Automated Estimation, Reduction, and Quality Assessment of Video Noise from Different Sources

Meisam Rakhshanfar

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy (Electrical Engineering) at

Concordia University

Montréal, Québec, Canada

April 2016

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Mr. Meisam Rakhshanfar**

Entitled: **Automated Estimation, Reduction, and Quality Assessment of Video Noise from Different Sources**

and submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy (Electrical Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Chair
Dr. Adam Krzyzak

_____ External Examiner
Dr. Eric Dubois

_____ Examiner
Dr. Thomas Fevens

_____ Examiner
Dr. Yousef R. Shayan

_____ Examiner
Dr. Wei-Ping Zhu

_____ Supervisor
Dr. Maria A. Amer

Approved by _____
Name of Dept. Chair, Chair
Department of Electrical and Computer Engineering

_____ 2016

_____
Dean Of ENCS, Dean
Faculty of Engineering and Computer Science

Abstract

Automated Estimation, Reduction, and Quality Assessment of Video Noise from Different Sources

Meisam Rakhshanfar, Ph.D.

Concordia University, 2016

Estimating and removing noise from video signals is important to increase either the visual quality of video signals or the performance of video processing algorithms such as compression or segmentation where noise estimation or reduction is a pre-processing step. To estimate and remove noise, effective methods use both spatial and temporal information to increase the reliability of signal extraction from noise. The objective of this thesis is to introduce a video system having three novel techniques to estimate and reduce video noise from different sources, both effectively and efficiently and assess video quality without considering a reference non-noisy video. The first (intensity-variances based homogeneity classification) technique estimates visual noise of different types in images and video signals. The noise can be white Gaussian noise, mixed Poissonian-Gaussian (signal-dependent white) noise, or processed (frequency-dependent) noise. The method is based on the classification of intensity-variances of signal patches in order to find homogeneous regions that best represent the noise signal in the input signal. The method assumes that noise is signal-independent in each intensity class. To find homogeneous regions, the method works on the downsampled input image and divides it into patches. Each patch is assigned to an intensity class, whereas outlier patches are rejected. Then the most homogeneous cluster is selected and its noise variance is considered as the peak of noise variance. To account for processed noise, we estimate the degree of spatial correlation. To account for temporal noise variations a stabilization process is proposed. We show that the proposed method competes related state-of-the-art in noise estimation.

The second technique provides solutions to remove real-world camera noise such as signal-independent, signal-dependent noise, and frequency-dependent noise. Firstly, we propose a noise

equalization method in intensity and frequency domain which enables a white Gaussian noise filter to handle real noise. Our experiments confirm the quality improvement under real noise while white Gaussian noise filter is used with our equalization method. Secondly, we propose a band-limited time-space video denoiser which reduces video noise of different types. This denoiser consists of: 1) intensity-domain noise equalization to account for signal dependency, 2) band-limited anti-blocking time-domain filtering of current frame using motion-compensated previous and subsequent frames, 3) spatial filtering combined with noise frequency equalizer to remove residual noise left from temporal filtering, and 4) intensity de-equalization to invert the first step. To decrease the chance of motion blur, temporal weights are calculated using two levels of error estimation; coarse (block-level) and fine (pixel-level). We correct the erroneous motion vectors by creating a homography from reliable motion vectors. To eliminate blockiness in block-based temporal filter, we propose three ideas: interpolation of block-level error, a band-limited filtering by subtracting the *back-signal* beforehand, and two-band motion compensation. The proposed time-space filter is parallelizable to be significantly accelerated by GPU. We show that the proposed method competes related state-of-the-art in video denoising.

The third (sparsity and dominant orientation quality index) technique is a new method to assess the quality of the denoised video frames without a reference (clean frames). In many image and video applications, a quantitative measure of image content, noise, and blur is required to facilitate quality assessment, when the ground-truth is not available. We propose a fast method to find the dominant orientation of image patches, which is used to decompose them into singular values. Combining singular values with the sparsity of the patch in the transform domain, we measure the possible image content and noise of the patches and of the whole image. To measure the effect of noise accurately, our method takes both low and high textured patches into account. Before analyzing the patches, we apply a shrinkage in the transform domain to increase the contrast of genuine image structure. We show that the proposed method is useful to select parameters of denoising algorithms automatically in different noise scenarios such as white Gaussian and real noise. Our objective and subjective results confirm the correspondence between the measured quality and the ground-truth and proposed method rivals related state-of-the-art approaches.

Acknowledgments

First and foremost I would like to express my deepest gratitude to my supervisor, Dr. Maria A. Amer, who has supported me throughout this research journey with her patience and provided me the flexibility of exploring various domains and ideas. I appreciate her knowledge and skills especially her experience with real industrial problems and for giving me the opportunity to gain experience from real challenges. I am thankful to her also for assisting in writing papers, patents, and this thesis.

I would like to thank the members of my doctoral and thesis committee Dr. Yousef R. Shayan, Dr. Thomas Fevens, and Dr. M. Omair Ahmad for their insightful comments which guided my research from various perspectives. My thanks go to the external examiner Dr. Eric Dubois for his availability and input. My thanks also go to Dr. Wei-Ping Zhu, the member of thesis defence committee, for his input and presence.

I would like to express my gratitude to TandemLaunch Inc. and wrnch Inc. for investing in my ideas and providing me equipment, software and test data. I appreciate the time and finance that they spent to make this thesis possible and industrial.

Finally, I take this opportunity to express the profound gratitude to my parents and family. They were always supporting me spiritually and materially and encouraging me with their best wishes.

Contents

List of Figures

List of Tables

List of abbreviations

BLAB ..... Band-limited anti-blocking filter.
BLTST ..... Band-limited time-space filter.
CCD ..... Charge-coupled device.
DCT ..... Discrete cosine transform.
DFT ..... Discrete Fourier transform.
GPU ..... Graphics processing unit.
LF ..... Low frequency.
HDR ..... High dynamic range.
HF ..... High frequency.
IVHC ..... Intensity-variance homogeneity classification.
JPEG ..... Joint photographic experts group.
KROCC ..... Kendall rank order correlation coefficient.
LMMSE ..... Linear minimum mean squared error.
MAD ..... Median absolute deviation.
MAE ..... Mean of absolute error.
MSE ..... Mean squared error.
MV ..... Motion vector.
NLF ..... Noise level function.
NR-IQA ..... No-reference image quality assessment.
PSD ..... Power spectrum density.
PSNR ..... Peak signal to noise ratio.
RTF ..... Recursive temporal filtering.
SCN ..... Spatially correlated noise.
SDSCN ..... Signal-dependent spatially correlated noise.
SDQI ..... Sparsity and dominant orientation quality index.
SDWN ..... Signal-dependent white noise.
SISCN ..... Signal-independent spatially correlated noise.
SROCC ..... Spearman rank order correlation coefficient.
STD ..... Standard deviation.
STF ..... Symmetric temporal filtering.
SVD ..... Singular value decomposition.
TMSP ..... Temporal mean squared power.
TSS ..... Three step search.
VST ..... Variance stabilization transform.
WGN ..... White Gaussian noise.

List of Symbols

i, j, k, l, m, n	General integer variables.
t	Time or frame number.
$\boldsymbol{c}^e, \boldsymbol{c}^r, \boldsymbol{c}^q$	Constants of estimation, reduction and quality assessment algorithm.
$\boldsymbol{T}^e, \boldsymbol{T}^r, \boldsymbol{T}^q$	Threshold constants in estimation, reduction and NR-IQA algorithm.
(I, F, F_t)	Noisy image, frame, and frame at time t.
I_{org}	Noise-free image or ground-truth.
$\mathbf{P}(\cdot), \mathbf{O}(\cdot)$	Processing in the capturing pipeline and estimation stabilizer process.
$(\boldsymbol{n}_d, \boldsymbol{n}_g, \boldsymbol{n}_q)$	Signal-dependent, signal-independent, and quantization noise.
$(\boldsymbol{n}_o, \boldsymbol{n}_p)$	Observed white and potentially processed noise.
$(\boldsymbol{n}_p, \boldsymbol{n}_\gamma)$	Processed signal-independent noise and distortion
$(\sigma_g^2, \sigma_d^2(I))$	Variance of signal-independent, signal-dependent noise.
$(\sigma_o^2, \sigma_p^2, \sigma_\gamma^2)$	Maximum variance of white, processed, and distortion noise.
$\boldsymbol{a}_l, \boldsymbol{a}_{max}$	Slope of NLF for l^{th} intensity class and its maximum possible value.
γ, γ_{max}	Processing degree for scales 0 and 1.
$\hat{\gamma}_0, \hat{\gamma}_1$	Processing degree and maximum value for moderately processed noise.
$L_l, \tilde{H}_{th}(l)$	l^{th} intensity class and its homogeneity threshold.
(R_e, W_e)	Image downscaling factor and downscaled patch size for noise estimator.
$(B_i, \tilde{B}_i)$	Original and downscaled image patch with size of $[R_e W_e]^2$ and W_e^2.
$(\sigma^2(\tilde{B}_i), \mu(\tilde{B}_i)$	Mean and variance of downscaled patch.
$\ddot{\Phi}(l,k), \Phi(l,k)$	k^{th} cluster of l^{th} class before and after outlier removal.
$\Gamma(l,k)$	cluster confidence value.
$\tilde{I}$	Downscaled image by factor of R_e.
$M_{\mathbf{I}}$	Number of intensity class.
$\tilde{H}_i$	Homogeneity value of downscaled patch.
$\hat{\Phi}$	Noise representative selected cluster.
$(\hat{\Omega}, \Omega)$	Estimated noise level function before and after regression.
$\bar{\mathbf{E}}_{\mathbf{LF}}, \bar{\mathbf{E}}_{\mathbf{HF}}, \mathbf{E}_{\mathbf{r}}$	Power of low-pass and high-pass filtered cluster and their ratio.
ω_j	j^{th} weight assigned to specific cluster.
$\zeta_{t-1,t}$	Scene change value at time t.
τ_i	median of absolute deviation for i^{th} patch.
$N_p\{\Phi(l,k)\}, N_p\{I\}$	Number of patches in $\Phi(l,k)$ and image I.
$\mu(l,k), \sigma^2(l,k)$	Mean of mean and mean of variance of $\Phi(l,k)$ patches.
$\ddot{\mu}(l,k), \ddot{\sigma}^2(l,k)$	Variance of mean and variance of variance of $\Phi(l,k)$ patches.
$\zeta_{t-1,t}$	Scene change between frame at time t and previous frame $t-1$.
I^l, F_t^l	Image and frame at l^{th} scale (F_t downscaled by factor of 2^l) and time t.
$\mathcal{J}^l$	HF content that is equal to $I^l - I^{l+1}$.

g^l ..... Restoration factor of multi-scale filtering for l^{th} scale.
$\nu(I)$ ..... Noise equalizer in intensity domain.
R ..... Temporal radius of recursive and symmetric temporal filter.
$\mathcal{B}_i, \vec{\mathcal{B}}_{m,i}$ ..... i^{th} block of reference frame and motion compensated from F_{t-m}.
$\mathbf{V}_{t,t+m}$ ..... Motion vector between frame F_t and F_{t+m}.
$\mathcal{G}_t$ ..... Output of temporal filter at the time (frame number) t.
W_r, W_q ..... Effective size of Coarse motion error detection and its quarter $W_q = \frac{W_r}{4}$.
$\mathbf{MC}(F, \mathbf{V})$ ..... Motion compensated frame F by vector $\mathbf{V}$.
$D_t, \vec{D}_t$ ..... Approximation subtracted frame and its motion compensated.
$\delta_m, \bar{\delta}_m, \hat{\delta}_m$ ..... Temporal error based on power and mean of block, and pixel.
$\mathbf{w}_m$ ..... m^{th} weight matrix in temporal filtering.
λ ..... Noise reduction factor by temporal filtering.
$\mathbf{b}_t, \hat{\mathbf{b}}_t$ ..... Back-signal at original size and downscaled by 4 scale.
$\mathcal{D}_t, \vec{\mathcal{D}}_t$ ..... Back-signal extracted frame at time t and its motion-compensated.
$S^r_{x,y}$ ..... Output of proposed bilateral filter with radius r at location (x, y).
r ..... r Spatial radius in spatial filtering.
$\mathbf{G}_x, \mathbf{G}_y, \mathbf{G}$ ..... Image gradient horizontally, vertically, and both (complex format).
N_q, W_q ..... Size of patch and block for DFT and SVD.
$\dot{\mathbf{x}}, \alpha^c, \mathbf{D}^c$ ..... Signal, coefficients, and dictionary related to 2D Transform.
$\hat{\alpha}^c$ ..... Sorted coefficients of 2-D Transform.
β ..... SVD sparsity value.
s_1, s_2 ..... Singular values along the dominant and perpendicular direction.
$\tilde{s}_1, \tilde{s}_2$ ..... Singular values after the shrinkage.
$\theta, \tilde{\theta}$ ..... Dominant direction of patch without and with shrinkage.
ψ ..... Probability of signal presence in the patch.
$\mathbf{Q}(I)$ ..... Measured quality index for the image I.
ξ^{-1} ..... Inverse sparsity of the patch.
ϕ_{gtm} ..... Quality of image according to Ground-truth.
$I_{\mathbf{opt}}$ ..... Output of NR-IQA based denoiser.
O_q ..... Overlapping size in the DFT shrinkage.
$\mathcal{N}_r$ ..... Image height or number of rows.
U_r, U_i ..... Orthonormal matrix of SVD.

Chapter 1

Introduction

1.1 Noise sources

Even most modern video capturing devices introduce random noise and video denoising is still an important feature in many video systems. Video signals may originate from different capturing sources such as consumer electronics devices (e.g., TV, cinema, or mobile phones), medical devices, or remote-sensing devices. Different video cameras produce different types of noise. A widely used sensor in video camera, specially in consumer electronics is the CCD (Charge-coupled device) sensor. Figure 1.1 illustrates a typical CCD camera pipeline. Noise is mostly added to the image [1–6] in the sensor layer mostly due to lack of photons. Noise at the sensor layer is modeled as summation of different types of noise (e.g., fixed pattern noise, dark current noise, shot noise, amplifier noise, and sampling noise) see [6]. Captured image at the sensor-layer becomes processed through a capturing pipeline to be displayed or stored. Noise characteristics change through the capturing pipeline as the noise becomes spatially correlated and compressed.

Noise at the sensor layer (before any digital processing) is additive and signal-dependent, however, when the dependency to signal is minor, noise can be modelled as additive and white Gaussian noise (WGN). Many noise reduction methods propose solutions to remove WGN in videos [7–19] and images [20–23]. For reduction, characteristics of noise should be estimated. Many methods are developed to estimate the variance of WGN in the images [24–35]. Noise in this and related work is additive.

Video or image noise after processing becomes spatially correlated (frequency-dependent), however, when the spatial correlation is minor noise can be assumed as white (equally distributed in all frequencies).

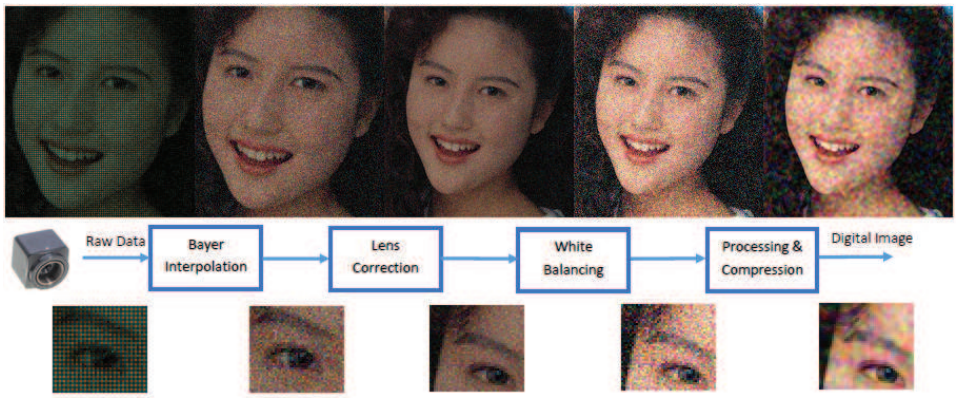

Figure 1.1: Typical digital camera imaging pipeline.

1.2 Applications

The most important application of video noise reduction is improving the visual quality. CCD cameras are integrated into mobile devices such as phones, tablets and laptops. Due to cost and size constraints, small lens and low-quality cameras are often used which introduce unpleasant noise. Video noise reduction helps the user to have a better experience in watching video signals. It also improves the compression rate resulting in less storage or higher channel capacity. Video noise reduction can be used to improve still image quality as well. In low light conditions we can produce a single image with higher quality by taking several photos in burst mode and denoising them. Video noise estimation and reduction is also used in post-production not only for removing noise but also for renoising. In movie production, consecutive sequences are captured in different lighting conditions. To integrate those sequences noise should be completely removed and added again (renoise) with same level and property to keep the consistency and avoid quality discontinuity. The important role of noise estimator in renoising applications is extracting noise property from one source to be added to another denoised source. One new application of video denoising is in video rendering. In video rendering the synthetic 3-D scene is created based on geometry, viewpoint,

texture, lighting, and shading information of the virtual scene. In video rendering number of sample points (or iterations) defines the quality of image and less iteration produces more noise. This is a tedious task and it takes several hours to create a high quality and high-resolution frame. A solution to increase the speed is using less number of iterations but video denoising to improve the quality. One important application of no-reference image quality assessment (NR-IQA) is the quality control of output [36–42]. In the development process of a computer vision software, NR-IQA method can be used to quantify the quality of the output and if the quality is less than an expected value the control system can report a failure or initiate correction. Another important application of NR-IQA is in automatic setting of camera parameters. Digital cameras often constantly capture images and check their quality to find the optimal point. Focal length, ISO and shutter speed are the parameters that usually are automatically found. NR-IQA methods are used for this matter. Another application of NR-IQA is in the burst-mode image capture in the mobile phones, where the image with the highest quality along all burst images should be selected.

1.3 Motivation

Several video denoising approaches are known to restore videos that have been degraded by random noise. Recent advances in estimation and reduction of noise have achieved remarkable results, however, the simplicity of their noise source model makes them impractical for real video noise. Mostly, noise is assumed a) white Gaussian and b) accurately pre-estimated. When the noise is overestimated the chance of information loss (blur) increases. One essential objective of all video denoising methods is to prevent blur. When the properties of the noise deviate from whiteness, the efficiency of algorithms that are designed for white noise degrades. In practice, noise is often spatially-correlated or non-white. For methods that handle spatially correlated noise (SCN), input parameter should be tuned according to the video content to achieve desirable results. Data sampled at the sensor layer becomes processed in a long capturing pipeline. Demosaicing, color correction, filtering, and quantization are examples of processing in the capturing pipeline which change the noise properties. In addition to noise properties, in the development of a video denoising software,

computation cost and modularity should be considered. Computationally or memory-wise expensive methods are not practical. Hardware such as graphics processing unit (GPU) can be used to accelerate the heavy processing operations, however, algorithms should be highly *parallelizable* to benefit from this. In video processing applications a trade-off between performance (e.g., quality) and cost (e.g., speed) is desired. A modular system, i.e., a system with functionality independent modules, helps to adjust the performance-cost point by changing certain components or parameters. For instance motion estimation/compensation complexity can be adjusted according to the hardware/timing budget. Optical flow motion estimation are the pioneers algorithms but time consuming; on the other hand, block-matching approaches are fast but they introduce blocking artifacts (*blockiness*). The human visual system easily recognizes the blockiness and perceives it as non-natural content, thus, we aim at denoising algorithm with the least possible blockiness. Here, are the list of other problems and motivations to do this research.

1.4 Requirement for an effective video denoising system

Complete model: To handle a wide range of real types of noise we need to use a comprehensive noise model that can detect the noise characteristics such as power, type and non-uniformity model. Most existing denoising algorithms deal with grayscale video signals or propose the same processing procedure for chrominance channels. In real-world cases which the chrominance channels are usually subsampled, this is not efficient.

Automated framework: We require a framework that is fully automated. Conventional methods need manual intervention, such as defining or verifying the input values and formats or noise profiles. Many parameters should be estimated in order to have an accurate model of noise. In noise estimation, parameters such as the level of noise in all channels spatially and temporally, degree of spatial correlation in all channels, the model of noise variation over the intensities, scene change, frame replication, format of the input, and noise boundaries based on the metadata is estimated. Based on these estimates, the noise reduction is accomplished. This makes the framework much more complicated than the conventional noise reduction methods that estimate a single value for noise variance for grayscale channel using only spatial information.

Error sensitivity: Unlike the frameworks where a user selects or verifies the input parameters, the automated framework is not always reliable. We require a system that works automatically for all selected test dataset, however, the dataset is limited and there is possibility of estimation error which should be taken into account. Thus, by using a wide variation of test points we aim for a system that detects estimation error and compensate that.

Modularity: A modular framework, i.e., a framework with functionally independent modules, provides the feature to adjust the performance-cost point by changing certain components or parameters. For instance, many video denoising algorithm use a motion estimation and compensation engine that is independent to their main algorithm and can be adjusted according to the hardware and timing budget. Instead of one-piece sophisticated framework, we require a modular system with many components that each provide high quality and as needed they can be replaced by lower quality but faster equivalents. Our system design can provide reasonable variation of speed versus quality that can be tuned based on the hardware resources.

Quality assessment: Most of the denoising approaches use PSNR as the ground-truth to measure the quality and adjust algorithm parameters. We require both objective and subjective evaluation for parameter adjustment and quality assessments. A no-reference quality assessment method is required to provide results that correspond with human perception. We require objective measures at the first level of assessment, however, many undesired artifacts such as blocking, posterization, ringing cannot fully be detected by quantitative measures such as PSNR. Thus, we require to verify the results by subjective evaluation as well. Using a wide variation of datasets and extensive subjective and objective experiments our system should automatically find the optimal denoising point.

Reliability: The degree of reliability and effectiveness of the system should be proportional to number and diversity of test data that it can handle automatically. Ineffective frameworks handle special cases such as simple motion and noise. Real test data is required to make a system sufficiently effective. Unlike conventional denoising methods that are tuned for specific data and noise, we should consider special cases such as handling replicated frames, noise overestimation and HDR formats. Although it makes the system more complex and

slower, we should address two challenging HDR problems, high-precision and extreme non-uniformity of the noise.

Speed: Processing time is a factor that is not considered in most of the pioneer methods. For some of the state-of-the-art methods it takes several hours to process a one-minute high-resolution video clip. Parallel processing (e.g., GPU and vector processor) is an important feature to accelerate the processing. In addition to finding a reasonable speed-quality point algorithm-wise, we require a system with highest possible parallel units.

Quality control: In the development of a denoising software, having the control on the quality, speed, and memory is essential. In multi-platform implementation such as CPU-GPU implementation, we need to have a parity between platforms so they produce identical results. We have implemented test units that automatically test all sub-modules individually for both CPU and GPU and under three operating systems (Windows, Mac OS, and Linux) and reports implementation errors, memory speed and quality of output.

1.5 Summary of contributions

The contributions of this thesis are 1) an automated solution for video denoising that comprises estimation and reduction of three types of noise: additive white Gaussian noise, Poissonian-Gaussian (white signal-dependent) noise, and processed Poissonian-Gaussian (spatially correlated signal-dependent) noise, and 2) a method to assess the quality of the image using sparsity and dominant orientation of patches when no reference (clean) frame exists. So far we have published three papers [43–45] to video noise estimation, video noise reduction, and no-reference image quality assessment. Two journal papers (one on noise estimation [TIP-13994-2015] and one on quality assessment [TIP-14249-2015]) are under review. Three patents [46–48] were filed regarding to noise estimation, reduction, and quality assessment. The details of contribution for each method is provided in the each related chapter.

1.6 Research and system evolution

We have started our research considering simplified noise model additive and white Gaussian noise (WGN) and studied related state-of-the-art noise estimation methods. We selected the pioneers in providing practical solution and analyzed their flaws and improved them. However, when we were provided with real-noise data from industry (where the noise was signal-dependent and spatially correlated) we realized our method model is inaccurate. Consequently, we modified our framework and developed an intensity-variance based classification methods that uses connectivity of patches to estimate the noise. We developed that method to estimate the power of noise before and after processing and also the noise level function (NLF). When we analyzed more data, we realized our model is accurate for specific type of correlated noise. To solve this problem we developed a method to estimate the degree of processing (spatial correlation) of the noise in different image scales. As we processed more data, we faced more problems such as noise estimator could find only the noise level function below a certain baseline (which is estimated based on the target cluster). However, in special cases, noise level in certain intensities is higher than the baseline. The goal was to fix this problem without changing the efficiency of noise estimator in normal cases, which we were able to handle that. Another spatial case was sequences with two identical consecutive frames, which that by detecting identical frame and exclude them from estimation and reduction process. The HDR contents were also problematic, where the dependency of noise to signal is extreme and our model based on limited slope is not suitable. We have developed a forward and backward tone-mapping to balance the noise level function in this case.

For noise reduction we started the research with a state-of-the-art fast recursive temporal filter (RTF) under WGN and we attempted enhancing it by addressing its flaws. The first step was combining of block and pixel based error detection to address motion blur. The second step was developing a collaboration of symmetric temporal filtering (STF) and spatial filter since noise reduction by RTF is not uniform reduced temporally and spatially. The first implementation of spatial filter was based on partial differential equations, however, the results was not satisfactory which led us to implementing a dual-domain spatial filter. The speed and quality of implemented video

filter was satisfactory under WGN, however, under spatially correlated noise the artifacts were unpleasant. Three main artifacts were blocking, posterization, and ringing (i.e., frequency- domain artifacts). For solving the blocking we firstly used the idea of deblocking that is used in most of the video decoders. The problem with deblocking is that it leaves the stripes of filtering effect at the edges of blocks which was undesired. Thus, we proposed the two-band motion compensation. Although this solution eliminates most of blocking, we enhanced the visual quality by developing a band-limited temporal filtering in order to exclude the very low frequency (LF) content that is very likely to be signal. We used the same (excluding the very LF content from processing) idea in the spatial filtering to address the posterization. We solved the ringing effect by using two iteration spatial domain filtering. In the first we find the sharp results that has the ringing, in the second iteration we use the first iteration to steer (guide) the second iteration. In analyzing new video dataset we realized when there is a large translational motion and the resolution is high (e.g., 1920×1080) the results of motion estimation for some blocks are not reliable. However, the motion was translational and not complex to estimate. The reason was in pyramid (multi-resolution) motion estimation the error from the top of pyramid propagated to the bottom and creates many erroneous motion vectors. To address that we developed a method to find the reliable motion vectors and create a homography based on that and correct the motion faulty motion vectors. The challenges in homography creation were how to define reliable motion vectors and how to relate the blocks to different reliable motion vectors.

For testing the quality of denoised videos where the reference was not available we started using a NR-IQA to assess our quality. However, we realized the selected NR-IQA tends to select blurry results as higher quality. Therefore, we decided to develop a method for more reliable blind quality assessment. In developing a NR-IQA, we started with an entropy based idea that was working based on scattering the image structure and measuring the amount of entropy that is increased. In addition of being slow, the performance of this method degrades as the noise becomes more spatially correlated. Thus, developed a new method based on the sparsity and dominant direction of the patches which is better corresponds to human visual system compared to other methods.

In the development procedure of noise estimation, noise reduction and NR-IQA algorithms we faced the following main implementation issues.

Memory optimization: Memory consumption in computer video processing and vision algorithms is less taken into account. However, this should be considered in our application since the input video can be high-resolution (e.g., 4K ultra high definition) and the memory resource is limited. Assuming the processing is done in a 4 byte floating point with radius of 5 frames, for a 4K (3840×2160) color video we need more than one GigaByte of memory just to store temporal data. As an example, for the noise estimator we reduced the memory usage by finalizing the processing in the luma channel and reuse the same allocated memory for chroma channels.

GPU implementation : The GPU implementation is not as straightforward as CPU and the debugging is very challenging. The processing is done in parallel and in addition of dividing the algorithm into separate parallel units many parameters (e.g., cache size, number of local workers, number of global workers) should be investigated to reach the maximum speed. One of the challenging problems was implementing the motion estimation on the GPU. As an example to accelerate the motion estimation we experimented three different implementations. In the first idea, we assigned one thread per block to compute the block matching cost in parallel. The approach is easy to implement and debug. However, since the location of matching blocks are random, the memory access is not continuous and the performance decreases significantly. The more complex idea was assigning multiple threads per block to compute the block matching criteria. This improves the results, however, the acceleration rate is not desirable. To reach maximum bandwidth of continuous memory access, we developed even more complex idea that parallel threads compute the 3 cost for each block at the same time. Another problem was that some GPUs they cannot handle vector processing unless they are correctly aligned in the memory. Assuming there are 4 values that we want to add a constant to them with one operand. The vector operations will work only when the first element is located in the memory with the offset of zero otherwise the code cannot be executed.

Asymmetric temporal filtering: Our temporal denoising filter is developed based on a symmetric temporal filtering. We use forward motion estimation data to guide the backward motion estimation which increases the total motion estimation speed. However, for generality our

framework should support asymmetric window which we needed to solve the problem of all combination of forward and backward frames. The major challenge was when the number of the backward frames was larger than forward. In this case, there is no pre-estimated motion vector to guide the backward motion and the motion should be estimated without any guide.

Automated testing: One of the challenges in developing a video processing software is the debugging. We need to assure that by modifying the code we do not introduce any problem such as degrading the results, slowing down the algorithm, or consuming an unexpected amount of memory. This procedure should be done for all platforms and operating systems which is tedious to be done manually. We have developed an automated testing system that checks all modules individually for all platforms and generates the results and compare it with expected (or previous) results and reports the problems.

