Video Denoising with Local Linear Denoising and Non-Local Means

Emil Vardar and Lovisa Byman

Abstract—Denoising of images is a well-studied subject, and some very sophisticated methods to denoise images already exist. The denoising methods used for images could be used to denoise videos frame by frame, but we wanted to investigate how the similarities between different frames could be taken advantage of to improve the denoising of videos. In this study, we implemented four different denoising methods for videos, namely an averaging of successive frames, an averaging of successive frames with Gaussian distributed weights, a non-local means algorithm that only searches for similar patches in the same frame, and a non-local means algorithm which searches for similar patches in the same frame as well as in other frames in the video. Each of the methods was tested for different parameters to find the best parameters. The results from each method were then compared with the state-of-the-art Video BM3D method. The results show that the two averaging methods perform very poorly. The two non-local means methods perform much better, but none of our methods can come close, regarding PSNR performance, to Video BM3D.

Index Terms—Video Denoising, Local Linear Denoising, Non-local Means Denoising, VBM3D

1 Introduction

The number of taken and shared images increase at a fantastic rate with each day going by. The most important reason for this is that it is easier to take pictures nowadays, as the cameras have shrunk enough to fit in our cellphones. We can take out our cellphones from our pockets, and by pressing three to four buttons, we can take a high-quality picture. However, a tremendous amount of calculations happen in the background after one pushes the shutter button. Unfortunately, the images also get corrupted by Gaussian noise while these calculations happen. Primarily, the analog circuitry in the camera is the reason for the Gaussian noise [1]. Therefore, even today, restoring the ground truth image from a noisy image is a much-researched topic.

To obtain a denoised image from a noisy image is called denoising. In this project, we will look at how to denoise a video. A video is a bunch of images shown one after another at a high rate. Therefore, many image denoising techniques can be adapted to video denoising with some minor changes which we discuss in this paper.

2 Related Work

There are already many image denoising algorithms that elegantly remove Gaussian noise. One of them is the non-local mean denoiser. This denoising algorithm is about searching for similar patches in the image and combining these in a sensibly way so that the noise is eliminated [2]. The non-local means algorithm takes advantage of the high redundancy in a natural image. In [2], Buades et al. show that the non-local means algorithm successfully reduces noise while keeping the sharp edges. However, as mentioned in [3], the complexity for the non-local means algorithm when searching for similar patches over the whole image is too big to realize it in any practical application. Therefore, the search area must be limited to a particular neighborhood size [3], [4]. In this project, we will extend this approach to videos. We will search for similar patches not only in the current frame but also in the neighboring frames to reduce the noise even further. We will compare this method with the one where each frame is denoised separately with a non-local mean algorithm. We will discuss and go into depth about these in Section 3.3 and 3.4.

Some known video denoisers are, for example, VBM3D, 3DWTF, and WRSTF. Dabov et al. show in [5] that VBM3D outperforms 3DWTF and WRSTF. Therefore, in this project, we will compare our results to VBM3D to see how much better or worse our simple algorithms are compared to one of the state-of-the-art video denoisers. We will not detail the theory behind the VBM3D in this paper. For that, we refer to [5], and [6].

We will also use two inferior algorithms. The first one is only averaging the pixel values over different frames. This approach we call local averaging denoiser in this text. Furthermore, we will extend this to use Gaussian weights for the nearby frames, and this we call local Gaussian denoiser. The goal with these is to show how much better the sophisticated algorithms such as non-local means or VBM3D algorithms are compared to these basic approaches.

In the last couple of years, machine learning approaches have been used for video denoising, as in many other research areas. One of them is given in [7]. In this article, the authors show that their algorithm based on deep neural network gives a slightly better PSNR than VBM3D. Even though there are slightly better methods than VBM3D, we will compare our results to VBM3D in this work since VBM3D is a highly accepted algorithm.

