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:

$$\widetilde{I_1}^{corr} = \text{diag}(\widetilde{I_2})\boldsymbol{\Psi\theta_m} + \boldsymbol{\Psi\theta_a} + \boldsymbol{\eta}. \quad (2)$$

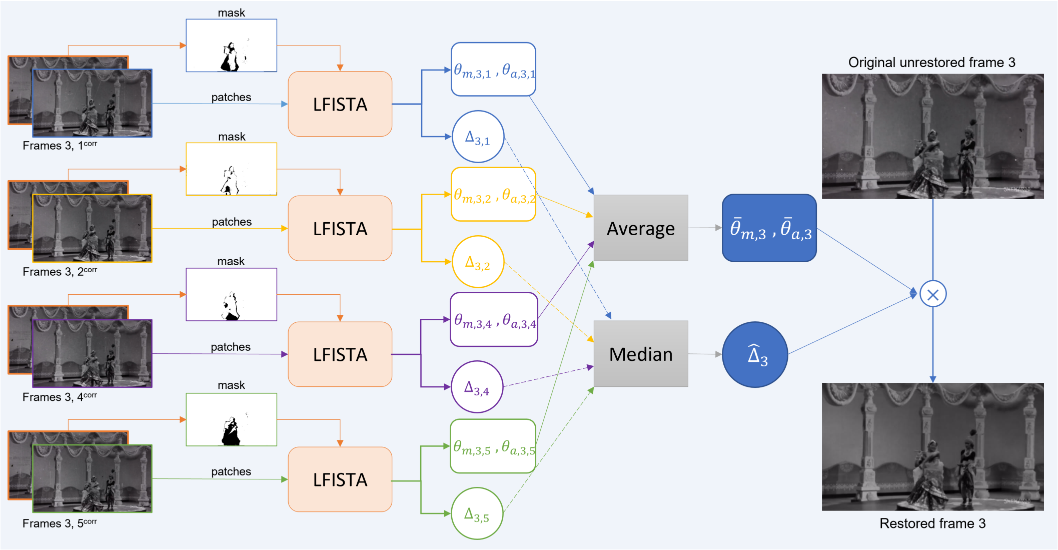

Figure 1. Overview of the LFISTA Algorithm: restoration of frame 3 using its neighboring frames $\mathcal{T}(3) := \{1, 2, 4, 5\}$. For more details including definitions of flicker coefficients $\bar{\boldsymbol{\theta}}_{\boldsymbol{m,3}}, \bar{\boldsymbol{\theta}}_{\boldsymbol{a,3}}, \boldsymbol{\theta}_{\boldsymbol{m,i,j}},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

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

where $\boldsymbol{\Delta_{2,1}} := \text{diag}(\boldsymbol{\Delta_2})\boldsymbol{\Psi}\boldsymbol{\theta_m} - \boldsymbol{\Delta_1^{corr}}$. 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:

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

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):

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

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 $p \geq 1$:

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

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:

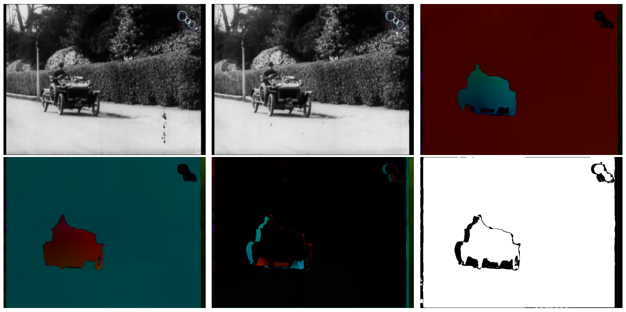

Figure 2. Mask creation. Left to right, top to bottom: Frames I_1, I_2; flow fields u_{12}, u_{21}; Magnitude of $\vec{r}$ and Occlusion Mask $\boldsymbol{M_{12}}$. (Note: Flow images are brightened for better visualization). Notice how the blotch region on the road is not masked out.

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

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 $p \geq 1$:

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

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:

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

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 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:

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

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      $\boldsymbol{y_{k+1}} = \hat{\boldsymbol{\Theta}}_{\boldsymbol{k}} - \frac{1}{L}\boldsymbol{W}^T\boldsymbol{\Phi}^T(\boldsymbol{\Phi}\boldsymbol{W}\hat{\boldsymbol{\Theta}}_{\boldsymbol{k}} - \boldsymbol{x})$;
4      $\boldsymbol{z_{k+1}} = \text{soft}_{\lambda,L}(\boldsymbol{y_{k+1}})$; Soft thresh.
5      $t_{k+1} = \frac{1+\sqrt{1+4t_k^2}}{2}$
6      $\hat{\boldsymbol{\Theta}}_{\boldsymbol{k+1}} = \boldsymbol{z_{k+1}} + \frac{t_k-1}{t_{k+1}}(\boldsymbol{z_{k+1}} - \boldsymbol{z_k})$; Momentum
```

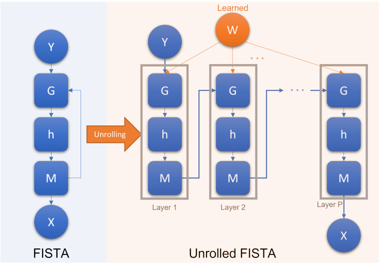

Figure 3. **Left:** Original FISTA algorithm. The three steps of FISTA iteration are shown as: 1) G: Gradient step, 2) h: Thresholding step, 3) M: Momentum step. **Right:** Unrolled version of the FISTA algorithm, where a fixed number of iterations of FISTA are unrolled. $\boldsymbol{W}$ can be partly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

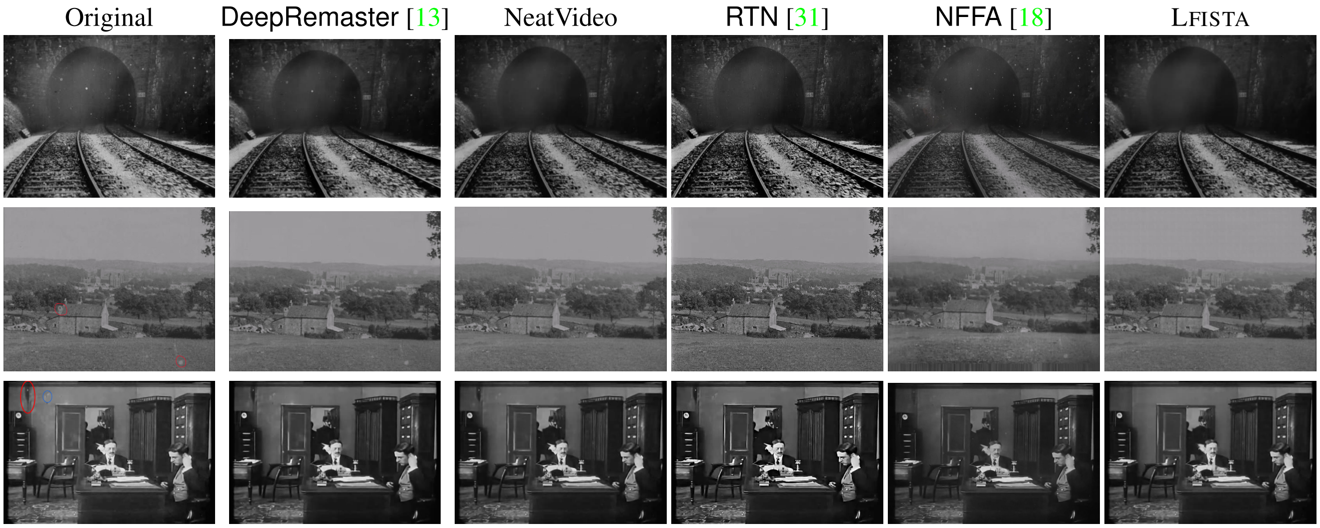

Figure 4. Video restoration performance: Blotches marked out by red/blue borders. Zoom into the pdf for better viewing. Notice the superior blotch removal performance of our method. For deflickering results in video form, see supplemental material at [4].

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

Supplemental Material: 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,satish.mulleti@gmail.com, ajitvr@cse.iitb.ac.in

1. Contents

This document contains supplemental material for the main paper. The contents include:

1. Results on the guitar video referred to in the Experiments section of the main paper - see Sec. 2.
2. Some details regarding the procedure to implement LFISTA are included in Sec. 3.
3. An ablation study for LFISTA by varying the window-size W of the temporal smoothing process is presented in Sec. 4.
4. Quantitative results on synthetically generated videos with no-reference and full-reference quality measures are included in Sec. 5.
5. Quantitative results from Sec. 4 of the main paper are included in Sec. 7 for *each* video separately.
6. Quantitative comparisons between LFISTA and NFFA are presented in Sec. 6.
7. Qualitative Results (not in this document, but the accompanying video *'revised_supplemental_video_542.mp4'* available at the link https://drive.google.com/file/d/1-a0akBb6YVCqZGtCfvTxKB78fCyguA39/view?usp=sharing): This video shows restoration results on two videos each from the datasets *D1*, *D2* and four videos (including one color video) from *D3*. The datasets *D1*, *D2*, *D3* are described in Sec. 4.1 of the main paper. In addition, we also show a result on the (self-acquired) guitar video. In the video file, results obtained using our LFISTA method are shown alongside the original (to be restored) video, as well as results obtained using RTN [7], DeepRemaster [2], OldPhoto [8] + TS [3] and NeatVideo. The superior performance of the proposed technique is very clear from the video results - from the point of view of flicker as well as blotch removal. Note that for all competing methods, we have used the code and parameter settings provided by the respective authors.

2. Results on Guitar Video

Dataset	Method	OFE1	OFE2	OFEP
Guitar	Original	0.021452	0.02017	0.067409
	OldPhoto + TS	0.02078	0.01998	0.07714
	DeepRemaster [2]	0.019439	0.018503	0.04392
	RTN [7]	0.007589	0.008533	0.019989
	NeatVideo	0.016328	0.013719	0.054162
	LFISTA	**0.003602**	**0.004247**	**0.008376**

Table 1. Values of OFE1, OFE2, OFEP metrics (lower is better), computed on the Guitar video from Fig. 1, for various competing methods. In all videos, intensity range was $[0, 1]$.

This is a result on the guitar video in Fig. 1. The video is gathered at 240 fps from a mobile phone and exhibits flicker artifacts. The numerical scores for different methods for this video are presented in Table 1. Please see the accompanying video file *'revised_supplemental_video_542.mp4'* for the restoration results in video format.

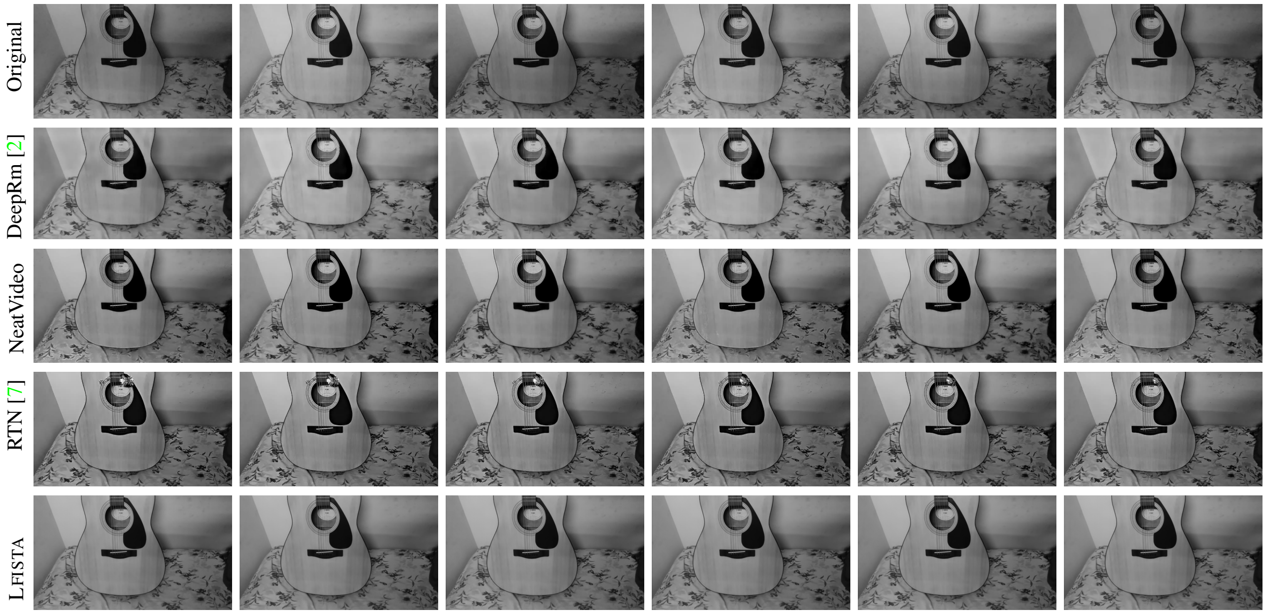

Figure 1. Comparison of deflickering performance of different methods on a high speed (240 fps) video gathered from a mobile phone. See accompanying video results, and numerical scores in Table 1.

	OFE1↓	OFE2↓	OFEP↓	BRISQUE↓	NIQE↓
Orig.	0.01	0.01198	0.044	43.54	21.29
$W = 1$	0.0102	0.0115	0.038	44.89	18.8877
$W = 3$	0.0072	0.0091	0.0237	45.088	19.033
$W = 5$	0.0062	0.008	0.022	45.294	19.037
$W = 7$	0.005753	0.0075	0.021916	45.4	18.96
$W = 9$	0.0055	0.0072	0.022	45.46	18.94

Table 2. Ablation study on the effect of temporal smoothing radius W on the LFISTA technique.

3. Details of the video used to train the network for LFISTA

Refer to Sec. 4.2 of the main paper. For implementing LFISTA, the network was trained (in an unsupervised manner) 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 extracted from the Youtube video at https://www.youtube.com/watch?v=LXb3EKWsInQ (converted to grayscale). 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 video clips were part of the datasets *D1-D3* on which our algorithm was tested.

4. Ablation Study

We present an ablation study to determine the effect of the temporal smoothing radius W on our LFISTA technique. The results of this study are presented in Table 2. All experiments were carried out on 10 videos from the *D3* dataset. We generally observed that $W = 5$ (i.e. 5 frames on either side of the reference frame) yielded the best results visually and hence used that value in our experiments. Larger values of W lowered the metric values as seen in the table, but caused slight blurring of moving object boundaries. This is due to errors in optical flow computed between distant frames (due to increase in W). On the other hand, lower values of W did not sufficiently erase flicker or blotches.

Method	OFE1↓	OFE2↓	OFEP↓	BRISQUE [5]↓	NIQE [6]↓	SM [1]↑	SSIM↑	PSNR↑
Original	0.0144	0.015174	0.142124	26.61	18.59	1	0.924	25.56
DeepRm [2]	0.011	0.0135	0.0899	31.81	18.84	1.23	0.92	25.86
NeatVideo	0.006	0.008	0.0563	35.57	17.576	1.58	0.9086	**27.03976**
RTN [7]	0.009	0.011	0.066	**19.96**	17.92	1.12	0.8529	24.386
NFFA [4]	**0.00455**	**0.006456**	0.115	35.32	**15.64**	1.4249	0.512	14.88
LFISTA	0.0072	0.00844	**0.0477**	33.43	17.477	**1.69**	**0.9258**	26.524