1.7 Thesis structure

The rest of thesis is structured in 7 chapters as follows: chapter 2 presents our model for the noise variance and the noise level function; chapter 3 discusses related methods to noise estimation (section 3.1), noise reduction (section 3.2), and NR-IQA (section 3.3); chapter 4 presents the proposed noise estimation where section 4.2 presents the details of algorithm, section 4.3 discusses application specific adaptation, and section 4.4 demonstrates objective and subjective results; chapter 5 explains our method for transforming a WGN filter to handle SDSCN where sections 5.2 and 5.3 present our solutions to address signal and frequency dependency, and section 5.4 provides experimental results; chapter 6 presents the proposed band-limited time-space video filter where section 6.2 discusses our proposed temporal filtering, section 6.3 explains our proposed dual-domain spatial filtering, section 6.4 discusses application oriented adaptation of our video filter, and section 6.5 gives objective and subjective results; chapter 7 presents our proposed NR-IQA method where section 7.3 presents the details of algorithm and section 7.4 demonstrates objective and subjective results; chapter 8 provides conclusion and directions for future work.

Chapter 2

Noise Modeling

2.1 Noise variance

The input noisy video frame (or still image) I can be modeled as, $I = I_{org} + \boldsymbol{n}_d + \boldsymbol{n}_g + \boldsymbol{n}_q$, where I_{org}, $\boldsymbol{n}_d$, $\boldsymbol{n}_g$, and, $\boldsymbol{n}_q$ are the noise-free image, white signal-dependent noise, white signal-independent noise, and, quantization noise respectively. With modern camera technology $\boldsymbol{n}_q$ can be ignored since it is very small compared to $\boldsymbol{n}_o = \boldsymbol{n}_d + \boldsymbol{n}_g$. $\boldsymbol{n}_d$ and $\boldsymbol{n}_g$ are assumed zero-mean random variables with variances $\sigma_d^2(I)$ and σ_g^2, respectively. (For simplicity of notation, we use the symbol I to refer to either a whole image or to an intensity of that image; this will be clear from the context). The NLF of the image intensity I can be assumed,

$$\sigma^2(I) = \sigma_d^2(I) + \sigma_g^2. \tag{1}$$

We define $\sigma_o^2 = \max(\sigma^2(I))$ as the *peak* of $\sigma^2(I)$. We also define the normalized noise level function $\tilde{\Omega}(I)$ as

$$\sigma^2(I) = \sigma_o^2 \cdot \tilde{\Omega}(I)\ ,\ \tilde{\Omega}(I) \leq 1. \tag{2}$$

When a video application, e.g., motion detection, requires a single noise variance, the best descriptive value is the maximum level, since a boundary can be effectively designated to discriminate between signal and noise. If $\sigma_d^2(I) = 0$ then $\tilde{\Omega}(I) = 1$ and $\sigma^2(I) = \sigma_g^2$ and is WGN. In this case σ_o^2 is the variance of WGN. If the video application requires $\sigma^2(I)$, our method estimates it with

notation of $\Omega(I)$.

Although noise at the sensor layer (i.e., $\boldsymbol{n}_o$) is white, it often becomes non-white due to post-capture processing. A processing module $\mathbf{P}(\cdot)$ such as interpolation, quantization, and filtering is applied on the image to increase the resolution or decrease the entropy. $\mathbf{P}(\cdot)$ makes its input I spatially correlated and the observed image I' becomes processed as

$$I' = \mathbf{P}(I) = \mathbf{P}(I_{org} + \boldsymbol{n}_o) = I_{org} + \boldsymbol{n}_p, \tag{3}$$

where $\boldsymbol{n}_p$ contains the correlated Poissonian-Gaussian noise and the distortion noise. By distortion noise we mean image structure that is destructed by $\mathbf{P}(\cdot)$. In video capturing pipeline, usually the inter-frame information is processed. Meaning, the post-capture processing $\mathbf{P}(\cdot)$ is applied on the difference between the reference frame and motion-compensated adjacent frame. Assuming the motion is ideally compensated, the difference contains mostly noise. In this case we can assume that $\boldsymbol{n}_p = \mathbf{P}(\boldsymbol{n}_o)$. If $\mathbf{P}(I) = I$ (i.e., there is no processing) then $\boldsymbol{n}_p$ is white. Depending on $\sigma_d^2(I) = 0$ or $\mathbf{P}(I) = I$, 4 types of noise can be defined, 1) WGN when $\sigma_d^2(I) = 0$ and $\mathbf{P}(I) = I$, 2) signal-dependent white noise (SDWN) when $\sigma_d^2(I) \neq 0$ and $\mathbf{P}(I) = I$, 3) signal-independent spatially correlated noise (SISCN) when $\sigma_d^2(I) = 0$ and $\mathbf{P}(I) \neq I$, 4) signal-dependent spatially correlated noise (SDSCN) when $\sigma_d^2(I) \neq 0$ and $\mathbf{P}(I) \neq I$. In this thesis by signal-dependent noise (SDN) we mean SDWN and by spatially correlated noise (SCN) we mean SISCN. Let us consider σ_p^2 as the peak of the variance of $\boldsymbol{n}_p$ along the all intensities. In (17), we estimate σ_p^2 as the peak of the level function of the *observed* video noise $\boldsymbol{n}_p$. Under SDWN where $\mathbf{P}(I) \neq I$, σ_p^2 is in fact the peak variance σ_o^2. Under SISCN and SDSCN, the peak variance σ_o^2 is estimated from σ_p^2 using (4).

When SDWN becomes processed, we model the resulting image as $I' = I_{org} + \boldsymbol{n}_p$ with $\boldsymbol{n}_p$ as the SDSCN and peak variance σ_p^2. We model the before in-camera processing image I as $I = I' + \boldsymbol{n}_\gamma$ with $\boldsymbol{n}_\gamma$ as the distortion noise and peak variance σ_γ^2. We thus differentiate here between SDWN $\boldsymbol{n}_o$, SDSCN $\boldsymbol{n}_p$, and distortion noise $\boldsymbol{n}_\gamma$, where $\boldsymbol{n}_o = \boldsymbol{n}_p + \boldsymbol{n}_\gamma$. Let $1 \leq \gamma \leq \gamma_{max}$ be the degree (power) of processing on σ_o^2. We estimate,

$$\sigma_o^2 = \gamma \cdot \sigma_p^2. \tag{4}$$

$\gamma = 1$ means $\mathbf{P}(I) = I$ and the observed noise is SDWN; $\gamma \leq \gamma_{max}$ means I was *moderately* processed, as shown in Figure 4.6 of section 4.2.4. Estimation of γ is based on the some assumption between LF and HF noise. Theoretically, as γ increases its estimation becomes harder since noise and image signal becomes similar. If $\gamma > \gamma_{max}$ the estimation error becomes intolerable. Practically, based on our method for estimation of γ $\gamma_{max} = 3.5$ (see Figure 4.6). *Heavily processed* means the nature of SDWN was heavily changed resulting in large σ_γ^2 compared to σ_p^2, i.e., $\sigma_\gamma^2 \gg \sigma_p^2$ since the mean absolute difference of I and I' is large. Processing technologies such as Bayer pattern interpolation, noise removal, bit-rate reduction, and resolution enlargement, are being increasingly embedded in digital cameras. For example, spatial filtering is used to decrease the bit-rate. Accurate data about in-camera processing is not available, in many cameras, however, processing can be bypassed manually, which allows to explore statistical properties of noise before and after processing. Our study shows that the low-power high frequency components of the noise (compared to noise power) are eliminated. As a result, low-frequency (LF) and impulse shaped noise remains. Fig. 2.1 shows parts of two images taken under the same condition in raw and processed image mode. This figure also shows the frequency spectrum of noise in both modes. We studied the noise using homogeneous image regions that we manually selected from 35 images taken by 7 different cameras (Canon EOS 6D, Fujifilm x100, Nikon D700, Olympus E-5, Panasonic LX7, Samsung NX200, Sony RX100). As we can see, filtering changes the frequency spectrum of the noise and makes it processed (frequency-dependent). In many video processing applications, estimation of the noise level before the in-camera filtering is desirable for accurate processing. Such estimation is challenging since some of noise frequency components are removed and calculation of the pre-processing (original) noise level by its current power (e.g., variance of homogeneous patches) is no longer accurate.

Often in the video capturing pipeline, built-in cameras or codecs filters make the noise processed and the whiteness property (i.e., independent and identically distributed frequency domain coefficients) is no longer realistic. These built-in filters are usually considered to reduce the bit-rate and not to remove the noise. Since most of the entropy is taken by high-frequency (HF) noise components, bit-rate adaptive codecs remove some part of (usually low-power) HF noise and leave undesired LF noise. Fig. 2.2 shows a part of homogeneous frame corrupted with real video noise

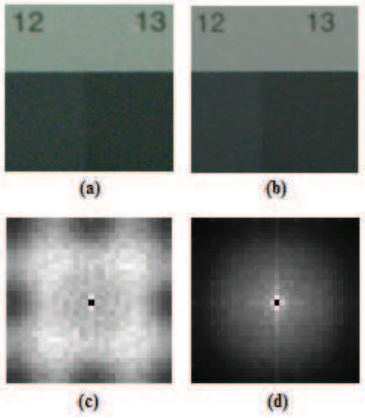

Figure 2.1: Images captured with the same camera in raw (a) and processed mode (b). Average of noise frequency magnitude of 35 different images taken by 7 cameras in raw (c) and processed mode (d).

and its power spectral density (PSD). Figure 2.3 shows an example of 1-D frequency spectrum for moderately and heavily processed. In case of heavily processed noise (i.e., $\gamma > \gamma_{max}$) the difference between power of LF and HF is significant and most of noise energy is concentrated in the LF components. By downsampling the image noise becomes more uniform in frequency domain (see Figure 2.4). Thus, the moderately processed noise model $\gamma \leq \gamma_{max}$ can be applied on the downsampled image and σ_p can be estimated in downsampled scale. In sections 4.2 and 5.3 we show how we estimate and use σ_p in different scales (resolutions) in the denoising process. The statistical properties of noise at the finest scale is different to coarse scale. In processed noise condition, at finest scale, noise is spatially correlated and as the scale becomes coarser noise becomes less correlated and closer to white. We employ this property to treat LF and HF differently employing different image scales using the fact that image signal and noise can be represented in different frequencies by decomposing image into a different scales. Figure 2.5 shows an example of 2-D SCN in three scales. Noise in coarsest scale can be assumed as white noise since the energy (magnitude) of noise is equally distributed in all frequencies.

2.2 Noise level function

A better adaptation of video processing applications to noise can be achieved by considering the NLF instead of a single value. However, as there is no guarantee that pure noise (signal-free) pixels

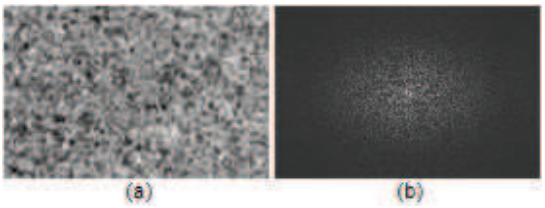

Figure 2.2: The left column (a) shows the part of a real noisy frame which is heavily processed and the right column (b) shows its frequency spectrum.

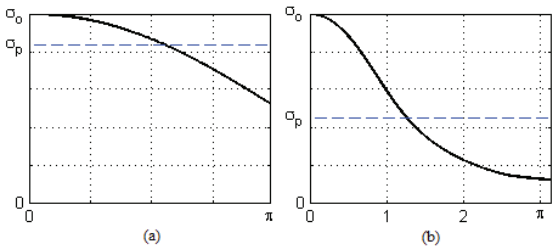

Figure 2.3: Example for 1-D spectrum of (a) moderately processed versus (b) heavily processed noise.

are available for all intensities, NLF estimation is challenging. The NLF strongly depends on camera and capture settings [6] as illustrates in Fig. 2.6.

Let the input noisy image I be divided into M_{I} sub-intensity classes. A piecewise linear function, see Fig. 2.6(c), can approximate the NLF in intensity class l as follow,

$$\sigma_l^2(I) = \boldsymbol{a}_l \cdot \sigma_{rep_l}^2 (I - I_{rep_l}) + \sigma_{rep_l}^2, \tag{5}$$

where $l \in \{1, \ldots, M_{\mathrm{I}}\}$, $I \in \{I_l^{\min}, I_l^{\max}\}$. $I_l^{\min}$ and $I_l^{\max}$ define the class boundaries, $\sigma_{rep_l}^2$ is a representative point of $\sigma_l^2(I)$ and I_{rep_l} is its corresponding intensity. $\sigma_{rep_l}^2$ can be, for example, the median of $\sigma_l^2(I)$. $\boldsymbol{a}_l$ represents the slope of a line approximating the NLF in the class l as illustrated in Fig. 2.6(c). If M_{I} is appropriately selected, $|\boldsymbol{a}_l|$ does not exceed $\boldsymbol{a}_{max} \geq \max(|\boldsymbol{a}_l|)$, which we estimated experimentally in analyzing different images and cameras. With $\max(|I - I_{rep_l}|) = \frac{1}{M_{\mathrm{I}}}$,

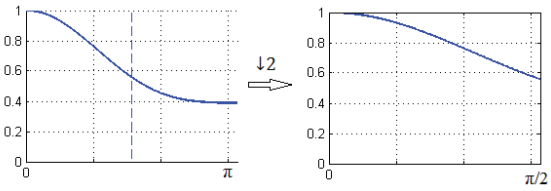

Figure 2.4: Spatially correlated noise becomes less correlated by downsampling.

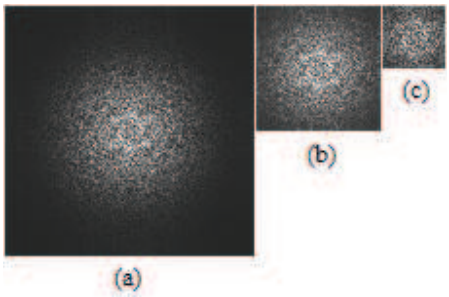

Figure 2.5: (a) Spectral density of heavily processed noise, (b) is (a) is downscaled by 2 using 3×3 anti-aliasing filter, and (c) is (b) downscaled by 2 using 3×3 anti-aliasing filter. (c) can be assumed as WGN.

$\sigma^2_{rep_l}$, and $|\boldsymbol{a}_l| \leq \boldsymbol{a}_{max}$,

$$\sigma^2_l(I) \leq \sigma^2_{max_l} = \sigma^2_{rep_l}\boldsymbol{a}_{max} \cdot \max(|I - I_{rep_l}|) + \sigma^2_{rep_l}. \tag{6}$$

The patches with variances greater than $\sigma^2_{max_l}$ do not fit in the NLF curve and should be rejected as non-homogeneous patches. This can thus be used to target homogeneous patches, as shown in section 4.2.2, where we use $\boldsymbol{a}_{max}$ to locate patches that fit into the linear approximation of NLF. In section 4.2.5 we propose an approximation of the NLF.

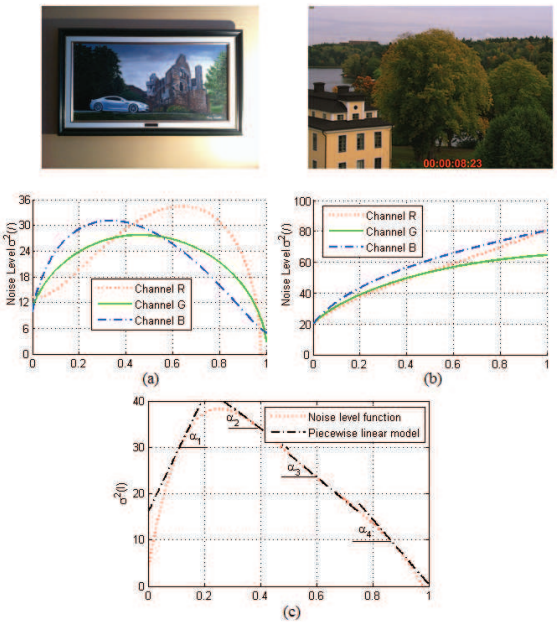

Figure 2.6: NLF approximation:(a) and (b) show two sample images and their NLF in RGB channels.(c) shows piecewise linear modeling of NLF.

Chapter 3

Related Work

In this chapter we review related works on video noise estimation, video noise reduction, and no-reference quality assessment.

3.1 Noise estimation

WGN estimation techniques can be categorized into filter-based, transform-based, edge-based, and patch-based methods. Filter-based techniques [25, 26] first smooth the image using a spatial filter and then estimate the noise from the difference between the noisy and smoothed images. In such methods, spatial filters are designed based on parameters that represent the image noise. Transform (wavelet or discrete cosine transform DCT) based methods [10, 11, 27, 28, 49–51] extract the noise from the diagonal band coefficients. [50] proposed a statistical approach to analyze the DCT filtered image and suggested that the change in kurtosis values results from the input noise. They proposed a model using this effect to estimate the noise level in real-world images. Although the global processing makes transform-based methods robust, their edge-noise differentiation lead to inaccuracy in low noise levels or high structured images. [50] aims to solve this problem by applying a block-based transform. [51] uses self-similarity of image blocks, where similar blocks are represented in 3-D form via a 3-D DCT transform. The noise variance is estimated from high-frequency components assuming image structure is concentrated in low frequencies. Edge-based methods [6, 29, 30]

select homogeneous segments via edge-detection. In patch-based methods [32–35], noise estimation relies on identifying pure noise patches (usually blocks) and averaging the patch variances. Overall local methods that deal with subsets of images (i.e. homogeneous segments or patches) are more accurate, since they exclude image structures more efficiently. Using a single patch as a reference makes [24] less accurate high noise some noise levels [31] adds particle filter to make it more robust and reduce the processing time. [32] utilizes local and global data to increase robustness.

In [33], a threshold adaptive *Sobel* edge detection selects the target patches, then averages the convolutions over the selected blocks for accurate estimation. Based on principal component analysis [34] first finds the smallest eigenvalue of the image block covariance matrix and then estimates the noise variance. Gradient covariance matrix is used in [35] to select "weak" textured patches through an iterative process to estimate the noise variance. Patch size is critical for patch-based methods. A smaller patch is better for low level of the noise, while, larger patch makes the estimation more accurate in higher noise level. For all patch sizes, estimation is error prone under processed noise; however by taking more low frequency components into account, larger patches are less erroneous. By adapting the patch size in these estimators to image resolution, it is more likely to find noisy (signal-free) patches, which consequently increases the performance. Logically finding image subsets with lower energy under WGN conditions leads to accurate results. However, under SDWN conditions underestimation normally occurs. Under WGN, [33–35] outperform others, however noise underestimation under signal-dependent noise makes them impractical for real-world applications.

SDWN estimation methods express the noise as a function of image brightness. The main focuses of related work is to first simplify the variance-intensity function and second to estimate the function parameters using many candidates as fitting points. In [52, 53], the NLF is defined as a linear function $\sigma^2(I) = aI + b$ and the goal is to estimate the constants a and b. Wavelet domain [52] and DCT [53] analysis are used to localize the smooth regions. Based on the variance of selected regions, each point of curve is considered to perform the maximum likelihood fitting. [54] estimates noise variation parameters using maximum likelihood estimator. This iterative procedure brings up the initial value selection and convergence problems. The same idea is applied in [6] by using a piecewise smooth image model. After image segmentation, the estimated variance of

each segment is considered as an overestimate of the noise level. Then the lower envelope variance samples versus mean of each segment is computed and based on that the noise level function by a curve fitting is calculated. After image segmentation, the estimated variance of each segment is considered as an overestimate of the noise level. Then the lower envelope variance samples versus mean of each segment is computed and based on that, the noise level function by a curve fitting is calculated. In [55], particle filters are used as a structure analyzer to detect homogeneous blocks, which are grouped to estimate noise levels for various image intensities with confidences. Then, the noise level function is estimated from the incomplete and noisy estimated samples by solving its sparse representation under a trained basis. The curve fitting using many variance-intensity pairs, requires enormous computations, which is not practical for many application especially when the curve estimation is needed to be presented as a single value. As a special case of SDWN with zero dependency, WGN cases are not examined in these NLF estimation methods. In [56], a variance stabilization transform (VST) converts the properties of the noise into WGN. Instead of processing the "Gaussianized" image and inverting back to Poisson model, a Poisson denoising method is applied to avoid an inverted VST.

Spatially correlated noise (SCN) is not yet an active research and few estimation methods exist. In [57], first, candidate patches are selected using their gradient energy. Then, the 3-D Fourier analysis of current frame and other motion-compensated frames is used to estimate the amplitude of noise. A wider assumption is in [58] by considering both frequency and signal dependency. In this method, the similarity between patches and neighborhood is the criterion to differentiate the noise and image structure. Using the exhaustive search, candidate patches are selected and noise is estimated in each DCT coefficient. [43] assumes noise is white Gaussian or white signal-dependent but it does not estimate the NLF.

3.2 Video noise reduction

Video noise reduction methods utilize the correlation between pixels temporally and spatially to extract signal and remove noise. They fall into three general groups, temporal, spatial, and spatio-temporal filters. Most of temporal and spatio-temporal filters utilize motion estimation to increase

their efficiency. In this section we review the related work for main components of video noise reduction.

3.2.1 Temporal and time-space filters

Temporal and time-space video filters can be classified based on two criteria 1) how the temporal information is fed into the filter and 2) what domain (transform or pixel) the filter use. According to the first criterion, filters can be classified into two categories: filters that operate on the original frames (prior and posterior) [8, 12, 14, 15] and ones use already filtered frames [7, 9, 17, 18]. The former use symmetric temporal window and the latter use a recursive framework. Recursive temporal filters (RTF) [9, 13, 59] are widely used due to the simplicity of their structure using few previous denoised frames. Using only previous frames, RTF does not introduce lag and recursive structure makes it efficient when there is no motion or accurately estimated. However, it takes several frames for a RTF to become effective. Besides, depending on when the sequence starts it produces different results. In some applications, (e.g., post-production) exact results need to be reproducible regardless of start time. Symmetric temporal filter (STF) [8, 12, 44] are more complex but address these problems.

The second criterion divides video filters into transform or pixel domain. Many high-performance transform (e.g., Wavelet or DCT) domain methods [8, 9, 11–16, 19, 60, 61] have been introduced to achieve a sparse representation of the video signal. High performance video denoising algorithm VBM3D [8] groups a 3-D data array which is formed by stacking together blocks found similar to the currently processed one. [15] is an advanced VBM3D by going a step further by proposing the VBM4D which stacks similar 3-D spatio-temporal volumes instead of 2-D blocks to form four-dimensional (4-D) data arrays. In [12], based on the spatio-temporal Gaussian scale mixture (ST-GSM) model, local correlation between the wavelet coefficients of noise-free video sequences across both space and time is captured. Then the Bayesian least square estimation is applied to accomplish the video denoising. Computation of these methods is costly. Moreover, the noise model is oversimplified which makes them unsuitable for real-world (such as consumer electronics) applications. A recently developed RF3D [61] made a wide assumption by proposing a denoising method for jointly corrupted by non-white random noise and fixed-pattern noise. The signal is extracted

based on both spatial and temporal correlations on a 3-D spatiotemporal transform domain.

Pixel-domain video filtering approaches [17, 18, 59, 62–68], utilizing motion estimation techniques, are generally faster by performing pixel-level operations. In such methods, a 3-D window of blocks along the temporal axis or the estimated motion trajectory is utilized for filtering of each pixel value. In Pixel-domain the challenges are the blocking artifacts and how to use the spatial information. Block-based temporal approaches are simple to implement but they create blockiness. The blockiness is more intensified when the motion compensation also is also block-based. In some video filters such as [65, 68] spatial information is not employed to keep the framework simple and fast, however, the residual noise makes the noise reduction inconsistent over the frame especially in complex motion. Multi-hypothesis motion-compensated filter (MHMCF) presented in [65] uses linear minimum mean squared error (LMMSE) of non-overlapping block to calculate the averaging weights. Its coarse (low-resolution) estimation of error using large blocks (e.g., 16×16), leads to motion blur and blocking artifacts in complex motion. [69] applies MHMCF to color video denoising, where the video denoising is performed in a noise adaptive color space different from traditional YUV color space. This leads to a more accurate estimation, however, due to chroma subsampling in codecs, noise adaptive color space is not realistic in many applications. [18] used the same scheme of color conversion in [69] but all channels are taken into account to increase the reliability of weight estimation. [68] simplifies the temporal motion to global camera motion. They perform the denoising by estimating the homography flow and applying the temporal aggregation using the multi-scale fusion. Another class of pixel-domain video filter use spatial filters when the temporal information is not reliable. In [59] hard decision is used to combine temporal and bilateral filter. Computational costly non-local mean is used in [67] by employing random K-nearest neighbor blocks where temporal and spatial blocks are treated in the same way. Authors of [66] used the complex BM3D [20] filter as the spatial support. [70] combined the outputs of wavelet-based local Wiener and adaptive bilateral filtering to be used as the backup spatial filter. In [67], noise assumed to be structured (non-white) but the motion is considered to be accurately estimated. [44] proposes a high-performance video filter to handle signal-dependent noise but it assumes that noise is white.

3.2.2 Spatial filtering

Spatial filters can be divided into three categories pixel domain, transform domain, and hybrid framework. One of the classic method which is equivalent to solving anisotropic heat diffusion equation (a second-order linear PDE) is anisotropic diffusion. To keep sharp edges, anisotropic diffusion can be performed using gradient of pixels, where in low gradient pixels, they get blurred with neighboring pixels [22] and in high gradient pixels edges are preserved and a monotonically decreasing function defines the blurring factor. A more sophisticated way of choosing this function is discussed in [71]. Compared to simple Gaussian filtering, anisotropic diffusion smooths out noise while keeping edges. However, it tends to over blur the image and sharpen the boundary with many texture details lost. More advanced partial differential equations (PDEs) have been developed so that a specific regularization process is designed for a given (user-defined) underlying local smoothing geometry [72], preserving more texture details than the classical anisotropic diffusion methods. Bilateral filtering is an alternative way of adapting Gaussian filtering to preserve edges [21], where both space and range distances are taken into account. The relationship between bilateral filtering and anisotropic diffusion is derived in [73]. A fast bilateral filtering algorithm is also proposed in [74]. Bilateral filtering has been widely adopted as a simple algorithm for denoising, for example, video denoising in [75]. However, it cannot handle impulse-shaped noise, and it also has the tendency to over smooth and to sharpen edges. If both the scene and camera are static, we can simply take multiple pictures and use the mean to remove the noise. This method is impractical for a single image, but a temporal mean can be computed from a spatial mean as long as there are enough similar patterns in the single image. Similar patterns can be found to a query patch and take the mean or other statistics to estimate the true pixel value, for example, in [76]. A more complicated formulation of this approach is through sparse coding of the noisy input [77]. Non-local methods are an exciting innovation and work well for texture-like images containing many repeated patterns. However, compared to other denoising algorithms that have n2 complexity, where n is the image width, these algorithms have n4 time complexity, which is unaffordable for real-world applications.

In transform domain (wavelet) filtering methods, natural image is decomposed into multi-scale-oriented sub-bands and highly kurtotic marginal distributions is observed [78, 79]. To enforce the

marginal distribution and have high kurtosis, low-amplitude values is suppressed while retaining high-amplitude values, a technique known as coring [80, 81]. In [82] the joint distribution of wavelets was found to be dependent. A joint coring technique is developed to infer the wavelet coefficients in a small neighborhood across different orientation and scale sub-bands simultaneously. The typical joint distribution for denoising is a Gaussian scale mixture (GSM) model [83]. In addition, wavelet-domain hidden Markov models have been applied to image denoising with promising results [84, 85]. Although the wavelet-based method is popular and dominant in denoising, it is hard to remove the ringing artifacts of wavelet reconstruction. In other words, wavelet based methods tend to introduce additional edges or structures in the denoised image. Besides, these methods are not well adapted to signal-dependent noise yet. In video application residual noise left after filtering varies pixel to pixel. This fact makes the wavelet-domain spatial filters not suitable to remove residual noise, since a single threshold cannot be defined to suppress noisy coefficients.

BM3D [20] and DDID [23] are the state-of-the-art methods which offer hybrid architecture by combining both pixel and transform domains. BM3D used a combination of 3D block matching and wavelet shrinkage, however, this implementation due to complexity is not trivial. DDID, on the other hand, proposes simple structure by combining bilateral filtering for the pixel domain filtering and short-time Fourier transform for the transform domain filtering (wavelet shrinkage). Although the DDID implementation is very simple, the computation time is excessive and not practical for video applications.

3.2.3 Motion estimation

Motion estimation is widely used in video enhancement and compression standards and can roughly divided into block-matching and optical flow techniques. Optical flow techniques are computationally complex and are less used in denoising methods. State-of-the-art optical flow algorithm [86] integrates the local method [87] into the global total variation framework [88]. [89] proposes a variational approach based on a theory for warping. [90] uses median filtering to denoise the flow after every warping step to improve accuracy.

Block-matching algorithms are fast and efficient under noise and used frequently in denoising

techniques. It is used to find a block which is most similar to a current block within a predefined search area in a reference frame, and it affects the enhancer performance and the overall computation time. The straight forward method to accomplish this procedure is full-search block matching algorithm (FSBMA), but it requires lots of operations due to its high complexity. The reduction of the number of search steps in order to increase the overall speed is required. The fast FSBMA, including the successive elimination algorithm [91–93], partial distortion elimination [94], the winner-update algorithm [95], and the advanced diamond search algorithm (DSA) [96] are proposed to reduce the computational heavy overload of FSBMA while maintaining its quality. In addition, in order to enhance the accurateness of DSA, several new algorithms, such as motion vector (MV) field adaptive search technique (MVFAST) [97], and enhanced predictive zonal search (EPZS) [98] are proposed. MVFAST is an improved DSA in both motion error and speed up by initially considering a small set of predictors. In DSA, only a large moving diamond pattern was considered, while MVFAST also presented a smaller moving diamond. PMVFAST uses basically the same architecture and patterns as MVFAST does, but a significant difference of PMVFAST compared to MVFAST is the way the small versus the large diamond is selected. Unlike MVFAST where motion was characterized as low, medium, or high by considering the largest motion vector candidate, PMVFAST improves the overall speed of the algorithm by using the large diamond less often. Furthermore, EPZS that improves upon PMVFAST by considering several other additional predictors in the generalized predictor selection phase of PMVFAST. EPZS also selects a more robust and efficient adaptive threshold calculation whereas, the pattern of the search is considerably simplified. However, the early termination of the search procedure leads to the poor performance. An architecture, which combines PMVFAST and EPZS, is developed, and it can be configured to support different search patterns, and independent sum of absolute difference (SAD) computations [99].

