsymbol{\theta_{m,k,j}}$, $\boldsymbol{\theta_{a,k,j}}$ and $\boldsymbol{\Delta_{k,j}}$ via Eqn. 7. By construction, this mask will have the value 0 in R_o and 1 elsewhere including in R_b. For example, consider two neighboring frames $\widetilde{I}_1$ and $\widetilde{I}_2$. To find regions of occlusions between these two frames, we consider the so called residual flow $\vec{r}(x, y)$, which is defined to be the sum of the forward optical flow (u_{12}) vector from $\widetilde{I}_1$ to $\widetilde{I}_2$ and the backward optical flow vector (u_{21}) from $\widetilde{I}_2$ to $\widetilde{I}_1$ at corresponding locations. This is given as:

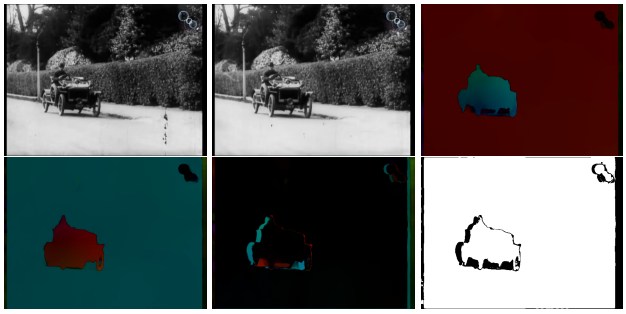

Figure 2. Mask creation. Left to right, top to bottom: Frames I_1, I_2; flow fields u_{12}, u_{21}; Magnitude of $\vec{r}$ and Occlusion Mask $\boldsymbol{M_{12}}$. (Note: Flow images are brightened for better visualization). Notice how the blotch region on the road is not masked out.

$$\vec{r}(x, y) := u_{12}(x, y) + u_{21}(x + u^x_{12}(x, y), y + u^y_{12}(x, y)). \quad (11)$$

The idea is that $\vec{r}(x, y)$ will be close to zero at the pixels with valid optical flow vectors (i.e. with valid corresponding points in $\widetilde{I}_1, \widetilde{I}_2$), as the forward and backward flow vectors will have equal magnitude but opposite directions. However fast moving objects which are present in the foreground will be associated with regions of occlusions, mostly at the boundary of those objects. Points from $\widetilde{I}_1$, which are absent in $\widetilde{I}_2$ due to the occlusions, will possess motion vectors nearly equal to the background motion vectors. However, there will be a fast-moving object at their corresponding location in $\widetilde{I}_2$ with a very different motion vector. Hence, the residual flow will be non-zero. We set the mask values at points (x, y) whose residual magnitude $\|\vec{r}(x, y)\|_2$ exceeds some threshold τ_r, to 0, and set the mask value at remaining pixels to be 1.

To construct a mask efficiently, it is crucial to distinguish between R_o and R_b. To this end, we use the smoothness priors in optical flow, as exploited by different optical flow estimation algorithms including RAFT [29]. Due to this smoothness prior, the optical flow at pixels in R_b will be very similar to that of the background or surrounding regions, if the blotches occur in regions of smooth motion. Due to this property, the residual flow vectors in R_b will have a magnitude close to 0, and hence those pixels will have a mask value of 1 (i.e., they are not masked out). On

the other hand, this similarity does not occur in R_o, as argued earlier. An example of the mask estimation procedure is shown in Fig. 2. Notice for example, that the black blotch on the road present in $\widetilde{I}_1$ is not masked away because it has optical flow equal to the background in both the forward and backward optical flow. This mask $\boldsymbol{M_{k,j}}$ (a binary vector of size $n \times 1$) is estimated for every pair of frames $(\widetilde{I}_k, \widetilde{I}_j)$ under consideration, with $j \in \mathcal{T}(k)$. The mask $\boldsymbol{M_{k,j}}$ is multiplied element-wise with the pair of frames under consideration: $\widetilde{I}_j^{\,corr}$ and $\widetilde{I}_k$. This effectively ignores the pixels in R_o, leading to the following cost function with $p \geq 1$:

$$J_3(\boldsymbol{\theta_m}, \boldsymbol{\theta_a}, \boldsymbol{\Delta}) := \|\boldsymbol{M_{k,j}} \circ \widetilde{I}_j^{\,corr} - \text{diag}(\boldsymbol{M_{k,j}})(\text{diag}(\widetilde{I}_k)\boldsymbol{\Psi\theta_m} + \boldsymbol{\Psi\theta_a} + \boldsymbol{\Delta})\|_p^p + \lambda\|\boldsymbol{\theta_m}\|_1 + \lambda\|\boldsymbol{\theta_a}\|_1 + \lambda\|\boldsymbol{\Delta}\|_1, \quad (12)$$

where $\circ$ represents element-wise multiplication. (Compare this to Eqn. 7.) Note again that at pixel (x, y) where $\boldsymbol{M_{k,j}}(x, y) = 0$, the value of $\boldsymbol{\Delta}(x, y)$ will be meaningless. In our experiments, we observed it to be zero consistently due to the presence of the $\|\boldsymbol{\Delta}\|_1$ term in $J_3(.)$.

This framework from Eqn. 12 is a novel method of correcting both flicker and blotches, when compared to previous methods such as [15, 26, 30] or recent methods [13, 31]. It is intuitive and uses simple convex optimization. Our method in fact *inherently* couples deflickering and deblotching as seen in Eqns. 7 and 12, where $\boldsymbol{\theta_{m,k,j}}, \boldsymbol{\theta_{a,k,j}}, \boldsymbol{\Delta_{k,j}}$ are *jointly* estimated in a robust LASSO framework. Here $\boldsymbol{\Delta_{k,j}}$ is defined in Eqn. 6, which shows that it depends on $\boldsymbol{\Delta_j}, \boldsymbol{\Delta_k}$ and $\boldsymbol{\theta_{m,k,j}}$. As both frames $\tilde{I}_j$ and $\tilde{I}_k$ could contain blotches, separate estimation of the flicker coefficients and $\boldsymbol{\Delta_{k,j}}$ is *not* possible. The method cannot be interpreted or implemented as deflickering followed by deblotching or vice versa. Note that from Eqn. 10, we see that the restoration process requires $\boldsymbol{\theta_m}, \boldsymbol{\theta_a}$ as well as $\boldsymbol{\Delta_k}$. The method of obtaining $\boldsymbol{\Delta_k}$ from $\boldsymbol{\Delta_{k,j}}$ in Eqns. 8, 9 is a novel contribution to the deblotching literature. It cannot be accomplished by spatial/temporal median filtering on blotchy videos, which will not work well due to possibly large area of some blotches.

3. Learned FISTA

The cost function in Eqn. 12 is minimized using the well-known FISTA algorithm [6]. However, FISTA, despite its fast convergence, is too slow for video processing, especially for large frame sizes. In this section, we describe an unrolled neural network-based approach that mimics the workings of FISTA, but with additional tunability and significantly greater speed. For convenience, we express the cost function in Eqn. 12 in the following way for $p = 2$, where $\mathbb{I}$ stands for the $n \times n$ identity matrix and t stands for the transpose of a vector:

$$J_4(\boldsymbol{\Theta}) := \|\boldsymbol{x} - \boldsymbol{\Phi W \Theta}\|_2^2 + \lambda\|\boldsymbol{\Theta}\|_1, \text{ where}$$
$$\boldsymbol{x} := \boldsymbol{M_{k,j}} \circ \widetilde{I}_j^{\,corr}, \boldsymbol{\Theta} := (\boldsymbol{\theta_m}^t|\boldsymbol{\theta_a}^t|\boldsymbol{\Delta}^t)^t,$$
$$\boldsymbol{\Phi} := \text{diag}(\boldsymbol{M_{k,j}}) \begin{bmatrix}\text{diag}(\widetilde{I}_k) & \mathbb{I} & \mathbb{I}\end{bmatrix},$$
$$\boldsymbol{W} := \begin{bmatrix}\boldsymbol{\Psi} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Psi} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} & \mathbb{I}\end{bmatrix}. \quad (13)$$

