Advanced Image Processing Project Report

Atulya Kumar Verma Sahil Jain

June 2020

1 Introduction

In this project we have implemented the video denoising through low-rank matrix completion as described in the paper [1].

2 Problem Statement

The objective of the project and the paper is to perform video denoising, knowing very little or nothing about the noise structure and intensities. We are given a set of consecutive video frames noised with mixture of poisson, gaussian and impulsive noises, of unknown intensities. And problem is to recover the original video frames from it.

3 Problem Formulation and Solution Overview

Consider the noisy video V consisting of K frames, $V = \{v_k\}_{k=1}^{k=K}$. Each frame can be written as

$$v_k = u_k + g_k \tag{1}$$

where u_k is the underlying clean frame(image) and g_k is the noise. Our goal is to recover $U = \{u_k\}_{k=1}^{k=K}$, given V, by removing the noise g_k, where the noise can be from mixed distribution(s).

The solution of the above problem is as follows -

1. To exploit the temporal redundancy in a video, consider a patch with dimensions $n \times n \times ch$ centered at pixel j from k-th frame, $p_{j,k}$ as the reference patch, where n is some small value, say 8 and ch represents the number of color channels, 3 for RGB image and 1 for B/W image. We search for patches similar to the reference patch in this as well as neighbouring frames using the algorithms described in [3], and obtain, say, m total patches $\{p_{i,j,k}\}_{i=1}^{i=m}$. Let $p_{i,j,k}(:)$ denotes the vectorised version of

$p_{i,j,k}$, define an $(n^2 * ch) \times m$ patch matrix $P_{j,k}$ by horizontally concatenating all vectorised patches as

$$P_{j,k} = [p_{1,j,k}(:)\ p_{2,j,k}(:)\ ...\ p_{m,j,k}(:)] \tag{2}$$

Then, we can rewrite equation (1) as

$$P_{j,k} = Q_{j,k} + N_{j,k} \tag{3}$$

where $Q_{j,k}$ denotes the patch matrix obtained from the clean frames u_k and $N_{j,k}$ denotes the noise.

Since similar patches would have similar underlying structure, we expect $Q_{j,k}$ to have a low rank, and by choosing only relatively less noisy elements from our $P_{j,k}$ matrix as Ω for recovery, low-rank matrix completion techniques can be utilized to recover $Q_{j,k}$, which is further explained below.

2. Having obtained $P_{j,k}$ and Ω(the set of reliable points), we want to recover the matrix $Q_{j,k}$ under the constraints that it has low rank, and that for elements belonging to the set Ω, the values should be close to the corresponding values in $P_{j,k}$. Thus, our problem has been formulated as that of low-rank matrix completion under noise, where given certain reliable points Ω, we would like to complete the matrix $Q_{j,k}$.
There exist various algorithms like Fixed Point Iterative Algorithm [5], Alternating Direction Algorithm [6], Singular Value Thresholding(SVT) [4] etc for solving the above problem. We employ the first two algorithms in this project.

4 Patch Finding Algorithm

As described in the paper [3], we have used block matching techniques and used the Mean Absolute Difference (MAD) values to compare the quality of match between the reference patch and some other patch. We have implemented both exhaustive and fast algorithms for comparison's sake. For the matching algorithms to work, it is imperative that impulsive noises are removed from the images, as they can affect the matching result significantly. For this we have used the RAMF adaptive median filtering technique [2]. After this we have considered reference patches at some regular intervals, and searched the neighbouring frames for similar patches $p_{i,j,k}$'s. Once the similar patches are stacked in $P_{j,k}$ matrix (say $N \times M$), reliable points from this are selected in the Ω matrix. This is done by calculating $\bar{\sigma}$ as:

$$\bar{\sigma} = (\text{average variance of all rows of } P_{j,k})^{\frac{1}{2}}$$

$2\bar{\sigma}$ is used as the threshold to select reliable points. The points that were changed in the Median Filter Algorithm are also removed from this Ω matrix. Next step is to recover $P_{j,k}$ matrix from the reliable points in Ω for which we have used the following algorithms.