Another reduced computational complexity architectures are introduced by decreasing the number of computations using the hierarchical motion vector search algorithms (HMVSA), including three-step search (TSS) [100], and four-step search [101], which separate the estimation process into several levels, and the numbers of levels is fixed. Although HMVSA is a fast method, it is less accurate than FSBMA, especially when the motion field is large and complex. An advanced HMVSA

is developed to solve this problem, multi-resolution motion estimation algorithm (MMEA), which is based on initial coarse estimation and then refine it. In [102] propose a method to detect and correct the outlier MVs at the end of each pass. Conventional MMEAs are usually implemented in two types. In one variable search, the area at each level is used [103] and MV candidate is obtained from a large search area at the coarse level and the candidate becomes the search center of the next level, which has a smaller search area. A larger search area corresponds to a more accurate MV, but the extent of motion may increase with the search area. Therefore, the first MV candidate does not guarantee an accurate estimate, and yields an incorrect result at the next levels. Although [104, 105] apply a constant search area to partially solve this problem, the MVs may be less robust against noise. These MMEAs fall easily into the local minimum by choosing single MV candidate, so several algorithms that combine the scheme with a multiple MV candidate search have been proposed [106–108]. Due to multiple MV candidates for local searches at each level their performance is close to FSBMA, however they have a high computational cost. MMEA start with an initial coarse estimation and then refine it. They are efficient in both small and large motions since motion vector (MV) candidates are obtained from the coarse levels and the candidate becomes the search center of the next level. The problem of these methods is that the error propagates into finer levels when estimation falls into local minima in a coarse level. Therefore, a procedure to detect the failures and compensate them is essential, as we address in the proposed method.

3.3 No-reference image quality assessment

Image and video quality suffers from noise, blur, and compression artifacts. During the capturing process, noise from different sources is added to image and video content. It is essential to reduce the noise for enhancing the quality, reducing the bit-rate, or improving the performance of subsequent image processing tasks. Blur may be introduced to an image either during capture or due to processing such as denoising. In order to evaluate the performance of a denoiser or a deblurrer, a quantitative measure of quality is required. In many practical cases where the reference image is not available, the role of quality measurement techniques is more highlighted. During the development

process of image enhancement algorithms, the importance of no-reference image-quality assessment (NR-IQA) becomes clear. The effect of changing the parameters, or the algorithm (e.g., in an optimizing process), should be studied and the output quality needs to be verified. Subjective evaluation is tedious, especially when the test dataset is large. Thus, an automated quality measurement is necessary. Although it seems not possible to build an image-quality assessment that replaces the human visual system in all situations, many assessment methods are designed and successful at achieving reliable results at least in confined conditions (e.g., limited range of noise and image structure). NR-IQA techniques aim to distinguish image structure (e.g., salient geometric features) from distortion (e.g., noise and blur) and quantify the overall image distortion without a ground-truth according to visual perception [109], sharpness, and noise. As a consequence, parameters of image or video processing methods, such as noise estimation, noise removal, and deblurring can be optimized based on overall quality. By measuring the quality of the final output, with a recursive procedure, quality of current output is compared with previous outputs to find the optimal point. In addition of parameter selection, NR-IQA methods can be used to classify images based on their quality. As an example, among several captured images, such as in the burst mode, the one with the highest quality can be selected as the image of interest.

Many NR-IQA methods have been introduced, however, presenting a technique that works on a large set of data and distortion type (e.g., noise, blur, and compression artifact) is still an open problem. In this thesis, we propose a sparsity and dominant orientation based (SDQI) method that can be used 1) to optimize parameters of image enhancement algorithms and 2) to verify the quality of enhancement algorithms. We assume noise may have different types such as Gaussian or processed (non-white). To do this, we quantify the genuine image content based on the sparsity of local gradients using singular value decomposition (SVD) and discrete Fourier transform (DFT). SVD is applied to find the orientation dominancy of the image gradient patches. For a more accurate estimation of orientation, a shrinkage (i.e., suppressing the small coefficient) in the transform domain is first applied on the gradient image. To address multi-orientation patches, where one orientation is not dominant, we employ DFT to detect image structure, which increases the reliability of signal detection. To compute the SVD, instead of recursive matrix operations, we propose a faster method simpler to implement.

The diversity of NR-IQA methods using various image processing principles makes them difficult to be categorized. Based on their applications some focus on specific types of distortion (e.g., white Gaussian noise) and others consider different potential distortions. In [110], the unbiased risk estimate is proposed to calculate the distortion cost (e.g., mean squared error, MSE) for the enhancement application. It assumes the noise is additive and WGN, and accurate estimation of the noise variance is available. The technique proposed in [111] detects both noise and blur in one step based on the image anisotropy and measures the visual quality based on the variance of the entropy. The optimal performance of this method is achieved when the degradation is globally uniform and in the case of non-uniform noise or blur its performance decreases. [112] presents two separate pipelines for estimation of noise and blur. The noise is assumed to be WGN implying the high-frequency part of the noise exists, which is not accurate under real (e.g., processed non-white) noise. [38, 39, 42, 113–115] are developed to substitute human visual system to classify images (e.g., detection of blurred versus non-blurred in digital photography). Based on their applications, these methods are designed to be more sensitive to blur and less to noise. Thus, their performance decreases in the presence of the noise because the detection of edge versus high-frequency noise becomes challenging. Just noticeable blur (JNB) [38] is introduced to express the presence of blur around an edge using Sobel operator on local patches. [39] computes the cumulative probability of blur detection (CPBD) by classifying the blocks into edges and smooth areas. [37] developed a blind image quality assessment (BIQI) method, which utilizes support vector machine classifier to define the quality index (QI) based on subband coefficients of wavelet transform. The image is first ranked in each category of the degradations: JPEG and JPEG2000 compression, white noise, Gaussian blur, and fast fading; the final QI is estimated by combining all ranks. Blind/referenceless spatial quality evaluator (BRISQUE) [42] is a less computational complex method compared to [37] where, instead of wavelet, it employs Gaussian filter to extract low and high-frequency image components. Local phase coherence (LPC) of the wavelet image coefficients is employed in [40] to evaluate the image sharpness. The authors assume blur affects the LPC relationship near sharp image features and the degradation of LPC strength is employed to compute the image sharpness. The authors of [116] assess the image blur using a combination of natural scene statistics, multi-resolution decomposition, and machine learning. Based on training a probabilistic support vector

machine, blur is measured using gradient histogram features. In [41], the method S3 evaluates the sharpness by combining both spectral (Fourier domain) and spatial (pixel domain) sharpness measurements. Spectral measure is based on the slope of the local magnitude spectrum and spatial measure is based on local maximum total variation. In [45], pixel scattering is used to determine the image content, relying upon the fact that the noisier or blurrier the image is, the less entropy change is made by scattering the pixels. Its performance decreases as the noise properties deviate from WGN since entropy becomes inaccurate for measuring the quality. MetricQ [36] is a local method that made a wider noise assumption. SVD of the local gradient is employed to exploit the sharpness and noise of patches. SVD is used to estimate the dominant direction and its perpendicular direction and energy of both are considered to estimate the quality of each patch based on estimated signal to noise ratio. The average of quality values of patches that contain relative high quality is considered as output quality. Although MetricQ addresses other types of noise in addition of WGN, such as processed noise, the impact of noise in quality measurement is less emphasized by excluding noisy patches from the process.

Chapter 4

Homogeneity Classification Based Noise Estimation

4.1 Overview

Noise measurement is required in many image and video processing techniques, e.g., enhancement and segmentation, as adapting their parameters to the noise level can significantly improve their accuracy. Noise is added to an image from different sources [1–3] such as sensor (fixed pattern noise, dark current noise, shot noise, and amplifier noise), post-filtering (processed noise), and compression (quantization noise). In digital cameras, noise is signal-dependent due to physical properties of sensors and frequency-dependent due to post-capture filtering or Bayer interpolation. As discussed in the chapter 2, we assume image and video noise is additive and we classify it into: WGN, both frequency and signal-independent, signal-dependent white SDWN or (*Poissonian-Gaussian*), signal-independent but spatially correlated SISCN (or *processed WGN*), and signal-dependent and spatially correlated SDSCN (or *processed Poissonian-Gaussian*). SDSCN statistically is frequency and signal dependent, i.e., non-white Gaussian for a particular intensity.

Many noise estimation approaches assume the noise is Gaussian, which is not accurate in practical video applications, where video noise is signal-dependent. Techniques that estimate signal-dependent noise, on the other hand, do not handle Gaussian noise. Furthermore, noise estimation approaches rely on the assumption that high frequency components of the noise exist, which makes

them fail in real-world non-white (processed) noise. This is even more problematic in approaches using small patches (e.g., 5×5 pixels) [31–35, 52] because the probability to find a small patch with a variance much less than the noise power is higher than in large patch. We propose a method to estimate image and video noise of different types: WGN, SDWN, SISCN, and SDSCN. Our method also estimates the noise level function (NLF) of these noises. We do so by classification of intensity-variances of image patches in order to find homogeneous regions that best represent the noise. We assume the noise variance is a piecewise linear function of intensity in each intensity class. To find noise representative regions, noisy (signal-free) patches are first nominated in each intensity class. Next, clusters of connected patches are weighted where the weights are calculated based on the degree of similarity to the noise model. The highest ranked cluster defines the peak noise variance and other selected clusters are used to approximate the NLF. The more information, such as temporal data and camera settings, we incorporate, the more reliable the estimation becomes. To account for processed noise, (i.e., remaining after in-camera processing), we consider the ratio of low to high frequency energies. We address noise variations along video signals using a temporal stabilization of the estimated noise. Objective and subjective simulations demonstrate that the proposed method well outperforms, both in accuracy and speed, known noise estimation techniques. Our contribution is a method 1) operating on an image or a video signal in gray-scale or color space; 2) estimating the variance of WGN, SDWN, SISCN, and SDSCN automatically; 3) estimating the noise level function NLF, i.e., the relation between the noise variance and the intensities of the input noisy signal; 4) relating the input noisy signal and its down-sampled version to a) differentiate noise from image structure, b) adapt the patch size for intensity classification, and c) accelerate the estimation; 5) ranking noise representative regions (clusters) based a) on intra-frame (spatial) features including intensity, spatial relation (connectivity and neighborhood dependency), low-high frequency relation, size, and margins, and b) on inter-frame (temporal) features including temporal difference between patch signal in neighboring frames and difference between current estimate and estimates from previous frames; 6) integrating capture settings, if available as metadata, and user input of offline applications such as post production. 7) measuring the level of spatial correlation and adapt the input estimation parameter to that.

In the following, section 4.2 presents the proposed method and section 4.4 gives objective and

subjective results.

4.2 Proposed noise estimation method

The proposed method is based on the classification of intensity (or color) variances of signal patches (blocks) in order to find homogeneous regions that best represent the noise. We assume that noise variance is linear, with limited slope, to the intensity in a class. To find homogeneous regions, the method works on the down-sampled input image and divides it into patches. Each patch is assigned to an intensity class, whereas outlier patches are rejected. Clusters of connected patches in each class are formed and some weights are assigned to them. Then, the most homogeneous cluster is selected and the mean variance of patches of this cluster is considered as the noise variance peak of the input noisy signal. To account for processed noise, an adjustment procedure is proposed based on the ratio of low to high frequency energies. To account for noise variations along video signals, a temporal stabilization of the estimated noise is proposed. The block diagram in Figure 4.1 shows our noise estimator within one image or video frame without temporal considerations. Figure 4.2 shows how the method is stabilized using temporal processing in video. The proposed noise estimation based on intensity-variance homogeneity classification (IVHC) can be summarized as in Algorithm 1. In the remainder of this section, section 4.2.1 builds homogeneous patches; section 4.2.2 classifies patches; section 4.2.3 builds clusters of connected patches and estimate the noise peak variance; section 4.2.4 estimates parameters of processed noise; section 4.2.5 approximates the NLF; section 4.2.6 temporally stabilizes the estimate; sections 4.2.7 and 4.2.8 compute intra-frame and inter-frame weights; section 4.3.1 extends the method to camera settings and user input.

4.2.1 Homogeneity guided patches

Homogeneous patches are image blocks $\tilde{B}_i$ of size $W_e \times W_e$ where here i is the single index of the block, where the blocks are scanned column-wise,

$$\tilde{B}_i = \left\{ \tilde{I}(x,y) \;\middle|\; \frac{i}{\mathcal{N}_r} \leq x \leq \frac{i}{\mathcal{N}_r} + W_e - 1 \;, \mathrm{mod}(i,\mathcal{N}_r) \leq y \leq \mathrm{mod}(i,\mathcal{N}_r) + W_e - 1 \right\}, \quad (7)$$

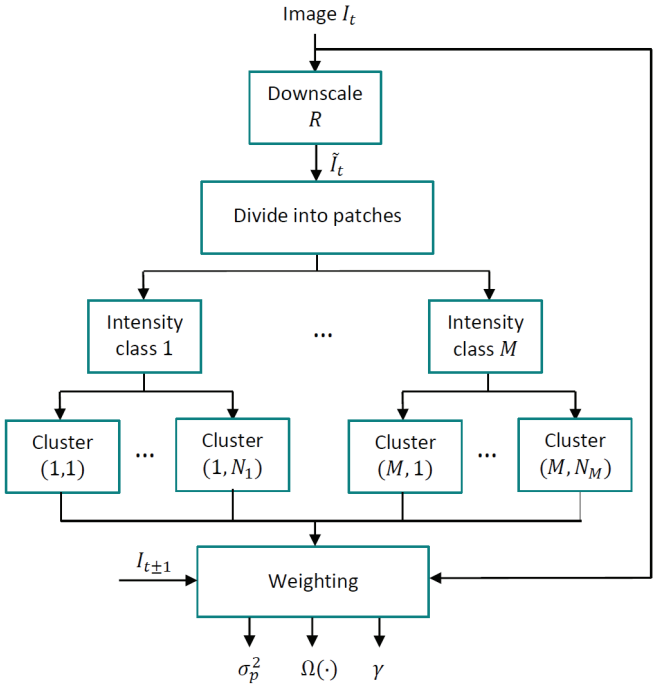

Figure 4.1: Intra-frame block diagram of the proposed estimator operating spatially within one image or video frame. $I_{t\pm1}$ is either preceding or subsequent frame (see section 4.2.8) and is used only for video frame.

where $\tilde{I}(x, y)$ is the downsampled version of the input noisy image at the spatial location (x, y), mod() is the modulus after division, and $\mathcal{N}_r$ is the image height (number of rows). After decomposing the image into non-overlapped patches, the noise $\boldsymbol{n}_i$ of each patch can be described as $\tilde{B}_i = \dot{B}_i + \boldsymbol{n}_i$, where $\tilde{B}_i$ is the observed patch corrupted by independent and identically-distributed (i.i.d) zero-mean Gaussian noise $\boldsymbol{n}_i$, and $\dot{B}_i$ is the original non-noisy image patch. The variance $\sigma^2(\tilde{B}_i)$ of a patch represents the level of homogeneity $\tilde{H}_i$ of $\tilde{B}_i$,

$$\tilde{H}_i = \sigma^2(\tilde{B}_i) = \frac{\sum \left(\tilde{B}_i - \mu(\tilde{B}_i)\right)^2}{W_e^2 - 1}\ ; \quad \mu(\tilde{B}_i) = \frac{\sum \tilde{B}_i}{W_e^2}, \tag{8}$$

where $\sum \tilde{B}_i$ is the summation of pixels of $\tilde{B}_i$. A small $\tilde{H}_i$ expresses high patch homogeneity. Under SDWN conditions, noise is i.i.d for each intensity level. If an image is classified into classes of

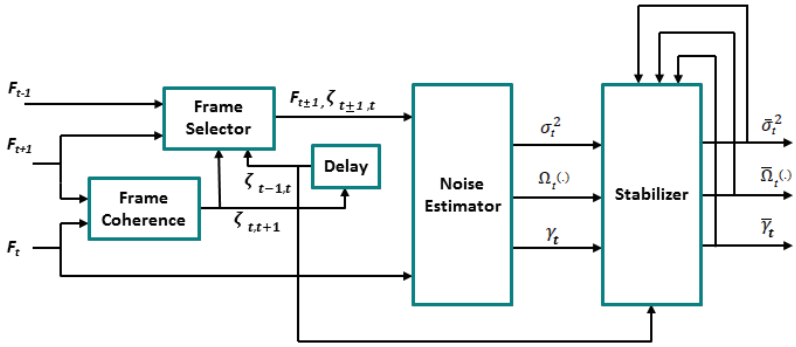

Figure 4.2: Block diagram of the proposed estimator operating spatio-temporally in a video signal. The noise estimator block is shown in Figure 4.1

Algorithm 1 IVHC based noise estimation

1: **Downscale** the noisy image I to $\tilde{I}$ & divide $\tilde{I}$ into $R_e \times R_e$ patches: (10).
2: **Assign** each patch a class number: (9).
3: **Find** the target connected clusters in each class in $\tilde{I}$: (11).
4: **Find** the corresponding cluster $\ddot{\Phi}(l,k)$ in I remove outliers: (14).
5: **Calculate** weights for the clusters: $\omega_1(l,k) \cdots \omega_{11}(l,k)$.
6: **Find** the noise-representative cluster $\hat{\Phi}$: (16).
7: **Compute** the noise variance σ_p^2 of selected cluster $\hat{\Phi}$: (17).
8: **Estimate** the noise level function $\Omega(.)$: (20).
9: **Estimate** the in-camera processing degree γ: (19).
10: **Compute** the pre-filter noise σ_o^2: (4).
11: **Stabilize** the estimates σ_p^2, $\Omega(.)$, and γ temporally: (21).

patches with same intensity level, the $\tilde{H}_i$ homogeneity model can be applied to each class. Assuming M_{I} intensity classes, $\tilde{L}_l$ represents the patches of the lth intensity class,

$$\tilde{L}_l = \left\{ \tilde{B}_i \;\middle|\; I_l^{\min} \leq \mu(\tilde{B}_i) \leq I_l^{\max} \right\}, \quad l \in \{1 : M_{\mathrm{I}}\}\,. \tag{9}$$

For $M_{\mathrm{I}} = 4$, $I_l^{\min} = \{0, 0.17, 0.4, 0.82\}$ and $I_l^{\max} = \{0.2, 0.45, 0.84, 1\}$ are vectors defining lower and upper bounds of class intensity. It is possible that a patch belongs to two intensity classes and therefore clusters can overlap (see Figure 4.3).

4.2.2 Adaptive patch classification

Images contain statistically more low frequencies than high frequencies. But small image patches show more high frequencies than low frequencies. Thus small patches have the advantage of better signal-noise differentiation. Large image patches, on the other side, are less likely to fall in the local

minima especially when noise is processed. To benefit from both, we propose image downscaling with rate R_e with a $R_e \times R_e$ grid averaging as the block-based averaging filter,

$$\tilde{I}(x,y) = \frac{1}{R_e^2} \sum_{i,j=0}^{R_e-1} I(xR_e + i, yR_e + j), \tag{10}$$

where I and $\tilde{I}$ are the observed and down-sampled images. This gives small patches in $\tilde{I}$ and large patches in I. Furthermore, the processed noise converges to white in the down-scaled image. Other desirable effects of downscaling are: 1) noise estimation parameters can be fixed for a lowest possible resolution of the images (note that R_e varies depending on the input image resolution) and 2) since the down-scaled image contains more low frequencies, the signal to noise ratio is higher. Assuming $\tilde{L}$ represents the set of patches in $\tilde{I}$; we binary classify the patches of the lth intensity class in $\tilde{I}$ into $\tilde{L}_l = \left\{\tilde{L}_l^0, \tilde{L}_l^1\right\}$, where $\tilde{L}_l^1$ are the target patches as

$$\tilde{L}_l^1 = \left\{ \tilde{B}_i \;\middle|\; \tilde{H}_i \leq \tilde{H}_{th}(l)\ ,\ \tilde{B}_i \in \tilde{L}_l \right\}. \tag{11}$$

(11) uses the homogeneity values $\tilde{H}_i$ and a threshold value $\tilde{H}_{th(l)}$ to binary classify $\tilde{L}_l$. Assuming the maximum value of the slopes $\boldsymbol{a}_l$ of the NLF in (5) is $\boldsymbol{a}_{\max}$. We define $\tilde{H}_{th}(l)$ as,

$$\tilde{H}_{th}(l) = \boldsymbol{a}_{\max} \tilde{H}_{\text{med}}(l) + \boldsymbol{c}_b^e, \tag{12}$$

where $\boldsymbol{c}_b^e = 1$ and $\boldsymbol{a}_{\max} = 3$. To calculate $\tilde{H}_{\text{med}}(l)$ we first divide $\tilde{L}_l$ into three sub-classes, then we find the minimum $\tilde{H}_i$ in each sub-class and finally we find the median of the three values. When class l contains overexposed or underexposed patches, $\tilde{H}_{\text{med}}(l)$ becomes very small. Therefore, the offset $\boldsymbol{c}_b^e$ is considered to include noisy patches. Figure 4.3 shows sample target patches and their connectivity with $M_{\text{I}} = 4$. Spatial information from horizontal and vertical connectivity can be used to form patch clusters as explained next.

4.2.3 Cluster selection and peak variance estimation

Due to complexity of noise and image structure, the variance based classification (11) by itself does not describe the noise in the image. In addition to statistical analysis, we use a spatial analysis

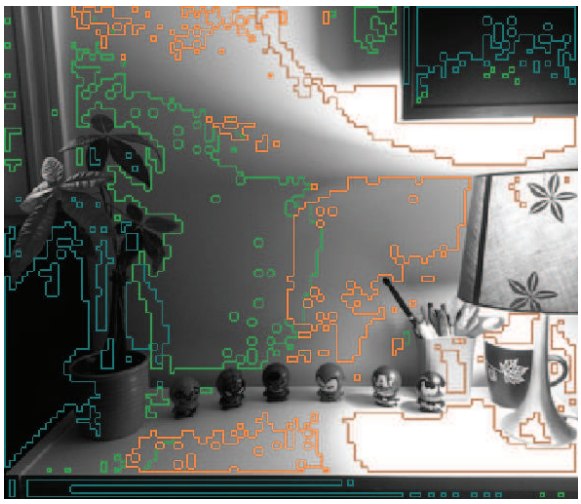

Figure 4.3: Target patches: different intensity classes are shown with different colors; each class consists of several clusters of different sizes.

to extract a more reliable noise descriptor. We use connectivity of patches in both horizontal and vertical directions to form clusters of similar patches. Next, for each cluster of connected patches in the down-sampled image $\tilde{I}$, we first find the corresponding connected patches B_i (with size of $R_eW_e \times R_eW_e$) from the cluster $\ddot{\Phi}(l,k)$ in the input noisy image I and then eliminate the outliers of cluster based on their mean and variance. Finally, we assess each cluster (after outlier removal) based on the intra- and inter-frame weights ω_1 to ω_{11}. $\ddot{\Phi}(l,k)$ represents the kth cluster of connected patches in the class l before outlier removal.

Outlier removal

The removal of outliers in each cluster is based on Euclidean distance of both the mean and the variance. For each cluster the patch with higher probability of homogeneity is defined as the reference patch and patches out of certain Euclidean distance are removed. Assuming $\ddot{\Phi}(l,k)$ represents the kth cluster of connected patches in the class l before outlier removal, we define the reference value of variance and mean of each cluster as,

$$\sigma^2_{ref}(l,k) = \min\{\sigma^2_{B_i}\}\ ,\ B_i \in \ddot{\Phi}(l,k), \quad \mu_{ref}(l,k) = \text{mean}\left[B_{ref}(l,k)\right],$$
$$B_{ref}(l,k) = \arg\min_{B_i \in \ddot{\Phi}(l,k)} \{\sigma^2_{B_i}\}, \tag{13}$$

where $B_{ref}(l,k)$ is the patch with the minimum variance in $\ddot{\Phi}(l,k)$ and its variance $\sigma^2_{ref}(l,k)$ and mean $\mu_{ref}(l,k)$ are considered references. By defining two intervals using two thresholds, the cluster after outlier removal is,

$$\Phi(l,k) = \Big\{ B_i \;\Big|\; |\sigma^2_{B_i} - \sigma^2_{ref}(l,k)| \leq \boldsymbol{T}^e_{\sigma}(l,k) \wedge |\mu_{B_i} - \mu_{ref}(l,k)| \leq \boldsymbol{T}^e_{\mu}(l,k) \wedge B_i \in \ddot{\Phi}(l,k) \Big\} \tag{14}$$

where $\boldsymbol{T}^e_{\sigma}(l,k)$ and $\boldsymbol{T}^e_{\mu}(l,k)$ are the variance and the mean thresholds that are directly proportional to $\sigma^2_{ref}(l,k)$ as,

$$\boldsymbol{T}^e_{\sigma}(l,k) = \boldsymbol{c}^e_{\sigma} \cdot \sigma^2_{ref}(l,k); \quad \boldsymbol{T}^e_{\mu}(l,k) = \boldsymbol{c}^e_{\mu} \cdot \frac{\sigma_{ref}(l,k)}{R_e \cdot W_e}, \tag{15}$$

where $\boldsymbol{c}^e_{\sigma} = 3$ and $\boldsymbol{c}^e_{\mu} = 4$. These parameters are selected based on the maximum possible slope $\boldsymbol{a}_{\max}$.

Cluster ranking

For each outlier-reduced connected cluster $\Phi(l,k)$, we first compute the weights $\omega_j(l,k)$ and then select the final homogeneous cluster $\hat{\Phi}$ by examining up to 11 criteria such as low and high frequency relationship, size of cluster, and variation noise power in each cluster. Based on each criterion we assign a weight to each cluster and the summation of all weights define the highest ranked cluster as

$$\hat{\Phi} = \underset{\Phi(l,k)}{\arg\max} \left(\sum_{j=1}^{11} \omega_j(l,k) \right). \tag{16}$$

We define the weights in section 4.2.7. We define the peak noise level σ^2_p in the input noisy image as the average of the patch variances in the cluster ranked highest $\hat{\Phi}$, i.e., best represents random noise,

$$\sigma^2_p = \frac{\sum \sigma^2_{B_i}}{N_p\{\hat{\Phi}\}} \;,\; B_i \in \hat{\Phi}, \tag{17}$$

where $N_p\{\hat{\Phi}\}$ is the number of patches in $\hat{\Phi}$. σ^2_p is the *peak* variance because we give higher weights to cluster with higher variances. Estimates of $\{0 \leq \omega_j(l,k) \leq 1\}$ are proposed in sections 4.2.7-4.2.8. Figure 4.4 shows selected clusters in the different intensity classes of Figure 4.3.

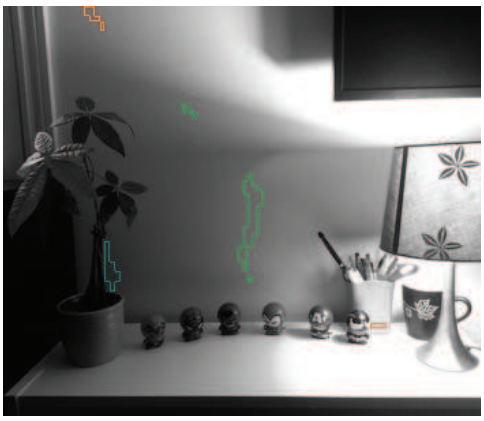

Figure 4.4: Highest-ranked clusters in different intensity classes, $M_{\mathrm{I}} = 4$.

4.2.4 Moderately processed noise estimation

In processed images, the assumption that the noise is frequency-independent in each homogeneous cluster is incorrect. In such situations, the variance of selected cluster σ_p^2 (17) does not represent the true level of the noise in the unprocessed noisy image because some frequency components of the noise have been removed. In many applications such as enhancement, the level of the unprocessed (original) noise is required. To estimate this original noise, the relation between low and high frequency components is necessary to trace the deviation from whiteness because we assume that the degree of noise removal in high frequency and low frequency is different. Let $\bar{\mathrm{E}}_{\mathrm{LF}}$ represents the variance of low-pass filtered pixels of $\Phi(l,k)$ and $\bar{\mathrm{E}}_{\mathrm{HF}}$ represents the median of the power of high-pass filtered pixels of $\Phi(l,k)$. We define,

$$\mathrm{E_r} = \frac{\bar{\mathrm{E}}_{\mathrm{LF}}}{\bar{\mathrm{E}}_{\mathrm{HF}}} = \frac{\boldsymbol{c}_e^e \cdot \mathrm{Var}\left\{h_{lp} * \Phi(l,k)\right\}}{\mathrm{Median}\left\{|h_{hp} * \Phi(l,k)|^2\right\}} \tag{18}$$

where $*$ is convolution, h_{lp} is a 3×3 moving average filter, and $h_{hp} = \mathbf{1} - h_{lp}$ a high-pass filter. $\mathbf{1}$ has zero elements except one at the center. With the given low-pass filter, according to the median of Chi-squared distribution $\boldsymbol{c}_e^e = 8(1-\frac{2}{9})^3 = 3.7$. The ratio $\mathrm{E_r}$ increases with spatial filtering occurs. We select $\bar{\mathrm{E}}_{\mathrm{HF}}$ as the median energy because high-frequency noise after filtering has an impulse shape and is divided into high and low levels. In many cameras, the filtering process is optional which allowed us to study the effect of this filtering on processed noise. Figure 4.5 shows the low-to-high ratio of homogeneous regions in different raw and processed images. The more

noise deviates from whiteness, the higher $\mathrm{E_r}$ is.

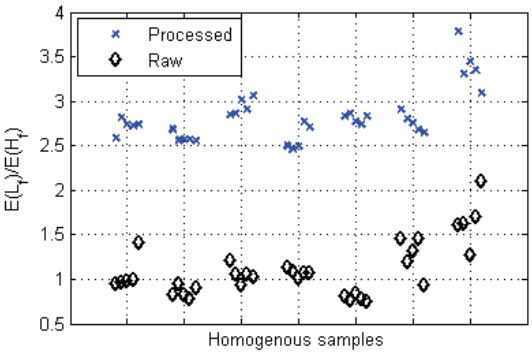

Figure 4.5: Low-to-High frequency power ratio of homogeneous regions in raw and processed images taken by 7 different cameras (Canon EOS 6D, Fujifilm x100, Nikon D700, Olympus E-5, Panasonic LX7, Samsung NX200, Sony RX100). Homogeneous regions are manually selected.