3 Theory

To see how good the different denoising methods are, we first separate a video into its frames. Then we add zero-mean Gaussian noise to each of these frames with different standard deviations since we assume that the ground

truth video does not contain noise. We used three different videos in this project, each with 150 frames. The reason we used three different videos was so that a random factor in one specific video does not happen to effect the denoising results. Furthermore, we only assume that the videos contain Gaussian noise and no other noises such as Poisson noise, quantization noise, etc. After we have added Gaussian noise, the noisy frames were denoised with the denoisers mentioned in the following subsections and with VBM3D. We underline that we *did not* implement any code for VBM3D but used the code given in [8]. We could not find any source code for VBM3D in Python, and instead we used MATLAB for evaluating the results for VBM3D while the other denoisers were evaluated with Python. However, since our project aims to compare how well the different algorithms work in the sense of quality (PSNR and visual comparison) and not how fast they are, it will not make any difference to have codes in different programming environments.

After each frame in the video had been denoised, we compared the video to the ground truth video (the video before noise was added) by calculating the PSNR, see Section 3.5. Finally, the denoised frames were combined to construct the denoised video for visual comparison.

3.1 Local Averaging Denoiser

The local averaging denoiser is perhaps the most basic denoising algorithm one can think of. The logic goes that since a video consists of many images (frames) per second, the nearby frames will not differ from each other that much. Therefore, we can average the value at the same pixel position over nearby frames. Under the assumption that all of the K frames are aligned with no differences, the averaged signal to noise ratio (SNR) will be

$$SNR_{avg} = \frac{\sqrt{K}\mu}{\sigma} \tag{1}$$

where each of the K images has an SNR of μ/σ. Here we see that the SNR increases with increased K and therefore averaging over nearby frames can increase the SNR, if the assumption that all of the frames are aligned with no differences holds.

Let us assume that we average over 3 frames ($K = 3$). That is we average the values of pixel (k, j) in frame $i - 1$, i, and $i + 1$ as shown in Figure 1. Mathematically, we can

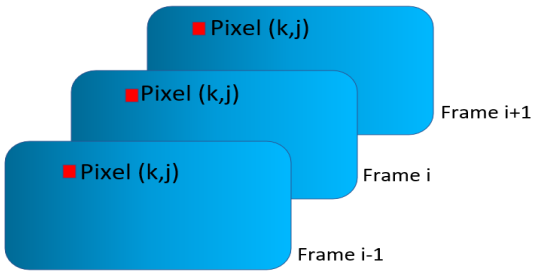


Fig. 1: Local linear denoiser representation.

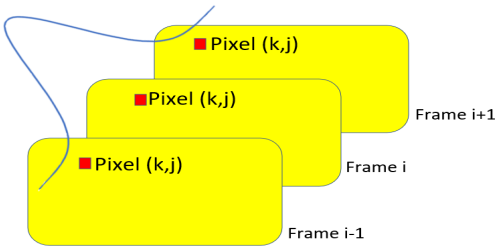


Fig. 2: Local Gaussian denoiser representation.

formulate this as

$$\text{Denoised}^t_{(k,j)} = \frac{1}{K} \sum_{t=i-\lfloor K/2 \rfloor}^{i+\lfloor K/2 \rfloor} \text{noisy}^t_{(k,j)} \tag{2}$$

where Denoised^t stands for the t:th denoised frame and noisy^t stands for the t:th noisy frame. This is of course done for all combinations (k, j).

It is easy to see that this causes a problem on the first and last frames. For example, assume $K = 5$ and $i = 1$. Then we need to average frames -1, 0, 1, 2, and 3. However, there are no frames -1 and 0. We take as many frames as possible on the first and last frames to counteract this problem. In the example above, this corresponds to only averaging frames 1, 2, and 3 in order to get the denoised frame for $i = 1$. Therefore, the edge frames will be denoised a little worse than the others. However, a few frames at the beginning and end on a long video with tens of thousands of frames will not even be noticeable, and therefore, there is no need to make a complex edge case denoiser.