LFISTA-SHARP	0.0077	0.0094	0.0505	30.937	16.84	1.35	0.903	24.775

Table 3. Comparative results on synthetic data (different evaluation measures averaged across 10 videos).

Method	OFE1↓	OFE2↓	OFEP↓	BRISQUE↓	NIQE↓	SM↑
Original	0.009882	0.012696	0.071894	**42.9**	19.37	1.0
NFFA [4]	0.006918	0.0101	0.064	46.56	19.07	**1.413925**
LFISTA	**0.005654**	**0.007826**	**0.034621**	51.86	**17.79**	1.366949

Table 4. Quantitative comparison (values of different measures averaged across 60 videos) between LFISTA and NFFA.

5. Results on Synthetic Data

We performed a restoration study on a set of 10 clean and temporally consistent videos of size 640×360 gathered from https://www.videvo.net/. These videos were degraded synthetically using (1) blotch templates as mentioned in the github repository associated with [8], and (2) flicker induced by multiplying each frame with a number generated from $\text{Uniform}(0.8, 1.2)$. The degraded videos were restored by various competing methods and no-reference as well as full-reference quality measures are included in Tab. 3. All measures are averaged over all 10 videos. The full-reference measures are SSIM (structured similarity index) and PSNR (peak signal to noise ratio). We observed that NFFA did not remove blotches and hence produced lower SSIM and PSNR values as compared to our method. RTN again yielded high frequency artifacts. Results on a sample synthetic video are included in the accompanying video file *'revised_supplemental_video_542.mp4'*.

6. Comparisons with NFFA

We present a quantitative comparison between LFISTA and NFFA [4] on a subset of the dataset from [4]. The subset contains 60 video clips from old movies with flicker artifacts. The results are presented in Table 4. From these results, we see that the performance of LFISTA and NFFA are comparable on various metrics, even on a dataset released by their paper.

7. Quantitative Results

This section contains Table 5 which shows numerical results for each video from the datasets *D1*, *D2*, *D3* described in Sec. 4.1 of the main paper. The actual weblink for every video is also included in the table (wherever applicable). The comparison metrics OFE1, OFE2 and OFEP are defined in Sec. 4 of the main paper.

Table 5. Detailed quantitative comparison of our method on various video sequences to existing methods: RTN [7], DeepRemaster [2], OldPhoto [8] + TS [3] and NeatVideo. For each video sequence, a weblink is included.

Dataset	Video	Method	OFE1	OFE2	OFEP	BRISQUE	NIQE	SM	Link
D1	Example 1	Original	0.01246	0.01577	0.07096	40.65	24.39	1.00000	link
		OldPhoto + TS	0.01124	0.01512	0.14651	25.11	15.16	0.85119	link
		Deepremaster	0.01148	0.01630	0.06289	42.74	23.20	0.90080	link
		NeatVideo	0.00765	0.01101	0.05364	46.73	23.06	1.13249	link
		RTN	0.00889	0.01347	0.06570	26.19	22.20	0.89253	link
		NFFA	0.01164	0.01649	0.01556	49.48	16.01	0.98754	link
		LFISTA	0.00886	0.01227	0.06328	45.49	23.50	1.08302	link
		LFISTA-SHARP	0.00917	0.01253	0.08898	41.46	15.61	0.97680	link

De l'importance des chars dans la victoire de 1918	Original	0.01287	0.01860	0.06435	45.62	22.04	1.00000	link
	OldPhoto + TS	0.01017	0.02123	0.09963	51.89	19.27	1.04633	link
	Deepremaster	0.01158	0.01572	0.04588	45.86	22.41	0.94057	link
	NeatVideo	0.01192	0.01155	0.02944	48.23	22.32	1.09836	link
	RTN	0.01319	0.01680	0.07725	25.03	20.03	0.61454	link
	NFFA	0.00621	0.00897	0.06121	50.64	21.39	1.97109	link
	LFISTA	0.00856	0.01394	0.02839	46.61	22.09	1.09591	link
	LFISTA-SHARP	0.01015	0.01548	0.05036	45.50	19.56	0.96152	link
Example 2	Original	0.00931	0.01222	0.06804	42.18	23.43	1.00000	link
	OldPhoto + TS	0.01014	0.01253	0.11534	39.97	15.42	0.72417	link
	Deepremaster	0.00893	0.01247	0.04466	48.73	20.33	0.93340	link
	NeatVideo	0.00532	0.00805	0.02696	48.76	20.67	1.02866	link
	RTN	0.00833	0.01052	0.04645	36.19	20.53	0.84309	link
	NFFA	0.00884	0.01639	0.02880	51.00	17.16	0.50311	link
	LFISTA	0.00558	0.00741	0.02884	51.52	20.69	1.02429	link
	LFISTA-SHARP	0.00626	0.00844	0.04124	45.77	17.30	0.91943	link
Example 3	Original	0.00757	0.01185	0.01939	55.36	17.71	1.00000	link
	OldPhoto + TS	0.00643	0.00908	0.03009	51.78	20.06	1.39112	link
	Deepremaster	0.00757	0.01217	0.01581	58.89	23.20	0.98448	link
	NeatVideo	0.00529	0.00856	0.01232	58.75	23.26	1.36637	link
	RTN	0.01034	0.01343	0.02846	36.27	22.75	0.90917	link
	NFFA	0.00478	0.00638	0.01193	54.72	18.88	2.07086	link
	LFISTA	0.00515	0.00779	0.01115	56.37	23.70	1.52130	link
	LFISTA-SHARP	0.00625	0.00903	0.01483	55.30	17.74	1.27480	link
Le Mort qui tue (film, 1913) Sequence 1	Original	0.00723	0.01104	0.02058	47.06	18.71	1.00000	link
	OldPhoto + TS	0.00624	0.00848	0.02773	54.84	16.84	1.16491	link
	Deepremaster	0.00750	0.01125	0.01916	49.05	20.12	0.98674	link
	NeatVideo	0.00540	0.00821	0.01571	48.21	20.28	1.15251	link
	RTN	0.00893	0.01144	0.02731	26.93	20.20	0.77193	link
	NFFA	0.00576	0.00752	0.02191	47.30	17.63	1.28934	link
	LFISTA	0.00497	0.00685	0.01228	47.44	20.02	1.39167	link
	LFISTA-SHARP	0.00593	0.00802	0.01440	43.92	18.19	1.09858	link
Le Mort qui tue (film, 1913) Sequence 2	Original	0.00495	0.00734	0.01795	47.78	20.31	1.00000	link
	OldPhoto + TS	0.00500	0.00660	0.02015	59.13	15.97	0.99781	link
	Deepremaster	0.00550	0.00776	0.01688	49.84	20.40	0.97629	link
	NeatVideo	0.00431	0.00623	0.01456	52.20	20.43	1.12097	link
	RTN	0.00699	0.00900	0.01952	33.47	21.64	0.79640	link
	NFFA	0.00503	0.00651	0.02292	51.90	20.19	1.57615	link
	LFISTA	0.00419	0.00573	0.00900	49.29	20.34	1.39048	link
	LFISTA-SHARP	0.00503	0.00681	0.01026	46.10	18.58	1.08330	link
Le Mort qui tue (film, 1913) Sequence 3	Original	0.01003	0.01413	0.02122	80.75	19.33	1.00000	link
	OldPhoto + TS	0.00917	0.01235	0.03227	68.57	16.91	1.18858	link
	Deepremaster	0.01024	0.01426	0.01875	78.47	20.84	1.02114	link
	NeatVideo	0.00880	0.01285	0.01683	82.19	21.34	1.06073	link
	RTN	0.01149	0.01419	0.03016	57.65	21.39	0.72196	link
	NFFA	0.00802	0.01037	0.02541	78.65	21.05	0.97118	link
	LFISTA	0.00710	0.01050	0.01272	74.51	21.35	1.28823	link
	LFISTA-SHARP	0.00853	0.01179	0.01466	74.29	18.39	0.97191	link

	Le Mort qui tue (film, 1913) Sequence 4	Original	0.00724	0.01139	0.02006	42.85	21.54	1.00000	link
		OldPhoto + TS	0.00572	0.00794	0.02810	50.15	18.18	1.37910	link
		Deepremaster	0.00760	0.01179	0.01870	42.38	20.48	1.00270	link