To approximate the processing degree γ of (4), we have studied the effect of applying anisotropic diffusion [22] and bilateral filters [21] on synthetic WGN. Figure 4.6 shows the relation between $\bar{\mathrm{E}}_{\mathrm{LF}}$ and $\bar{\mathrm{E}}_{\mathrm{HF}}$ and how $\mathrm{E_r}$ relates to γ. We propose linear approximation of γ as

$$\gamma = 1.4\mathrm{E_r}. \tag{19}$$

We temporally stabilized γ as in section 4.2.6. As shown in Fig. 4.6(b) at $\gamma \approx 3.5$, the approximation becomes less accurate.

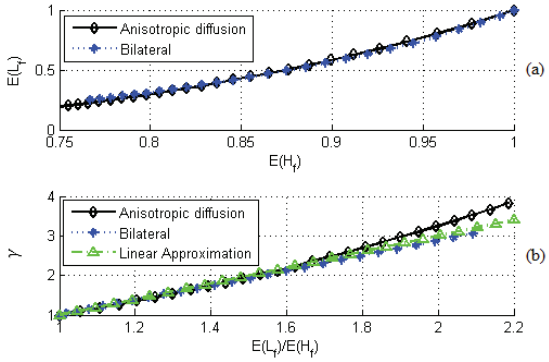

Figure 4.6: Relation between the filter strength and low-to-high average frequency power ratio (a). Linear approximating γ using the low-to-high ratio (b).

4.2.5 Noise level function approximation

We estimate the NLF based on the peak noise variance σ_p^2 of the selected cluster $\hat{\Phi}$ defined in (17) and employ other outlier-removed clusters $\Phi(l,k)$ to approximate the NLF. First, we set all the initial NLF curve $\hat{\Omega}(.)$ to σ_p^2, which means the noise level is identical in all intensities (Gaussian). Then, we update the $\hat{\Omega}(.)$ based on $N_p\{\Phi(l,k)\}$ the size (i.e., number of patches) and on $\sigma^2(l,k)$ the average of the variances of cluster $\Phi(l,k)$. We assign a weight (confidence) $\Gamma(l,k)$ to $\sigma^2(l,k)$: the larger $N_p\{\Phi(l,k)\}$ is, the better $\sigma^2(l,k)$ represents the noise at intensity $\mu(l,k)$, meaning the closer $\Gamma(l,k)$ should be to 1. The point-wise NLF $\hat{\Omega}(.)$ is then,

$$\hat{\Omega}\left(\mu\left(l,k\right)\right) = \min\left(\sigma_p^2, \frac{1}{\Gamma(l,k)} \cdot \sigma^2(l,k)\right). \tag{20}$$

$\Gamma(l,k) = 1 - \exp(-\frac{N_p\{\Phi(l,k)\}}{\boldsymbol{c}_\Gamma^e})$ meaning clusters with smaller number of patches, are less reliable. $\boldsymbol{c}_\Gamma^e = 5$ calculated numerically: let the large clusters with 15 (or more) patches be completely reliable, i.e., $\Gamma(l,k) = 1$, then from the 3σ rule $\boldsymbol{c}_\Gamma^e = 5$. Finally, the continuous NLF $\Omega(.)$ can be approximated from $\hat{\Omega}(.)$ by applying a regression analysis, e.g., curve fitting as illustrated in Fig. 4.7 using *polyfit* of *Matlab*. Under WGN, $\hat{\Omega}(\mu(l,k))$ is constant equal to σ_p^2. Under SDSCN, $\hat{\Omega}(\mu(l,k))$ is reduced by factor γ but the normalized NLF shape is not altered. Thus, with $\sigma_o^2 = \gamma \cdot \sigma_p^2$ as in (4) under SDWN in each cluster the proposed method can estimate the NLF whether the noise is processed or white.

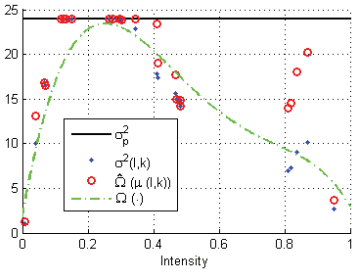

Figure 4.7: Illustration of NLF approximation.

4.2.6 Estimate temporal stabilization

In many video applications, instability of noise level is intolerable, unless the temporal coherence between frame is very small e.g., a scene change. Let $\zeta_{t-1,t}$ represent the similarity between the current I_t and previous frame I_{t-1}; $0 \leq \zeta_{t-1,t} \leq 1$. ζ determines how the statistical properties of new observation (i.e., image) are related to previous observations. Consider a process (such as median) $\mathbf{O}_i(\sigma^2_{t-i}, ..., \sigma^2_{t-1}, \sigma^2_t)$ to filter out outliers from the set of current σ^2_t and previous estimates $\{\sigma^2_{t-i}\}$. When $\zeta_{t-1,t} = 1$, the accurate estimate should be $\mathbf{O}_i(\sigma^2_{t-i}, ..., \sigma^2_{t-1}, \sigma^2_t)$; when $\zeta_{t-1,t} = 0$, the accurate estimate is σ^2_t itself. So we propose the following linear stabilization,

$$\bar{\sigma}^2_t = \mathbf{O}_i(\sigma^2_{t-i}, ..., \sigma^2_{t-1}, \sigma^2_t) \cdot \zeta_{t-1,t} + (1 - \zeta_{t-1,t}) \cdot \sigma^2_t \tag{21}$$

where, $\bar{\sigma}^2_t$ is the stabilized final noise variance in I_t. Note σ^2_t in (21) is σ^2_p in (17) at time t. The stabilization process can be performed on both γ and the NLF to get $\bar{\gamma}^2_t$ and $\bar{\Omega}_t(.)$.

4.2.7 Intra-frame weighting

Noise in low frequencies

Image signal is more concentrated in low frequencies, however noise is equally distributed. Down-sampled versus input images can be exploited to analyze noise in the low-frequency components. The variance of finite Gaussian samples follows a *scaled chi-squared* distribution. But here we utilize an approximation benefiting the normalized Euclidean distance,

$$\omega_1(l,k) = \exp(-\boldsymbol{c}^e_1 \cdot \frac{(\sigma^2(l,k) - R^2_e \cdot \tilde{\sigma}^2(l,k))^2}{(\sigma^2(l,k))^2}), \tag{22}$$

where exp(.) symbolizes the exponential function, $\sigma^2(l,k)$ and $\tilde{\sigma}^2(l,k)$ are the average of variances of the input and down-sampled patches in the cluster after outlier removal $\Phi(l,k)$. The positive constant $\boldsymbol{c}^e_1$ (e.g., 0.4) varies depending on the R_e and the W_e. Low values of $\omega_1(l,k)$ account for image structure, which the signal is concentrated in low frequencies.

Noise in high frequencies

The dependency of neighboring pixels is another criterion to extract image structure. The median absolute deviation (MAD) in the horizontal, vertical and diagonal directions expresses this dependency as

$$\begin{aligned} \tau_i = \quad & \text{median}\,\{|B_i(m,n+1) - B_i(m,n)|, \\ & |B_i(m+1,n) - B_i(m,n)|, \\ & |B_i(m+1,n+1) - B_i(m,n)|\}\,, \end{aligned} \tag{23}$$

where $0 \leq m,n \leq R_e \cdot W_e - 2$. According to half-normal distribution $\sigma^2_{B_i} = 2\text{erf}^{-1}(0.5) \cdot \tau_i^2 = 1.1\tau_i^2$, where erf^{-1} is the inverse error function. We profit from this property to extract the likelihood function of neighborhood dependency. Assuming for each $\Phi(l,k)$, $\tau(l,k)$ is the average of τ_i of the blocks in the $\Phi(l,k)$. Under WGN we define the following likelihood function,

$$\omega_2(l,k) = \exp(-\boldsymbol{c}_2^e \cdot R_e^2 \frac{(\sigma^2(l,k) - 1.1\tau^2(l,k))^2}{(\sigma^2(l,k))^2}). \tag{24}$$

For a Gaussian random variable, $\boldsymbol{c}_2^e$ can be computed by numerical analysis, however, we considered a more relaxed value $\boldsymbol{c}_2^e = 0.2$ to handle both unprocessed and processed noise. Low values of $\omega_2(l,k)$ mean a strong neighboring dependency, which is a hint of image structure. In case of white noise, we analyze the MAD versus variance to estimate if the patch contains structure. Thus, in final estimation step we use $1.1\tau^2(l,k)$ instead of $\sigma^2(l,k)$ for patches with structure.

Size of the cluster

The target patches are more concentrated in homogeneous regions and the size of the homogeneous region should be large enough to precisely represent the noise statistics. Therefore, larger cluster has a higher probability of presenting the homogeneous regions. However, a linear relationship between cluster size and the corresponding weight is not advantageous, since once it is past a certain size, sufficient noise information can be obtained. We propose the following the weight for the size of the cluster,

$$\omega_3(l,k) = 1 - \exp(-\boldsymbol{c}_3^e \cdot \frac{N_p\{\Phi(l,k)\}}{N_p\{I\}}), \tag{25}$$

where $N_p\{\Phi(l,k)\}$ and $N_p\{I\}$ are the number of patches in $\Phi(l,k)$ and the input image, respectively. We compute $\boldsymbol{c}_3^e$ numerically: assuming we divide the image into a grid of 5×5 and each grid unit, containing 4% of the image, is large enough to give $\omega_3(l,k) = 1$; with the 3σ rule, $\boldsymbol{c}_3^e = \frac{3}{0.04} = 75$.

Variance of means and variance of variances

In a homogeneous cluster with relatively large number of pixels in each patch, the normalized value of the variance of variances $\ddot{\sigma}^2(l,k)$ and variance of means $\ddot{\mu}(l,k)$ of $\{B_i \in \Phi(l,k)\}$, should be small. And so we propose,

$$\omega_4(l,k) = \omega_3(l,k)\text{exp}(-\frac{\ddot{\sigma}^2(l,k)}{\sigma^4(l,k)}), \tag{26}$$

$$\omega_5(l,k) = \omega_3(l,k)\text{exp}(-\frac{\ddot{\mu}(l,k)}{\sigma^2(l,k)}), \tag{27}$$

where

$$\ddot{\sigma}^2(l,k) = \frac{\sum\left(\sigma_{B_i}^2 - \sigma^2(l,k)\right)^2}{(N_p\{\Phi(l,k)\})^2 - 1}, \ddot{\mu}(l,k) = \frac{\sum\left(\mu_{B_i} - \mu(l,k)\right)^2}{(N_p\{\Phi(l,k)\})^2 - 1}.$$

In equations (26) and (27) $\omega_4(l,k)$ and $\omega_5(l,k)$ are directly proportional to $\omega_3(l,k)$. Without this, it is probable to assign high values to $\omega_4(l,k)$ and $\omega_5(l,k)$ when the cluster has a small number of patches even though it is not homogeneous. Uniformity of mean and variance describes cluster homogeneity and leads to high value of $\omega_4(l,k)$ and $\omega_5(l,k)$.

Intensity margins

Excluding the intensity extremes from the estimation procedure can be problematic when the signal margins are informative. For instance, the elimination of dark intensities in an underexposed image leads to the removal of the majority of data and, consequently, inaccurate estimation. We propose thus negative weights to margins,

$$\omega_6(l,k) = -(\frac{\max(\mu(l,k) - \mathsf{I}_\text{H}, 0)}{1 - \mathsf{I}_\text{H}} + \frac{\max(\mathsf{I}_\text{L} - \mu(l,k), 0)}{\mathsf{I}_\text{L}}), \tag{28}$$

where $\mathrm{l_H} = 0.9$ and $\mathrm{l_L} = 0.06$.

Variance margins

There are cases where underexposed or overexposed image parts with very low variances are not observed in the intensity margins. On the other hand, extremely high variances signify image structure. For consumer electronic related applications, the PSNR usually is not below a certain value (e.g., 22dB). Thus, similar to intensity margins, variance margins also affect the homogeneity characterization. We propose thus the following weight,

$$\omega_7(l,k) = -\exp\left(-\frac{\sigma^2(l,k)}{\sigma^2_{min}}\right) - \exp\left(-\frac{\delta(l,k)}{\sigma^2_{max}}\right), \tag{29}$$

where $\delta(l,k) = \max(\sigma^2(l,k) - \sigma^2_{max}, 0)$, $\sigma^2_{min} = 5$ and $\sigma^2_{max} = 200$ are variance margins.

Maximum noise level

Under SDWN, the maximum noise level distinguishes the signal and noise boundary. Hence, the maximum noise level and the corresponding intensity can be used to estimate the NLF. As a result, the $\Phi(l,k)$ with the maximum level of the noise should be ranked higher. However, some consideration should be taken into account in order to exclude clusters containing image structures for this weighting procedure. The basic assumption that noise variance slope is limited helps to restrict the maximum level of noise in each intensity class. So,

$$\sigma^2_{peak}(l) = \min\left\{\boldsymbol{a}_{\max}\text{median}\left[\sigma^2(l,k)\right],\ \max\left[\sigma^2(l,k)\right]\right\}, \tag{30}$$

where $\sigma^2_{peak}(l)$ is the expected peak of noise in the class l. By outlining a valid noise variance interval, the weight can be defined as follows,

$$\omega_8(l,k) = \exp\left(-\frac{\left[\sigma^2_{peak}(l) - \sigma^2(l,k)\right]^2}{\sigma^4(l,k)}\right). \tag{31}$$

Clipping factor

Due to bit-depth limitations, the intensity values of the input images are clipped in low and high margins. We propose a weight according to 3σ bound,

$$\begin{aligned} \omega_9(l,k) &= \exp(-\tfrac{\mu_{clip}^2}{2\sigma^2(l,k)}) - 1; \\ \mu_{clip} = \quad & \max\left[\mu(l,k) + 3\sigma(l,k) - 1, 0\right] + \\ & \max\left[\mu(l,k) - 3\sigma(l,k), 0\right], \end{aligned} \tag{32}$$

where 1 and 0 are maximum and minimum intensity. If all pixels are in the 3σ bound, $\mu_{clip} = 0$.

4.2.8 Inter-frame weighting

Utilizing only spatial data in video signals may lead to estimation uncertainty, especially in processed noise, where the relation between low and high frequency components deviates from WGN, which in turn makes structure and noise differentiation more challenging. Another issue to consider in video is robust estimation over time especially in joint video noise estimation and enhancement applications.

Temporal error weighting

Assume $B_{(i,t)}$ is ith patch in the noisy frame I_t at time t, and $B_{(i,t+m)}$ is corresponding patch in the adjacent noisy frame at time $t+m$, where $m = \pm 1$. Based on which adjacent frame (previous or following) has less temporal error for whole frame m is set to -1 or $+1$. Assuming the noise level does not change through time the matching (or temporal consistency) factor can be defined as,

$$\omega_{10}(l,k) = \sum \exp\left(-\frac{\left(\sigma_{(B_i,t)} - \sigma_{(B_i,t+m)}\right)^2}{\sigma^2_{(B_i,t)}}\right), \tag{33}$$

where $B_{(i,t)} \in \Phi_t(l,k)$, and $\Phi_t(l,k)$ is the kth connected cluster of class l in I_t. Since the homogeneity detection is applied on the input noisy image, there is no guarantee that the temporal $B_{(i,t+m)}$ is also homogeneous. Therefore, high temporal error of few patches should not significantly affect $\omega_{10}(l,k)$. For this, we analyze each patch error and aggregate all matching degrees. This is more reliable than assessing the aggregated variances.

Previous estimates weighting

In video applications, noise estimation should be stable through time and coarse noise level jumps are only acceptable when there is a scene (or lighting) change. Therefore, the cluster with the variance closer to previous observation is more likely to be the target cluster. Assuming σ^2_{t-1} is the estimated noise σ^2_p for previous frame, we define the following to add temporal robustness,

$$\omega_{11}(l,k) = \zeta_{t-1,t}\text{exp}\left(-\frac{[(\sigma_{t-1}-\sigma(l,k)]^2}{\sigma^2_{t-1}}\right) , \tag{34}$$

where $0 \leq \zeta_{t-1,t} \leq 1$ measures scene change estimated at patch level. Assuming the temporally matched patches have the mean error less than $2\sigma^2_{max}/(W^2_e)$, the ratio of temporally matched patches to the whole patches defines the $\zeta_{t-1,t}$. Note that (34) guides the estimator to find the most similar homogeneous region in I_{t-1}.

4.3 Application specific adaptation

In the course of our research and development, there were application (industrial) specific aspects of estimation requiring solutions. We developed the following solutions to these issues.

4.3.1 Camera settings and user input

For a specific digital camera, the noise type and level can be desirably modeled using camera parameters such as ISO, shutter speed, aperture, and flash on/off. However, creating a model for each camera requires excessive data processing. Also such (meta) data can be lost for example, due to format conversion and image transferring. Thus, we cannot only rely on the camera or capturing properties to estimate the noise; however, these data, if available, can support the selection of homogeneous regions and thereby increase estimation robustness. Assuming based on camera settings we can find the range of noise level, patch selection threshold $\tilde{H}_{th}(l)$ in (12) can be modified according to this range. We can also use variance margin weights in (29) to reject out of range values. We will show related results in the experimental section.

In some video applications such as post-production, users require manual intervention to adjust

the noise level for their specific needs. Assuming user knowledge about the noise level can define the valid noise range, the variance margin used in (29) can be used to reject the out of range clusters.

4.3.2 Heavily processed noise

In section 4.2.4 we presented a method to estimate the processing degree γ for moderately processed noise. With moderately processed noise model we can estimate the amount of original noise level before processing. This model is useful in removing moderately processed noise by applying a WGN filter on the original image scale. However, if the noise is heavily processed, we need to apply the WGN filter on the different image scales. In chapter 2, we explained that if we downscale the image the noise becomes less spatially correlated and we use this feature to remove noise. For heavily processed noise, we need the knowledge of inter-scale noise statistics. Assuming we have detected the homogeneous region $\Phi(l,k)$ contains no signal, we model the noise using three values, the STD of pixel values in homogeneous region σ_p, $\hat{\gamma}_0$ deviation from whiteness at the original resolution, and $\hat{\gamma}_1$ deviation from whiteness at the downscaled by 2 resolution. In sections 5.3 and 6.2.6 we have explained how we use these parameters to adapt our filters to SCN. To define $\hat{\gamma}_0$ and $\hat{\gamma}_1$, let us assume a 3×3 Gaussian filter h_a as the anti-aliasing filter in the downsampling process. We downscale $\Phi(l,k)$ by 2 and 4 and compute their STDs as $\sigma_{p,1}$ and $\sigma_{p,2}$. We define $\hat{\gamma}_0$ and $\hat{\gamma}_1$ as,

$$\hat{\gamma}_0 = \frac{\sigma_{p,1}}{\sigma_p \sqrt{\sum h_a^2}} \ , \ \hat{\gamma}_1 = \frac{\sigma_{p,2}}{\sigma_{p,1} \sqrt{\sum h_a^2}}. \tag{35}$$

If the noise is WGN $\hat{\gamma}_0 = \hat{\gamma}_1 = 1$. If the noise is processed, $\hat{\gamma}_0 > 1$ and if the noise of downscaled image is also processed (i.e., the radius of spatial correlation is higher than 1) $\hat{\gamma}_1 > 1$.

4.3.3 Tone-mapping of high dynamic range video

The assumption that absolute slope NLF is smaller than $\boldsymbol{a}_{\max}$ is not accurate for HDR images. Due to nature of HDR capturing, the level of noise rapidly changes in the intensity domain. Thus, we propose a forward tone-mapping before noise estimation to equalize the noise level. After applying the noise estimation and reduction, we apply the backward tone-mapping to get the original histogram. Typical tone-mapping algorithms use the minimum and maximum value of pixels to

estimate the tone-mapping function. It is common that the minimum and maximum value changes significantly between frames and different tone-mapping function lead to tone shift between frames which makes the temporal data useless. Besides most of the typical tone-mapping algorithms are not easily invertible and consequently cannot be used for denoising purposes. We have analyzed a large set of HDR data and we concluded that the mean of the frame is a reliable value to estimate the tone-mapping function and if there is a temporal coherency between the frames, mean does not significantly change. We propose a simple tone-mapping function using a gamma correction with a fixed value ($\frac{1}{2}$) followed by a scaling based on the mean of the image. For an input image I we propose the following tone-mapping function

$$\ddot{I} = I^{\frac{1}{2}} \frac{\boldsymbol{c}_{tm}^{e}}{\textbf{MEAN}(I^{\frac{1}{2}})}, \tag{36}$$

where $\ddot{I}$ is the tone-mapped output $\boldsymbol{c}_{tm}^{e}$ is extracted experimentally and set to 0.4.

4.3.4 Adapting algorithm constants to noise types

The proposed algorithm includes constants that are defined to give the best results when the noise is not heavily processed. If we adapt those constants to the degree of spatial correlation we can increase the estimation accuracy. This degree does not need to be accurate since we use it only to tune the parameters of algorithm that provides an accurate estimation. In order to pre-estimate the degree of processing, we propose to first find the patch with minimum variance and maximum noise in HF as in ω_2. With both criteria, the patch is likely to include only noise. Then we calculate ω_2 for that patch and denote it as $\hat{\omega_2}$. $\hat{\omega_2}$ is the pre-estimated degree of processing and $\hat{\omega_2} \ll 1$ is a hint for heavily processed noise. The important constant, which becomes adaptive to pre-estimated degree of processing, is $\boldsymbol{c}_{\mu}^{e}$ in the outlier removal (15). When the noise in a cluster becomes processed, the variance of patch decreases while its mean remains unchanged and the relation between mean and variance deviates from our model. We propose to use ω_2 of the hint patch, thus we modify $\boldsymbol{c}_{\mu}^{e} = 4$ to $\frac{4}{\hat{\omega_2}}$ to adapt the constant to pre-estimated degree of processing.

4.4 Experimental results

In the following sections, we first discuss the parameter selection and then we evaluate the quality of the proposed estimation of WGN, SDWN, SDSCN, and NLF separately, and we show how camera settings and user input improve the estimation. We also discuss implementation issues.

4.4.1 Parameter selection

The down-sampling rate R_e is a function of image resolution. For example, $R_e = 2$ for low resolution (less than 720p) and $R_e = 3$ for higher resolutions. As a result, noise estimation parameters become resolution independent. We have set the down-sampled patch size W_e to 5. The higher the number of classes M_{I} is, the better the NLF can be approximated. The downside is, however, too small classes and invalid statistics, such as $\sigma^2_{rep_l}$. Experimentally, best value for M_{I} is either 3 or 4. We used $M_{\mathrm{I}} = 4$. All constant parameters used in the proposed weights are given directly after their respective equations, and we have used the same set of values in all results in this work.

The proposed homogeneous cluster selection can be performed either on one channel of a color space or on each channel separately. Normally the Y channel is less manipulated in capturing process and therefore noise property assumptions in it are more realistic. Our observation confirms that adapting the estimation to Y channel leads to better video denoising. We, therefore, use estimated target cluster in the Y as a guide to select corresponding patches in chroma. Utilizing these patches, we calculate the properties of chroma noise, i.e., σ_p^2 and γ according to (17) and (19). Due to space constraint, simulation results here are given for the Y channel.

Target patches in (11) can be recalculated in a second iteration by adapting the $\tilde{H}_{\min}(l)$ to σ_p^2 (estimated in first iteration). A finer estimation can be performed by limiting the bound meaning smaller value for $\boldsymbol{a}_{\max}$. The rest of the method is the same as in the first iteration. The complexity of a second iteration is very minor and much less than the first one since patch statistics are already computed. However, our tests show that a second iteration improves the estimation results slightly, not justifying iterative estimation.

4.4.2 White Gaussian noise (WGN)

We have selected six state-of-the-art approaches Yang2010 [33], [34, 35, 50], Tian2012 [32], and Ghazal2011 [31] and evaluated their performance on 14 test images as in Fig.4.8. We generated noisy images by adding a zero-mean WGN to the ground-truth, with 4 levels of standard deviation, from 4 to 16 with the step of 4 and we run 10 Monte-Carlo experiments for each noise level. Table 4.1 demonstrates mean of absolute errors of related and proposed method which outperforms. The average variance of the error for our method compared to related methods is similar and is not given here. Method [34] and [35] give the closest results. Fig.4.9 also shows examples of selected homogeneous clusters.

Figure 4.8: Test images for WGN experiment: *Lena*, *Barbara*, *Boat*, *Peppers*, and ten images from the *TID2008* database.

Table 4.1: WGN: Absolute estimation error averaged over test images in Fig. 4.8.

Noise STD	Ref [33]	Ref [34]	Ref[35]	Ref[50]	Ref [32]	Ref [31]	Ours
4	0.69	0.25	0.23	0.80	0.82	0.51	**0.22**
8	0.46	0.17	**0.15**	0.72	0.50	0.33	**0.15**
12	0.31	0.15	0.15	0.93	0.73	0.33	**0.14**
16	0.22	0.16	0.24	1.21	0.78	0.42	**0.15**

Figure 4.9: Homogeneity selection under WGN $\sigma = 8$ (a) and $\sigma = 4$ (b).

We also tested the proposed method in video signals and Fig. 4.10 shows average result of noise estimation with and without using temporal data for the first 100 frames of two sequences. Collaboration of inter-frame weighting (33), (34) and temporal stabilization (21) improves the estimation. In this figure, we also compare to [35] as closest related work from Table 4.1.

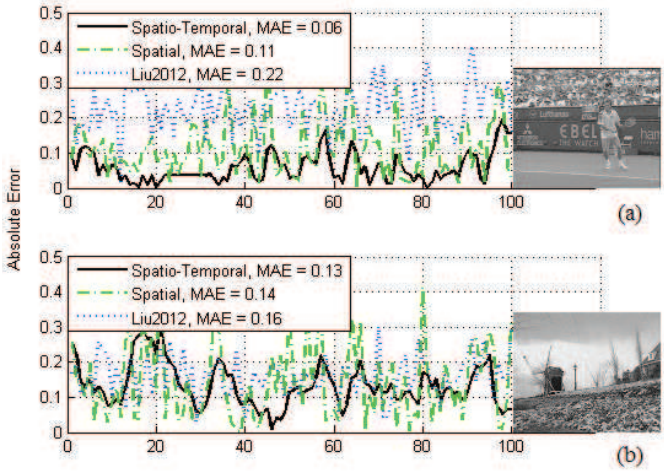

Figure 4.10: Stability of the proposed method in video signals under WGN $\sigma = 8$ with and without temporal weights. We give the mean of absolute error (MAE) over 100 frames of the *Stefan* and *Flower* sequences. Both inter-frame weighting (33), (34) and estimate stabilization (21) led to better estimate compared to Liu2012 [35].

4.4.3 Signal-dependent white noise

To evaluate the performance of the proposed estimation of SDWN, we tested six state-of-the-art approaches Yang2010 [33], [34, 35, 50], Tian2012 [32], and Ghazal2011 [31] on seven real-world test images see Fig.4.11, *intotree* from *SVT HD Test Set*, *tears* from *Mango Blender* and 5 other real-world noisy images that were taken in raw mode, where noise is visibly signal-dependent. To objectively evaluate the SDWN estimator without a reference frame, we combine the denoising method BM3D[20] with noise levels provided from ours and related estimators. The output performance is verified through the no-reference quality index MetricQ [36]. Table 4.2 compares MetricQ of denoised images with a higher value indicating better quality. The proposed method yields higher quality than related methods, where [32] and [50] achieve closest results. IVHC avoids underestimation by selecting the cluster with higher variance. Fig.4.12 shows examples of selected homogeneous clusters and Fig.4.13 shows visual comparison of noisy and noise-reduced image parts. As

can be seen, by using IVHC noise is better removed.

Table 4.2: MetricQ comparison of SDWN removal.

Image	Ref [33]	Ref [34]	Ref[35]	Ref[50]	Ref [32]	Ref [31]	Ours
Church	10.35	7.90	10.10	8.41	10.69	10.70	**11.08**
Intotree	9.34	7.71	7.24	8.98	10.56	10.06	**11.49**
Painting1	22.48	17.19	20.37	25.20	22.26	21.57	**25.27**
Painting2	19.58	15.62	16.86	20.14	20.67	20.11	**21.83**
Office	12.08	10.01	10.18	11.93	11.60	10.61	**13.10**
Room	11.06	9.56	10.31	11.18	10.84	10.01	**12.49**
Tears	12.05	11.09	10.89	11.22	12.23	12.02	**14.14**
Average	13.85	11.30	12.28	13.87	14.12	13.58	**15.63**

Figure 4.11: Real-world images corrupted with SDWN: *room* (1296×968), *painting1* (1296×968), *painting2* (1296×968), *church* (1296×968), *intotree* (1920×1080), *tears* (1600×1080) and *office* (1400×1080).

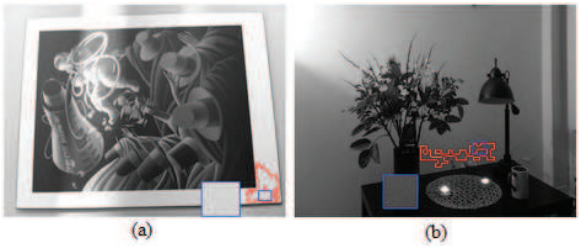

Figure 4.12: Examples of homogeneity selection for real SDWN.

We have also evaluated our SDWN estimator to denoise video signals using BM3D. Fig. 4.14 confirms the better quality of our method compared to closest related methods (from Table 4.2) for 150 frames of the *intotree* sequence.