The FISTA algorithm used to optimize the cost function in Eqn. 13 is summarized in Alg. 1 [6]. The work in [11] showed that *learning* the internal parts of the FISTA algorithm (via a neural network) can reduce the computational time by a factor more than 20. Steps $3-6$ of Alg. 1 represent the gradient step followed by soft thresholding and momentum steps. These can be represented as a single (learned) layer of a neural network. P consecutive iterations of FISTA can be interpreted as the cascading of P such layers together. This forms a deep feed-forward network with P layers as shown in Fig. 3. Such a network can be trained in a supervised or unsupervised manner using real world data. A single layer of FISTA contains multiple parameters like $\boldsymbol{W}, \boldsymbol{\Phi}, \lambda$ which can be learned from input data. See Fig. 3. Such a deep network which mimics several iterations of FISTA is termed 'Learned FISTA' (LFISTA). In

Algorithm 1: FISTA

Input: $\boldsymbol{x}, \boldsymbol{\Phi}, \boldsymbol{W}, P, \lambda$

Output: $\hat{\boldsymbol{\Theta}}$

1 $\hat{\boldsymbol{\Theta}}_{\boldsymbol{0}} := \boldsymbol{0}, t := 1, L := \text{Maxeig}(\boldsymbol{W}^T\boldsymbol{W})$; *Init.*

2 **for** $k := 0, 1, ..., P-1$ **do**

3 $\quad \boldsymbol{y_{k+1}} = \hat{\boldsymbol{\Theta}}_{\boldsymbol{k}} - \frac{1}{L}\boldsymbol{W}^T\boldsymbol{\Phi}^T(\boldsymbol{\Phi W}\hat{\boldsymbol{\Theta}}_{\boldsymbol{k}} - \boldsymbol{x})$;

4 $\quad \boldsymbol{z_{k+1}} = \text{soft}_{\lambda,L}(\boldsymbol{y_{k+1}})$; *Soft thresh.*

5 $\quad t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$

6 $\quad \hat{\boldsymbol{\Theta}}_{\boldsymbol{k+1}} = \boldsymbol{z_{k+1}} + \frac{t_k - 1}{t_{k+1}}(\boldsymbol{z_{k+1}} - \boldsymbol{z_k})$; *Momentum*

the FISTA algorithm given in Alg. 1, the basis matrix $\boldsymbol{\Psi}$, which is used to create $\boldsymbol{W}$ in Eqn. 13, can also be learned via backpropagation. In [11], it is shown that the learning of $\boldsymbol{\Psi}$ improves the convergence speed significantly. Given the structure of $\boldsymbol{W}$ in Eqn. 13, it is sufficient and also efficient to learn only $\boldsymbol{\Psi}$. Essentially, $\boldsymbol{\Psi}$ is learned to minimize the unsupervised loss function $\mathcal{L}(\boldsymbol{\Psi}^L, \lambda)$ over N training examples $\{(\widetilde{I_{j,1}}, \widetilde{I_{j,2}})\}_{j=1}^{N_T}$ as shown below:

$$\mathcal{L}(\boldsymbol{\Psi}^L, \lambda) = \frac{1}{N_T}\sum_{j=1}^{N_T}\|\boldsymbol{x} - \boldsymbol{\Phi W}\hat{\boldsymbol{\Theta}}_{\boldsymbol{j}}\|_2^2. \quad (14)$$

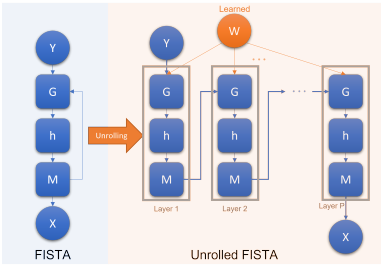

Figure 3. **Left:** Original FISTA algorithm. The three steps of FISTA iteration are shown as: 1) G: Gradient step, 2) h: Thresholding step, 3) M: Momentum step. **Right:** Unrolled version of the FISTA algorithm, where a fixed number of iterations of FISTA are unrolled. $\boldsymbol{W}$ can be partly learned from the data - see Sec. 3 and Eqn. 13.