3.2 Local Gaussian Denoiser

The local Gaussian denoiser is an extension of the local averaging denoiser. The difference is that instead of averaging the pixel values directly, we attach a Gaussian weight to the nearby frames (only for those who are going to be used in the averaging) depending on how far they are from the frame in focus, see Figure 2. The frames close to frame i will have bigger weights and frames further away from the i:th frame will have smaller weights. The mathematical formulation for this can be given such as

$$\text{DG}^t_{(k,j)} = \frac{1}{\sum_t w_t} \sum_{t=i-\lfloor K/2 \rfloor}^{i+\lfloor K/2 \rfloor} w_t \cdot \text{noisy}^t_{(k,j)} \tag{3}$$

where DG stands for Denoised Gaussian and

$$w_t = \frac{1}{\sigma_{gauss}\sqrt{2\pi}} \exp\left(-\frac{(t-i)^2}{2\sigma^2_{gauss}}\right) \tag{4}$$

Observe that a similar problem arises with this method near the edge frames. However, we solve it similarly as mentioned in Section 3.1. We only average over as many frames as possible on each side of the frame in focus.

The local gaussian denoiser approach is expected to be better than the local averaging denoiser approach since closer frames will look more alike while frames further away in the time domain will look less similar.

3.3 Non-local Means Denoiser in Spatial Domain

As mentioned in Section 1 non-local means takes advantage of the high redundancy on a natural image [2]. That is, in a natural image, there will be many similar regions at different positions in the same image (see Figure 3). These similar regions can be combined using non-local means to reduce the noise while keeping the sharpness of the image. Observe that similar regions do not need to be close to each other. They can be far apart, hence the name "non-local". For example, in Figure 3 the patch p is very similar to patch q_1 and q_2. Therefore, these two should have higher weights for denoising. On the other hand, q_3, which is very different from p, should have minimal weight (approximately 0).

Let us now take a look at the mathematical formulation of the non-local means. The information given below is mainly from [2]. Let us assume that we have a discrete noisy image $v = \{v(i)|i \in I\}$ where I contains all the pixels in the image and i is the current pixel. Then the non-local means denoised pixel i for the noisy image is denoted with $NL(v)(i)$ and given by

$$NL(v)(i) = \sum_{j \in I} w(i,j)v(j). \tag{5}$$

where $w(i,j)$ are the weights and depends on the similarity between the area around pixel i and j. Furthermore, $v(j)$ is the the noisy pixel j. Observe here that $j \in I$. That is we compare i with all the other pixels in the image. The weights are given by

$$w(i,j) = \frac{1}{Z(i)} \exp\left(-\frac{||v(\mathcal{N}_i) - v(\mathcal{N}_j)||^2_{2,a}}{h^2}\right) \tag{6}$$

where $\mathcal{N}_i$ is a neighborhood subset of I. This is to prevent the high complexity of going through all the pixels in the image as mentioned in [3]. By limiting ourselves to a subset of neighborhoods the complexity decreases significantly. Furthermore, h controls the decay of the weights and $Z(i)$ is

$$Z(i) = \sum_j \exp\left(-\frac{||v(\mathcal{N}_i) - v(\mathcal{N}_j)||^2_{2,a}}{h^2}\right) \tag{7}$$

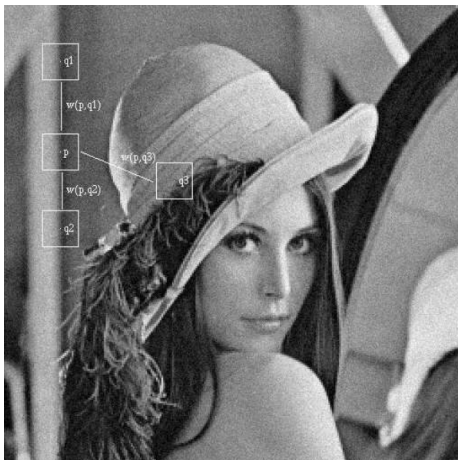


Fig. 3: Non-local means representation. Image taken from [2].