4.4.4 Signal-dependent spatially correlated noise

If the observed noise is SDSCN, downscaling has the effect of converging it to white. This in turn leads to better patch selection under processed noise. Moreover, since our method uses a large patch

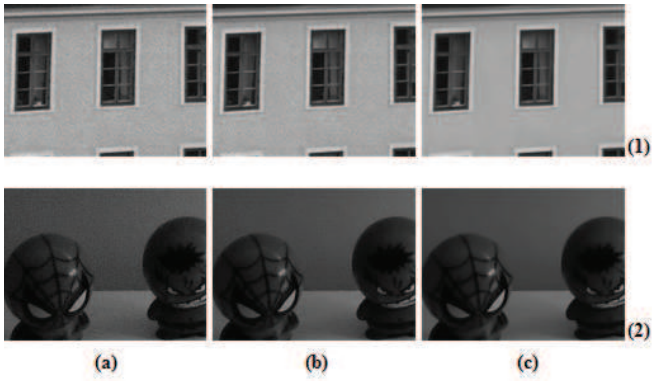

Figure 4.13: Real-noise removal examples using BM3D. (a) original. (b) noise estimated using [33]. (c) noise estimated using IVHC. Noise is left in (b) while it is efficiently removed in (c).

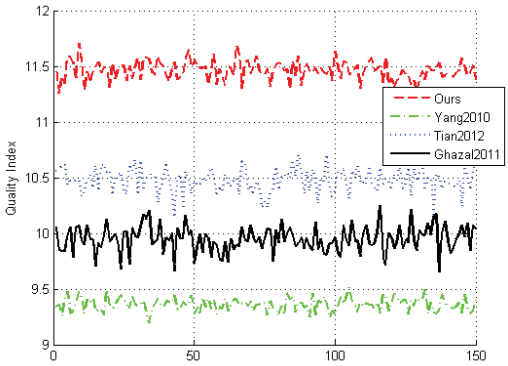

Figure 4.14: MetricQ of real noise removal using different noise estimators Yang2010 [33], Tian2012 [32], Ghazal2011 [31] and ours for *Intotree* sequence.

size, it leads to include more low frequencies and more realistic estimation. Fig. 4.15 shows better performance of the proposed method with γ adjustment in (4), and compared to the related method [35] (which we selected since it is closest to our method under $\sigma = 8$ in Table 4.1). To evaluate our method under real-world processed noise, we chose 6 images (4 from *iPhone 5* and 2 from *iPhone 6*) and apply BM3D[20] using noise levels provided by [34], [35], and proposed IVHC. Table 4.3 and Fig. 4.16 show that objectively and subjectively noise is better removed based on IVHC.

Table 4.3: Real-world processed noise removal using BM3D for 6 images captured by smartphones.

Method:	Ref [34]	Ref [35]	Ours
Average MetricQ:	13.95	15.34	18.77

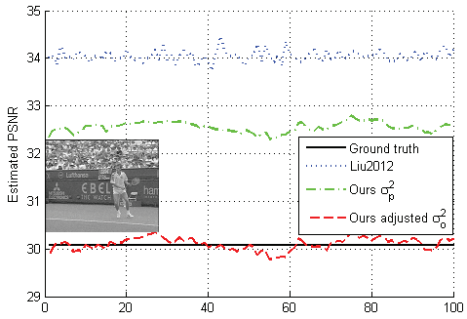

Figure 4.15: Processed synthetic noise in *Stefan* video: σ_p^2 and $\sigma_o^2 = \gamma\sigma_p^2$ in PSNR (original WGN $\sigma = 8$ then filtered by bilateral filter [21]) compared to Liu2012 [35].

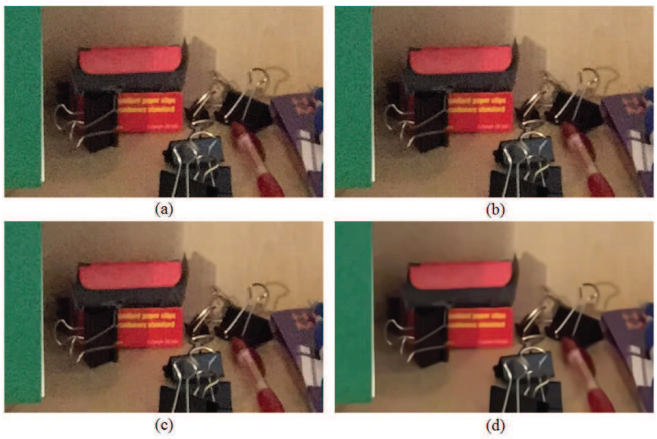

Figure 4.16: Real-world processed noise removal using BM3D: (a) original image has MetricQ=14.61 , (b) denoised based on [35] has MetricQ=15.77, (c) denoised based on [34] has MetricQ=18.15, and (d) denoised based on IVHC has MetricQ=23.32. Noise is much better removed in (d).

4.4.5 Noise level function

We applied the proposed NLF estimation on images with synthetic and real SDWN. The ground-truth for real SDWN images has been extracted manually (i.e., subjectively extracted homogeneous regions). Two state-of-the-art methods [6] and [52] are selected for comparison. Fig. 4.17 shows NLF results and Table 4.4 shows the root mean squared error (RMSE) and the maximum error comparison. Proposed IVHC has a better performance of finding the noise level peak especially when the level is greater in higher intensities (e.g., *Intotree* signal).

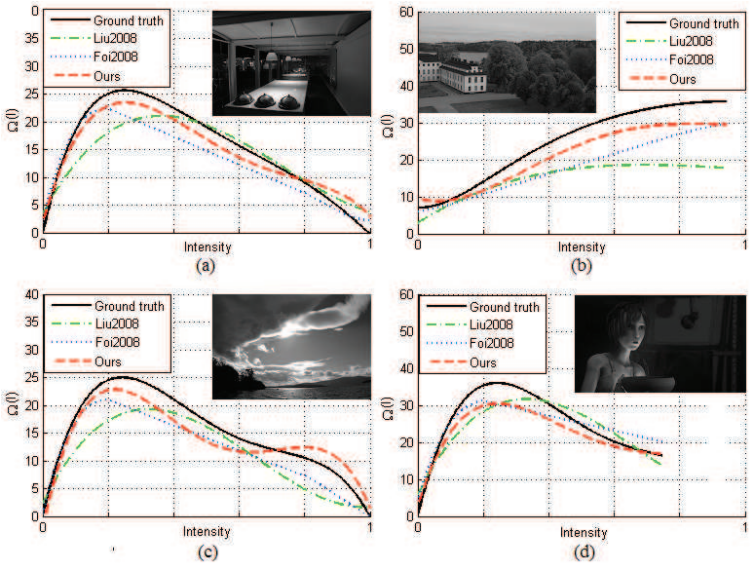

Figure 4.17: Estimated NLF for SRx100II (a), Intotree (b), Salpha77 (c) and Sintel (d). Noise in (a) and (b) is real and in (c) and (d) synthetically added. Liu2008 [6], Foi2008 [52], and ours are compared to ground-truth.

Table 4.4: RMSE and maximum of error of NLF in noisy images *SRx100II* (real), *Intotree* (real), *Salpha77* (synthetic) and *Sintel* (synthetic).

Image	Ref [6] Liu2008		Ref [52] Foi2008		Ours	
	RMSE	MAX	RMSE	MAX	RMSE	MAX
SRx100II	3.41	7.29	3.30	5.47	**1.99**	**3.35**
Intotree	11.17	17.87	7.31	9.95	**4.10**	**6.04**
Salpha77	4.40	7.45	3.38	5.63	**2.52**	**3.89**
Sintel	3.88	7.44	3.49	6.03	**3.55**	**5.59**
Average	5.71	10.01	4.37	6.77	**3.04**	**4.72**

4.4.6 Camera settings and user input

The more image information is provided, the more reliable estimation can be performed. Capturing properties if available as a meta-data can be useful for guiding the cluster selection procedure. To test this, we have selected 10 highly-textured images taken by a mobile camera (*Samsung S5*) in the burst mode without motion. First, we manually found the ground-truth peak of the noise by analyzing the homogeneous patches and temporal difference of burst mode captured images. Second, we applied our noise estimator using only Intra-frame weights and the estimated PSNR when compared the ground-truth show an average estimation error of 1.2 dB. In the last step, we have adapted both the patch selection threshold $\tilde{H}_{th}(l)$ in (12) and variance margin weight $\omega_7(l,k)$

in (29) to the meta-data brightness value and ISO. This led to more reliable estimation with average error of 0.34dB in PSNR.

Performance of image and video processing methods improves if expertise of their users can be integrated. Our method easily allows for such integration, for example, if the user of an offline application can define possible noise range, the proposed variance margin (29) can be used to reject the out of range clusters.

4.4.7 Implementation issues

The source codes of [33–35, 50] was obtained from the authors' websites. All, but [33], are implemented using *Matlab* combined with *MEX* functions. [33] is a pure *MEX* code. We have implemented [32], [31], and our method using *Matlab* combined with *MEX* functions. For a fair comparison to [33], we have implemented our method using pure *MEX* functions. We measured the processing time of related methods using a 3.07 GHz, i7 CPU. Table 4.5 shows the results. The proposed method is significantly faster than the related methods (in both *Matlab* and *MEX*). This is mainly because our method deals with down-sampled images.

Table 4.5: Average of elapsed time in seconds to process 10 HD (1920×1080) frames from *intotree* sequence.

Ref [34] *Matlab*	Ref[35] *Matlab*	Ref[50] *Matlab*	Ref [32] *Matlab*	Ref [31] *Matlab*	Ours *Matlab*	Ref [33] *Mex*	Ours *Mex*
17.70	7.65	9.73	21.01	10.06	0.98	1.15	0.03

4.5 Conclusion

Noise estimation methods typically assume video noise is white Gaussian. This thesis bridges the gap between the relatively well studied white Gaussian noise and the more complicated white signal-dependent and non-white processed types. We proposed a noise estimation method that widens noise assumptions based on the classification of intensities (or color) and on the extraction of weights using statistical noise property and homogeneous regions in the images. The classification of intensities into connected clusters of homogeneous patches allowed us to well approximate the noise level function. We estimated the degree of processed versus white noise as a ratio of low

to high frequency energies in the input image. Another important feature of our method is its use of both the input noisy image and its down-scaled version. This allowed better differentiation of noise and structure and fast processing. We have shown that the developed visual noise estimation method robustly handles different type of visual noise: white Gaussian, white Poissonian-Gaussian, and processed (non-white) that are visible in real-world video signals. Our simulation results showed the superiority of the proposed method both in accuracy and speed.

For the real-world experiment, simulation results have been tested for very challenging sequences. Simulation results in this thesis are given for the gray-level format of test video sequences. However, we have tested our method on color sequences and it also outperforms related work.

Chapter 5

Transformation of WGN Filter to Handle SDSCN

5.1 Overview

Recent advances in video denoising [8–10, 12, 15, 17–19] oversimplifies noise model assuming WGN, however, as discussed in the chapter 2 noise is often signal-dependent spatially correlated (SDSCN). This led us to the problem of how we can use these effective methods to remove SDSCN. Thus, we propose an approach that converts a WGN filter into a filter which able to remove SDSCN. This approach comprises four steps; 1- equalization of noise level in the intensity domain, 2- equalization of noise in the frequency domain using the property of the noise in different image scales, 3- remove the resulting noise using any WGN denoiser, and de-equalization to get the original histogram. Our approach removes the SDSCN using any WGN filter. To make the noise WGN we convert SDSCN to WGN by equalizing the noise in both intensity and frequency domains (see Figure 5.2). We use an invertible transform to map pixel intensity into another histogram where noise becomes signal-independent. Thus, we can apply a WGN filter to remove noise and then convert back the intensities into the original histogram as shown in the Figure 5.1. In order to address the non-uniformity of the noise level in frequency domain we propose a multi-scale WGN filtering.

In the following, section 5.2 discusses our proposed method to equalize the noise level in the intensity domain, section 5.3 explains our proposed noise level equalizer in the frequency domain,

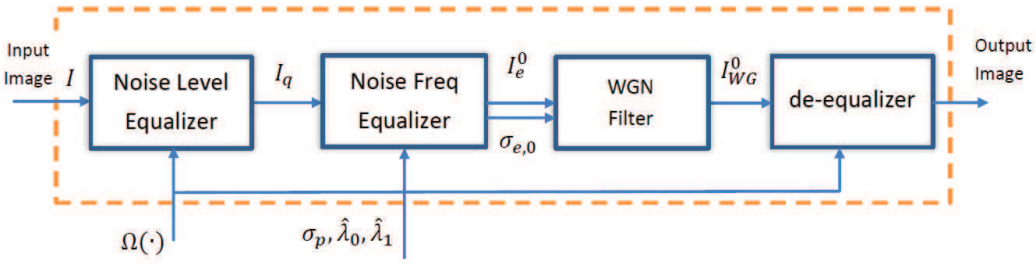

Figure 5.1: The proposed approach to address signal-dependent and spatially correlated noise.

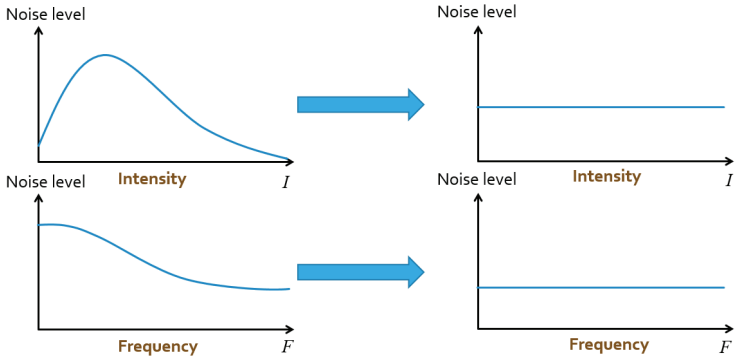

Figure 5.2: Noise level equalization in both intensity and frequency domains.

and section 5.4 presents simulation results to validate our approach.

5.2 Noise level equalization (SDSCN to SCN)

We propose an invertible transform that maps the intensity values to another histogram where the noise becomes signal-independent. The forward transform (noise equalizer) equalizes the level of noise for all intensities. The inverse transform de-equalizes the pixel intensities to create the same histogram as the original input. Once the noise is equalized, we can use signal-independent denoising algorithms to remove signal-dependent noise optimally (see Figure 5.1).

Let us consider a noise equalizer $\nu(I)$ that maps the intensity value I to a noise equalized $I_q = \nu(I)$. $\nu(\cdot)$ should be invertible (Figure 5.1), i.e., $\nu^{-1}(\nu(I)) = I$ where $\nu^{-1}(\cdot)$ is the de-equalizer. The NLF $\Omega(I)$ defines the variation of noise variance and we define $\Omega^*(I) = \sqrt{\Omega(I)}$ as a functions defines the STD of noise (see Figure 5.3). Let us consider a linear function for both $\nu(I)$ and $\Omega^*(I)$ as $\nu(I) = c_\nu I$ and $\Omega^*(I) = c_\Omega I$ where c_ν and c_Ω are constants and $\nu^{-1}(I) = \frac{I}{c_\nu}$ is

the de-equalizer functions. Then,

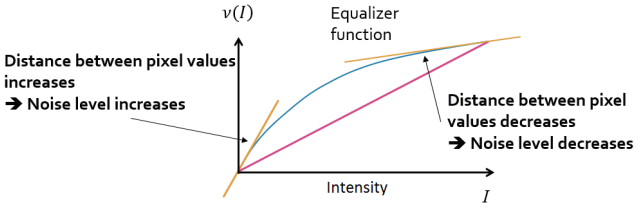

Figure 5.3: Principle of noise level equalizer. The slope of intensity mapping function changes the noise level.

$$\Omega^*(\nu(I)) = \Omega^*(c_\nu I) = c_\nu \Omega^*(I). \tag{37}$$

Assuming we divide the whole intensity range into small intervals where both $\nu(I)$ and $\Omega^*(I)$ can be assumed as linear function of I for each interval. Then c_ν and c_Ω becomes local slopes. In this case, (37) means the slope of $\nu(\cdot)$ defines the STD of noise at intensity I. We use this property to define the noise equalizer. Considering $\frac{\partial \nu(I)}{\partial I}$ as the slope of $\nu(I)$ at the intensity I and our objective is to have a fixed noise level for all intensities we can write,

$$\Omega^*(\nu(I)) = \frac{\partial \nu(I)}{\partial I}\Omega^*(I) = \sigma^*_{eq}. \tag{38}$$

where, σ^*_{eq} varies according to Ω^* curve and represents the STD of the noise after equalization. (38) can be rewritten as,

$$\nu(I) = \int_0^I \frac{\sigma^*_{eq}}{\Omega^*(I)} dI, \tag{39}$$

and the slope of $\nu(I)$ becomes $\frac{\sigma^*_{eq}}{\Omega^*(I)}$. As described in chapter 2 we use $\sigma_p = \text{MAX}(\Omega^*(I))$ as the input parameter of WGN filter. Thus, our objective is to equalize the noise and at the same time keep the noise level at σ_p. That is $\sigma^*_{eq} = \sigma_p$. Since $\Omega^*(I) > 0$, $\nu(I)$ is strictly increasing and therefore invertible, however, a definite form of the integral in the (39) is not available. In many cases such as our proposed noise estimator, $\Omega^*(I)$ is estimated numerically and thus $\nu(I)$ can be estimated numerically as well. We use a piecewise linear model to describe the $\Omega^*(I)$. We divide the intensity interval, $[0, 1]$ into N_{eq} equal sub-interval and for each we model the $\Omega^*(I)$ as a linear function. Let us consider the $z_i = \Omega^*(\frac{i}{N_{eq}})$ where $0 \leq i \leq N_{eq}$. For i^{th} interval the line becomes $a_i I + b_i$ where,

$a_i = (z_{i+1} - z_i)N_{eq}$ and $b_i = z_i - (z_{i+1} - z_i)i$. Thus for each interval the equalization function (39) becomes,

$$\nu(I) = \int \frac{\sigma^*_{eq}}{\Omega^*(I)} dI = \tag{40}$$
$$\sigma^*_{eq} \times \begin{cases} \frac{1}{a_i}\mathbf{ln}(a_i I + b_i) + C_i, & \text{if } |a_i| > \boldsymbol{T}^r_{eq} \\ (\frac{a_i}{2z_i z_{i+1}})(I - \frac{i}{N_{eq}})^2 + \frac{1}{z_i}(I - \frac{i}{N_{eq}}) + C_i, & \text{otherwise} \end{cases} \quad z_i \leq I \leq z_{i+1},$$

where $\mathbf{ln}(\cdot)$ is the natural logarithm function and $\boldsymbol{T}^r_{eq}$ is constant. When $a_i = 0$ or very small, for instance $|a| \leq \boldsymbol{T}^r_{eq}$ then ($\frac{1}{z_i} \approx \frac{1}{z_{i+1}}$) the and logarithmic function can be approximated due to the fact that,

$$\lim_{a_i \to 0} \frac{\mathbf{ln}(\frac{z_{i+1}}{z_i})}{a_i} = \frac{z_{i+1} - z_i}{a_i z_i}. \tag{41}$$

To define C_i in (40), we consider two conditions, the zero point and the continuity. Since the equalizer should not add any offset so $\nu(0) = 0$, and C_0 can be defined according to (40) $C_i = -\frac{\mathbf{ln}(b_i)}{a_i}$ when $|a_i| > \boldsymbol{T}^r_{eq}$ and $C_i = 0$ otherwise. Since $\Omega^*(\cdot)$ is continuous, $\nu(\cdot)$ is continuous. From the continuity condition the last point of each interval should be equal to first interval of next interval as follows,

$$\mathbf{ln}(a_{i+1}\frac{i}{N_{eq}} + b_{i+1}) + C_{i+1} = \mathbf{ln}(a_i \frac{i}{N_{eq}} + b_i) + C_i, \quad |a_i|, |a_{i+1}| > \boldsymbol{T}^r_{eq}, \tag{42}$$

Thus, C_{i+1} can be calculated from C_i. Starting from C_0, all C_i can be calculated. To define $\boldsymbol{T}^r_{eq}$ we consider the case that **ln** can be linearly approximated i.e., $\mathbf{ln}(a_i I + b_i) \approx \mathbf{ln}(b_i) + \frac{a_i I}{b_i}$ or $\frac{a_i I}{b_i} \leq 0.1$ or $\frac{|z_{i+1} - z_i|}{z_i} = \frac{a_i}{N_{eq} z_i} \leq 0.1$. In our NLF model we assume a limited slope a_i (see chapter 2). This forces the z_i to have values higher than a limit. We consider this limit as $\frac{\sigma_p}{2.5}$ (equal to 8dB lower than σ_p). Thus,

$$\boldsymbol{T}^r_{eq} = \frac{N_{eq}\sigma_p}{25}. \tag{43}$$

Figure 5.4 shows a random noise level function $\Omega^*(I)$ (top left) and the equalizer derivative (top right) which is $\frac{\sigma^*_{eq}}{\Omega^*(I)}$ and the corresponding equalizer, $\nu(I)$ compared to a neutral mapping which does not change the input, i.e. $\nu(I) = I$.

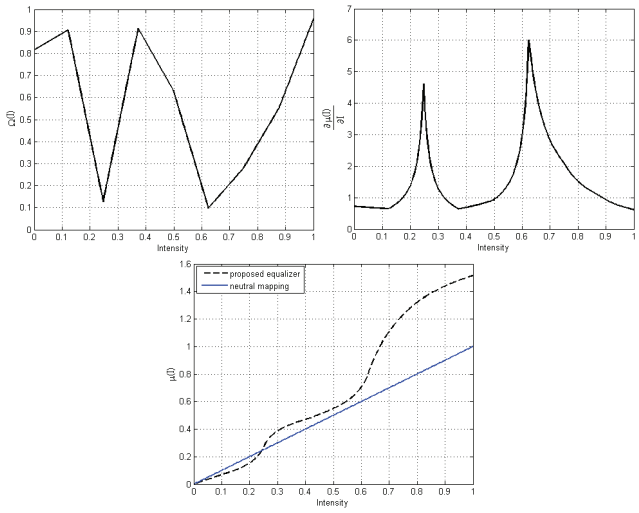

Figure 5.4: Top left is a random NLF $\Omega^*(I)$ with $N_{eq} = 8$. Top right is the derivative of equalizer function and bottom is the corresponding equalizer function compared to a neutral mapping.

The inverse equalizer function $\nu^{-1}(I)$ of (40) is,

$$\nu^{-1}(I\sigma^*_{eq}) = \begin{cases} \frac{\mathbf{exp}(Ia_i - C_i a_i) - b(k)}{a_i}, & \text{if } |a_i| > \boldsymbol{T}^r_{eq} \\ \frac{I - \nu(\frac{i}{N_{eq}})}{[\nu(\frac{i+1}{N_{eq}}) - \nu(\frac{i}{N_{eq}})]N_{eq}} + \frac{i}{N_{eq}}, & \text{otherwise} \end{cases} \quad \nu(z_i) \leq I \leq \nu(z_{i+1}), \tag{44}$$

where $\nu^{-1}(\nu(I)) = I$.

In computing the forward noise equalizer we start from the subsampled $\Omega^*(I)$, i.e., z_i and we compute the a_i and b_i. Using the zero-point and continuity conditions we find the C_i. Considering that $i = \lfloor N_{eq} I \rfloor$ then we can compute $\nu(I)$ via (39). The backward (inverse) de-equalizer, however, is more complex since finding the i requires to check all the points to meet the $\nu(z_i) \leq I \leq \nu(z_{i+1})$. Firstly, we should compute the $\nu(z_i)$ (or use the already computed in the forward) and search for the i that lies between $\mu(z_i)$ and $\nu(z_{i+1})$. Since $\nu(z_i)$ is strictly increasing this search requires $\mathbf{log}_2(N_{eq})$ operations. By increasing the N_{eq} the complexity of search increases, however, the $\mathbf{ln}(\cdot)$ and $\mathbf{exp}(\cdot)$ are less used considering $|a_i| \leq \boldsymbol{T}^r_{eq}$ condition for all point. Note that $\boldsymbol{T}^r_{eq}$ increases as N_{eq} increases. In our implementation, we use $N_{eq} = 8$ which is sufficient for non-HDR images since the slope of the NLF is not large.

5.3 Noise frequency equalization to handle SCN

Noise in I_q is SCN. In order to denoise I_q, denoiser should treat the noise differently in different frequencies. In pixel domain, HF components of an image is represented in fine image scale (or high resolution) and LF ones in coarse scale (or low resolution). We assume noise in coarsest scale is white and the energy (here magnitude) of noise is equally distributed in all frequencies. We use this property to equalize the noise power in all other scales. Assuming a transform can equalize the noise power for all intensities and frequencies, we can use a WGN filter to remove the transformed noise. Assuming noise is equalized for all intensities, we now propose a method to transform SCN to WGN. Let us assume we decompose the image I^l into downsampled LF I^{l+1} and HF $I^l - I^{l+1}$ (see Figure 5.5). In SCN the energy of noise is mostly concentrated in the LF. Assuming the STD of LF noise, which is WGN, is $\sigma_{p,l+1}$, we suppress the noise in I^{l+1} using a WGN filter $I^{l+1}_{WG} = \mathbf{WGD}(I^{l+1}, \sigma_{p,l+1})$. Thus, $I^{l+1}_{WG} + I^l - I^{l+1}$ will contain only HF noise. In order to make the noise frequency level equalized, we need to restore back part of LF noise as

$$I^l_e = I^l + g^l \left[I^{l+1}_{WG} - I^{l+1} \right], \;\; 0 \leq g^l \leq 1, \tag{45}$$

where I^l_e is the image with equalized noise power in frequency components and g^l is the restoration factor of multi-scale filtering. $\mathbf{WGD}(\cdot)$ can be any spatial or temporal filter such as those we proposed in sections 6.2 and 6.3. Figure 5.5 shows the pyramid of noise equalization using two levels of decompositions in 1-D. A heavily processed noise can be decomposed into different moderately processed noises and we can use a WGN filter to remove each of these noises. The problem is now to first find g^l in (45) to obtain I^l_e and second to find the STD of noise in the I^l_e to remove noise in I^l_e afterward.

Figures 5.6 and 5.7 shows the block diagram of proposed SCN removal using WGN filter using one and two levels of decompositions. To find STD of noise $\sigma_{e,l}$, assuming we decompose the image into $l_{\max}$ scales where $l_{\max}$ contains WGN. For each scale l, we require the STD of noise $\sigma_{e,l}$. This requires calculating $l_{\max}$ STDs which are calculated based on $\{\hat{\gamma}_0, \hat{\gamma}_1, \ldots, \hat{\gamma}_{l_{\max}-1}\}$, $\hat{\gamma}_l \geq 1$ as discussed in section 4.3.2. Downsampling rate defines how the spectrum is divided into LF and HF. In order to have maximum information in both LF and HF bands, we propose dividing the spectrum

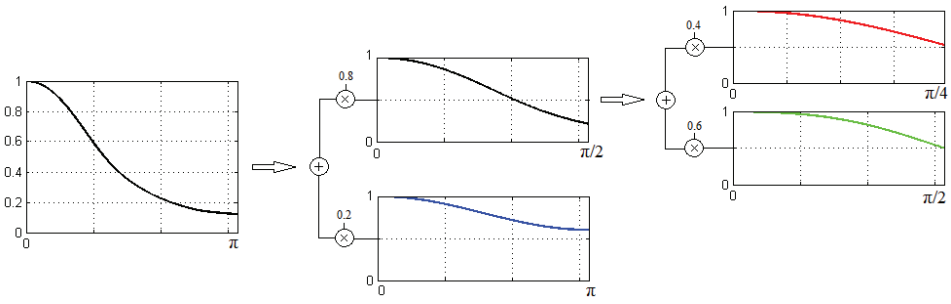

Figure 5.5: Example for noise frequency equalization. Highly frequency-dependent noise (left) is decomposed to three (right and middle-bottom) less frequency-dependent noise.

equally with downscaling by 2. In 1-D downsampling, depending on the starting index (i.e., 0 or 1) results in two different options. In 2-D downsampling, depending on the starting index (horizontally or vertically), 4 possibilities exist. We take all information into account by rearranging the 4 images according to Figure 5.8. Figure 5.9 shows the downsampled and rearranged *Lena*. At each level level of $\hat{\gamma}_l$ defines the degree of spatial correlation for noise at the l^{th} scale of image. In practice after maximum 2 level of decomposition, noise becomes white, thus we consider $l_{\max} = 2$. The parameters of noise frequency equalizer becomes σ_p, $\hat{\gamma}_0$, and $\hat{\gamma}_1$ (see Figure 5.1).

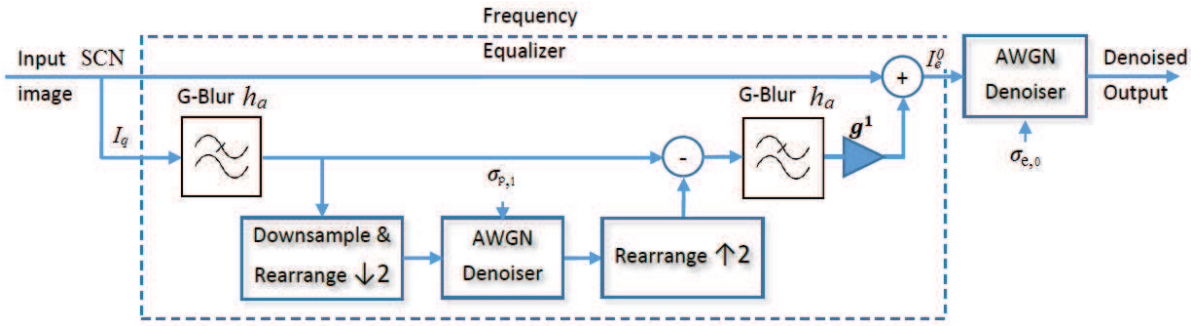

Figure 5.6: Proposed SCN filter using WGN filter using 1 levels of decompositions.