Here $\boldsymbol{x}, \boldsymbol{\Phi}$ is constructed using $\widetilde{I_{j,1}}, \widetilde{I_{j,2}}$ respectively, $\boldsymbol{W}$ is constructed using the original DCT basis matrix, but the coefficient vector $\hat{\boldsymbol{\Theta}}_{\boldsymbol{j}}$ of the jth image is estimated using FISTA with the *learned* matrix which we denote as $\boldsymbol{\Psi}^L$. Overall, the primary advantage of such an unrolling approach is that we retain *interpretability* while improving the computational cost of the algorithm via the neural network. FISTA would take around 10 hours to process a typical $320 \times 480 \times 100$ video with $W = 5$ in our dataset, whereas LFISTA brought down this cost to $\sim$5 mins. We wish to emphasize that the matrices $\boldsymbol{\Psi}, \boldsymbol{\Psi}^L$ used in LFISTA in Eqns. 13, 14 are of size $n \times n$, i.e. full orthonormal matrices (even though the $\boldsymbol{\Psi}$ matrix in Eqn. 7 in of size $n \times K, K < n$). An additional advantage of LFISTA over FISTA for our specific problem is that the value of K need not be tuned in the former. Moreover, λ is set to be a learnable parameter to minimize the loss function in Eqn. 14. Here, the parameters $\hat{\boldsymbol{\Theta}}_{\boldsymbol{j}}$ are estimated using FISTA which uses $\boldsymbol{\Psi}^L$ and λ to estimate the best coefficients. During training, the forward pass executes the usual FISTA algorithm which uses $\{\boldsymbol{\Psi}^L, \lambda\}$ and in the backward pass, $\boldsymbol{\Psi}^L$ and λ are adjusted to minimize the loss function.

4. Experiments

4.1. Datasets

There are no standardized datasets available for the problem of restoration of vintage video sequences. Hence we performed our experiments on the following three datasets: (Refer to [4] for details such as video size, fps and actual URLs for each dataset.) (1) Dataset *D1*, 10 videos: This consists of old movies used by previous papers on vintage video restoration. This includes four video sequences from the film *Le mort qui tue*(1913/14) used in [27], a video sequence from [2] used in [30], and five video sequences shared by the authors of [31] on [32]. (2) Dataset *D2*, 4 videos: This consists of high frame rate videos (150-190 fps) with flicker from [25]. (3) Dataset *D3*: This is a collection of 41 videos from Youtube, containing both blotch and flicker artifacts. Many of these videos were downloaded from the *British Film Institute* Youtube channel [1]. Most of these were around 30 secs. in duration. However this collection contained one Youtube video of around 3.5 mins.

4.2. Implementation Details

The complete framwork of the proposed LFISTA-based method is presented in Fig. 1. The number of layers in the proposed LFISTA model was set to $P = 50$. The network was trained on patches of size 32×32, as opposed to entire frames in order to reduce the number of parameters being learned. The training patches were extracted from randomly selected pairs of frames from 52 good quality video clips collected from Youtube. In every pair, both the frames were located within a temporal neighborhood of radius $W = 5$ from each other. The video clips were synthetically degraded by blotch artifacts with average diameter of 10 pixels. We emphasize that *none* of these videos were part of the datasets *D1*-*D3* on which our algorithm was tested. For training LFISTA, we set $p = 2$. Our model was trained for 35 epochs with the Adam optimizer. We noticed in our experiments that the value of the term $\boldsymbol{\Psi\theta_a}$ from Eqn. 12 was usually too small to be relevant and hence could be ignored. In fact, the best results were obtained without using $\boldsymbol{\Psi\theta_a}$ while training LFISTA. In our experiments, we used $W = 5$ for *D1*,*D3* and $W = 10$ for *D2* (an ablation study is presented in [4]). The value of τ_r was set to 0.05. This value was selected from residual vector fields (normalized to contain values from 0 to 1) obtained from three training videos, so as to best distinguish between blotch regions (R_b) and regions of optical flow errors/occlusions (R_o). The value of λ in Eqn. 13 was learned within the LFISTA framework. For video restoration of test sequences, the 32×32 patches from each frame were chosen in an overlapping fashion with a stride of 16 pixels, with averaging in overlapping pixels to remove any patch seam artifacts while producing the final restored set of frames. Most of our test videos were grayscale. In case of color videos, we converted the RGB values to the HSV color space and performed restoration only on the V channel, leaving H, S intact.