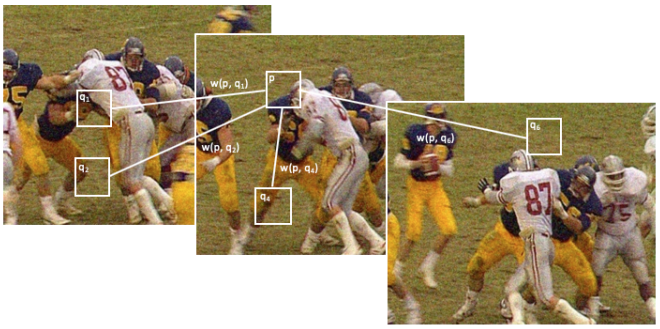


Fig. 4: Representation of non-local means in spatial and temporal domain.

In the non-local means with only spatial denoiser, we use this image denoising on each frame separately. That is, there is no inter-frame information used for denoising. In Section 3.4 we will discuss how to extend this to use inter-frame information.

For implementing the non-local means denoisers, in this project, we used the `denoise_nl_means()` function from the `skimage.restoration` library in Python [9]. Even though we could implement this function ourselves, we abstained since the function from the `skimage.restoration` package works far faster and is more reliable than any implementation we could come up with in the given time constraint.

3.4 Non-local Means Denoiser in Spatial and Temporal Domain

The non-local means denoiser in the spatial and temporal domain works similarly to the non-local means denoiser in only the spatial domain. The main difference is that it searches for similar patches not only in the same frame but in other frames as well, see Figure 4. Due to frames close in time often being very similar, the number of similar patches increases by searching in a neighborhood that spans over the temporal domain as well.

The weights $w(i,j)$ are determined in the exact same way as in Section 3.3, Equation 6 and 7.

Again, let v denote a noisy frame, and i denote the current pixel. Also let J contain all pixels in K consecutive frames, centered around the current frame. The denoised pixel i of the noisy image is then given as

$$NL_{\text{st}}(v)(i) = \sum_{j \in J} w(i,j)v(j), \tag{8}$$

where NL_{st} denotes the denoised frame using the non-local means with search neighborhood spanning over both the spatial and the temporal domain.

As for the local averaging denoiser and the local gaussian denoiser, the first few and the last few frames will pose an issue since the neighborhood in the temporal domain will span outside of the video. This is handled in the exact same way for this methods as for the previous methods, i.e. only using as many frames as possible near the edges.

Another difference from the non-local means in only the spatial domain is that we could not use the func-

tion from `skimage.restoration` directly in the non-local means in both the temporal and the spatial domain. Instead we needed to separate each color channel, denoise each color channel separately using a search area spanning over multiple frames, and then fuse the color channels. By doing it this way, we could still use the function from the `skimage.restoration` package.

3.5 Peak Signal to Noise Ratio

In this project we mainly use the peak signal to noise ratio (PSNR) to quantitatively compare the denoised videos. The PSNR per frame is given by [10]

$$PSNR_{\text{frame}} = 10\log_{10}(\frac{\max I_{GT}^2}{MSE}) \tag{9}$$

where

$$MSE = \frac{1}{3mn}\sum_{c=1}^{3}\sum_{i=1}^{m}\sum_{j=1}^{n}[I_{GT}(i,j,c) - I_d(i,j,c)]^2. \tag{10}$$

Here I_d denotes the denoised image, I_{GT} denotes the ground truth (original) image. Observe that we have three summations. These summations from left to right are because we have three color channels, m pixels vertically and n pixels horizontally. I.e., we compare each pixel in each color channel with the original value and sum their squared difference. We want this summation to be as small as possible. Because when MSE is small, the PSNR is large, and high PSNR is correlated with high similarity. Moreover, the PSNR for the entire video is calculated as the mean of the PSNRs for each frame.

4 Results, Analysis and Evaluation

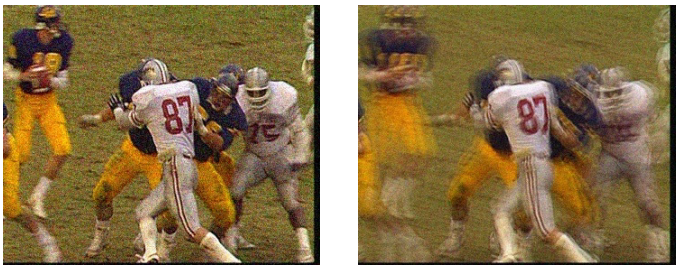
Each of our implemented methods is evaluated using different parameters in the following section. Furthermore, they have been compared to each other and the VBM3D method. All results presented here can be reconstructed by using the code given in [11].