		NeatVideo	0.00550	0.00933	0.01252	47.17	20.30	1.15613	link
		RTN	0.00910	0.01173	0.02811	16.14	20.50	0.82047	link
		NFFA	0.00813	0.01123	0.02723	75.14	21.72	0.96234	link
		LFISTA	0.00503	0.00755	0.01133	40.90	20.23	1.48842	link
		LFISTA-SHARP	0.00591	0.00858	0.01328	37.71	18.92	1.20482	link
	Example 4	Original	0.01747	0.01760	0.12250	36.50	19.47	1.00000	link
		OldPhoto + TS	0.01749	0.01851	0.22618	34.08	19.12	0.77074	link
		Deepremaster	0.01410	0.01721	0.09121	43.05	23.23	0.94377	link
		NeatVideo	0.00513	0.00760	0.04535	48.04	23.13	1.13304	link
		RTN	0.01090	0.01339	0.08504	18.94	23.38	0.77152	link
		NFFA	0.00769	0.01103	0.04253	46.39	19.72	0.96726	link
		LFISTA	0.00714	0.00912	0.05776	47.93	22.87	1.11591	link
		LFISTA-SHARP	0.00755	0.00973	0.10641	46.25	19.05	0.95865	link
	The study of the Channel Tunnel project	Original	0.01108	0.01796	0.08044	31.23	23.52	1.00000	link
		OldPhoto + TS	0.00716	0.00973	0.05002	49.27	15.75	1.29625	link
		Deepremaster	0.00889	0.01595	0.04574	41.44	21.13	0.98691	link
		NeatVideo	0.00756	0.01777	0.01928	47.22	21.64	1.07383	link
		RTN	0.00817	0.01517	0.04237	20.81	22.04	0.80364	link
		NFFA	0.00403	0.00561	0.07413	42.36	18.81	2.01658	link
		LFISTA	0.00710	0.01479	0.02649	41.92	21.90	1.08459	link
		LFISTA-SHARP	0.00593	0.01019	0.02844	38.67	16.54	0.93687	link
	Example 5	Original	0.00833	0.01252	0.01956	34.64	18.29	1.00000	link
		OldPhoto + TS	0.00824	0.01078	0.03401	37.99	17.44	1.13718	link
		Deepremaster	0.00810	0.01255	0.01625	40.65	24.54	1.02161	link
		NeatVideo	0.00565	0.00894	0.01443	38.94	24.87	1.25574	link
		RTN	0.01066	0.01329	0.02866	15.36	25.06	0.75405	link
		NFFA	0.00525	0.00679	0.01434	36.21	18.48	1.57343	link
		LFISTA	0.00547	0.00849	0.01229	37.17	24.94	1.32683	link
		LFISTA-SHARP	0.00683	0.00991	0.01520	33.45	18.67	1.03981	link
D2	Example 1 (190 fps)	Original	0.01619	0.01184	0.01350	17.17	22.06	1.00000	link
		OldPhoto + TS	0.01240	0.01011	0.01359	27.86	16.61	1.20848	link
		Deepremaster	0.01132	0.01108	0.03390	28.22	21.56	0.89122	link
		NeatVideo	0.00517	0.00434	0.00576	19.99	22.12	2.76306	link
		RTN	0.00979	0.01204	0.02491	40.99	21.43	0.71483	link
		NFFA	0.00344	0.00447	0.00832	28.26	22.64	1.25105	link
		LFISTA	0.00431	0.00420	0.00413	26.54	22.21	2.95666	link
		LFISTA-SHARP	0.00417	0.00463	0.00343	25.61	19.57	2.31342	link
	Example 2 (150 fps)	Original	0.01298	0.01163	0.00807	20.57	20.57	1.00000	link
		OldPhoto + TS	0.01060	0.01028	0.01879	29.37	15.82	1.10486	link
		Deepremaster	0.00949	0.01054	0.01312	36.94	20.56	1.06048	link
		NeatVideo	0.00489	0.00507	0.00412	27.10	20.94	1.11053	link
		RTN	0.00807	0.00950	0.01063	27.39	18.49	0.91092	link
		NFFA	0.00433	0.00564	0.01024	40.85	17.65	0.89206	link
		LFISTA	0.00427	0.00470	0.00369	37.11	20.98	1.15442	link
		LFISTA-SHARP	0.00464	0.00553	0.00606	34.95	15.25	1.03161	link
	Guitar Video (240 fps)	Original	0.02145	0.02017	0.06741	31.97	14.35	1.00000	-

		OldPhoto + TS	0.02078	0.01998	0.07714	30.55	11.99	1.00773	-
		Deepremaster	0.01944	0.01850	0.04392	36.89	15.18	1.50771	-
		NeatVideo	0.01633	0.01372	0.05416	31.81	13.81	2.37171	-
		RTN	0.00759	0.00853	0.01999	16.26	13.28	2.96772	-
		NFFA	0.00396	0.00514	0.01700	40.85	16.22	4.26872	-
		LFISTA	0.00360	0.00425	0.00838	36.59	14.86	7.61568	-
		LFISTA-SHARP	0.00415	0.00500	0.01061	32.46	15.54	5.92946	-
	Example 4 (190 fps)	Original	0.01625	0.01281	0.01840	22.80	22.13	1.00000	link
		OldPhoto + TS	0.01189	0.01082	0.01979	25.79	12.81	1.10204	link
		Deepremaster	0.01205	0.01232	0.04981	32.86	22.49	0.85032	link
		NeatVideo	0.00683	0.00687	0.01308	28.05	22.10	1.26899	link
		RTN	0.00947	0.01120	0.02162	16.08	20.60	0.81068	link
		NFFA	0.00359	0.00465	0.00794	34.28	15.28	2.67800	link
		LFISTA	0.00588	0.00642	0.01014	31.24	22.19	1.32682	link
		LFISTA-SHARP	0.00478	0.00543	0.00508	30.48	13.76	1.10209	link
	Example 3 (150 fps)	Original	0.00407	0.00424	0.00697	17.06	22.42	1.00000	link
		OldPhoto + TS	0.00514	0.00618	0.00571	18.28	18.22	0.49762	link
		Deepremaster	0.00428	0.00499	0.01008	27.06	22.60	0.87568	link
		NeatVideo	0.00292	0.00310	0.00390	22.34	22.34	1.18771	link
		RTN	0.00743	0.00941	0.02793	27.40	20.94	0.39044	link
		NFFA	0.00269	0.00324	0.00452	32.31	28.62	0.95253	link
		LFISTA	0.00248	0.00277	0.00295	30.10	22.24	1.44467	link
		LFISTA-SHARP	0.00248	0.00306	0.00297	27.17	27.47	1.14998	link
D3	The Dog Outwits the Kidnapper (1908) Sequence 1	Original	0.00758	0.01172	0.03481	40.51	20.38	1.00000	link
		OldPhoto + TS	0.00956	0.01280	0.05017	27.67	17.78	0.89654	link
		Deepremaster	0.00844	0.01220	0.02907	40.42	20.75	0.97080	link
		NeatVideo	0.00664	0.01042	0.02230	42.19	20.50	1.03256	link
		RTN	0.01112	0.01328	0.04765	20.43	18.81	0.62921	link
		NFFA	0.00697	0.00967	0.04653	43.17	17.39	1.32844	link
		LFISTA	0.00607	0.00830	0.01985	37.52	20.41	1.05341	link
		LFISTA-SHARP	0.00734	0.00981	0.02310	35.88	16.80	0.72825	link
	The Dog Outwits the Kidnapper (1908) Sequence 2	Original	0.00952	0.01526	0.02332	33.03	21.31	1.00000	link
		OldPhoto + TS	0.01015	0.01490	0.03460	19.51	19.55	0.99608	link
		Deepremaster	0.00999	0.01539	0.02307	31.32	21.08	0.99461	link
		NeatVideo	0.00849	0.01388	0.01905	33.45	21.34	1.11652	link
		RTN	0.01121	0.01408	0.04036	10.54	20.71	0.77373	link
		NFFA	0.00676	0.00904	0.02513	32.32	23.64	1.56936	link
		LFISTA	0.00642	0.01019	0.01453	33.15	20.96	1.49502	link
		LFISTA-SHARP	0.00771	0.01137	0.01695	32.68	21.32	1.09239	link
	The Dog Outwits the Kidnapper (1908) Sequence 3	Original	0.00949	0.01416	0.02990	43.28	20.96	1.00000	link
		OldPhoto + TS	0.00996	0.01398	0.03916	33.06	17.34	0.98803	link
		Deepremaster	0.00988	0.01420	0.02749	43.63	20.90	0.99290	link
		NeatVideo	0.00578	0.00927	0.01656	45.91	21.16	1.38328	link
		RTN	0.01062	0.01313	0.04445	20.45	21.06	0.80632	link
		NFFA	0.00680	0.00890	0.03337	43.47	17.31	1.20447	link
		LFISTA	0.00631	0.00935	0.01685	43.13	20.98	1.37794	link
		LFISTA-SHARP	0.00751	0.01051	0.02064	41.67	16.47	1.02737	link

	The Dog Outwits the Kidnapper (1908) Sequence 4	Original	0.00812	0.01272	0.02913	36.10	22.10	1.00000	link
		OldPhoto + TS	0.00866	0.01231	0.03944	24.72	17.84	0.93871	link
		Deepremaster	0.00850	0.01288	0.02549	36.48	21.67	0.96929	link
		NeatVideo	0.00501	0.00759	0.01990	38.22	22.24	1.08419	link
		RTN	0.01008	0.01223	0.04340	15.93	20.29	0.68978	link
		NFFA	0.00617	0.00820	0.03414	43.53	18.34	1.02252	link
		LFISTA	0.00558	0.00770	0.01648	35.02	21.64	1.11579	link