4.3. Comparison Baselines and Metrics:

We compared the performance of our LFISTA-based method to (i) RTN, the state of the art RTN-based approach in [31], (ii) DeepRemaster, a recent method from [13] which performs only blotch removal and no deflickering, (iii) a commerical software called NeatVideo [3], (iv) a combination of the 'OldPhoto' method from [33] to independently remove blotches in individual video frames followed by the temporal stabilization algorithm from [16] to remove flicker, referred to as OldPhoto+TS, and (v) a very

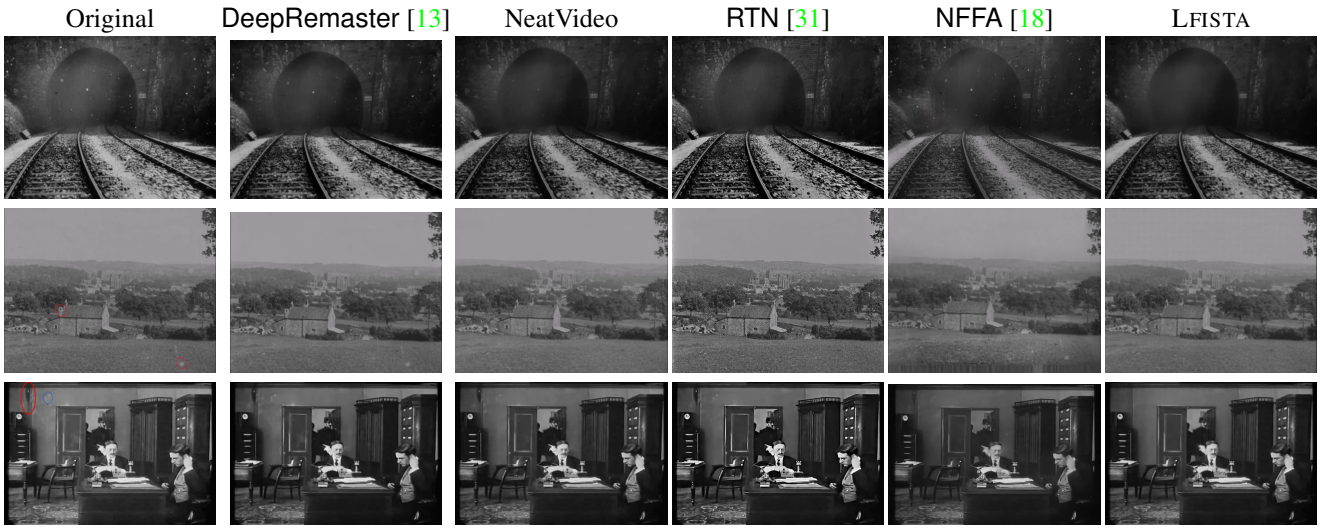

Figure 4. Video restoration performance: Blotches marked out by red/blue borders. Zoom into the pdf for better viewing. Notice the superior blotch removal performance of our method. For deflickering results in video form, see supplemental material at [4].

recent blind deflickering method [18] based on the concept of neural network based filtering with a flawed atlas, referred to as NFFA. For RTN, DeepRemaster and NFFA, we used the code as well as the neural network models provided by the respective authors without changing any parameter settings. NeatVideo requires manual selection of a blotch size parameter, and contrast thresholds for global and local flicker, and blotches. This can be tedious and error-prone, especially in case of artifacts with intensity/size that varies within a frame or across frames in a video. We do not report results using FISTA because the computational cost was more than 100 times that of LFISTA. We also report results on a version of LFISTA with frame-wise unsharp masking, which we refer to as LFISTA-SHARP.