5 Matrix Completion Algorithms

5.1 Fixed Point Iterative Algorithm [5]

The objective of the algorithm is to find a low rank matrix Q which approximates the matrix P whose elements are partially known. The chosen minimization problem is as follows -

$$\min_{Q} \|Q\|_* \text{ s.t. } \|Q|_\Omega - P|_\Omega\|_F^2 \leq \#(\Omega)\hat{\sigma}^2 \tag{4}$$

where $\|\cdot\|_*$ denotes the nuclear norm of a matrix (sum of non-zero singular values), $\#(\Omega)$ is the size of the set Ω, $\hat{\sigma}$ is the estimate of standard deviation of noise obtained by calculating the average of the variances of all elements $\in \Omega$ on each row, and $M|_\Omega$ is the projection operator for a matrix Mk given by -

$$M|_\Omega = \begin{cases} M(i,j), & \text{if}(i,j) \in \Omega \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

As may be noted, and has been shown [7, 8, 9], this minimization problem is a convex relaxation of the following problem -

$$\min_{Q} \text{ rank}(Q) \text{ s.t. } \|Q|_\Omega - P|_\Omega\|_F^2 \leq \#(\Omega)\hat{\sigma}^2 \tag{6}$$

We choose to solve the Lagrangian version of the above equation formulated as

$$\min_{Q} \frac{1}{2}\|Q|_\Omega - P|_\Omega\|_F^2 + \mu\|Q\|_* \tag{7}$$

where μ is the Lagrangian parameter, empirically chosen as $\mu = \hat{\sigma}\sqrt{p}(\sqrt{n_1} + \sqrt{n_2})$[9], where $n_1 \times n_2$ is the size of the patch matrix ($n_1 = n * n * ch$ and $n_2 = m$ in our case) and p is the ratio of the number of pixels in Ω to the total number of pixels in the patch matrix $n_1 * n_2$.

The fixed point iterative algorithm update is as follows -

1. Initialize $Q^{(0)} := 0$
2. While convergence criterion is not met, repeat the following -

   $$R^{(k)} := Q^{(k)} - \tau * (Q^{(k)} - P)|_\Omega$$
   $$Q^{(k+1)} := D_{\tau\mu}(R^{(k)})$$

   where μ and $1 \leq \tau \leq 2$ are pre-defined parameters, D is the shrinkage operator defined as -

   $$D_\tau(X) = U\Sigma_\tau V^T \tag{8}$$

   where $X = U\Sigma V^T$ is the SVD for X and $\Sigma_\tau = \text{diag}(\max(\sigma_i - \tau, 0))$.
3. Output $Q := Q^{(k)}$.

5.2 Alternating Direction Algorithm [6]

The objective of the algorithm is to find nonnegative matrices X and Y, such that their product XY approximates a nonnegative data matrix M whose elements are partially known (to a certain accuracy). The Nonnegative Matrix Factorisation/Completion (NMFC) problem is stated as follows:

$$\min_{X,Y}\{||P_\Omega(XY-M)||_F^2 : X \in \mathbb{R}^{m\times q}, Y \in \mathbb{R}^{q\times n}, X_{i,j} \geq 0, Y_{i,j} \geq 0, \forall i,j\}$$

Where P_Ω is the projection operator defined as follows

$$P_\Omega(Z)_{i,j} = \begin{cases} Z_{i,j}, \text{ if } \Omega_{i,j} = 1 \\ 0, \text{ otherwise} \end{cases}$$

This problem is formulated as follows:

$$\min_{U,V,X,Y,Z}\{\frac{1}{2}||XY-Z||_F^2 : X = U, Y = V, U \geq 0, V \geq 0, P_\Omega(Z-M) = 0\}$$

The augmented Lagrangian of the above mentioned problem is:

$$\mathbb{L}({}_A(X,Y,Z,U,V,\Lambda,\Pi) = \frac{1}{2}||XY-Z||_F^2 + \Lambda*(X-U) + \Pi*(Y-V) + \frac{\alpha}{2}||X-U||_F^2 + \frac{\beta}{2}||Y-V||_F^2$$

where $\Lambda \in \mathbb{R}^{m\times q}, \Pi \in \mathbb{R}^{q\times n}$ are Lagrangian multipliers, α, $\beta > 0$ are penalty parameters and X and Y are initialised with random nonnegative values. The alternating direction method for the above is derived by successively minimizing $\mathbb{L}_A$ with respect to X, Y, Z, U, V, one at a time while fixing others at their most recent values, i.e,

$$X_{k+1} \leftarrow \arg\min \ \mathbb{L}_A(X, Y_k, Z_k, U_k, V_k, \Lambda_k, \Pi_k),$$

$$Y_{k+1} \leftarrow \arg\min \ \mathbb{L}_A(X_{k+1}, Y, Z_k, U_k, V_k, \Lambda_k, \Pi_k),$$

$$Z_{k+1} \leftarrow \arg\min_{\mathbb{P}_\Omega(Z-M)=0} \mathbb{L}_A(X_{k+1}, Y_{k+1}, Z, U_k, V_k, \Lambda_k, \Pi_k),$$

$$U_{k+1} \leftarrow \arg\min_{U\geq 0} \mathbb{L}_A(X_{k+1}, Y_{k+1}, Z_{k+1}, U, V_k, \Lambda_k, \Pi_k),$$

$$V_{k+1} \leftarrow \arg\min_{V\geq 0} \mathbb{L}_A(X_{k+1}, Y_{k+1}, Z_{k+1}, U_{k+1}, V, \Lambda_k, \Pi_k),$$

and then updating the multipliers Λ and Π. The closed formula for the above minimisations are as mentioned in the paper [6] These iterations are done until the stopping criteria is reached.