4.1 Comparing parameters

4.1.1 Local Averaging Denoiser

The main parameter that can be changed in the local averaging denoiser is the number of frames to average. The results of changing this parameter was tested by averaging 3, 5, or 7 frames for three different noise levels, $\sigma = 0.05$, $\sigma = 0.1$ and $\sigma = 0.15$ on three different videos. The results are listed in Table 1. The best result for each noise level σ is shown in green (this will be used in the following tables as well). To be able to visually compare the results, one example frame can be seen in Figure 5.

From both the PSNRs and the example frame it is clear that the averaging algorithm does not perform very well in improving the quality of the video. Some noise is removed, but the consecutive frames are too different (contrary to thought), which makes the artifacts from movement in the scene very large.



(a) Noisy frame (b) $K = 3$

(c) $K = 5$ (d) $K = 7$

Fig. 5: An example frame from average denoising, using a noise level of $\sigma = 0.1$.

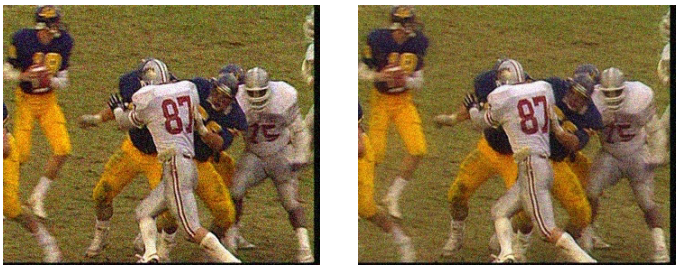
4.1.2 Local Gaussian Denoiser

For the Gaussian denoiser, two parameters can be changed; the number of frames included in the averaging and the standard deviation of the Gaussian kernel. The effects of these parameters were tested in the same way as the number of frames for the averaging algorithm, using three different noise levels and three different videos. The Gaussian denoiser was first tested by including five frames in the averaging for three different standard deviations of the Gaussian kernel, namely $\sigma_{\text{Gauss}} = 0.5$, 1.0, and 5.0. The results from these tests are presented in Table 2. It was then tested with 3, 5, or 7 frames included in the averaging, with the standard deviation of the Gaussian distribution fixed at 1.0. The results are presented in Table 3. An example frame for the local Gaussian denoiser is presented in Figure 6.

The results show that for higher noise levels, it is beneficial to increase the standard deviation of the Gaussian weights and increase the number of frames included in the averaging. However, it is more beneficial for lower noise levels to have a small standard deviation and a few frames included in the averaging. The reason for this is that increasing the number of frames or the standard deviation of the Gaussian weights reduces noise more but also increases

TABLE 1: PSNRs of the average denoising method averaging different number of frames.

# of frames being averaged		3	5	7
football	σ= 0.05	22.46 dB	21.26 dB	20.59 dB
	σ= 0.1	20.83 dB	20.38 dB	20.01 dB
	σ= 0.15	19.16 dB	19.34 dB	19.25 dB
crew	σ= 0.05	27.27 dB	26.65 dB	25.86 dB
	σ= 0.1	23.61 dB	24.32 dB	24.29 dB
	σ= 0.15	20.95 dB	22.23 dB	22.66 dB
city	σ= 0.05	26.03 dB	24.92 dB	24.15 dB
	σ= 0.1	22.98 dB	23.24 dB	23.06 dB
	σ= 0.15	20.41 dB	21.42 dB	21.73 dB



(a) Noisy frame (b) $K = 5$, $\sigma_{\text{Gauss}} = 0.5$

(c) $K = 3$, $\sigma_{\text{Gauss}} = 1$ (d) $K = 5$, $\sigma_{\text{Gauss}} = 1$

Fig. 6: An example frame from average denoising with Gaussian distributed weights, using a noise level of $\sigma = 0.1$ and parameters specified below each image.

the motion artifacts in the image, which is beneficial for high noise levels but not for smaller noise levels.