As no ground truth is available for reference, all competing methods were evaluated based on the following intuitive optical flow based no-reference video quality metrics: OFE1, OFEP, OFE2, which are defined specifically for the task of calibrating the performance of video restoration. OFE1 is computed using the following formula for a video I: $\texttt{OFE1}(I) = \sum_{i=2}^{T} \|\boldsymbol{M_{i,i-1}} \circ (I_i - I_{i-1}^{corr})\|_2^2 / [(T-1)\|\boldsymbol{M_{i,i-1}}\|_1]$, where I_{i-1}^{corr} denotes the frame I_{i-1} obtained *after* warping it with the optical flow $u_{i-1,i}$ from I_{i-1} to I_i, and $\boldsymbol{M_{i,i-1}}$ denotes the binary mask as described in Sec. 2.3. OFEP (OFE Perceptual) is computed as: $\texttt{OFEP}(I) = \sum_{i=2}^{T} \|\boldsymbol{M_{i,i-1}} \circ (\phi(I_i) - \phi(I_{i-1}^{corr}))\|_2^2 / [(T-1)\|\boldsymbol{M_{i,i-1}}\|_1]$, where $\phi(.)$ denotes features obtained from a pre-trained VGG classification network [28]. These features are known to correlate very well with human perception [36]. OFE2 is defined similarly as OFE1, but using a neighborhood of radius $W = 5$ around every Wth frame of the video. The formula is $\texttt{OFE2}(I) = \sum_{i=W+1, i\%W=1}^{T} \sum_{j \in \mathcal{T}(i), j \neq i} \|\boldsymbol{M_{i,j}} \circ (I_j - I_i^{corr})\|_2^2 / [(T/W - 1)(2W-1)\|\boldsymbol{M_{i,j}}\|_1]$. The basic motivation behind OFE1, OFE2, OFEP is that they measure the difference between consecutive frames after aligning them, ignoring regions of occlusion. Clearly, these OFE measures would be lower for deflickered and deblotched videos, as compared to the original ones. Metrics very similar to OFE1, OFE2, OFEP have been used in [16] (See Eqns. 1,2,3) and [18] as well. In addition, we use the following measures for evaluation: a video smoothness measure SM from [10, Eqn. 10], and the popular no-reference image quality measures BRISQUE [21] and NIQE [22] which have been widely used in image restoration tasks. We note that the latter two are are generic *single-frame* measures, and therefore not suitable for evaluating deflickering which has a *temporal* aspect.

4.4. Results

Numerical results for all competing methods are shown in Table 1. Some pictorial results for blotch removal and deflickering are presented in Fig. 4 and Fig. 1 of [4] respectively. From these, we see that our method LFISTA outperforms the state of the art techniques RTN [31] and DeepRemaster [13] in terms of temporal measures such as SM, OFE1, OFE2 and OFEP. It also outperforms NFFA [18] in most cases despite requiring no explicit training. To appreciate the quality of deflickering, which is a temporal phenomenon, the restored videos with all methods can be found in the suppl. mat. [4]. We observed that RTN performs high quality restoration with sharper images than with LFISTA, but tends to create high-frequency visual artifacts since its neural network is also trained to enhance contrast. This

Dataset	Method	OFE1↓	OFE2↓	OFEP↓	BRISQUE [21]↓	NIQE [22]↓	SM [10]↑
D1	Orig.	0.0098	0.01367	0.0477	45.87	20.79	1.0
	DeepRm [13]	0.0092	0.0134	0.036	49.18	21.8	0.9725
	NeatVideo	0.0066	0.01	0.0237	51.49	21.9	1.143
	RTN [31]	0.0097	0.013	0.0435	**28.45**	21.79	0.79
	Old.+TS [16]	0.0088	0.012	0.074	47.52	**17.28**	1.086
	NFFA [18]	0.0073	0.0107	**0.0225**	47.51	17.98	1.281
	LFISTA	**0.006**	**0.009**	0.025	49.01	21.97	**1.256**
	LFISTA-SHARP	0.007	0.01	0.036	46.21	18.05	1.0388
D2	Orig.	0.014	0.012	0.023	**19.4**	21.79	1.0
	DeepRm [13]	0.0113	0.0114	0.03	31.27	21.8	1.037
	NeatVideo	0.0072	0.0066	0.0162	24.37	21.87	1.74
	RTN [31]	0.00847	0.01	0.021	27.96	20.36	1.1589
	Old.+TS [16]	0.012	0.011	0.027	25.32	**15.86**	0.984
	NFFA [18]	**0.0036**	0.0046	0.0096	35.31	20.08	2.008
	LFISTA	0.0041	**0.004466**	0.0059	31.25	21.9	**2.899**
	LFISTA-SHARP	**0.004**	0.0047	**0.0056**	30.13	18.32	2.305
D3	Orig.	0.008	0.0119	0.0393	45.93	21.55	1.0
	DeepRm [13]	0.0085	0.0123	0.0336	45.28	21.75	0.965
	NeatVideo	**0.0056**	0.0087	0.0234	48.81	21.8	1.2
	RTN [31]	0.0093	0.0117	0.0446	**28.78**	20.97	0.77
	Old.+TS [16]	0.0089	0.012	0.053	34.71	17.02	0.92
	NFFA [18]	0.0062	**0.00825**	0.04295	43.18	17.39	**1.407**
	LFISTA	**0.00569**	**0.0082**	**0.021**	46.22	21.57	1.274
	LFISTA-SHARP	0.0067	0.009	0.027	42.97	**16.83**	0.966