		LFISTA-SHARP	0.00667	0.00899	0.02142	32.33	18.74	0.85639	link
	The Dog Outwits the Kidnapper (1908) Sequence 5	Original	0.01088	0.01797	0.04111	35.53	20.26	1.00000	link
		OldPhoto + TS	0.01123	0.01677	0.05730	21.00	15.64	1.08419	link
		Deepremaster	0.01123	0.01767	0.03237	35.40	21.01	1.02365	link
		NeatVideo	0.00787	0.01321	0.02700	37.05	20.33	1.40940	link
		RTN	0.01142	0.01464	0.04891	16.61	19.30	0.99579	link
		NFFA	0.00633	0.00852	0.03484	35.58	15.31	2.24099	link
		LFISTA	0.00715	0.01184	0.01781	33.32	20.30	1.62353	link
		LFISTA-SHARP	0.00833	0.01273	0.02327	30.16	16.31	1.30894	link
	The Dog Outwits the Kidnapper (1908) Sequence 6	Original	0.00812	0.01299	0.02723	39.55	19.68	1.00000	link
		OldPhoto + TS	0.00980	0.01338	0.04132	30.38	18.88	0.89377	link
		Deepremaster	0.00878	0.01330	0.02521	40.15	19.69	0.97111	link
		NeatVideo	0.00502	0.00795	0.01773	41.44	19.76	1.09725	link
		RTN	0.01157	0.01377	0.04229	25.01	18.38	0.66748	link
		NFFA	0.00656	0.00897	0.03397	37.23	16.88	1.24093	link
		LFISTA	0.00591	0.00844	0.01671	38.15	19.69	1.08761	link
		LFISTA-SHARP	0.00727	0.00997	0.01904	37.47	19.22	0.80084	link
	The Dog Outwits the Kidnapper (1908) Sequence 7	Original	0.00747	0.01192	0.03485	42.84	21.67	1.00000	link
		OldPhoto + TS	0.00790	0.01103	0.04300	32.35	17.62	1.00291	link
		Deepremaster	0.00778	0.01203	0.02843	43.43	21.82	0.99744	link
		NeatVideo	0.00504	0.00756	0.02372	46.01	21.75	1.16513	link
		RTN	0.00960	0.01160	0.04403	19.42	20.68	0.65195	link
		NFFA	0.00590	0.00763	0.03888	44.30	16.60	0.95972	link
		LFISTA	0.00523	0.00734	0.01618	39.68	21.46	1.10698	link
		LFISTA-SHARP	0.00638	0.00865	0.02076	35.13	16.57	0.84180	link
	The Dog Outwits the Kidnapper (1908) Sequence 8	Original	0.00929	0.01466	0.03262	35.55	20.15	1.00000	link
		OldPhoto + TS	0.00998	0.01409	0.04639	22.35	15.85	0.96370	link
		Deepremaster	0.00968	0.01471	0.02779	35.20	20.34	0.99441	link
		NeatVideo	0.00522	0.00853	0.01748	38.67	20.26	1.23861	link
		RTN	0.01076	0.01311	0.04173	14.32	19.54	0.67675	link
		NFFA	0.00614	0.00893	0.03723	44.23	17.35	0.98234	link
		LFISTA	0.00596	0.00900	0.01818	33.55	20.24	1.22483	link
		LFISTA-SHARP	0.00716	0.01027	0.02047	29.64	16.93	0.85714	link
	The Dog Outwits the Kidnapper (1908) Sequence 9	Original	0.00714	0.01112	0.02902	36.90	19.87	1.00000	link

	OldPhoto + TS	0.00947	0.01259	0.04422	24.37	17.33	0.86835	link
	Deepremaster	0.00801	0.01160	0.02609	37.29	20.25	0.98959	link
	NeatVideo	0.00564	0.00833	0.02174	38.62	19.95	1.07314	link
	RTN	0.01132	0.01343	0.04983	16.91	18.37	0.60024	link
	NFFA	0.00638	0.00855	0.03526	44.53	17.56	1.11593	link
	LFISTA	0.00605	0.00824	0.01738	34.35	19.93	1.10902	link
	LFISTA-SHARP	0.00751	0.00989	0.02093	33.42	15.66	0.78885	link
Kanch Ki Gudiya 1961	Original	0.00638	0.00801	0.02851	50.79	23.34	1.00000	link
	OldPhoto + TS	0.00600	0.00759	0.03232	46.67	15.81	1.21837	link
	Deepremaster	0.00689	0.00907	0.02524	47.80	23.51	0.91302	link
	NeatVideo	0.00513	0.00681	0.01860	53.73	22.95	1.04101	link
	RTN	0.00707	0.00874	0.03061	38.62	21.82	0.66838	link
	NFFA	0.00608	0.00884	0.04613	46.21	15.67	1.53962	link
	LFISTA	0.00480	0.00641	0.01709	47.75	23.17	1.07537	link
	LFISTA-SHARP	0.00580	0.00752	0.01995	45.24	13.81	0.81795	link
Indian song	Original	0.00459	0.00724	0.02613	50.54	25.97	1.00000	link
	OldPhoto + TS	0.00479	0.00664	0.03380	48.02	15.43	0.92676	link
	Deepremaster	0.00464	0.00733	0.02343	54.35	26.27	0.92952	link
	NeatVideo	0.00297	0.00447	0.01529	55.44	26.13	1.13874	link
	RTN	0.00529	0.00716	0.03307	34.66	23.75	0.82591	link
	NFFA	0.00599	0.00892	0.03551	45.23	17.23	1.30013	link
	LFISTA	0.00318	0.00466	0.01503	52.58	26.02	1.15312	link
	LFISTA-SHARP	0.00374	0.00527	0.02228	50.71	16.72	0.95135	link
Mabel and Fatty's Married Life 1915 Sequence 1	Original	0.00600	0.01068	0.01644	55.21	21.57	1.00000	link
	OldPhoto + TS	0.00706	0.01091	0.02478	36.77	21.30	0.92290	link
	Deepremaster	0.00692	0.01125	0.01847	49.04	20.67	0.95763	link
	NeatVideo	0.00426	0.00738	0.01006	56.30	21.60	1.43142	link
	RTN	0.00948	0.01223	0.04130	22.80	18.91	0.76779	link
	NFFA	0.00609	0.00865	0.02347	46.35	18.25	1.29017	link
	LFISTA	0.00504	0.00764	0.01229	53.12	20.62	1.38248	link
	LFISTA-SHARP	0.00612	0.00890	0.01522	50.16	19.60	1.03407	link
Mabel and Fatty's Married Life 1915 Sequence 2	Original	0.01495	0.01940	0.04758	56.14	23.45	1.00000	link
	OldPhoto + TS	0.01476	0.01792	0.07980	40.56	16.93	0.89677	link
	Deepremaster	0.01467	0.01958	0.04359	54.56	23.59	0.93974	link
	NeatVideo	0.00981	0.01365	0.02344	59.19	23.42	1.19818	link
	RTN	0.01082	0.01446	0.05544	34.96	22.71	0.84073	link
	NFFA	0.00902	0.01273	0.04476	44.23	18.52	0.92633	link
	LFISTA	0.01046	0.01379	0.03352	55.88	22.96	1.11548	link
	LFISTA-SHARP	0.01121	0.01410	0.05195	49.04	17.56	0.90382	link
Alphonse and Gaston, no. 3	Original	0.00954	0.01568	0.04477	55.41	20.34	1.00000	link
	OldPhoto + TS	0.01091	0.01607	0.06589	39.09	20.07	0.90077	link
	Deepremaster	0.01032	0.01585	0.04415	54.06	20.27	0.95327	link
	NeatVideo	0.00654	0.01060	0.02928	57.66	20.61	1.26671	link
	RTN	0.01135	0.01459	0.05284	35.08	19.06	0.73488	link
	NFFA	0.00902	0.01205	0.07683	53.47	19.28	1.38516	link
	LFISTA	0.00699	0.01045	0.03144	52.91	20.26	1.19557	link
	LFISTA-SHARP	0.00833	0.01206	0.04074	48.81	18.71	0.85416	link

A gesture fight in Hester Street	Original	0.01125	0.01877	0.05761	50.30	22.11	1.00000	link
	OldPhoto + TS	0.01252	0.01850	0.08699	32.56	17.83	0.87244	link
	Deepremaster	0.01237	0.01905	0.05704	47.00	22.43	0.94530	link
	NeatVideo	0.00926	0.01583	0.03429	51.85	22.80	1.14264	link
	RTN	0.01344	0.01760	0.06561	28.23	20.09	0.77724	link
	NFFA	0.00592	0.01134	0.03515	45.32	17.45	1.13514	link
	LFISTA	0.00971	0.01597	0.04231	48.79	22.67	1.11672	link
	LFISTA-SHARP	0.01111	0.01648	0.06266	44.71	18.73	0.83937	link
Panama Pacific Expo World's Fair in San Francisco 1915 Sequence 1	Original	0.00634	0.00906	0.07257	63.01	24.18	1.00000	link
	OldPhoto + TS	0.00666	0.00895	0.06888	52.53	17.04	0.94053	link
	Deepremaster	0.00622	0.00932	0.03642	63.46	24.28	0.96163	link
	NeatVideo	0.00411	0.00628	0.01271	61.09	23.08	1.44503	link
	RTN	0.00648	0.00839	0.02731	44.21	25.03	0.80439	link
	NFFA	0.00449	0.00615	0.06476	46.44	16.23	1.27850	link
	LFISTA	0.00420	0.00587	0.01213	57.57	22.37	1.47812	link
	LFISTA-SHARP	0.00494	0.00671	0.01594	54.69	15.36	1.14781	link