4.1.3 Non-local Means Denoiser in Spatial Domain

The non-local means method where the search area only contains patches within the same frame was tested using different patch sizes and different patch distances. The denoising method was first tested with a fixed patch size of 5 but different patch distances, see Table 4. The method was also tested for a fixed patch distance of 5, but for different patch sizes, see Table 5. An example frame with some different parameters can be seen in Figure 7.

From the PSNRs, it is possible to tell that increasing the patch distance improves the PSNRs for the videos with high noise levels but decreases the PSNR for low noise levels. Increasing the patch distance should remove more noise, but also blur the video more than using smaller patch distances which explains why using a greater patch distance is beneficial for videos with a lot of noise but less beneficial for videos with less noise.

TABLE 2: PSNR of averaging with Gaussian distributed weights using different standard deviations for the Gaussian distribution

# of frames		5	5	5
Std of Gaussian distribution		0.5	1.0	5.0
football	σ= 0.05	26.73 dB	23.50 dB	21.36 dB
	σ= 0.1	21.74 dB	21.72 dB	20.46 dB
	σ= 0.15	18.61 dB	19.94 dB	19.40 dB
crew	σ= 0.05	27.63 dB	28.08 dB	26.75 dB
	σ= 0.1	22.01 dB	24.33 dB	24.38 dB
	σ= 0.15	18.80 dB	21.62 dB	22.26 dB
city	σ= 0.05	27.49 dB	26.87 dB	25.03 dB
	σ= 0.1	21.81 dB	23.73 dB	23.31 dB
	σ= 0.15	18.43 dB	21.11 dB	21.47 dB

TABLE 3: PSNR of averaging with Gaussian distributed weights including different number of frames in the averaging.

# of frames		3	5	7
Std of Gaussian distribution		1.0	1.0	1.0
football	σ= 0.05	23.77 dB	23.50 dB	23.48 dB
	σ= 0.1	21.61 dB	21.71 dB	21.73 dB
	σ= 0.15	19.59 dB	19.94 dB	19.97 dB
crew	σ= 0.05	27.94 dB	28.08 dB	28.09 dB
	σ= 0.1	23.76 dB	24.33 dB	24.37 dB
	σ= 0.15	20.91 dB	21.62 dB	21.68 dB
city	σ= 0.05	26.95 dB	26.89 dB	26.86 dB
	σ= 0.1	23.28 dB	23.73 dB	23.76 dB
	σ= 0.15	20.45 dB	21.10 dB	21.16 dB

TABLE 4: PSNRs of spatial non-local means denoiser using different patch distances.

Patch size		5	5	5
Patch distance		3	5	7
football	σ= 0.05	31.83 dB	31.01 dB	30.41 dB
	σ= 0.1	25.76 dB	27.52 dB	28.19 dB
	σ= 0.15	17.59 dB	18.13 dB	18.66 dB
crew	σ= 0.05	32.13 dB	31.03 dB	30.28 dB
	σ= 0.1	26.09 dB	28.01 dB	28.71 dB
	σ= 0.15	17.69 dB	18.29 dB	18.85 dB
city	σ= 0.05	29.44 dB	28.35 dB	27.68 dB
	σ= 0.1	24.30 dB	25.89 dB	26.60 dB
	σ= 0.15	16.86 dB	17.12 dB	17.40 dB

The PSNRs also shows that the optimal patch size for $\sigma = 0.05$ is at 5, but for the higher noise levels the optimal patch size is 3. With a larger patch size, the neighborhood around one patch has to be more similar for the weights to be big. If there are enough similar patches in the search area this should improve the denoising, but if there are not enough similar patches, it will instead make the image blurry because a lot of patches will have a similar weight.