Table 1. Average values of various evaluation measures computed on datasets *D1*-*D3*, for all competing methods. In all videos, intensity range was $[0, 1]$. See [4] for restored videos and individual scores.

tends to amplify noise and create some strong shock edges, which change position from frame to frame giving an impression of local flicker. These effects are particularly visible in *D2* which contains high speed videos, and in some videos in *D3*. LFISTA does not exhibit these artifacts. It is possible that RTN would not have exhibited these artifacts if the contrast enhancement part were to be excluded, but we do not have access to their code or network model which excludes this part, and hence no further comparison can be performed. NFFA removes flicker well, but its outputs have lower contrast than ours, and it does not remove blotches. DeepRemaster does not remove flicker (as it is not designed to remove flicker) though it generally removes blotches quite well. NeatVideo removes flicker and blotches quite well, but requires manual parameter selection. We again emphasize that the datasets *D1* and *D2* were created by other papers in the literature and shared by the respective authors [25, 31]. For LFISTA, we observed that inclusion of the mask $\boldsymbol{M_{i,j}}$ was very important, and excluding it produced visibly less optimal results. In addition to these results, we also report comparisons on synthetically degraded videos in Sec. 5 of [4]. Moreover, in Sec. 6 of [4], we also compare results of LFISTA and NFFA [18] on a dataset of 60 video clips from old movies released in [18].

5. Conclusion

We have presented a simple, model-based, interpretable method for video restoration (deflickering and deblotching) based on model-based learning. We have not attempted to perform colorization as it is not necessary for restoration, and a separate colorization module can be easily added. Our method is computationally efficient and outperforms state of the art techniques on publicly shared as well as our own datasets in terms of numerical scores and visual quality. One limitation of our method is that the computation, though efficient ($\sim$ 5 mins. for a $320 \times 480 \times 100$ video with $W = 5$ on a 2.8 GHz AMD CPU with A6000 GPUs), is not real time. Maybe, better unrolling models could be incorporated [5], to improve computational cost. Moreover, the performance of our algorithm on blotches that lie across motion boundaries, could be improved. Finally, vintage videos also suffer from spatial artifacts such as non-rigid jitters [9, 24], along with flicker and blotches. A combined engine to jointly remove all three types of artifacts is an interesting avenue for future work.

References

[1] British film institute youtube channel. https://www.youtube.com/channel/UC9dGqMAPJRpA7M0oSdgtQgg. 6

[2] Institut national de l'audiovisuel. https://www.ina.fr/. 6

[3] Neat video: best noise reduction for digital video. https://www.neatvideo.com/. 6

[4] Supplemental material. Uploaded on conference portal. 6, 7, 8

[5] P. Ablin, T. Moreau, M. Massias, and A. Gramfort. Learning step sizes for unfolded sparse coding. In *NeurIPS*, 2019. 8

[6] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009. 4, 5

[7] N. Bonneel, J. Tompkin, K. Sunkavalli, D. Sun, S. Paris, and H. Pfister. Blind video temporal consistency. *ACM Transactions on Graphics (TOG)*, 34(6):1–9, 2015. 2

[8] J. Delon and A. Desolneux. Stabilization of flicker-like effects in image sequences through local contrast correction. *SIAM Journal on Imaging Sciences*, 3(4):703–734, 2010. 1

[9] G. Dong, A. R. Patrone, O. Scherzer, and O. Oktem. Infinite dimensional optimization models and PDEs for dejittering. In *Scale Space and Variational Methods in Computer Vision*, pages 678–689, 2015. 8

[10] G. Eilertsen, R. Mantiuk, and J. Unger. Single-frame regularization for temporally stable cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11176–11185, 2019. 7, 8