Panama Pacific Expo World's Fair in San Francisco 1915 Sequence 2	Original	0.00999	0.01370	0.03390	49.73	22.57	1.00000	link
	OldPhoto + TS	0.01106	0.01441	0.05109	32.56	18.75	0.84466	link
	Deepremaster	0.00941	0.01370	0.02535	46.21	22.51	1.06345	link
	NeatVideo	0.00566	0.00818	0.02087	51.10	22.74	1.52531	link
	RTN	0.00816	0.01026	0.03228	31.65	23.40	0.92725	link
	NFFA	0.00508	0.00677	0.02495	47.15	16.66	1.50904	link
	LFISTA	0.00500	0.00680	0.01379	48.39	22.45	1.76780	link
	LFISTA-SHARP	0.00607	0.00796	0.01699	42.25	17.29	1.29414	link
Panama Pacific Expo World's Fair in San Francisco 1915 Sequence 3	Original	0.00719	0.01021	0.04281	51.58	24.37	1.00000	link
	OldPhoto + TS	0.00894	0.01170	0.05708	36.49	21.80	0.79807	link
	Deepremaster	0.00729	0.01046	0.03307	48.61	23.86	1.03678	link
	NeatVideo	0.00519	0.00752	0.02552	54.17	24.43	1.28871	link
	RTN	0.00805	0.01031	0.04099	34.68	24.82	0.91092	link
	NFFA	0.00570	0.00729	0.03605	46.24	17.24	1.57865	link
	LFISTA	0.00486	0.00701	0.01864	47.73	23.62	1.44974	link
	LFISTA-SHARP	0.00583	0.00809	0.02187	43.06	21.86	1.13982	link
RaviShankar	Original	0.00650	0.00634	0.03299	62.71	22.66	1.00000	link
	OldPhoto + TS	0.00620	0.00804	0.06452	53.11	15.92	0.71124	link
	Deepremaster	0.00524	0.00771	0.02913	65.06	23.17	0.94563	link
	NeatVideo	0.00416	0.00550	0.02348	64.52	22.85	1.00750	link
	RTN	0.00553	0.00712	0.04434	37.67	23.44	0.81588	link
	NFFA	0.00477	0.00686	0.03893	45.23	16.92	1.00876	link
	LFISTA	0.00483	0.00540	0.02019	66.04	22.69	1.02279	link
	LFISTA-SHARP	0.00466	0.00564	0.03337	61.20	16.74	0.84520	link
Indian song #2	Original	0.00878	0.00970	0.10937	40.90	24.29	1.00000	link
	OldPhoto + TS	0.01009	0.01136	0.14336	23.47	18.33	0.77925	link
	Deepremaster	0.00794	0.00979	0.05651	37.87	24.04	0.91490	link

		NeatVideo	0.00634	0.00732	0.04388	41.21	23.52	1.06481	link
		RTN	0.00858	0.00962	0.05805	23.41	22.51	0.74619	link
		NFFA	0.00666	0.00862	0.09048	45.96	17.89	1.08447	link
		LFISTA	0.00602	0.00681	0.03861	42.82	23.62	1.08581	link
		LFISTA-SHARP	0.00698	0.00815	0.05130	38.88	17.74	0.86957	link
	Old Norse Vikings Festival (1927) - Britain on Film (Total duration of 1 min 6 sec divided into 4 sequences)	Original	0.00724	0.01126	0.02988	54.85	19.59	1.00000	link
		OldPhoto + TS	0.00831	0.01158	0.04094	43.61	14.66	0.91406	link
		Deepremaster	0.00841	0.01229	0.03117	54.85	19.59	0.95918	link
		NeatVideo	0.00531	0.00814	0.02239	55.93	19.89	1.15724	link
		RTN	0.00821	0.01068	0.03753	35.83	19.05	0.78252	link
		NFFA	0.00342	0.00935	0.04524	42.35	18.34	1.95324	link
		LFISTA	0.00535	0.00819	0.02005	52.97	20.03	1.17650	link
		LFISTA-SHARP	0.00624	0.00867	0.02361	53.07	14.76	0.87038	link
	St. Andrew's Wells (1920) (Total duration of 9 min 23 sec divided into 20 sequences)	Original	0.00480	0.00666	0.02836	46.68	20.94	1.00000	link
		OldPhoto + TS	0.00675	0.00886	0.03109	43.88	15.36	0.78870	link
		Deepremaster	0.00589	0.00873	0.02648	59.69	20.96	0.90276	link
		NeatVideo	0.00303	0.00461	0.01320	66.13	21.40	1.08616	link
		RTN	0.00706	0.00901	0.02883	63.17	20.25	0.89063	link
		NFFA	0.00314	0.00446	0.01066	75.82	17.52	2.22192	link
		LFISTA	0.00343	0.00479	0.01278	67.82	21.24	1.12401	link
		LFISTA-SHARP	0.00483	0.00684	0.01750	54.92	15.85	0.93678	link
	Wedding of Miss Carrie Alexander, Faversham 1913 (Total duration of 3 min 59 sec divided into 15 sequences)	Original	0.00783	0.01248	0.04102	38.35	20.38	1.00000	link
		OldPhoto + TS	0.00818	0.01170	0.05849	33.64	14.86	0.90821	link
		Deepremaster	0.00853	0.01272	0.05187	38.35	20.38	0.94495	link
		NeatVideo	0.00607	0.00968	0.03228	46.01	20.78	1.03253	link
		RTN	0.00941	0.01193	0.05800	30.39	19.30	0.76856	link
		NFFA	0.00471	0.00693	0.03692	42.01	15.30	2.26568	link
		LFISTA	0.00578	0.00886	0.03215	43.38	20.67	1.09870	link
		LFISTA-SHARP	0.00672	0.00974	0.04536	44.04	15.07	0.86946	link
	New Zealanders Win At Rugby (1916) - Britain on Film Sequence 1	Original	0.00731	0.01037	0.01839	40.46	20.13	1.00000	link
		OldPhoto + TS	0.00839	0.01085	0.02517	27.53	13.08	0.90075	link
		Deepremaster	0.00772	0.01110	0.01714	36.15	20.33	0.87210	link
		NeatVideo	0.00460	0.00737	0.01357	41.68	20.22	1.47734	link
		RTN	0.00886	0.01082	0.02822	19.22	19.91	0.77250	link
		NFFA	0.00478	0.00614	0.01821	39.38	15.41	1.89204	link
		LFISTA	0.00438	0.00663	0.01024	38.45	20.20	1.64308	link

		LFISTA-SHARP	0.00614	0.00845	0.01178	39.55	13.75	1.16966	link
	New Zealanders Win At Rugby (1916) - Britain on Film Sequence 2	Original	0.00659	0.00994	0.02161	41.94	20.67	1.00000	link
		OldPhoto + TS	0.00812	0.01079	0.02945	26.87	14.96	0.81714	link
		Deepremaster	0.00730	0.01060	0.02030	38.15	20.62	0.95685	link
		NeatVideo	0.00452	0.00678	0.01774	43.53	20.72	1.23579	link
		RTN	0.00897	0.01071	0.03180	20.12	20.77	0.64280	link
		NFFA	0.00543	0.00705	0.02460	38.13	16.23	1.15081	link
		LFISTA	0.00464	0.00664	0.01421	40.22	20.54	1.34741	link
		LFISTA-SHARP	0.00643	0.00850	0.01634	39.45	15.26	0.89828	link
	Round the Wirral with a Movie Camera (1934) Sequence 1	Original	0.00937	0.01058	0.05061	44.55	18.95	1.00000	link
		OldPhoto + TS	0.00806	0.00936	0.06387	46.19	16.34	1.02347	link
		Deepremaster	0.00806	0.01003	0.04004	39.35	22.19	1.00013	link
		NeatVideo	0.00470	0.00612	0.03231	50.13	22.65	1.19531	link
		RTN	0.00917	0.01042	0.05190	33.65	23.13	0.89387	link
		NFFA	0.00466	0.00584	0.03340	42.34	17.35	2.28079	link
		LFISTA	0.00493	0.00631	0.02071	47.83	22.75	1.77048	link
		LFISTA-SHARP	0.00595	0.00746	0.02387	49.21	13.43	1.35691	link
	Round the Wirral with a Movie Camera (1934) Sequence 2	Original	0.00788	0.00974	0.05494	37.86	18.82	1.00000	link
		OldPhoto + TS	0.00749	0.00909	0.06142	45.88	14.37	1.11008	link
		Deepremaster	0.00823	0.01095	0.04825	37.01	19.95	0.96934	link
		NeatVideo	0.00529	0.00662	0.04308	40.63	20.36	1.30269	link
		RTN	0.00922	0.01039	0.06163	48.14	20.20	0.77131	link
		NFFA	0.00459	0.00596	0.03502	43.92	16.20	1.84327	link
		LFISTA	0.00535	0.00683	0.03252	40.54	20.07	1.37521	link
		LFISTA-SHARP	0.00646	0.00800	0.03953	39.39	13.29	1.03091	link
	The Kiss in the Tunnel (BFI National Archive)	Original	0.00705	0.01195	0.07099	43.54	20.42	1.00000	link
		OldPhoto + TS	0.00868	0.01300	0.08410	17.83	17.13	0.81189	link
		Deepremaster	0.00885	0.01251	0.07146	38.01	20.54	0.98911	link
		NeatVideo	0.00609	0.00907	0.04080	40.55	21.59	1.17375	link
		RTN	0.00964	0.01180	0.06094	30.53	21.00	0.71899	link