6 Results

In all the following experiments, the number of patches selected per frame is set to 5, the patch size has been fixed to $8 \times 8 \times 3$(3 is the number of channels), the reference interval(the stride) has been chosen to be 4, the maximum window size provided to median filter is 5(this is actually in one direction, so in effect we have a maximum window size of 11), and we are considering patches only within a boundary(searchArea) of 7 pixels of our reference patch, and we are only considering neighbouring frames upto a distance(neighbourhood) of 5 from the current frame; unless mentioned otherwise. Consider the noise levels as denoted by $\{\sigma, \kappa, s\}$, where σ is the standard deviation of the 0-mean Gaussian noise, $\kappa *$ (image matrix) is the variance and mean of the Poisson noise, which is added to the image matrix, and s denotes the fraction of pixels corrupted by Impulsive noise [1].

While doing patch finding, we are calculating MAD only based on the pixels of the reference patch unchanged by the median filter, including this detail didn't alter our results by too much, so we just chose to include it.

(a) Original (b) Noisy (c) Filtered (d) Recovered

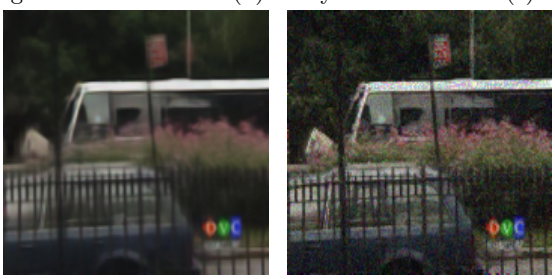
Figure 1: Fixed Point Iteration(FPI - Alg 1) on 50 frames

In 1 we set the neighbourhood to 10 frames and searchArea to 50 pixels, and the noise levels are set to $\{\sigma = 10, \kappa = 0.3, s = 0.3\}$. Since we chose a high searchArea, the algorithm took 12 hours to converge, and the reported PSNRs and MSEs are as follows -

	Noisy	Filtered	Recovered
PSNR	9.9986	22.6898	22.0388
MSE per pixel	0.1000	0.0054	0.0063

Table 1

(a) Original (b) Noisy (c) Filtered



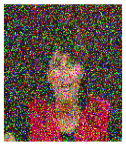
(d) FPI (e) ADM

Figure 2: 20 frames - $\{\sigma = 15, \kappa = 0.3, s = 0.3\}$

	Noisy	Filtered	Alg-1	Alg-2
PSNR	9.7316	20.1180	21.1524	22.2825
MSE per pixel	0.1064	0.0097	0.0077	0.0059

Table 2

(a) Original



(b) Noisy

(c) Filtered

(d) FPI

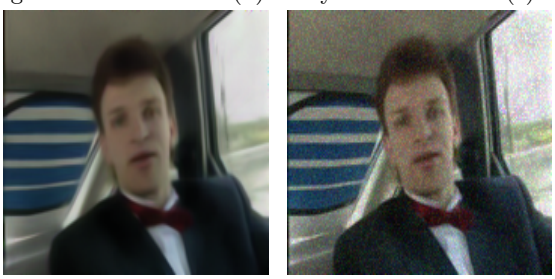
(e) ADM

Figure 3: 15 frames - $\{\sigma = 10, \kappa = 0.3, s = 0.3\}$

	Noisy	Filtered	Alg-1	Alg-2
PSNR	9.5	23.42	23.26	28.13
MSE per pixel	0.107	0.0045	0.0047	0.0015

Table 3

(a) Original (b) Noisy (c) Filtered



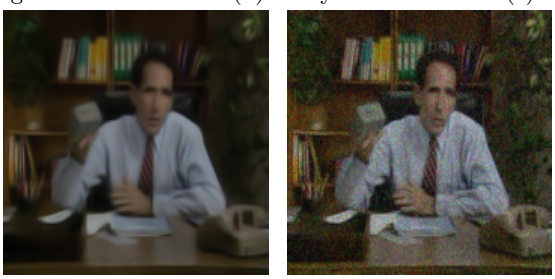
(d) FPI (e) ADM

Figure 4: 15 frames - $\{\sigma = 15, \kappa = 0.3, s = 0.3\}$

	Noisy	Filtered	Alg-1	Alg-2
PSNR	9.92	21.75	22.17	25.01
MSE per pixel	0.098	0.0068	0.0053	0.0032