4.1.4 Non-local Means Denoiser in Spatial and Temporal Domain

Due to the non-local means in the spatial and temporal domain being very computational inefficient and taking a very long time to run, it was only tested for one noise level and only changing the neighborhood in the temporal domain. The other parameters possible to change are the same as for the non-local means in the spatial domain and the results of changing these parameters are thus expected to have a similar result. The results from the tests for the non-local means algorithm in the spatial and the temporal

TABLE 5: PSNRs of spatial non-local means denoiser using different patch sizes.

Patch size		3	5	7
Patch distance		5	5	5
football	σ= 0.05	30.33 dB	31.01 dB	30.92 dB
	σ= 0.1	28.14 dB	27.52 dB	26.37 dB
	σ= 0.15	21.47 dB	18.13 dB	17.17 dB
crew	σ= 0.05	30.60 dB	31.03 dB	30.90 dB
	σ= 0.1	28.60 dB	28.01 dB	26.76 dB
	σ= 0.15	21.76 dB	18.29 dB	17.23 dB
city	σ= 0.05	28.29 dB	28.35 dB	28.16 dB
	σ= 0.1	26.97 dB	25.89 dB	24.48 dB
	σ= 0.15	20.32 dB	17.12 dB	16.61 dB

(a) Noisy frame

(b) PD = 3, PS = 5

(c) PD = 5, PS = 3

(d) PD = 5, PD = 5

Fig. 7: An example frame from non-local means denoising in the spatial domain, using a noise level of $\sigma = 0.1$.

(a) Noisy frame

(b) TN = 3

(c) TN = 5

(d) TN = 7

Fig. 8: An example frame from non-local means denoising in the spatial and temporal domain, using a noise level of $\sigma = 0.1$.

TABLE 6: PSNRs of spatial and temporal non-local means denoiser using different neighborhood sizes in the temporal domain.

Neighborhood size		3	5	7
football	σ= 0.1	27.17 dB	26.55 dB	25.81 dB
crew	σ= 0.1	28.28 dB	28.04 dB	27.80 dB
city	σ= 0.1	26.04 dB	25.90 dB	25.89 dB

domain are presented in Table 6 and an example frame can be seen in Figure 8.

In this case, increasing the neighborhood size decreases the PSNR. As for the patch distances for the non-local means in only the spatial domain, increasing the neighborhood in the temporal domain reduces more noise, but also blurs the video more, which is a possible explanation to the decrease in PSNR when the neighborhood gets larger.

4.2 Comparing Methods

To compare the different methods to each other, each method's highest PSNR for each video and each level of noise are summarized in Table 7. In the table, results using the Video BM3D method for denoising the videos have also been included for comparison.

In general, the two averaging algorithms show a similar result in terms of PSNR. The averaging method with weights performs better at low noise levels, but the averaging method without weights performs better for two out of the three tested videos for higher noise levels. However, both of the methods perform poorly and introduce many movement artifacts. The two variants of the non-local means method perform a lot better than the averaging methods. A lot of the noise is removed without introducing artifacts from the movements. The major issue with these methods is that some areas become blurry, e.g., the grass in the football video. This happens to a larger extent for the non-local means in both the spatial and the temporal domain than for the non-local means in only the spatial domain. Even though the non-local means methods perform a lot better than the averaging algorithms, they still can not compete with the state-of-the-art VBM3D method, which gives a much higher PSNR for the videos.

5 Discussion and Conclusion

As expected the averaging methods did not perform as well as the non-local means methods. This was true for images as well, and was not surprising. One thing that did surprise us about the result is that the non-local means in the spatial domain performed better than the non-local means in the temporal and the spatial domain. However, by looking at the example frames in Figure 7 and Figure 8 it is possible to see that the background after the non-local means in the spatial and temporal domain is more blurry than for the non-local means in only the spatial domain, which is one explanation for the lower PSNRs. Another aspect is that the parameters for the non-local means in the temporal and spatial domain might be worse than the parameters for the non-local means in the spatial domain since the first one was not tested with as many different sets of parameters.

5.1 Limitations

One limitation of this study is the small number of tests of the non-local means algorithm, which searches for patches in both temporal and the spatial domain. To improve the study results, the method should be tested with other parameters and other noise levels, but due to the computational complexity, long-running times, and the project's time limit, this was not possible in our case. So, the fundamental limitation is that the non-local means in the spatial and temporal domain is too slow.