		NFFA	0.00749	0.00906	0.10538	30.57	18.88	1.75659	link
		LFISTA	0.00581	0.00838	0.03179	39.52	19.90	1.26722	link
		LFISTA-SHARP	0.00692	0.00957	0.03380	31.45	17.03	0.89180	link
	Indian song 2 (color video)	Original	0.00535	0.01111	0.02937	51.81	22.83	1.00000	link
		OldPhoto + TS	0.00788	0.01043	0.04409	44.20	16.03	0.96605	link
		Deepremaster	0.00891	0.01122	0.03012	53.02	23.01	0.92529	link
		NeatVideo	0.00471	0.01297	0.02181	54.82	22.88	1.03584	link
		RTN	0.00706	0.01264	0.04920	29.81	22.00	0.83803	link
		NFFA	0.00692	0.01153	0.03831	47.24	22.31	1.15552	link
		LFISTA	0.00552	0.01139	0.02546	61.63	22.52	1.10329	link
		LFISTA-SHARP	0.00568	0.00973	0.03493	58.03	17.68	1.01544	link

Atlas	1907 Canada 001.mp4	Original	0.00656	0.00963	0.04941	53.14	15.33	1.00000	-
		NFFA	0.00567	0.00804	0.05193	60.05	16.54	1.31205	-
		LFISTA	0.00596	0.00795	0.02978	52.51	13.55	1.11208	-
	1907 Canada 002.mp4	Original	0.00655	0.00990	0.03896	42.25	15.87	1.00000	-
		NFFA	0.00522	0.00722	0.04537	52.25	18.09	1.40957	-
		LFISTA	0.00618	0.00826	0.03330	43.27	13.83	1.07833	-
	1928 Manchester 001.mp4	Original	0.00338	0.00660	0.01119	69.95	17.50	1.00000	-
		NFFA	0.00293	0.00456	0.01538	68.61	16.37	1.10633	-
		LFISTA	0.00327	0.00512	0.00899	67.39	17.91	1.25818	-
	97377 001.mp4	Original	0.01031	0.01261	0.16277	46.09	16.62	1.00000	-
		NFFA	0.00623	0.00782	0.08886	55.23	16.36	0.96032	-
		LFISTA	0.00569	0.00667	0.05563	63.81	16.42	1.12171	-
	97377 002.mp4	Original	0.00755	0.01224	0.14563	64.87	17.74	1.00000	-
		NFFA	0.00448	0.00844	0.09901	66.66	15.54	1.47109	-
		LFISTA	0.00524	0.00680	0.04370	82.17	17.30	1.13870	-
	97377 003.mp4	Original	0.00920	0.01208	0.12446	47.69	20.66	1.00000	-
		NFFA	0.00561	0.00712	0.05758	53.96	17.06	1.20441	-
		LFISTA	0.00632	0.00686	0.05572	56.15	16.69	1.07617	-
	97377 004.mp4	Original	0.00983	0.01266	0.08101	38.41	16.48	1.00000	-
		NFFA	0.00758	0.00943	0.07279	43.06	15.47	0.51097	-
		LFISTA	0.00519	0.00679	0.03835	49.97	16.38	1.03681	-
	africa speaks 001.mp4	Original	0.00799	0.01100	0.07598	48.83	20.39	1.00000	-
		NFFA	0.00588	0.00776	0.07315	51.22	17.75	1.58463	-
		LFISTA	0.00428	0.00624	0.02958	54.75	16.33	1.53632	-
	africa speaks 002.mp4	Original	0.00717	0.01057	0.07941	49.47	16.17	1.00000	-
		NFFA	0.00471	0.00605	0.07084	51.98	18.48	1.10735	-
		LFISTA	0.00568	0.00788	0.04136	49.82	14.76	1.15204	-
	africa speaks 003.mp4	Original	0.00703	0.01024	0.04267	50.30	19.04	1.00000	-
		NFFA	0.00430	0.00572	0.03252	52.07	16.82	1.92371	-
		LFISTA	0.00342	0.00497	0.01434	54.40	15.13	1.70497	-
	africa speaks 004.mp4	Original	0.00938	0.01358	0.06373	49.35	26.13	1.00000	-
		NFFA	0.00709	0.01061	0.06413	51.06	23.73	1.09605	-
		LFISTA	0.00462	0.00694	0.02846	59.94	22.18	1.22829	-
	africa speaks 005.mp4	Original	0.01563	0.01912	0.15502	42.99	33.84	1.00000	-
		NFFA	0.01403	0.01730	0.13652	44.90	32.27	1.23531	-
		LFISTA	0.00909	0.01158	0.07958	55.63	25.57	1.11964	-
	Around the world in 1896 001.mp4	Original	0.00757	0.01191	0.02822	56.66	16.06	1.00000	-
		NFFA	0.00636	0.00871	0.03808	60.42	16.48	1.38117	-
		LFISTA	0.00573	0.00942	0.01553	54.64	16.49	1.63312	-
	Around the world in 1896 002.mp4	Original	0.00583	0.00902	0.01604	55.45	18.74	1.00000	-
		NFFA	0.00474	0.00624	0.02129	56.41	17.65	1.30430	-
		LFISTA	0.00430	0.00615	0.01161	56.40	18.65	1.20295	-

born to the saddle 001.mp4	Original	0.00738	0.01039	0.03702	48.21	22.01	1.00000	-
	NFFA	0.00669	0.00919	0.04952	52.67	20.77	0.98386	-
	LFISTA	0.00661	0.00890	0.02226	49.14	20.76	1.16437	-
born to the saddle 002.mp4	Original	0.00768	0.00987	0.06188	47.28	15.27	1.00000	-
	NFFA	0.00665	0.00884	0.05388	48.25	15.41	1.19333	-
	LFISTA	0.00666	0.00856	0.02388	44.56	15.40	1.06856	-
CHRISTMAS 1932 001.mp4	Original	0.00770	0.00980	0.04614	45.52	18.65	1.00000	-
	NFFA	0.00626	0.00846	0.05160	50.71	18.71	1.08206	-
	LFISTA	0.00630	0.00806	0.03442	46.52	18.74	1.06803	-
CHRISTMAS 1932 002.mp4	Original	0.00731	0.00979	0.05395	52.36	17.61	1.00000	-
	NFFA	0.00645	0.00832	0.06911	56.40	17.17	1.05773	-
	LFISTA	0.00670	0.00868	0.04531	52.19	16.43	1.03796	-
CHRISTMAS 1932 003.mp4	Original	0.00734	0.00975	0.04546	42.50	18.07	1.00000	-
	NFFA	0.00643	0.00969	0.05618	48.98	17.74	1.00560	-
	LFISTA	0.00640	0.00787	0.03186	44.30	18.23	1.07297	-
danger ahead 001.mp4	Original	0.01089	0.01440	0.04271	41.25	26.76	1.00000	-
	NFFA	0.00822	0.01069	0.04418	42.05	28.30	1.28042	-
	LFISTA	0.00478	0.00715	0.01426	45.42	27.07	1.77432	-
danger ahead 002.mp4	Original	0.01368	0.01685	0.12669	42.45	18.22	1.00000	-
	NFFA	0.01407	0.01830	0.11298	42.95	24.19	0.98231	-
	LFISTA	0.01155	0.01362	0.08778	46.09	17.51	1.06368	-
danger ahead 003.mp4	Original	0.00963	0.01357	0.04641	36.61	22.03	1.00000	-
	NFFA	0.00727	0.01010	0.04324	37.72	22.37	1.17014	-
	LFISTA	0.00778	0.01064	0.03397	37.21	21.38	1.18385	-
danger ahead 004.mp4	Original	0.01077	0.01323	0.05223	41.27	19.97	1.00000	-
	NFFA	0.01205	0.01652	0.08719	44.52	21.38	0.63058	-
	LFISTA	0.00832	0.01061	0.03139	42.08	18.91	1.07064	-
Die 001.mp4	Original	0.01181	0.01278	0.05411	34.48	20.29	1.00000	-
	NFFA	0.00528	0.00653	0.04732	39.19	20.68	1.62794	-
	LFISTA	0.00476	0.00598	0.02513	39.55	17.51	1.36538	-
Die 002.mp4	Original	0.01831	0.01940	0.13488	48.19	21.26	1.00000	-
	NFFA	0.00507	0.00638	0.08072	52.59	18.90	2.76358	-
	LFISTA	0.00619	0.00884	0.03672	64.35	14.80	1.99030	-
Die 003.mp4	Original	0.01945	0.02061	0.14431	45.44	22.73	1.00000	-
	NFFA	0.00567	0.00705	0.10667	49.79	18.12	2.85076	-
	LFISTA	0.00620	0.00866	0.04015	62.04	15.40	2.12312	-
HOF 001.mp4	Original	0.01359	0.01634	0.10772	30.20	15.46	1.00000	-
	NFFA	0.01080	0.01398	0.10083	37.52	16.85	1.06046	-
	LFISTA	0.00770	0.01004	0.06656	50.39	18.07	1.05154	-
HOF 2 001.mp4	Original	0.01751	0.01844	0.16415	15.90	16.33	1.00000	-
	NFFA	0.00991	0.01278	0.09328	18.92	18.41	0.91674	-
	LFISTA	0.00629	0.00819	0.06323	45.15	18.69	1.10806	-
HOF 2 002.mp4	Original	0.01397	0.01602	0.14445	19.54	12.60	1.00000	-
	NFFA	0.01027	0.01456	0.11648	27.91	14.58	1.07767	-

	LFISTA	0.00889	0.01059	0.07942	51.09	18.44	1.04599	-
HOF 2 003.mp4	Original	0.01439	0.01378	0.13499	50.44	12.16	1.00000	-
	NFFA	0.00502	0.00577	0.08117	53.90	12.02	1.30126	-
	LFISTA	0.00707	0.00840	0.05306	55.12	17.73	1.17605	-
HOF 2 004.mp4	Original	0.01347	0.01528	0.13841	27.81	14.07	1.00000	-
	NFFA	0.01014	0.01306	0.11866	34.87	16.39	1.18384	-