To find $\hat{\gamma}_l$, let us first define the downsampling process. SCN has a LF nature and downsampling process decreases the dependency to frequency (see chapter 2). We define the h_a as the anti-aliasing filter used for our downsampling process. We use 3×3 Gaussian filter (G-blur) for this purpose. Optimal sigma value for G-blur depends on the cut-off frequency and correlation of signal pixels. Our experiments show sigma of 0.75 is an optimal value. Assuming the noise is WGN in the l^{th} scale with power of $\sigma^2_{p,l}$, the noise variance of downsampled image becomes $\sigma^2_{p,l} \sum h_a^2$. We define

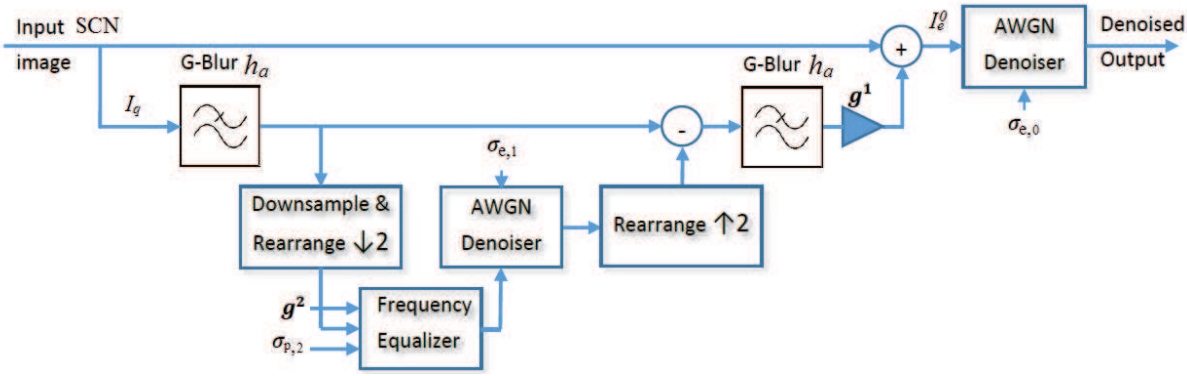

Figure 5.7: Proposed SCN filter using WGN filter using 2 levels of decompositions.

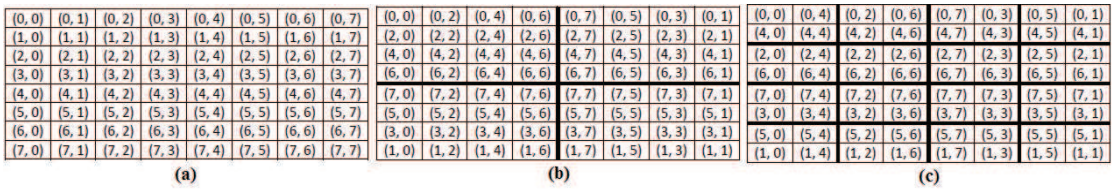

Figure 5.8: Downsampling and rearranging an 8×8 image. (a) shows original pixel positions. (b) shows pixel positions for 4 different downsampled (by 2) images. (b) shows pixel positions for 16 different downsampled (by 4) images.

the $\hat{\gamma}_l$ as a ratio between expected value (if the noise is WGN) and observed value as

$$\hat{\gamma}_l = \frac{\sigma_{p,l+1}}{\sigma_{p,l}\sqrt{\sum h_a^2}}. \tag{46}$$

If the noise is white, $\sigma^2_{p,l+1} = \sigma^2_{p,l}\sum h_a^2$ and $\hat{\gamma}_l = 1$ and as the noise becomes more spatially correlated $\hat{\gamma}_l$ increases.

In order to find g^l, we consider two margin constraints. In case of WGN all of filtering should

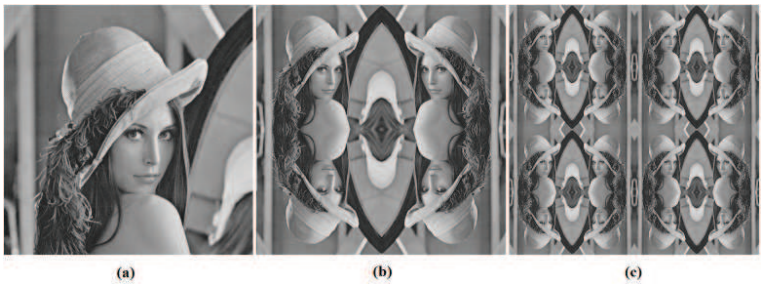

Figure 5.9: Downsampling and rearranging in for 2 scales (a) original image, (b) downsampled by 2 and rearranged, and (c) downsampled by 4 and rearranged.

be accomplished in finest scale, thus when $\hat{\gamma}_l = 1$ then $g^l = 0$. For when the noise is spatially correlated (by a G-blur h) according to (46) $\hat{\gamma}_l = \frac{\sqrt{\sum[h*h_a]^2}}{\sqrt{\sum h^2}\sqrt{\sum h_a^2}}$. In this case the anti-aliasing filter h_a is applied on the noise that is already processed. When $h = h_a$ most of noise power is concentrated in the scale $l+1$ and to equalize the noise, all of noise in this scale should be removed, i.e., $g^l = 0$. Now we define g^l using linear approximation as

$$g^l = \text{MIN}(\frac{1-(\hat{\gamma}_l)^{-1}}{1-\frac{\sum h_a^2}{\sqrt{\sum[h_a*h_a]^2}}}, 1), \tag{47}$$

where $\frac{\sum h_a^2}{\sqrt{\sum[h_a*h_a]^2}} = 0.56$.

After applying (45) noise in I_e^l is WGN. We need to find the power of equalized noise in I_e^l to set the second input parameter of WGN filter $\mathbf{WGD}(I_e^l)$. Here also we utilize two margins. When noise is WGN then $\sigma_{e,l} = \sigma_{p,l}$. When the noise is extremely correlated by h_a, $g^l = 1$ and the noise power originally was $\frac{\sigma_{p,l}^2}{\sum h_a^2}$. Assuming $\mathbf{WGD}(\cdot)$ is ideal, the noise at I_e^l is equal to applying the filter $h_a - h_a * h_a$ to original WGN noise. $h_a - h_a * h_a$ has the peak of frequency response 0.25. This means if the noise is WGN with STD of 1 filtered by h_a then $\sigma_{e,l} = 0.25$. However, the original STD is assumed to be $\frac{\sigma_{p,l}^2}{\sum h_a^2}$. Thus, $\sigma_{e,l} = \frac{\sigma_{p,l}}{4\sqrt{\sum h_a^2}}$. Since $\mathbf{WGD}(\cdot)$ can not be ideal we consider $\frac{1.2\sigma_{p,l}}{4\sqrt{\sum h_a^2}}$.We then estimate the input parameter of $\mathbf{WGD}(\cdot)$, noise STD $\sigma_{e,l}$ at l^{th} scale as

$$\sigma_{e,l} = [\boldsymbol{c}_w^r + (1-\boldsymbol{c}_w^r)(\hat{\gamma}_l)^{-1}]\sigma_{p,l}, \tag{48}$$

where $\boldsymbol{c}_w^r = 0.4$. In case of $\hat{\gamma}_l = 1$ noise is WGN and the input should be the STD of WGN, i.e., $\sigma_{e,l} = \sigma_{p,l}$. Since the number of taps in h_a is small (3×3) and introduces aliasing, to compensate for that we propose a second lowpass filtering before equalization and we modify (45) as

$$I_e^l = I^l + g^l \cdot h_a * \left[I_{WG}^{l+1} - I^{l+1}\right], \;\; 0 \le g^l \le 1, \tag{49}$$

5.4 Experimental results

In this section we provide the experimental results for proposed SCN and signal-dependent noise removal approach using WGN filter.

5.4.1 Noise level equalization (SDSCN to SCN)

In order to test noise level equalization idea we have chosen 5 different NLF. Figure 5.10 shows the 5 different NLFs used in these experiments. We added signal-dependent noise with peak value $\sigma_p = 12$ to 16 images (*Lena*, *Barbara*, *Peppers* and 13 images from TID2008) using 5 NLFs in Figure 5.10. We used a high performance spatial filter DDID for denoising. The algorithm of DDID has a pixel based operation and for each pixel different noise value can be considered which makes the algorithm suitable for our experiments. We used three different ways to denoise the noisy images. Firstly, we used the maximum level of the noise $\sigma_p = 12$. Secondly, we used a per-pixel adapted noise meaning for each pixel it uses the noise level according to its intensity. Finally, we used the Figure 5.1 approach with equalizer and de-equalizer. The input noise in this case assumed to be uniform σ_p. Figure 5.11 shows the PSNR of three different denoising ways for 5 different NLFs. In average using equalizer increases the PSNR compared to adapted noise. These experiments verify that noise equalization can be used to address the signal-dependency without degrading the quality.

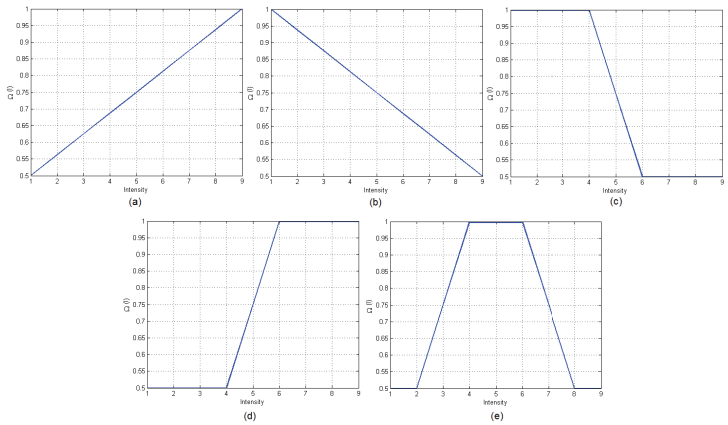

Figure 5.10: 5 different NLFs used to test the performance of noise equalizer. $\Omega_{eq} = 0.75$ for all cases except for (c) which is $\Omega_{eq} = 0.72$.

5.4.2 Handling SCN

In order to test our proposed SCN removal we have considered 3 state-of-the-art denoising methods, BM3D, VBM3D, and RF3D and we have compared the results. We have conducted two experiments for still image and video denoising. For still image experiments we have selected first frame from

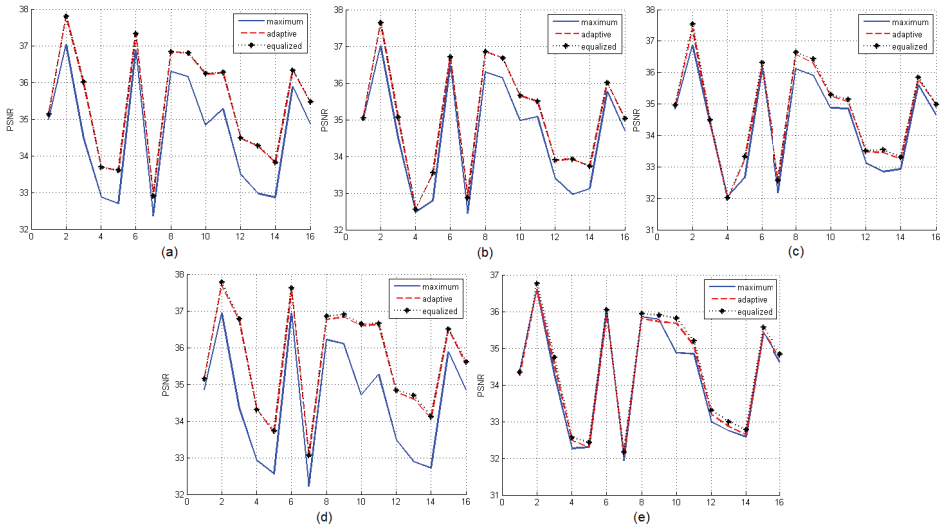

Figure 5.11: PSNR results for denoising of synthetic signal-dependent noise using different NLFs in Figure 5.10. In average using equalizer increases the PSNR compared to adapted noise. Horizontal axis is image number.

10 video sequences (see Figure 5.12) and we added SCN. To generate SCN we have filtered WGN using 3×3 Gaussian blur using two different sigma 0.55 and 0.65 and two different STD 15 and 20. We tuned the key input parameter of BM3D (noise STD) to find the highest performance. For our method the input parameters, however, are fixed and computed as described in the section 5.3. For instance the when sigma is 0.65 and STD is 20 $\sigma_p = 9.1$, $\hat{\gamma}_0 = 1.6$ and $\hat{\gamma}_1 = 1$. In this experiments the proposed method is used the BM3D as the WGN filter (see figure 5.7). Figure 5.13 compares the PSNR results for dataset-1. For both noise profile and all videos the proposed filter provide results with higher quality in PSNR since it can better remove SCN. Figure 5.15 compares the visual results for the SCN with sigma of 0.65.

The same experiment has been repeated for 10 video sequences *Bus*, *Flower*, *Foreman*, *News*, *ParkJoy*, *Rush hour*, *Soccer*, *Stefan*, *Stem*, and *Tennis*. We compared the denoising result of video denoiser VBM3D and proposed filter using VBM3D as the denoiser. VBM3D is tuned to provide highest possible quality in PSNR. We have added two different SCN with same statistics as the previous experiment. Figure 5.16 compares the average PSNR of 150 frames of 10 videos from dataset Figure 5.14. For all cases the proposed filter better removes SCN and provides higher PSNR. We have also tested the result of proposed filter that uses VBM3D as WGN filter compared

to RF3D that is designed to remove SCN. For RF3D we have changed the input parameter which is the power spectral density (PSD) of noise to reach the highest quality (PSNR). Figure 5.17 compares the results for 150 frames. The SCN in this experiment is the same as Figure 5.16 (b).

Figure 5.12: Still image dataset-1. First frame of 10 videos *Akiyo*, *Coastguard*, *Flower*, *Foreman*, *Hall*, *News*, *Sean*, *Stefan*, *Tennis*, and *Bus*.

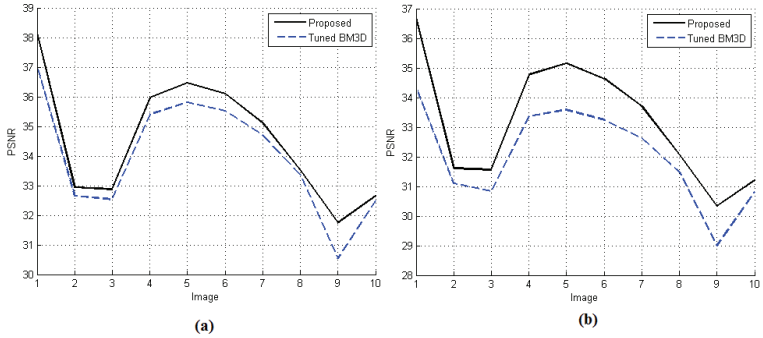

Figure 5.13: (a) average PSNR for denoised first frames using tuned BM3D and proposed under SCN (WGN with STD of 15 denoised by G-blur with sigma of 0.55). Average PSNR is 34.2dB for tuned BM3D and 33.6dB for proposed (b) PSNR for denoised frames using tuned BM3D and proposed under SCN (WGN with STD of 20 denoised by G-blur with sigma of 0.65). Average PSNR is 31.7dB for tuned BM3D and 32.8dB for ours.

Figure 5.14: First frame of our video dataset used in synthetic noise video experiments.

Figure 5.15: SCN removed by (a) tuned BM3D with PSNR=34.3dB (b) proposed with PSNR=36.7dB.

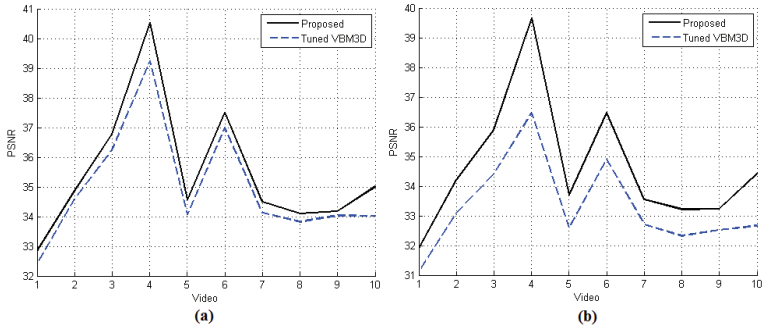

Figure 5.16: (a) average PSNR for denoised 150 frames of each using tuned VBM3D and proposed under SCN (WGN with STD of 15 denoised by G-blur with sigma of 0.55). Average PSNR is 34.6dB for tuned VBM3D and 35.06dB for proposed (b) PSNR for denoised frames using tuned VBM3D and proposed under SCN (WGN with STD of 20 denoised by G-blur with sigma of 0.65). Average PSNR is 33.0dB for tuned BM3D and 34.2dB for ours.

5.5 Conclusion

Many advances in removing WGN from videos encouraged us to develop a method that uses WGN filter to remove real noise which is often signal-dependent spatially correlated (SDSCN). We propose an approach that converts SDSCN to WGN using noise level equalization in both intensity and frequency domain. After this conversion, we use WGN filter to remove WGN and we apply de-equalization to get the original histogram. We use an invertible transform to map pixel intensity into another histogram where noise becomes signal-independent. In order to equalize the non-uniformity of the noise level in frequency domain we propose a multi-scale WGN filtering. Our results show that the proposed method is effective in removing SDSCN for many tested temporal, spatial and spatio-temporal denoising methods.

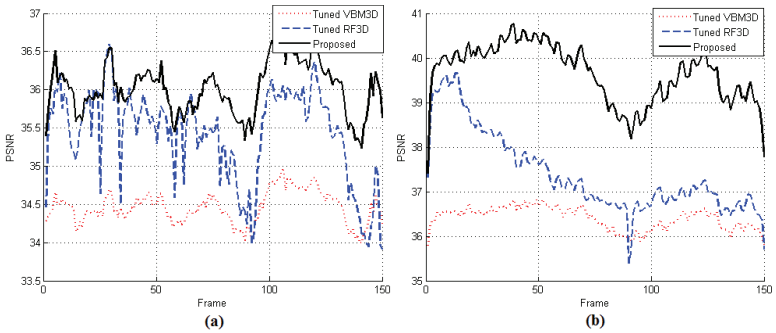

Figure 5.17: PSNR results under SCN noise for video sequences (a) *Foreman* (with the average PSNR of 34.4dB for tuned VBM3D, 35.3dB for tuned RF3D, and 36.0dB for proposed), and (b) *News* (with the average PSNR of 36.39dB for tuned VBM3D, 37.30 for tuned RF3D, and 39.60dB for proposed using VBM3D as WGN filter). SCN is the WGN filtered by G-blur with sigma of 0.65.

Chapter 6

Band-Limited Anti-Blocking Time-Space Filtering of Noise From Different Sources

6.1 Overview

In chapter 5, we proposed an approach to enable WGN filter to reduce SDSCN. Our approach can use any WGN filter, however, powerful video denoising techniques are either computationally complex or introduce blockiness due to block-based processing. To address this, we propose a fast band-limited anti-blocking temporal filter followed by a spatial filter to remove SDSCN. Our filter is band-limited which decreases the blocking artifact and increases the effectiveness of temporal filtering. Our temporal filter supports both recursive and symmetric temporal structure, however, for high-quality denoising, we propose using STF because of the following reasons. In high-quality applications (e.g., post-production) where processing lag (i.e., non-causal filter) and time is not critical but the quality is, it is essential to use maximum possible information such as data of forward frames (frames in future time). Another problem of recursive filter (RTF) is that depending on the starting point in time, the results will be different for a specific frame which is not acceptable in post-production application. Finally, in order to accurately estimate the amount of reduced noise

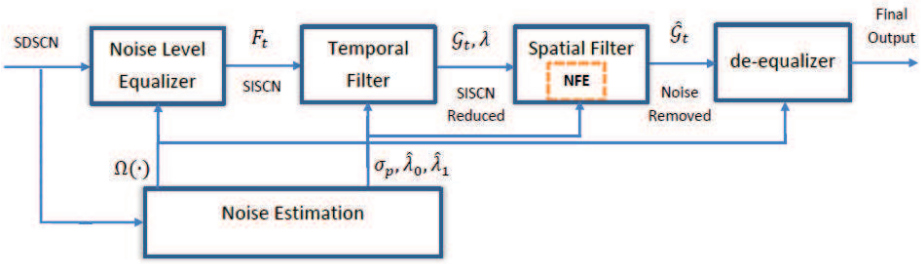

Figure 6.1: Simplified block diagram of proposed automated video denoiser system: noise estimation (chapter 4), transform SDSCN to WGN (chapter 5), and temporal and spatial filter (chapter 6).

after RTF, we need to store the temporal weights for many (theoretically infinite) previous frames which is not practical. The temporal information is not always useful (e.g., when motion estimation fails), thus we propose a spatial filtering to remove the residual noise left after temporal filtering. We use a combination of small-kernel and large-kernel pixel-domain and frequency-domain approaches to efficiently remove noise.

Figure 6.1 shows a simplified block-diagram for our automated video noise estimation and reduction. Our fast video denoiser comprises 1) SDSCN to SISCN transform, 2) motion estimation between the current frame and symmetric preceding and subsequent frames, 3) motion-compensated filtering using LMMSE estimator, 4) spatial filtering to remove residual noise left from temporal filtering , and 5) intensity de-equalization. We benefit from the speed of block-matching motion estimators, however, to minimize the blocking artifacts we utilize both band-limited filtering technique and two-band motion compensation. We also propose a procedure to correct the erroneous motion vectors by creating a homography from reliable motion vectors.

Our contribution is a time-space denoiser that 1) operates on a video signal in gray-scale or color space; 2) removes WGN, SDWN, SISCN, and SDSCN; 3) has an anti-blocking system in motion compensation and temporal error detection; 4) uses a fast dual (pixel and transform) domain spatial filter to estimate and remove residual noise of the temporal filter; 5) in-loop handles possible noise overestimation; 6) uses reliability factors to calculate weights in temporal filter; 7) corrects the erroneous motion vectors by creating a homography from reliable motion vectors; and 8) uses two-band motion compensation to eliminate blocking. The proposed band-limited time-space video filter BLTSF can be summarized as in Figure 6.2 and as in Algorithm 2.

Algorithm 2 Band-limited time-space filter BLTSF

1: **Estimate** the MVs in $2R$ preceding, and subsequent frames.
2: **Use** the *back-signal* $\mathbf{b}_t$ to find *fore-signal* $\mathcal{D}_{t+m}$ and motion-compensate one $\vec{\mathcal{D}}_{t+m}$.
3: **Compute** the coarse-level error probability using (71).
4: **Compute** the pixel-level motion error $\hat{\delta}_m$ and the estimator weights via (73).
5: **Filter** spatially the residual noise according to temporal reduction factor λ using (86).

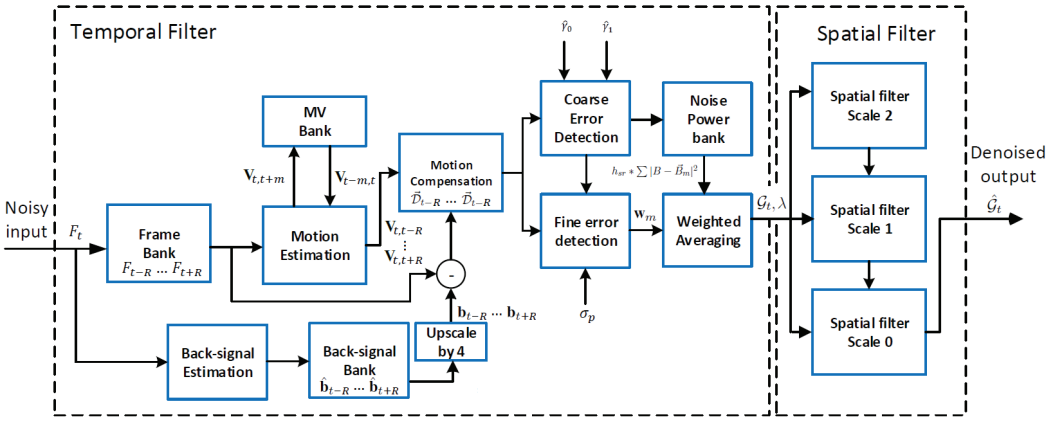

Figure 6.2: Block diagram of proposed time-space filter.

In the remainder of this chapter, section 6.2 discusses our proposed band-limited recursive and symmetric WGN and SCN temporal filtering, section 6.3 explains our proposed dual-domain WGN spatial filtering, section 6.4 presents application oriented adaptation of our filter, and section 6.5 gives objective and subjective results.

6.2 Band-limited anti-blocking temporal filter

In this section propose a band-limited anti-blocking temporal filter. To simplify the explanation, we first assume a recursive filter under WGN. Then, in section 6.2.5, we extend our approach to STF and in section 6.2.6 we modify our method to handle all noise types.

6.2.1 Recursive temporal filter principal

Recursive temporal filters have the advantage of using the information of many previous frames in a less complex structure compared to symmetric temporal filters. Linear minimum mean squared error

(LMMSE) filters are efficient temporal filters which use blocks of current frame and motion compensated and previously filtered frames to remove noise. The output is computed in pixel-domain by combining the multiple temporal predictions and the current noisy observation. The objective is to estimate the noise-free frame $\mathcal{G}_t$ from a noise-contaminated frame F_t at time t utilizing temporal information. LMMSE based RTF uses R previous filtered frames in a recursive structure. Assuming $\vec{\mathcal{G}}_{t-m}$ is a motion-compensated $\mathcal{G}_t$, LMMSE based RTF are defined as,

$$\mathcal{G}_t = \frac{\sum_{m=1}^{R} \mathbf{w}_m \cdot \vec{\mathcal{G}}_{t-m} + \mathbf{w}_0 F_t}{\sum_{m=0}^{R} \mathbf{w}_m} + \bar{F}_t, \quad m > 0, \quad \mathbf{w}_0 = 1; \tag{50}$$

where $\mathbf{w}_m$ is the averaging weights defined for each pixel and $\bar{F}_t$ is an offset adjustment. In block-based RTF, weights are calculated based on block-based temporal error, i.e., temporal difference between current frame and denoised previous frames. Considering $\mathcal{B}_i$ as i^{th} block in the F_t and $\vec{\mathcal{B}}_{m,i}$ as its corresponding motion compensated block from $\mathcal{G}_{t-m}$. The block-based motion error is defined based on sum of squared error as,

$$\delta_{m,i} = \text{MAX}\left(\frac{\sum |\mathcal{B}_i - \vec{\mathcal{B}}_{m,i}|^2}{W_r^2 \sigma_p^2} - 1, 0\right), \tag{51}$$

where $sigma_p$ is the STD of WGN. In an ideal case with perfect motion estimation $|\mathcal{B}_i - \vec{\mathcal{B}}_i|$ contains only noise. In this case, the expected value of $\sum |\mathcal{B}_i - \vec{\mathcal{B}}_i|^2$ is σ_p^2 (considering that $\vec{\mathcal{B}}_i$ is already filtered and noise-free). MVs are not always accurate and the higher the δ_m, the lower $\mathbf{w}_m$ becomes. δ_m is a 2-D matrix defines one value per block. In order to map the block matrix δ_m to the entire image, block-based approaches assign the value of each block to the corresponding pixels inside the block. This operation is equivalent to an interpolation using box-shaped kernel. Let us consider the operation $\mathbf{BXI}_n(\cdot)$ as a box-shaped interpolation, with interpolation factor of n (i.e., each pixel becomes a rectangle with size of n), then temporal weight in block-based approach is defined as,

$$\mathbf{w}_m = \frac{1}{\mathbf{BXI}_{W_r}(\delta_m)}, \quad \mathbf{w}_0 = 1, \tag{52}$$

where δ_m is a matrix presents $\delta_{m,i}$ for all blocks and W_r is the size of block. W_r a compromise between the number of pixels considered in error detection and accuracy of its spatial occurrence (spatial resolution). We set $W_r = 16$. The offset adjustment is,

$$\bar{F}_t = \sum_{m=1}^{R} \mathbf{w}_m \bar{e}_m, \ \bar{e}_{m,i} = \sum (\mathcal{B}_i - \vec{\mathcal{B}}_{m,i}) \tag{53}$$

where $\bar{e}_m$ is a matrix presents $\bar{e}_{m,i}$ for all blocks. $\sum(\mathcal{B}_i - \vec{\mathcal{B}}_{m,i})$ is the summation of temporal error $(\mathcal{B}_i - \vec{\mathcal{B}}_{m,i})$ for all pixels of block $\mathcal{B}_i$. $\bar{F}_t$ offset guarantees that the average value for each block does not vary after filtering. In (52) only the power of error $\sum |\mathcal{B}_i - \vec{\mathcal{B}}_{m,i}|^2$ is taken into account. If a motion is accurately estimated both mean and power of error should be relatively small. Our experiments show that power criteria is not enough for efficient detection of error. We define the error based on the mean of error as

$$\bar{\delta}_m = \text{MAX}\left(\frac{|\bar{e}_m|}{2W_r\sigma_p} - 1, 0\right). \tag{54}$$

The expected value of $|\bar{e}_m|$ is $\frac{2W_r\sigma_p}{\sqrt{\pi}}$, however, a less sensitive function to $|\bar{e}_m|$ is considered because the we use multiple criteria to detect error. By adding mean criteria (54) to (52) is modified as

$$\mathbf{w}_m = \frac{1}{\mathbf{BXI}_{W_r}(\delta_m + \bar{\delta}_m^2)}, \ \ \mathbf{w}_0 = 1. \tag{55}$$

In (52) pixels inside the block $W_r \times W_r$ are considered. In order to have an efficient filtering W_r should be relatively large e.g., $W_r \geq 16$. The downside is motion error is detected coarsely which decreases the efficiency when in a block MVs are accurate for subset of pixels and erroneous for the rest. This leads to lack of filtering for subset with accurate MV and motion blur for erroneous MV. A solution can be increasing the resolution of error detection by decreasing the W_r, however, using less number of pixels makes the error detection less reliable. To benefit from both we use information of two levels, pixel-level and block-level to detect the motion error. Fine error detection exploit the pixel-level error in order to define the role of each pixel in temporal filtering. To efficiently extract the neighborhood dependency of pixels, we apply a low-pass filter on the absolute of difference

frames (reference and motion-compensated) to compute the e_m as

$$e_m = h_{pr} * \left| F_t - \vec{\mathcal{G}}_{t-m} \right|, \tag{56}$$

where h_{pr} is a 5×5 Gaussian filter with sigma of 1.2. We define the pixel-level error $\hat{\delta}_m$ as

$$\hat{\delta}_m = \text{MAX}\left(\frac{e_m^2}{\boldsymbol{c}_p^r \sigma_p^2} - 1, 0 \right). \tag{57}$$

where $\boldsymbol{c}_p^r = 3$. Although pixel-level error detection is advantageous to represent high resolution error, few pixels cannot desirably extract the temporal error. Our idea is to combine pixel and coarse-level errors by integrating the coarse-level temporal reliability with pixel-level error. We modify (58) as

$$\mathbf{w}_m = \frac{1}{\mathbf{BXI}_{W_r}(\delta_m + \bar{\delta}_m^2) + \hat{\delta}_m + \boldsymbol{c}_o^r}, \quad \mathbf{w}_0 = 1, \tag{58}$$

where $\boldsymbol{c}_o^r$ is a constant. An important drawback of recursive filtering is when the propagation of error from previous times to proceeding time. This is due to assigning high weights to $\vec{\mathcal{G}}_{t-m}$. In order to decrease this effect we consider the constant $\boldsymbol{c}_o^r = .1$ that prevents $\mathbf{w}_m$ from getting high values.