TABLE 7: The PSNRs for all of the tested methods, using the optimal parameters for each video and noise level.

		Averaging	Gaussian	NLM spatial	NLM spatial and temporal	VBM3D
football	$\sigma = 0.05$	22.46 dB	26.73 dB	31.83 dB	-	35.24 dB
	$\sigma = 0.1$	20.83 dB	21.74 dB	28.19 dB	27.17 dB	31.25 dB
	$\sigma = 0.15$	19.34 dB	19.97 dB	21.47 dB	-	28.89 dB
crew	$\sigma = 0.05$	27.27 dB	28.09 dB	32.13 dB	-	35.80 dB
	$\sigma = 0.1$	24.32 dB	24.38 dB	28.71 dB	28.28 dB	32.29 dB
	$\sigma = 0.15$	22.66 dB	22.26 dB	21.76 dB	-	30.23 dB
city	$\sigma = 0.05$	26.03 dB	27.49 dB	29.44 dB	-	34.87 dB
	$\sigma = 0.1$	23.24 dB	23.76 dB	26.97 dB	26.04 dB	31.20 dB
	$\sigma = 0.15$	21.73 dB	21.47 dB	20.32 dB	-	29.05 dB

5.2 Future work

As mentioned a couple of times already, our approach is computationally heavy. So one improvement to our algorithm could be to make it more efficient so it can run at least as fast as the VBM3D algorithm, which is both faster and gives better results.

Another thing one can look into is to test the non-local means in the spatial and temporal domain on videos with a higher frame rate. This should improve the methods that use information from multiple frames since successive frames would be more similar at a higher frame rate.

5.3 Conclusion

Since the VBM3D method performs so much better than the less sophisticated algorithms we implemented in this project (and even with suggested improvements in Section 5.2 made to the proposed methods) it is not likely that the methods will be able to compare to the VBM3D, the conclusion is that it does not seem promising to improve the less sophisticated methods used in this project further.

REFERENCES

[1] A. E. Ilesanmi and T. O. Ilesanmi, "Methods for image denoising using convolutional neural network: a review," *Complex & Intelligent Systems*, vol. 7, no. 5, pp. 2179–2198, 2021.

[2] A. Buades, B. Coll, and J. M. Morel, "On image denoising methods," *CMLA Preprint*, vol. 5, pp. 19–26, 2004.

[3] M. Mahmoudi and G. Sapiro, "Fast image and video denoising via nonlocal means of similar neighborhoods," *IEEE signal processing letters*, vol. 12, no. 12, pp. 839–842, 2005.

[4] G. Wetzstein, "Lecture notes in Digital Photography II, The Image Processing Pipeline," January 2022.

[5] K. Dabov, A. Foi, and K. Egiazarian, "Video denoising by sparse 3d transform-domain collaborative filtering," in *2007 15th European Signal Processing Conference*. IEEE, 2007, pp. 145–149.

[6] T. Ehret and P. Arias, "Implementation of the vbm3d video denoising method and some variants," *arXiv preprint arXiv:2001.01802*, 2020.

[7] X. Xu, M. Li, W. Sun, and M.-H. Yang, "Learning spatial and spatio-temporal pixel aggregations for image and video denoising," *IEEE Transactions on Image Processing*, vol. 29, pp. 7153–7165, 2020.

[8] M. Maggioni and A. Foi, *V-BM4D software for video denoising*, ver. 1.0, Tampere University of Technology, 2014. [Online]. Available: https://webpages.tuni.fi/foi/GCF-BM3D/

[9] "scikit-image: image processing in python, module: restoration," Accessed Mar. 2, 2022 [Online]. [Online]. Available: https://scikit-image.org/docs/stable/api/skimage.restoration.html#skimage.restoration.denoise_nl_means

[10] M. Nishimura, "Problem Session 2," January 2022, zoom lecture.

[11] E. Vardar and E. Byman, *EE367_project*, Stanford University, 2022. [Online]. Available: https://github.com/Vardar98/EE367_project