[11] K. Gregor and Y. LeCun. Learning fast approximations of sparse coding. In *ICML*, pages 399–406, 2010. 5

[12] T. Hastie, R. Tibshirani, and M. Wainwright. *Statistical Learning with Sparsity: The LASSO and Generalizations*. CRC Press, 2015. 3

[13] S. Iizuka and E. Simo-Serra. Deepremaster: Temporal source-reference attention networks for comprehensive video enhancement. *ACM Trans. Graph.*, 38(6), 2019. 2, 5, 6, 7, 8

[14] A. Kanj, H. Talbot, and R.-R. Luparello. Flicker removal and superpixel-based motion tracking for high speed videos. In *ICIP*, 2017. 1, 2

[15] A. Kokaram. *Motion Picture Restoration*. Springer Verlag, 1998. 1, 5

[16] W.-S. Lai, J.-B. Huang, O. Wang, E. Shechtman, E. Yumer, and M.-H. Yang. Learning blind video temporal consistency. In *ECCV*, 2018. 2, 6, 7, 8

[17] M. Lang, O. Wang, T. Aydin, A. Smolic, and M. Gross. Practical temporal consistency for image-based graphics applications. *ACM Transactions on Graphics (ToG)*, 31(4):1–8, 2012. 2

[18] C. Lei, X. Ren, Z. Zhang, and Q. Chen. Blind video deflickering by neural filtering with a flawed atlas. In *CVPR*, 2023. 2, 7, 8

[19] C. Lei, Y. Xing, and Q. Chen. Blind video temporal consistency via deep video prior. *Advances in Neural Information Processing Systems*, 33:1083–1093, 2020. 2

[20] C. Lei, Y. Xing, H. Ouyang, and Q. Chen. Deep video prior for video consistency and propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):356–371, 2022. 2

[21] A. Mittal, A.K. Moorthy, and A. C. Bovik. Blind/referenceless image spatial quality evaluator. In *Asilomar conference on signals, systems and computers*, 2011. 7, 8

[22] A. Mittal, R. Soundararajan, and A. C. Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal Processing Letters*, 20(3), 2012. 7, 8

[23] N. H. Nguyen and T. D. Tran. Robust lasso with missing and grossly corrupted observations. *IEEE Trans. Inf. Theory*, 59(4), 2013. 3

[24] M. Nikolova. One-iteration dejittering of digital video images. *Journal of Visual Communication and Image Representation*, 20(4):254–274, 2009. 8

[25] J. Nowisz, M. Kopania, and A. Przelaskowski. Realtime flicker removal for fast video streaming and detection of moving objects. *Multimedia Tools and Applications*, 80, 2021. 1, 6, 8

[26] F. Pitie. Removing flicker from old movies. Master's Thesis, University of Nice-Sophia Antipolis, https://tinyurl.com/3p8d5eej, 2002. 1, 2, 3, 5

[27] P. Schallauer, A. Pinz, and W. Haas. Automatic restoration algorithms for 35mm film. *J. Computer Vision Res*, 1(3):59–85, 1999. 6

[28] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 7

[29] Z. Teed and J. Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, pages 402–419. Springer, 2020. 2, 4

[30] P. van Roosmalen, R. L. Lagendijk, and J. Biemond. Correction of intensity flicker in old film sequences. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(7):1013–1019, 1999. 1, 5, 6

[31] Z. Wan, B. Zhang, D. Chen, and J. Liao. Bringing old films back to life. In *IEEE CVPR*, 2022. 2, 5, 6, 7, 8

[32] Z. Wan, B. Zhang, D. Chen, and J. Liao. Bringing old films back to life. http://raywzy.com/Old_Film/, 2022. 6

[33] Z. Wan, B. Zhang, D. Chen, P. Zhang, D. Chen, J. Liao, and F. Wen. Bringing old photos back to life. In *CVPR*, 2020. 6

[34] C.-H. Yao, C.-Y. Chang, and S.-Y. Chien. Occlusion-aware video temporal consistency. In *ACMMM*, pages 777–785, 2017. 2

[35] J. Zhang, L. Chen, P. Boufounos, and Y. Gu. On the theoretical analysis of cross validation in compressive sensing. In *ICASSP*, pages 3370–3374, 2014. 3

[36] R. Zhang, P. Isola, A.A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 7