6.2.2 Smooth filter weight

δ_m and $\bar{\delta}_m^2$ are calculated for each block which creates discontinuity in $\mathbf{w}_m$ at the edge of blocks in (58). Pixel based $\hat{\delta}_m$ decreases the discontinuity effect when $\delta_m + \bar{\delta}_m^2$ values are close at the edge of blocks. But when $\delta_m + \bar{\delta}_m^2$ values are significantly different, weight discontinuity creates blocking artifacts. In order to compensate discontinuity, we propose a method that guarantees the smooth change of weights between pixels. Let us assume instead of a block-based summation in the calculating $\sum |\mathcal{B}_i - \vec{\mathcal{B}}_i|^2$ and $|\sum(\mathcal{B}_i - \vec{\mathcal{B}}_i)|$ we use a continuous moving average with size of $W_r\times W_r$. Theoretically, moving average eliminates blockiness and by using the same number of pixels, the efficiency is equal or higher (due to spatial homogeneity). However, moving average filters have the drawback of assigning equal weights to all pixels in a block although a pixel can be spatially far from the center. If we use a Gaussian filter with larger center coefficient, then we

need a larger kernel (i.e., larger W_r) in order to have a same processing gain. In order to address blockiness but using smaller kernel size we propose using smaller block $W_q = \frac{W_r}{4}$ integrated with a Gaussian filter. Let us consider a block-based approach uses average of $W_q \times W_q$ blocks for error detection. In order to have the same processing gain compared to a $W_r \times W_r$ a Gaussian filter should average the result of surrounding blocks $W_q \times W_q$. The averaging factor of such a filter should be at least 16 to compensate the impact of using small block with $\frac{1}{16}$ number of pixels. Considering a Gaussian blur, at least a 5×5 kernel size is required. Let us denote this filter as h_{sr} then $\sum h_{sr}^2 = \frac{1}{16}$ which leads to a Gaussian filter with sigma of 1.2. Thus, we modify (51) and (54) as

$$\delta_m = \text{MAX}\left(\frac{h_{sr} * \sum |B - \vec{\mathcal{B}}_m|^2}{W_q^2 \sigma_p^2} - 1, 0\right), \bar{\delta}_m = \text{MAX}\left(\frac{4|h_{sr} * \bar{e}_m|}{2W_q \sigma_p} - 1, 0\right), \tag{59}$$

By using smaller block size W_q, spatial location of error can be more accurately detected and at the same time less blockiness appears by smooth filtering h_{sr}. We also propose to use smooth interpolation instead of box-shaped interpolation $\mathbf{BXI}_n(\cdot)$. We use *bilinear* interpolation which modifies (58) as

$$\mathbf{w}_m = \frac{1}{\mathbf{BLI}_{W_q}(\delta_m + \bar{\delta}_m^2) + \hat{\delta}_m + \boldsymbol{c}_o^r}, \tag{60}$$

where $\mathbf{BLI}_{W_q}(\cdot)$ is a *bilinear* interpolation process with interpolation rate of W_q. Figure 6.3 shows an example of block-based error detection used in MHMCF compared to smooth weight estimation in proposed.

The only part in the (50) that is still block-based is $\bar{F}_t$. We modify (53) by using a 15×15 Gaussian filter h_{rr} with sigma of 4 as

$$\bar{F}_t = h_{rr} * \frac{\sum_{m=1}^{R} \mathbf{w}_m \cdot (\vec{\mathcal{G}}_{t-m} - F_t)}{\sum_{m=0}^{R} \mathbf{w}_m}, \tag{61}$$

6.2.3 Band-limited filtering

In the proposed temporal weight calculation (60), weights are calculated spatially continuous (i.e., without sharp jump). However, it does not completely solve the blockiness problem since the motion

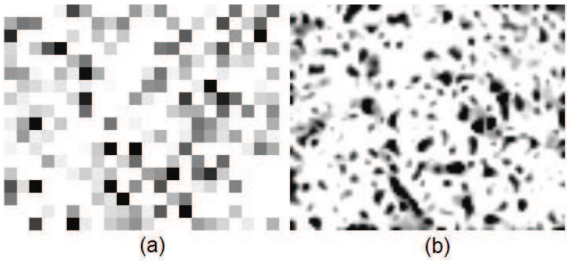

Figure 6.3: Temporal error detection for a *Foreman* video frame using (a) MHMCF that uses 16×16 blocks (b) the proposed smooth approach.

compensation is block-based. We propose a combination of two methods, band-limited temporal filtering and two-band motion compensation to minimize the blockiness. We first introduce our band-limited filtering. The LMMSE based filter (50) can be rewritten as,

$$\mathcal{G}_t = \frac{\sum_{m=1}^{R} \mathbf{w}_m \cdot (\vec{\mathcal{G}}_{t-m} - F_t)}{\sum_{m=0}^{R} \mathbf{w}_m} + F_t + \bar{F}_t, \quad m > 0, \quad \mathbf{w}_0 = 1; \tag{62}$$

In an ideal case with perfect motion estimation $\vec{\mathcal{G}}_{t-m} - F_t$ contains only noise. However, MVs are not always accurate. The more signal exists in $\vec{\mathcal{G}}_{t-m} - F_t$, the lower the $\mathbf{w}_m$ and the less effective (62) becomes. By removing signal from $\vec{\mathcal{G}}_{t-m} - F_t$ prior to temporal filtering the performance (62) increases. Since the blocking artifact is the result of MV discontinuity, in case that $\vec{\mathcal{G}}_{t-m} - F_t$ contains only noise no blocking will be appear. Inspired from this fact, we propose extracting signal prior to temporal filtering. One way can be applying a highly efficient spatial filter on F_t, however, spatial filters tend to keep sharp edges, even when they are noisy (trapped noise problem). Another way is to apply a strong lowpass filter on the F_t. In order to remove noise lowpass filter size should be large enough. This creates the problem of edge spread problem meaning the edge will be spread spatially to far places. We propose an intermediate solution using both ideas by applying an edge-stopping spatial filter on a coarse (blurred) approximation of F_t which decreases the edge spread problem and the chance of keeping noise in the sharp edges. Assuming F_t^2 is a downscaled F_t^2 using the 4×4 block-based averaging; we apply an edge-stopping spatial filter on F_t^2 which creates a coarse and noised removed approximation of F_t and contains LF only signal. In order to suppress the aliasing and release the trapped noise in the edges we apply a small kernel 3×3 Gaussian filter

h_{br} after edge-stopping spatial filter. In order to map the filtered F_t^2 results to original resolution we use the *bilinear* interpolation $\mathbf{BLI}_4(\cdot)$. We use the term *back-signal* for spatially filtered and interpolated F_t^2. *Back-signal* contains LF components is excluded from temporal processing which makes the temporal filter *band-limited* since it has no impact on the LF. Assuming $\mathbf{ESS}(\cdot,\cdot)$ is an edge stopping spatial filter taking two input parameters; the input image and the STD of noise, we compute the *back-signal* $\mathbf{b}_t$ as in

$$\mathbf{b}_t = \mathbf{BLI}_4(\hat{\mathbf{b}}_t)\ ,\ \hat{\mathbf{b}}_t = \mathbf{ESS}(F_t^2, \frac{\sigma_p}{2}) * h_{br}, \tag{63}$$

Theoretically, STD of noise in F_t^2 is $\frac{\sigma_p}{4}$ since each pixel of F_t^2 is the average of 16 F_t pixels. Generally spatial filters are designed to compromise between blur and noise but in (63) a noise-free *back-signal* is required and blur is not applicable. Thus, the noise removal strength is increased by factor of 2, i.e., $\frac{\sigma_p}{2}$. $\mathbf{ESS}(\cdot,\cdot)$ keeps the edges intact although they contain noise. However, using inter-frame information noise of strong edges can be removed temporally. Thus, we release the trapped noise and leave it for temporal filter to remove it. The role of h_{br} in (63) is to blur the edges and untie the noise from the edges. Once we obtained *back-signal*, we subtract it from the F_t and $\mathcal{G}_{t-m}$ before compensating the motion. The target of filtering becomes the band-limited *fore-signal* $\mathcal{D}_t = F_t - \mathbf{b}_t$ which contains image details and noise without very strong LF components. Prior to motion compensation of $\mathcal{G}_{t-m}$, we subtract its *back-signal* signal $\mathbf{b}_{t-m}$. We obtain the *fore-signal* for previous frames as $\mathcal{D}_{t-m} = \mathcal{G}_{t-m} - \mathbf{b}_{t-m}$ and we compensate the motion to create $\vec{\mathcal{D}}_{t-m}$. We modify (62) to a band-limited temporal filter as

$$\mathcal{G}_t = \frac{\sum_{m=1}^{R} \mathbf{w}_m \cdot (\vec{\mathcal{D}}_{t-m} - \mathcal{D}_t)}{\sum_{m=0}^{R} \mathbf{w}_m} + F_t + \bar{F}_t,\ \ m > 0,\ \ \mathbf{w}_0 = 1; \tag{64}$$

Assuming the $\mathbf{MC}(F)$ is the motion compensation process that compensates the frames F based on the estimated vectors then $\vec{\mathcal{D}}_{t-m} = \mathbf{MC}(\mathcal{D}_{t-m})$. $\vec{\mathcal{D}}_{t-m} - \mathcal{D}_t$ has less signal compared to $(\vec{\mathcal{G}}_{t-m} - F_t)$ which improves the performance of temporal filter and creates less artifacts. Since the size of F_t^2 is 16 times (4 in each dimension) smaller compared to F_t, generating and storing $\hat{\mathbf{b}}_t$ for

the radius of R is not expensive speed-wise and memory-wise. Thus, we can save and reuse them to increase the speed. We use a three iteration bilateral filter with radius of 2 as edge-stopping filter $\mathbf{ESS}(\cdot,\cdot)$ explained in section 6.3. By back-signal subtraction temporal error $(\vec{\mathcal{G}}_{t-m} - F_t)$ becomes $\vec{\mathcal{D}}_{t-m} - \mathcal{D}_t$ in (56).

6.2.4 Motion estimation and compensation

Block-matching motion estimation

Block matching motion estimation approaches are fast and high-performance under noise, which makes them suitable for denoising procedure. We use a fast multi-resolution block-matching approach to perform the motion estimation. In this approach, For each two consecutive frame, a Gaussian pyramid is generated and MVs are estimated in each level of resolution and the results of the previous level are used to set the initial search point. We start from $F_t = F_t^0$ and continue the downscaling process, until we reach a small number of pixels per image (e.g., 64x64). Then, using a block-matching technique we estimate the motion from the coarsest resolution to finest. For all levels, we use a three step search (TSS) [100]. In the final step, we check the validity of estimated vector by comparing the cost of estimated MV and the homography of MVs created from reliable MVs. We consider sum of absolute difference (SAD) as the cost function and the block with the least SAD is used to compute MV.

Homography-based motion correction

Block-matching motion estimation methods have the tendency to fall into local minima. This affects the performance of motion estimation especially when the motion is not complex (e.g., translational motion) which should be perfectly estimated. To address this problem, we propose detection of faulty MVs based on three steps: 1- detection of reliable MVs, 2- homography creation by expansion of reliable MVs to the whole frame, and 3- detection of the faulty MVs.

At first step we find the reliable MVs. To do so we use three criteria; 1) compensation gain 2) power of error and 3) repetition. We define a MV as reliable when it meets all three criteria. Assuming $\mathcal{B}_i$ is a particular block inside the reference frame F_t and $\vec{\mathcal{B}}_{m,i}$ is the corresponding

motion-compensated block of F_{t-m}, we define the motion compensation gain $\mathbf{g}_{cmp}$ as

$$\mathbf{g}_{cmp} = \frac{W_r^2 \text{VAR}(\mathcal{B}_i)}{\sum[\mathcal{B}_i - \vec{\mathcal{B}}_{m,i}]^2}, \tag{65}$$

where $\text{VAR}(\mathcal{B}_{F_t})$ is the variance of reference block $\mathcal{B}_i$ and W_r is size of block. For a block that contains only WGN the expected value of $\mathbf{g}_{cmp}$ is 0.5. We set a threshold $\mathbf{T}_{cmp} = 2$ so only MVs that $\mathbf{g}_{cmp} \geq \mathbf{T}_{cmp}$ meet the first criterion. The second criterion is the power of temporal difference $\sum[\mathcal{B}_i - \vec{\mathcal{B}}_{m,i}]^2$. We define a threshold $\mathbf{T}^r_{ptd}$ and remove the MVs that $\sum[\mathcal{B}_i - \vec{\mathcal{B}}_{m,i}]^2$ is higher than $\mathbf{T}^r_{ptd}$. To find $\mathbf{T}^r_{ptd}$ we look into those blocks which met the gain condition and we find the one with minimum power of error. Assuming the minimum power of error for all blocks that met the first criterion is $\hat{e}_{min}$, we define the threshold $\mathbf{T}^r_{ptd} = 4\hat{e}_{min}$ and remove MVs with the power of error higher than this value. The third criterion is the repetition of MVs. MVs that are not repeated are likely to be outliers. To check the repetition, for each MVs that passed the first two conditions we check the neighborhood MVs with radius of 2. MVs meet the third criterion that are repeated inside the neighborhood at least three times. At this point we have defined the reliable MVs that meet all three criteria. In the second step, we create the homography based on reliable MVs. To create the homography of MVs we diffuse reliable MVs to unreliable neighbours and we continue this procedure until all blocks are assigned with a reliable MV. At the final step, we compare the costs from homography and initially estimated MVs (using TSS) to find the which one has the least cost.

Two-band motion compensation

In spite of the performance and speed of block-matching algorithms, the discontinuity of MVs creates undesirable blocking artifacts which makes them less practical. This problem is more exposed under the processed noise where the original noisy frame is smooth and the blocking edges are more apparent. To address this problem we propose an efficient two-band motion compensation method. Assuming we decompose an image into a smooth LF and sharp HF bands in which LF band contains no strong edges. A perfect motion compensation of LF band should not contain any edges, however, discrete MVs create sharp edges from LF band. Instead of block-based compensation we propose a

method that guarantees that LF band of motion compensated frame also contains no edges. At the first step we decompose the input image F into LF ($\hat{F}_t^l = F_t * h_{mc}$) and HF ($\hat{F}_t^h = F - \hat{F}_t^l$) using a Gaussian filter h_{mc}. Motion compensation has the property of

$$\mathbf{MC}(F_t) = \mathbf{MC}(\hat{F}_t^l) + \mathbf{MC}(\hat{F}_t^h). \tag{66}$$

Ideally, $\mathbf{MC}(\hat{F}_t^l)$ should be smooth similar to $\hat{F}_t^l$, however, discontinuity of MVs creates sharp edges in border of blocks which is not desirable. If we separate h_{mc} into two lowpass filters such that $h_{mc} = \hat{h}_{mc} * \hat{h}_{mc}$, we propose the two-band motion compensation as,

$$\vec{F}_t = \Big[\mathbf{MC}(F_t * \hat{h}_{mc})\Big] * \hat{h}_{mc} + \mathbf{MC}(\hat{F}_t^h). \tag{67}$$

If MV is continuous the direct motion compensation (66) and proposed (67) becomes equal i.e., $\vec{F}_t = \mathbf{MC}(F_t)$. Otherwise, we make sure that $[\mathbf{MC}(F_t * \hat{h}_{mc})] * \hat{h}_{mc}$ has been smoothed by $\hat{h}_{mc}$. We use a 5×5 moving average filter for $\hat{h}_{mc}$ which makes the h_{mc} a 9×9 low-pass filter.

Speed optimization of motion estimation

Temporal filtering window in STF includes $2R+1$ frames which requires R forward and R backward and total $2R$ motion estimation per frame. This is very time-consuming when $R >> 1$. To reach the speed efficiency we define performing only one motion estimation per frame and compute the other MVs from that. Assuming $\mathbf{V}_{t,t+1}$ represents the MVs between two adjacent frames F_t and F_{t+1} where F_t is the reference frame. We calculate the other MVs for subsequent frames as

$$\mathbf{V}_{t,t+m} = \sum_{k=t}^{t+m-1} \mathbf{V}_{k,k+1} \; ; \; 1 < m \leq R. \tag{68}$$

Since we do not perform a sub-pixel motion estimation for $\mathbf{V}_{t,t+1}$, sub-pixel displacement can be accumulated and create a pixel displacement on $\mathbf{V}_{t,t+m}$ for $m > 1$. To compensate that we perform another motion estimation with small search radius (less than 4 pixels) using $\mathbf{V}_{t,t+m}$ in (68) as the initial search position. To reach the maximum speed in our framework we compute the backward MVs, i.e., MVs between F_t and preceding frames F_{t-m}, based on forward estimated MVs. We

store all the forward estimated MVs within the radius of R and we reuse them in the future time. Figure 6.4 shows the stored MVs (MV bank) for $R = 5$. At the time t forward motion estimation in the past, i.e., $\mathbf{V}_{t-m,t}$ with $1 \leq m \leq R$ defines the motion between frame reference frame F_t and preceding frames F_{t-m}.

t-5	t-4	t-3	t-2	t-1
$\mathbf{V}_{t-5,t-4}$	$\mathbf{V}_{t-4,t-3}$	$\mathbf{V}_{t-3,t-2}$	$\mathbf{V}_{t-2,t-1}$	$\mathbf{V}_{t-1,t-0}$
$\mathbf{V}_{t-5,t-3}$	$\mathbf{V}_{t-4,t-2}$	$\mathbf{V}_{t-3,t-1}$	$\mathbf{V}_{t-2,t-0}$	$\mathbf{V}_{t-1,t+1}$
$\mathbf{V}_{t-5,t-2}$	$\mathbf{V}_{t-4,t-1}$	$\mathbf{V}_{t-3,t-0}$	$\mathbf{V}_{t-2,t+1}$	$\mathbf{V}_{t-1,t+2}$
$\mathbf{V}_{t-5,t-1}$	$\mathbf{V}_{t-4,t-0}$	$\mathbf{V}_{t-3,t+1}$	$\mathbf{V}_{t-2,t+2}$	$\mathbf{V}_{t-1,t+3}$
$\mathbf{V}_{t-5,t-0}$	$\mathbf{V}_{t-4,t+1}$	$\mathbf{V}_{t-3,t+2}$	$\mathbf{V}_{t-2,t+3}$	$\mathbf{V}_{t-1,t+4}$

Figure 6.4: MVs stored in the MV bank within the radius $R = 5$. The estimated MVs in the past $t - R \leq t \leq t - 1$ is used at time t.

To convert forward MVs in the past, i.e., $\mathbf{V}_{t-m,t}$ to backward MVs in the time t, i.e., $\mathbf{V}_{t,t-m}$ we estimate the motion inversely. The only challenge is that block-matching algorithm is not a one-to-one function meaning two MVs may point to same location. Therefore, the inverse motion estimation operation may leave some blocks without MVs assigned to them. In this case, we use valid MVs of neighbor blocks to assign a MV to them. At the end of inverse operation we create homography and reconfirm the estimated MVs as described in section 6.2.4.

6.2.5 Extension to symmetric temporal filter

Unlike RTF that uses the previous denoised frames for filtering, in STF the noisy frames in past and future are used. Same as proposed band-limited RTF we propose to use *fore-signal* $\mathcal{D}_{t+m} = F_{t+m} - \mathcal{B}_{t+m}$ and its motion-compensated $\vec{D}_{t+m}$ to apply temporal filtering. The LMMSE based motion-compensated averaging in STF is based on filtering along temporal window with radius of R is defined as,

$$\mathcal{G}_t = \frac{\sum_{m=-R}^{R} \mathbf{w}_m \cdot (\vec{\mathcal{D}}_{t-m} - \mathcal{D}_t)}{\sum_{m=-R}^{R} \mathbf{w}_m} + F_t + \bar{F}_t, \quad \mathbf{w}_0 = 1. \tag{69}$$

We propose to use $R = 5$ for maximum quality; however, $1 \leq R \leq 5$ can be selected depending on the application, pipeline delay, and hardware limits. In calculation of weights in the RTF the basic

assumption is that the previous frames are noise-free. In the STF we assume the proceeding and subsequent frames are noisy with the same characteristics. Thus, the wight calculation procedure becomes different. However, we use the same concept of using three criteria using in the RTF, coarse (block-level) power of temporal error, coarse mean of temporal error, fine (pixel-level) temporal error. Similar to (59) the temporal power of error under WGN for block of data can be calculated as

$$\delta_m = \text{MAX}\left(\frac{h_{sr} * \sum |B - \vec{\mathcal{B}}_m|^2}{2W_q^2\sigma_p^2} - 1, 0\right), \bar{\delta}_m = \text{MAX}\left(\frac{4|h_{sr} * \bar{e}_m|}{2\sqrt{2}W_q\sigma_p} - 1, 0\right), \tag{70}$$

The constant 2 is considered since both B and $\vec{\mathcal{B}}_m$ are noisy. Based on the coarse level errors δ_m and $\bar{\delta}_m$, we compute the reliability of block-based temporal P_b based on both power and mean of error as

$$P_b = \exp\left(-\frac{\bar{\delta}_m^2}{2}\right) \cdot \frac{1}{1 + \delta_m^2}, \tag{71}$$

With the pixel-level error detection we define the role of each pixel in temporal filtering at a finer resolution.

$$\hat{\delta}_m = \text{MAX}\left(\frac{e_m^2}{\mathbf{BLI}_{W_q}(P_b)\sigma_p^2} - 1, 0\right), e_m = h_{pr} * \left|\mathcal{D}_t - \vec{\mathcal{D}}_{t+m}\right|, \tag{72}$$

and we define the temporal weights as

$$w_m = \frac{1}{1 + \hat{\delta}_m}, \tag{73}$$

6.2.6 Modification to handle SDSCN

As described in the section 5.2, we handle signal-dependency of noise by equalizing and de-equalizing process at the beginning and end of filtering pipeline. Thus, after equalization we assume noise is signal-independent. According to noise model in the chapter 2 and SCN estimation system in the section 4.3.2, the input parameters for video denoiser are 1- the power of the noise at the original scale σ_p^2, degree of spatial correlation at the original scale $\hat{\gamma}_0$, and scale 1 (downscaled by 2) $\hat{\gamma}_1$. Under SCN, block-level error detection based on power of error, i.e, δ_m in (59) and (70), and pixel-level error, i.e, δ_m in (57) and (72) are still valid since they are computed based on the power of the error. However, block-level error detection based on the mean of error i.e, $\bar{\delta}_m$ in (59) and (70)

is not valid. Under SCN, mean of error, cannot be calculated by only σ_p. According to the (46) the STD of the noise at the downscaled by 4 scale I^2 is

$$\sigma_{p,2} = \sigma_p \sum h_a^2 \cdot (\hat{\gamma}_0 \cdot \hat{\gamma}_1). \tag{74}$$

However the expected value for WGN is $\sigma_{p,2} = \sigma_p \sum h_a^2$. When the noise becomes spatially correlated, although, the power of noise changes, the mean of block-level noise remains intact. Assuming noise at the downscaled by 4 scale is WGN, we modify (59) and (70) to compute the block-level error mean as

$$\delta_m = \text{MAX}\left(\frac{h_{sr} * \sum |B - \vec{\mathcal{B}}_m|^2}{2W_q^2\sigma_p^2(\hat{\gamma}_0 \cdot \hat{\gamma}_1)} - 1, 0\right), \bar{\delta}_m = \text{MAX}\left(\frac{4|h_{sr} * \bar{e}_m|}{2\sqrt{2}W_q\sigma_p(\hat{\gamma}_0 \cdot \hat{\gamma}_1)} - 1, 0\right). \tag{75}$$

6.3 Dual-domain fast spatial noise filter

Spatial filters use the spatial correlation between pixels in order to estimate original value of pixel from a noise contaminated pixel. We assume noise is WGN, however, we consider that noise after temporal filtering is not uniformly reduced. For each pixel, based on the temporal error, different amount of filtering takes places and our spatial filter handles a varying noise level for each pixel (per-pixel noise level). We also propose a modification to our spatial filter to support SDSCN.

6.3.1 Principle

In most cases, where the temporal information is reliable most of processing is accomplished by temporal filter. Thus, using a time-consuming spatial filter is less justified and not needed. We have analyzed many pixel and transform domain methods such as iterative pixel-domain (e.g., anisotropic diffusion), large-kernel pixel-domain (e.g., bilateral filtering), small kernel transform-domain (e.g., DCT3$\times$3), and large-kernel transform-domain (e.g., DFT16$\times$16) methods. Our experiments show that the relation between efficiency and complexity in the spatial filtering in not linear. Figure 6.5 shows the MSE-complexity relation between different spatial filters. The slope of efficiency decreases as the complexity increases and the ideal MSE versus complexity curve can be obtained. Our objective is to design a filter that, first, fits on the ideal filtering curve and, second, looks natural

with minimum filtering artifacts. We have considered artifacts such as posterization, hammered surface, and ringing in the efficiency evaluation.

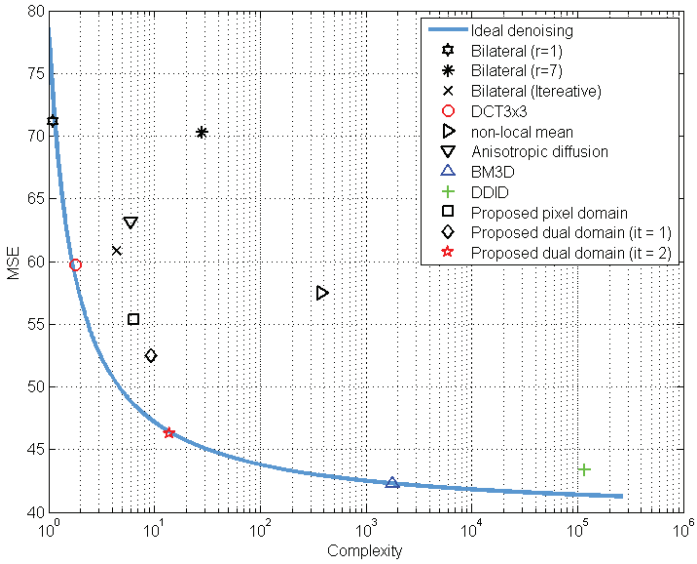

Figure 6.5: MSE-complexity curve and the hypothetical ideal denoising curve. We used 25dB noisy images of dataset in Figure 5.12.

Table 6.1 compares advantages and disadvantage of different spatial filtering methods. Small kernel transform domain are fast the most effective methods in preserving edges (see DCT3×3 in Figure (6.5). Assuming $\mathcal{DCT}_3(\cdot)$ and $\mathcal{DCT}_3^{-1}(\cdot)$ compute the 3×3 DCT of two dimensional pixels values. Assuming x and y are arbitrary position in the input image I, we define the DCT shrinkage as

$$S_p^1 = \mathcal{DCT}_3^{-1}\left(\mathcal{DCT}_3(I_p)\cdot\left[1-\mathbf{exp}(\frac{|\mathcal{DCT}_3(I_p)|^2}{c_{dct}\sigma_p^2}\right]\right) \quad , I_p = I(x\!:\!x+2, y\!:\!y+2), \tag{76}$$

where $S_p^1 = S^1(x\!:\!x+2, y\!:\!y+2)$ is part of shrunk output S^1 starts and ends at the spatial positions (x, y) and $(x+2, y+2)$. c_{dct} is a constant defines the shrinkage strength which is set to 1.8. When the magnitude of DCT coefficient is large compared to noise variance σ_p^2 the shrinkage has no effect otherwise it suppresses the noisy coefficients. Using a small neighborhood processing $\mathcal{DCT}_3(\cdot)$ is unable to remove LF noise. In order to remove LF noise we need to expand the considered neighborhood. We have employed the concept of bilateral filtering to remove LF noise. Let us first introduce bilateral and steered bilateral filtering.