Table 4

(a) Original (b) Noisy (c) Filtered



(d) FPI (e) ADM

Figure 5: 15 frames - $\{\sigma = 15, \kappa = 0.3, s = 0.3\}$

	Noisy	Filtered	Alg-1	Alg-2
PSNR	9.79	21.64	22.2	26.1
MSE per pixel	0.098	0.0069	00058	0.0017

Table 5

(a) Original (b) Noisy (c) Filtered

(d) FPI (e) ADM

Figure 6: 15 frames - $\{\sigma = 15, \kappa = 0.3, s = 0.3\}$

	Noisy	Filtered	Alg-1	Alg-2
PSNR	10.4983	21.8686	27.4008	25.5028
MSE per pixel	0.0892	0.0065	0.0018	0.0028

Table 6

s $\|\kappa$	10%	30%
10%	25.6169	25.5328
30%	24.8478	24.6903

Table 7: PSNR values for Alg-1 with $\sigma = 30$

s $\|\kappa$	10%	30%
10%	22.6579	22.5431
30%	22.1336	22.0554

Table 8: PSNR values for Alg-2 with $\sigma = 30$

7 Observations

1. We have implemented both the fast search algorithm (which uses subsampling of the pixels being matched) and the exhaustive algorithm for patch matching. But it appears that the 'fast' algorithm works much slower than the exhaustive search algorithm. For example, a task that took the exhaustive search 237s to complete, took 288s to complete using fast search. This is why, we have primarily used exhaustive search in our implementation.

2. The Alternating Direction Method (ADM) actually doesn't address the noise added to the frames. So the results although have higher PSNR values, they appear noisier than the ones recovered using Fixed Point Iteration (FPI) Method. However, the recovery quality of the ADM in terms of PSNR and MSE is superior to that of the FPI result. It also seems that it preserves more textures, as compared to the images from FPI which are rather blurred.

3. In the FPI algorithm we have used a regularisation parameter rgl, which ensures that the value of $\hat{\sigma}$ doesn't go too low. Its a small constant added to the square of $\hat{\sigma}$. Without this parameter, we observed that $\hat{\sigma}$ became 0 for some patch matrices and the recovery was very poor.

4. We implemented the Singular Value Thresholding Algorithm [4] for the matrix completion problem as well. But it appears to give very poor results as compared to the FPI or the ADM method.

5. The paper wrongly states that Poisson noise has zero mean, moreover in the paper the κ values should be in percentage to make any sense at all, otherwise most pixels will become very bright.

8 Link to the datasets

1. Car video: `https://media.xiph.org/video/derf/y4m/carphone_qcif.y4m`

2. Bus video: `https://media.xiph.org/video/derf/y4m/bus_qcif_15fps.y4m`

3. Salesman video: `https://media.xiph.org/video/derf/y4m/salesman_qcif.y4m`

4. Miss Am: `https://media.xiph.org/video/derf/y4m/miss_am_qcif.y4m`

5. Trevor video: `https://media.xiph.org/video/derf/y4m/trevor_qcif.y4m`

References

[1] Hui Ji, Chaoqiang Liu, Zuowei Shen and Yuhong Xu: Robust video denoising using Low rank matrix completion

[2] H. Hwang and R. A. Haddad: Adaptive median filters: new algorithms and results

[3] Bede Liu and Andre Zaccarin: New Fast Algorithms for the Estimation of Block Motion Vectors

[4] Jian-Feng Cai, Emmanuel J. Candes and Zuowei Shen: A Singular Value Thresholding Algorithm for Matrix Completion

[5] Shiqian Ma, Donald Goldfarb and Lifeng Chen: Fixed point and Bregman iterative methods for matrix rank minimization

[6] Yangyang Xu, Watao Yin, Zaiwen Wen and Yin Zhang: An alternating direction algorithm for matrix completion with nonnegative factors

[7] Emmanuel J. Candes and Benjamin Recht: Exact Matrix Completion via Convex Optimization

[8] Emmanuel J. Candes and Terence Tao: The Power of Convex Relaxation: Near-Optimal Matrix Completion

[9] Emmanuel J. Candes and Yaniv Plan: Matrix Completion With Noise