	LFISTA	0.00748	0.00906	0.07995	53.21	16.22	1.03620	-
HOF 2 005.mp4	Original	0.01634	0.01806	0.18749	28.37	14.49	1.00000	-
	NFFA	0.01023	0.01327	0.18524	40.51	16.30	1.48768	-
	LFISTA	0.00694	0.00895	0.09100	61.43	15.72	1.42178	-
HOF 2 006.mp4	Original	0.01171	0.01265	0.17687	23.87	18.27	1.00000	-
	NFFA	0.00897	0.01150	0.13411	30.83	20.89	1.01547	-
	LFISTA	0.00523	0.00604	0.07336	55.58	18.55	1.01421	-
HOF 2 007.mp4	Original	0.01281	0.01429	0.20788	24.37	12.75	1.00000	-
	NFFA	0.00782	0.01055	0.12501	34.17	16.55	0.46386	-
	LFISTA	0.00678	0.00744	0.10172	53.14	18.67	1.14741	-
HOF 2 008.mp4	Original	0.01466	0.01722	0.14806	14.09	18.31	1.00000	-
	NFFA	0.01081	0.01275	0.12739	16.36	21.22	0.84228	-
	LFISTA	0.00790	0.01117	0.05851	40.83	21.36	1.04698	-
HOF 3 001.mp4	Original	0.01098	0.01406	0.07749	37.65	17.16	1.00000	-
	NFFA	0.00624	0.00777	0.05479	38.57	18.66	1.02559	-
	LFISTA	0.00799	0.01183	0.03591	42.58	20.94	1.08173	-
HOF 3 002.mp4	Original	0.01702	0.01755	0.13910	6.81	13.96	1.00000	-
	NFFA	0.01093	0.01334	0.11227	6.06	15.37	1.00801	-
	LFISTA	0.00625	0.00753	0.05230	41.40	18.21	1.13110	-
HOF 3 003.mp4	Original	0.01617	0.01684	0.15215	7.59	14.47	1.00000	-
	NFFA	0.01101	0.01323	0.12487	3.39	15.91	1.22096	-
	LFISTA	0.00544	0.00662	0.04443	36.17	20.41	1.90534	-
IA 35000011 001 001.mp4	Original	0.00749	0.00889	0.06956	42.99	17.87	1.00000	-
	NFFA	0.00702	0.00923	0.07894	44.95	16.47	1.17441	-
	LFISTA	0.00645	0.00730	0.05454	49.74	15.96	1.00591	-
Lagent 001.mp4	Original	0.01087	0.01580	0.03281	43.56	20.03	1.00000	-
	NFFA	0.00669	0.00890	0.02692	44.89	20.28	4.73668	-
	LFISTA	0.00677	0.01036	0.01319	51.43	16.61	2.28647	-
Lagent 002.mp4	Original	0.00861	0.01227	0.02064	44.16	22.87	1.00000	-
	NFFA	0.00617	0.00825	0.02180	45.12	23.22	1.49480	-
	LFISTA	0.00471	0.00684	0.01079	44.71	21.67	1.45318	-
Lagent 003.mp4	Original	0.00886	0.01319	0.02224	32.66	18.63	1.00000	-
	NFFA	0.00582	0.00750	0.02290	34.83	17.72	1.29914	-
	LFISTA	0.00576	0.00903	0.01377	33.24	16.58	1.48611	-
Lagent 004.mp4	Original	0.00852	0.01305	0.02382	37.90	22.81	1.00000	-
	NFFA	0.00593	0.00832	0.02638	39.21	21.29	1.64946	-
	LFISTA	0.00513	0.00803	0.01442	40.31	21.02	1.60354	-
Lagent 005.mp4	Original	0.01013	0.01332	0.05195	45.68	24.16	1.00000	-
	NFFA	0.00819	0.01059	0.05497	46.71	23.45	1.22579	-
	LFISTA	0.00569	0.00827	0.02073	48.59	21.38	1.73138	-
Lagent 006.mp4	Original	0.00789	0.01217	0.02076	54.08	21.13	1.00000	-
	NFFA	0.00531	0.00691	0.01967	55.76	21.32	1.91077	-
	LFISTA	0.00450	0.00684	0.01176	60.48	18.27	1.57985	-
Lagent 007.mp4	Original	0.00833	0.01180	0.01970	42.24	21.89	1.00000	-
	NFFA	0.00562	0.00734	0.01901	41.95	21.50	1.62098	-
	LFISTA	0.00451	0.00704	0.00873	50.89	16.87	1.90928	-

	Scarlet Street 001.mp4	Original	0.00610	0.00843	0.01915	49.34	18.55	1.00000	-
		NFFA	0.00427	0.00571	0.01471	50.70	19.47	1.64251	-
		LFISTA	0.00401	0.00523	0.00959	50.51	15.64	1.37280	-
	SGL 4 001.mp4	Original	0.00950	0.01289	0.08031	31.18	18.68	1.00000	-
		NFFA	0.01095	0.04513	0.06774	33.54	18.74	0.68892	-
		LFISTA	0.00722	0.01550	0.04324	31.89	17.10	1.04968	-
	SGL 4 002.mp4	Original	0.01837	0.02159	0.06880	29.72	17.44	1.00000	-
		NFFA	0.02163	0.03015	0.06468	46.28	17.35	1.05982	-
		LFISTA	0.01423	0.01593	0.05510	33.47	16.96	1.05985	-
	SGL 4 003.mp4	Original	0.01343	0.01693	0.11222	36.67	18.56	1.00000	-
		NFFA	0.00924	0.01282	0.10144	45.11	17.19	1.04376	-
		LFISTA	0.00939	0.01236	0.08714	42.66	15.60	1.13837	-
	SGL 4 004.mp4	Original	0.00744	0.01013	0.08181	33.53	16.08	1.00000	-
		NFFA	0.00528	0.00742	0.05166	40.52	15.56	1.52285	-
		LFISTA	0.00518	0.00699	0.03397	42.42	15.85	1.16996	-
	TheFrontPage1931 001.mp4	Original	0.00620	0.00867	0.01950	57.95	18.64	1.00000	-
		NFFA	0.00457	0.00624	0.01708	61.86	18.57	0.83515	-
		LFISTA	0.00326	0.00487	0.00984	66.01	14.67	1.47641	-
	TheGoodforNothing 001.mp4	Original	0.01029	0.01325	0.04730	27.40	16.63	1.00000	-
		NFFA	0.00781	0.01011	0.05118	29.10	16.39	1.03019	-
		LFISTA	0.00566	0.00729	0.02576	38.37	14.48	1.22978	-
	Winter Scenes in Holland 001.mp4	Original	0.00734	0.00975	0.04546	42.50	18.07	1.00000	-
		NFFA	0.00646	0.00969	0.05756	48.68	17.19	0.99821	-
		LFISTA	0.00640	0.00787	0.03186	44.30	18.23	1.07297	-
	Winter Scenes in Holland 002.mp4	Original	0.00770	0.00980	0.04614	45.52	18.65	1.00000	-
		NFFA	0.00616	0.00839	0.05163	50.74	18.98	1.12887	-
		LFISTA	0.00630	0.00806	0.03442	46.52	18.74	1.06803	-
	Winter Scenes in Holland 003.mp4	Original	0.00694	0.01029	0.03109	45.68	15.30	1.00000	-
		NFFA	0.00570	0.00812	0.03252	48.59	15.76	0.78119	-
		LFISTA	0.00525	0.00772	0.01640	49.14	14.75	1.13670	-
	Winter Scenes in Holland 004.mp4	Original	0.00799	0.01156	0.05276	50.02	16.01	1.00000	-
		NFFA	0.00601	0.00830	0.06094	58.46	18.14	1.61320	-
		LFISTA	0.00526	0.00778	0.01860	54.57	15.78	1.80398	-
	Winter Scenes in Holland 005.mp4	Original	0.00997	0.01504	0.04406	36.05	17.97	1.00000	-
		NFFA	0.00790	0.01105	0.05589	36.22	18.10	1.07171	-
		LFISTA	0.00750	0.01106	0.01998	36.76	15.72	1.19862	-
	Winter Scenes in Holland 006.mp4	Original	0.00828	0.01197	0.05091	44.26	14.47	1.00000	-
		NFFA	0.00670	0.00919	0.06491	49.94	15.75	0.94599	-
		LFISTA	0.00682	0.00908	0.03559	45.90	14.51	1.33872	-
	Winter Scenes in Holland 007.mp4	Original	0.00749	0.00946	0.06015	52.08	17.65	1.00000	-
		NFFA	0.00646	0.00829	0.06970	56.30	17.16	1.03692	-
		LFISTA	0.00676	0.00845	0.05126	53.24	16.09	1.04254	-

References

[1] 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. 3

[2] S. Iizuka and E. Simo-Serra. Deepremaster: Temporal source-reference attention networks for comprehensive video enhancement. *ACM Trans. Graph.*, 38(6), 2019. 1, 2, 3

[3] W.-S. Lai, J.-B. Huang, O. Wang, E. Shechtman, E. Yumer, and M.-H. Yang. Learning blind video temporal consistency. In *ECCV*, 2018. 1, 3

[4] C. Lei, X. Ren, Z. Zhang, and Q. Chen. Blind video deflickering by neural filtering with a flawed atlas. In *CVPR*, 2023. 3

[5] A. Mittal, A.K. Moorthy, and A. C. Bovik. Blind/referenceless image spatial quality evaluator. In *Asilomar conference on signals, systems and computers*, 2011. 3

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

[7] Z. Wan, B. Zhang, D. Chen, and J. Liao. Bringing old films back to life. In *IEEE CVPR*, 2022. 1, 2, 3

[8] Z. Wan, B. Zhang, D. Chen, P. Zhang, D. Chen, J. Liao, and F. Wen. Bringing old photos back to life. In *CVPR*, 2020. 1, 3