Bilateral filtering [21] calculates the denoised high-contrast values for a pixel p using a bilateral kernel. Bilateral kernel is defined over a square neighborhood window N_p centered around every pixel p with window radius r.

$$\tilde{I}_p = \frac{\sum_{q \in N_p} k_{p,q} A_q}{\sum_{q \in N_p} k_{p,q}}, \tag{77}$$

$$k_{p,q} = e^{-\frac{|p-q|^2}{2\sigma_s^2}} e^{-\frac{(I_p - I_q)^2}{c_r \sigma_n^2}}. \tag{78}$$

The parameters σ_s and c_r shape the spatial and range kernels respectively and σ_n is the STD of noise. $\tilde{I}_p$ is the filtered output. In steered (or joint) bilateral, averaging weights $k_{p,q}$ are calculated from a guide image. Steered bilateral filtering calculates the denoised high-contrast value J_p for a pixel p using a joint bilateral filter. The Steered bilateral filter uses the guide image J to filter the noisy image I as

$$\tilde{I}_p = \frac{\sum_{q \in N_p} k_{p,q} A_q}{\sum_{q \in N_p} k_{p,q}}, \tag{79}$$

$$k_{p,q} = e^{-\frac{|p-q|^2}{2\sigma_s^2}} e^{-\frac{(J_p - J_q)^2}{c_r \sigma_n^2}}. \tag{80}$$

Weights are calculated using euclidean distance of both intensity difference and spatial difference. However, there are two differences between proposed methods and bilateral filter (77). First, we use subset of pixels iteratively in order to decrease complexity. Second, we decrease the strength of filtering as we increase the radius. Figure 6.7 shows pattern that is used in the proposed bilateral filtering. The 3×3 at the center pixels denoted by 1 are handled by DCT shrinkage and the 3 other iterations denoted by 2 to 4 are handled by iterative spatial filter. Assuming $S^{r-1}_{x,y}$ is the center pixel of input image S^{r-1} and r is the spatial radius, we define the proposed bilateral filtering as

$$S^r_{x,y} = \frac{\sum[u_{k_x,k_y} S^{r-1}_{(x+k_x r, y+k_y r)}] + S^{r-1}_{x,y}}{1 + \sum u_{k_x,k_y}}, k_x, k_r \in \{-1, 0, 1\}, r \in \{2, 4\}, \tag{81}$$

$$u_{k_x,k_y} = \mathbf{exp}\left(-\frac{r^2 |k_x^2 + k_y^2|^2}{c_{blt}}\right) \cdot \mathbf{exp}(-\frac{|S^r_{x,y} - S^r_{x+k_x r, y+k_y r}|^2}{2^{1-r}\sigma_p^2}), \tag{82}$$

where $S^{r-1}_{x,y}$ and $S^r_{x,y}$ are the input and output of proposed bilateral filter with radius r at the spatial location of x, y. As the filter proceed to higher r the noise power decreases and we have considered

the factor 2^{1-r} to scale σ_p^2. The output of proposed pixel-domain filtering, $S_{x,y}^r$ is fast and efficient in removing noise keeping high-contrast edges, however, it has three drawbacks, 1) removing low-contrast edges (texture blur), 2) introducing posterization, and 3) introducing hammered surface. In order to address low-contrast texture blur, we use DFT shrinkage to reconstruct destroyed textures. DFT is a powerful tool to detect textures since repeated pattern are appeared as a powerful DFT coefficient. In order to detect weak textures the size of processing window (kernel size) should be large enough, e.g., 16×16. Other than size the difference between DFT and DCT shrinkage is that DCT is applied on the input image but DFT is applied on the estimated noise (difference between noisy input and denoised output). This means the destroyed weak textures are extracted from estimated noise and added back to denoised image. Assuming $\mathcal{DFT}_{16}$ and $\mathcal{DFT}_{16}^{-1}$ are the two dimensional 16×16 DFT and inverse DFT transforms we define the DFT shrinkage as

$$S_p^5 = \mathcal{DFT}_{16}^{-1}\left(\mathcal{DFT}_{16}(I_p - S_p^4)\cdot\left[\mathbf{exp}(16^2\frac{c_{dft}\sigma_p^2}{|\mathcal{DFT}_{16}(I_p - S_p^4)|^2}\right]\right) + S_p^4,$$
$$I_p = I(x{:}x{+}15, y{:}y{+}15) \quad , \quad S_p^4 = S_p^4(x{:}x{+}15, y{:}y{+}15), \tag{83}$$

where c_{dft} is a constant and the higher value makes the shrinkage stronger. In order to increase the efficiency for both shrinkage operations (76) and (83) we use overlapped blocks with half size of the block, i.e., 1 and 8. c_{dft} set to 4 and 1 for strong and weak DFT shrinkage.

Table 6.1: Advantages and disadvantages different spatial filtering methods.

Method	Advantage	Disadvantage
Iterative pixel domain	**Fast**	Posterization
	Minor edge blur	Hammered surface
	No ringing & blocking	Texture blur
Large-kernel pixel domain	**No Posterization**	Slow
	No hammered surface	Edge blur
	No ringing & blocking	Texture blur
Small kernel transform domain	**Minor edge blur**	Hammered surface
	Fast	Texture blur
	No ringing & blocking	LF noise
Large kernel transform domain	**Texture preservation**	Slow
	Minor edge blur	Ringing & blocking
	No hammered surface	Impulse blur
Block-matching & non-local mean	**Texture preservation**	Very Slow
	No Posterization	Blur

Iterative approaches are efficient in keeping the strong edges but introduce posterization and hammered surface. Posterization happens when gradient of continuous shades converts into few value of discrete shades. Figure 6.6 shows the iterative pixel domain filtering compared to one iteration large kernel filter for a one dimension noisy gradient. In iterative approach, pixel values in different regions converge to a single value and create undesirable edges between different regions. Anisotropic processing (using pixel based edge-stopping) also convert noise into structure like patterns similar to hammered surface. In order to handle this problem we need to exclude the gradient of continuous shades (very LF content) from the filtering. This can be done by a large kernel bilateral filter (e.g., radius of 7) (see Figure 6.6). However, large kernel bilateral filter is costly since it accesses all surrounding pixels in a large neighbourhood. Instead, we propose a faster method that has the same efficiency. Since the goal is to exclude very LF contents from filtering, we replace large kernel bilateral filter by, 1) downscale by 4 and apply small kernel bilateral filter (e.g., radius of 2, and 2) upscale the results by 4. Let us assume the LF content is denoted by $\bar{S}$, we exclude $\bar{S}$ from filtering process by applying the proposed filter on the $I - \bar{S}$. Let us denote proposed filter (i.e., DCT shrinkage+iterative bilateral+strong DFT shrinkage) as $\mathbf{S}^{1-5}(\cdot)$, the output of proposed filter is defined as $\mathbf{S}^{1-5}(I - \bar{S}) + \bar{S}$ which is denoted as proposed with one iteration in Figure 6.5.

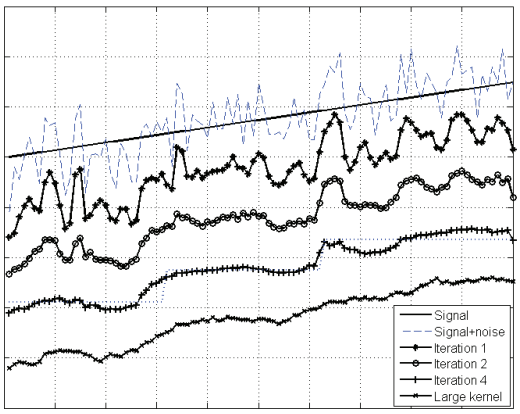

Figure 6.6: Posterization effect due to small kernel iterative filtering. Large kernel single iteration filters address this problem.

In the proposed single iteration method, the hammered surface and ringing artifacts are often visible. In order to solve this problem we propose a two iteration filtering. We have removed the unnecessary steps of second iteration and modified the proposed two iteration spatial filtering by

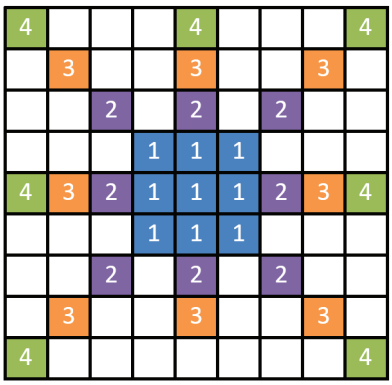

Figure 6.7: Proposed pattern for pixel-domain spatial filtering. The first iteration is a 3×3 DCT shrinkage. In the next steps, indicated pixels are considered for iterative bilateral.

adding two steps 1) adding a weak DFT shrinkage on the $\mathbf{S}^{1-5}(I - \bar{S})$ to minimize the hammered surface effect, and 2) apply a steered bilateral before the strong DFT shrinkage to address the ringing artifact. Frequency-domain processing, using a weak DFT shrinkage, provides a smooth denoising (no hammered effect) and because of being weak it does not introduce blur and ringing. However, it leaves some noise that is removed by proposed $\mathbf{S}^{1-5}(\cdot)$. The proposed two iteration spatial filtering is summarized in Algorithm 3. When there is per-pixel noise level, for instance when spatial filter is used to remove residual noise remained after temporal filter the input noise level σ_p^2 will have a per-pixel value. For pixel-domain operation such as (81) this can be done by noise level of each pixel as σ_p^2, however, for transform domain that uses block of pixels we need to use the per-block noise level. In this case, we use the average noise of block as the per-block noise level.

Algorithm 3 Proposed two iteration spatial filtering

1: **Downscale** the image by 4 and apply a single iteration bilateral filter.
2: **Upscale** results of previous step by 4 and subtract it by original image.
3: **Apply** a weak DFT (16×16) shrinkage of results previous step.
4: **Apply** a DCT shrinkage (3×3) on the results of previous step.
5: **for** r = 2:4 **do**
6: Iterative bilateral filtering using Figure 6.7 pattern with radius r.
7: **end for**
8: **Apply** a steered bilateral on the original noisy using results of previous step as the guide.
9: **Apply** a strong DFT (16×16) shrinkage on results previous step and add it to results of step 7.

6.3.2 Modification to handle SCN

We use the concept of described in the section 5.3 in order to handle SCN. As described in the section 5.3 to address SCN noise we use WGN filter in three different scales (see Figure 5.7). Since the processing is accomplished for three scales with same resolution, the processing time becomes tripled. In order to increase the speed, we propose instead of using all downscaled images with different starting indexes, we only consider the downscaled image with starting index of 0 for both row and column and use interpolation operation to map the downscaled to images to original resolution. Let use assume $\dot{I}^{l+1}$ is the downscaled image with starting index of 0. We modify the (45) as

$$I_e^l = I^{l+1} + g^l \cdot h_a * \left[\mathbf{BLI}_2 \left(\mathbf{WGD}(\dot{I}^{l+1}, \sigma_{p,l+1}) \right) - I^{l+1} \right] \quad 0 \leq g^l \leq 1, \tag{84}$$

where $\mathbf{BLI}_2$ is the bilinear interpolation with rate of 2.

6.3.3 Integration to temporal filter

Noise after temporal filtering is not uniformly reduced since for each pixel based the error different amount of temporal averaging takes place. Although noise is assumed spatially correlated, we assume noise is temporally independent and identically distributed. Thus, the power of the noise after temporal filtering in (69) is reduced by factor of λ as

$$\lambda = \frac{\sum\limits_{m=-R}^{R} w_m^2}{(\sum\limits_{m=-R}^{R} w_m)^2}, \tag{85}$$

Since $\lambda > 0$ noise is not fully removed especially when w_m is small (i.e., high temporal error). We use space-domain to remove residual noise based on power of the noise. Since λ is can be different for each pixel, we consider a spatial filter which handles per-pixel noise level to be integrated to our proposed video denoising pipeline. Let us denote the SCN optimized spatial filter as $\mathbf{SPF}(\mathcal{G}_t, \sigma_p, \hat{\gamma}_0, \hat{\gamma}_1, \lambda)$ which takes the temporally filtered $\mathcal{G}_t$ and 4 other inputs, noise descriptor parameters $\sigma_p, \hat{\gamma}_0, \hat{\gamma}_1$ and temporal filtering factor λ. Then, the output of proposed time-space algorithm becomes,

$$\hat{\mathcal{G}}_t = \mathbf{SPF}(\mathcal{G}_t, \sigma_p, \hat{\gamma}_0, \hat{\gamma}_1, \lambda). \tag{86}$$

6.4 Application specific adaptation

In the course of our research and development, there were application (industrial) specific aspects of denoising requiring solutions. We developed the following solutions to these issues.

6.4.1 Color video denoising

In denoising of color videos both luminance and chroma channels should be denoised. As shown in the [69] in denoising, the optimal color conversion can be found according to the noise at each channel. However, the main assumption is that noise is WGN in all channels and all channels have the same visual importance. However, practically both noise and content in chrominance channels are subsampled and the importance of luminance channel in the visual quality is much more than chrominance. Thus, we use YCbCr color conversion that is used in most of video and image codecs. A simple way of denoising chroma channels is applying the same time-space filter on all channels. In order to gain speed we propose an optimization with using the same motion vectors estimated for the luminance channels and weights calculated in (73) to apply temporal filtering. This provides a reliable temporal filtering in most cases, however, the only problematic case happens when the temporal data is reliable for the luminance channel is unreliable for chrominance channel. These inaccurate weights lead to motion blur in chrominance channel. To compensate this problem we propose detecting the motion blur by considering the difference between original frame and filtered one. If the difference is higher than expected value we compensate the error by restoring the original pixel and we use the spatial filter instead to remove noise. With the same procedure of calculating (73) we find the reliability of chroma filtering. The only difference is instead of using difference between two frames we use the difference between original noisy and filtered frame. Assuming $F_{t,cb}$ is an original chrominance channel and $\mathcal{G}_{t,cb}$ is the temporal output, we use $F_{t,cb} - \mathcal{G}_{t,cb}$ to find the reliability of temporal filtering according to (73). Assuming the $\hat{\mathbf{w}}_{cb}$ is the reliability we modify the $\mathcal{G}_{t,cb}$ by adding $(F_{t,cb} - \mathcal{G}_{t,cb})(1 - \hat{\mathbf{w}}_{cb})$ to it. When the temporal filtering is reliable (i.e., $\hat{\mathbf{w}}_{cb}$)

the output does not change and when the temporal filtering is unreliable (i.e., $\hat{\mathbf{w}}_{cb} = 0$) the output becomes $F_{t,cb}$.

6.4.2 Detection of noise power overestimation

Video noise filter often assume that noise has been accurately pre-estimated. Due to difficulty of differentiation between noise and image structure, noise overestimation is possible. We utilize coarse-level analysis to detect local noise overestimation. In (58) we use the local temporal error power and we propose to use that for local noise power assessment and detect noise power overestimation in (72). Due to high coherence between reference frame $\mathcal{D}_t$ and motion-compensated $\vec{\mathcal{D}}_{t\pm1}$, there is a high chance to have a temporal difference $(\vec{\mathcal{D}}_{t-m} - \mathcal{D}_t)$ containing only noise due to accuracy of MVs. Thus, we can first detect the noise level overestimation at the coarse-level analysis, during the processing of $\vec{\mathcal{D}}_{t\pm1}$, and then correct it for processing of $\vec{\mathcal{D}}_{t+m}$ when $|m| > 1$. Since the motion is larger and more complex at $|m| > 1$, the introduced motion blur/artifact are significantly stronger than $|m| = 1$. By correcting the noise level for $|m| > 1$ under overestimation situation, the potential motion blur can considerably decrease. To correct the overestimated noise we use the already computed average power of temporal difference in (58) for large number (equivalent to W_r) pixels. If the motion is accurately estimated it represents the average power of temporal noise. This means, if $h_{sr} * \sum |B - \vec{\mathcal{B}}_m|^2$ is less than the expected error $(\frac{W_r^2\sigma_p^2}{8})$ the noise is overestimated. In such case, we consider it as overestimated and we replace $(\frac{W_r^2\sigma_p^2}{8})$ by $h_{sr} * \sum |\mathcal{B} - \vec{\mathcal{B}}_m|^2$ that is calculated for the adjacent temporal frame. Thus, we store the computed $h_{sr} * \sum |\mathcal{B} - \vec{\mathcal{B}}_m|^2$ for adjacent temporal frames in the *Noise power bank* to be used in processing other motion-compensated frame.

6.4.3 Image naturalization and edge preserving

Our filter reduces the posterization and ringing artifacts significantly, however, according to our subjective evaluation, the result that contain slight high-frequency noise are more appealing since they look more natural. Thus, at the end of filtering pipeline we add the high-frequency noise to the output. As an optional filtering tool this high frequency noise can be done non-uniformly by adding more high-frequency to the edges and less to the non-edge pixels. We have implemented an edge

detection algorithm using Canny edge detector to detect the edges and deactivates the filtering for those edges. Although the edges may look noisy but the overall image looks sharper.

6.4.4 Burst mode optimization

We did some experiments for analyzing the burst photography, i.e., denoising the images taken by phone in the burst mode using the phone. In some phones we have access to raw data (see Figure 1.1). We started analyzing the noise in the both raw and JPEG files. The problem for the raw processing is that we need to perform other processing tasks such as demosaicing and white-balancing. Thus, we performed our analyses on the JPEG files. Since the processing resources are limited in the phone we search for less complex solutions. Our first idea was using metadata (exposure time and brightness value) to estimate the noise. We implemented a novel approach by using metadata that is stored in the (exchangeable image file format). We used 20 images taken in different condition to the calibrate the function that maps the metadata to an approximate noise value for specific phone.

Another idea was simplifying the motion estimation engine. We assumed that motion is burst mode image is small and simple (mostly translational). We have tested two ideas to decrease the amount of processing, block subsampling and pixel subsampling. In block subsampling our idea is to choose a subset of blocks with image features and do the motion estimation only for those blocks and create the homography. To do that we tested 2 ideas; First idea is feature extraction based on accurate motion estimation. Meaning, as we do the motion estimation in different resolutions we eliminate the blocks that motion estimation does not improved the signal to noise ratio. As we grow the resolution more blocks will be eliminated and at the highest resolution only few blocks remains. If the image is highly structured and the motion is simple this is an effective method. The second idea is fixing the complexity by setting the searching blocks to a constant number. This approach works better in the more complex motions with less image structure. The block subsampling approach is less efficient when image feature are concentrated in specific part of image. This situation makes the process of homography creation impossible. In pixel subsampling our idea is to use a subset of pixels for each block to compute the cost. We used this idea utilizing both small and large blocks and our experiments show using large blocks are more efficient. Pixel subsampling

is also more effective compared to block subsampling in low textured part of image. Therefore, we used pixel subsampling with large block for the motion estimation. We were able to denoise 4K image using 5 images on the Samsung S5 phone in 700ms.

6.4.5 Impulse noise removal

In consumer electronics applications the impulse noise such as salt and pepper is improbable. We have tested the effect of impulse noise in our framework. Since the proposed framework tends to keep the sharp edges, impulse noise which is similar to sharp edges cannot be removed. Thus, we used a 3×3 median filter to remove impulse noise. We have analyzed the where in the pipeline is better to place the median filter (where options are at the beginning, after temporal filtering, and at the end) and the experiment results show the beginning of the pipeline provides the best results. Impulse noise should be excluded from temporal filter since it degrades the effectiveness of temporal filtering. In order not to blur the edges we apply the 3×3 median filter only on the pixels that are corrupted by impulse noise. To detect corrupted pixels, for each pixel we find the pixel with closest value in the 3×3 neighborhood and indicate it if the difference is higher than threshold (e.g., 50).

6.5 Experimental results

To validate our denoising system, we have run the whole system and its sub-systems and algorithms on related dataset and examined their main parameters as discussed in the remainder of this section. For video experiments under synthetic noise we used ten 352×288 videos (except for 352×240 *Tennis*) (dataset Figure 5.14). The presented time-space video filter has been implemented and tested (without automated noise estimation) the results have been compared to state-of-the-art video denoising methods; DDID [23], BM3D [20], MHMCF [17], *STGSM* [12], VBM3D [8], VBM4D [15], and RF3D [61]. Different experiments have been conducted using synthetic and real noise. For the synthetic noise experiments, two noise types WGN, and processed WGN has been tested.

Table 6.2: WGN (PSNR = 30dB): Average error in PSNR (dB) for 150 frames.

	DDID [23]	BM3D [20]	MHMCF [17]	STGSM [12]	VBM3D [8]	VBM4D [15]	RF3D [61]	Ours R=2	Ours R=5
Bus	33.79	34.15	32.61	33.89	34.64	34.15	35.62	35.10	35.52
Flower	33.45	33.46	33.21	34.60	35.70	35.40	35.73	34.99	35.68
Foreman	37.29	37.01	35.01	37.03	38.28	38.07	38.20	37.92	38.37
News	37.30	37.92	37.29	39.51	41.75	41.02	41.88	40.16	41.12
Parkjoy	34.76	32.56	32.56	34.35	34.10	33.98	33.27	34.10	34.42
Rushhour	38.38	38.53	35.08	38.29	39.89	39.66	39.65	38.81	39.41
Soccer	34.10	36.08	34.17	36.21	36.68	36.58	37.72	37.07	37.44
Stefan	34.51	34.63	33.24	35.68	35.84	35.35	36.06	35.61	36.00
Stem	34.37	34.09	32.51	34.15	36.05	35.63	35.67	35.34	35.80
Tennis	31.88	33.74	34.14	34.00	36.22	35.70	32.02	35.65	35.85
Average MSE based	34.58	34.82	33.77	35.42	36.42	36.07	35.76	36.13	**36.57**

Table 6.3: WGN (PSNR = 25dB): Average error in PSNR (dB) for 150 frames.

	DDID [23]	BM3D [20]	MHMCF [17]	STGSM [12]	VBM3D [8]	VBM4D [15]	RF3D [61]	Ours R=2	Ours R=5
Bus	30.54	30.74	28.97	30.76	31.33	30.95	32.16	31.90	32.43
Flower	29.66	29.62	29.49	31.49	32.68	32.35	32.34	31.89	32.58
Foreman	34.87	34.52	32.16	34.87	35.96	35.64	35.78	35.21	35.66
News	34.24	34.66	33.83	37.06	38.92	37.94	38.72	36.90	37.91
Parkjoy	31.23	28.93	28.93	31.42	31.90	30.80	30.31	30.93	31.31
Rushhour	35.39	35.36	31.96	36.01	37.13	36.83	36.82	35.91	36.49
Soccer	31.22	33.25	30.92	33.81	33.18	33.52	34.45	33.97	34.30
Stefan	31.29	30.95	29.68	32.67	32.58	32.23	32.50	32.43	32.99
Stem	31.15	30.60	28.89	31.25	33.12	32.68	32.53	32.13	32.66
Tennis	28.54	30.77	30.72	31.27	33.31	32.58	30.69	32.61	32.93
Average MSE based	31.32	31.44	30.29	32.60	33.48	33.01	32.96	33.01	**33.52**

6.5.1 Time-space filter applied to WGN

We have evaluated the performance of BLTSF under the synthetic WGN. WGN with two PSNR level of 30dB and 25dB has been added to the gray-scale original frames and we denoised the noisy frames using state-of-the-art methods and the proposed with two temporal radii; $R = 2$ and $R = 5$. All the input parameters are according to STD of WGN are set. For RF3D we used a flat PSD since the noise is WGN. Table 6.2 and 6.3 demonstrate the average error in PSNR of filtered frames for all videos. The proposed method achieves competitive results in comparison with other methods.

6.5.2 Time-space filter applied to synthetic SCN

We have conducted many experiments to evaluate the performance of proposed method under processed noise. In the experiments we added synthetic processed noise to (dataset Figure 5.14)) and

we compared the denoising results of state-of-the-art filters with the proposed with two radii; $R = 2$ and $R = 5$. To generate processed noise, we analysed noise characteristics in real videos. To have a more realistic noise characteristic in addition of SCN we consider the effect of quantization. In practice noise becomes spatially correlated in both pixel and frequency domain. Thus, in addition of pixel-domain blurring (Gaussian filter) we use a DCT domain quantizer. We use a DCT based quantization by dividing the image into 8×8 blocks and using the quantization table defined in the JPEG standard [117]. Figure 6.8 shows the block diagram of processed noise generation.

Based on how much noise is spatially correlated we define two noise profiles with different level of spatial correlation. First SCN (profile-1) was generated by processing the WGN with STD of 12 using 3×3 G-Blur with sigma of 0.55 and compressed with JPEG quantization table with QF = 90. This noise profile generates noisy frames with PSNR = 32.6dB. Second SCN (profile-2) we processed the WGN (with STD of 17) using 3×3 G-Blur with sigma of 0.65 and QF = 75. This profile generates also noisy frames with PSNR = 32.6dB. Accordingly we set the input parameters of algorithm, $\sigma_p = 6$ and $\gamma_1 = 0.65$, and $\gamma_2 = 1$ for the profile-1 SCN, and $\sigma_p = 6$ and $\gamma_1 = 0.55$, and $\gamma_2 = 1$ for SCN. Figure 6.9 shows the a sample of these noise profiles in both pixel domain and frequency domain. For other methods we varied the input noise STD to the value that leads to highest PSNR. For the RF3D [61], the input parameter is the power spectral density (PSD). We used two-dimensional Gaussian lowpass filter with different sigma σ_{psd} to define the PSD with the highest PSNR. Table 6.4 and 6.5 compares the results under both SCN, profile-1 and profile-2. Although noise power is the same for both noise profiles, PSNR of profile-1 is higher than heavily processed noise for all method. This shows as the noise gets more correlated and structured the performance of all methods decreases. In average ours gives the highest PSNR followed by RF3D. Under heavily processed noise our method outperforms all for all videos except for the *Flower*. For the state-of-the-arts the problem is for fixed noise profile we have to change the input parameters to reach the highest PSNR but this is not case for the proposed method. For instance in Table 6.4, in denoising of *Tennis* with RF3D we set $\sigma_{psd} = 80$ which is almost a flat PSD which gives higher PSNR (2.4dB) compared to $\sigma_{psd} = 4$. Table. 6.6 compares the average of error in PSNR for all methods when the noise profile and the input parameter (σ_i and σ_{psd}) are set to a fixed value. σ_i and σ_{psd} are set to the values that gives the highest PSNR for most of test videos. Note that for

computing the PSNR values, first the average error power (MSE) is computed and then the PSNR is calculated for the average (MSE based average).

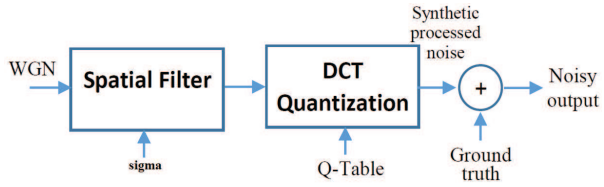

Figure 6.8: Block diagram of synthetic processed noise generation.

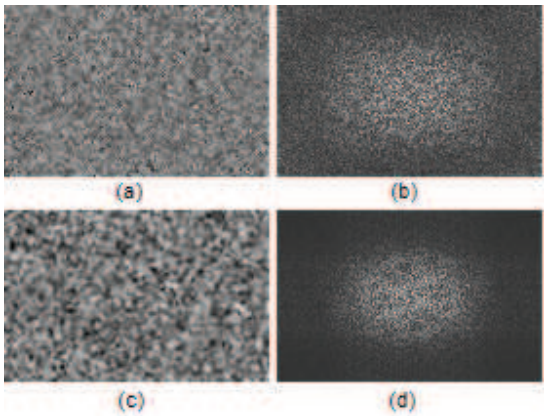

Figure 6.9: A sample of two synthetic noise profiles (a) and (c) are the pixel values of SCN according to profile-1 and profile-2. (b) and (d) are their frequency spectrum.

Figure 6.10 shows the error curve in time for two video sequences *Bus* and *Tennis* under heavily processed noise (Table 6.5). Figure 6.11 compares the visual results under heavily processed synthetic noise for the frame 100 of *Tennis*. Performance of VBM3D and RF3D significantly decreases under processed noise. MHMCF leaves noise and blocking when the motion is not accurately estimated.

6.5.3 Time-space filter applied to real noise

We have evaluated the proposed filter on real noisy video sequences captured by a digital camera under noisy conditions. For our method we obtained the σ_p, $\hat{\gamma}_1$, and $\hat{\gamma}_2$ by applying our noise estimator as in chapter 4. For other methods the optimal input parameters differs based on the motion and details of the video (see Tables 6.4 and 6.5). Thus, to have a fair comparison, we

Table 6.4: SCN (profile-1): Average error in PSNR (dB) for 150 frames. The input of all methods are tuned to give the highest PSNR. For ours the inputs are fixed ($\sigma_p = 6$, $\hat{\gamma}_0 = 1.4$, $\hat{\gamma}_1 = 1$).

	DDID [23]	BM3D [20]	MHMCF [17]	STGSM [12]	VBM3D [8]	VBM4D [15]	RF3D [61]	Ours R=2	Ours R=5
Bus	34.24 $\sigma_{inp} = 6$	34.47 $\sigma_{inp} = 6$	34.30 $\sigma_{inp} = 6$	33.65 $\sigma_{inp} = 6$	34.84 $\sigma_{inp} = 6$	34.46 $\sigma_{inp} = 6$	36.45 $\sigma_{psd} = 4$	36.09	**36.46**
Flower	34.80 $\sigma_{inp} = 8$	34.67 $\sigma_{inp} = 6$	35.34 $\sigma_{inp} = 6$	34.82 $\sigma_{inp} = 6$	36.45 $\sigma_{inp} = 8$	36.09 $\sigma_{inp} = 8$	**37.06** $\sigma_{psd} = 4$	36.37	36.81
Foreman	36.79 $\sigma_{inp} = 8$	36.36 $\sigma_{inp} = 8$	36.06 $\sigma_{inp} = 6$	35.73 $\sigma_{inp} = 6$	37.74 $\sigma_{inp} = 8$	37.65 $\sigma_{inp} = 8$	38.58 $\sigma_{psd} = 4$	38.33	**38.73**
News	37.33 $\sigma_{inp} = 8$	37.33 $\sigma_{inp} = 8$	38.25 $\sigma_{inp} = 6$	36.75 $\sigma_{inp} = 6$	41.40 $\sigma_{inp} = 8$	40.73 $\sigma_{inp} = 8$	**42.18** $\sigma_{psd} = 4$	40.62	41.38
Parkjoy	35.32 $\sigma_{inp} = 6$	35.25 $\sigma_{inp} = 6$	35.01 $\sigma_{inp} = 6$	34.03 $\sigma_{inp} = 6$	34.76 $\sigma_{inp} = 6$	34.73 $\sigma_{inp} = 6$	**34.83** $\sigma_{psd} = 10$	35.49	35.79
Rushhour	37.64 $\sigma_{inp} = 8$	37.46 $\sigma_{inp} = 8$	36.20 $\sigma_{inp} = 6$	36.54 $\sigma_{inp} = 6$	39.13 $\sigma_{inp} = 8$	38.97 $\sigma_{inp} = 8$	**39.78** $\sigma_{psd} = 4$	39.10	39.61
Soccer	34.23 $\sigma_{inp} = 6$	34.71 $\sigma_{inp} = 6$	35.51 $\sigma_{inp} = 6$	35.08 $\sigma_{inp} = 6$	36.20 $\sigma_{inp} = 6$	36.22 $\sigma_{inp} = 6$	**38.38** $\sigma_{psd} = 4$	37.73	38.08
Stefan	35.31 $\sigma_{inp} = 8$	35.34 $\sigma_{inp} = 8$	34.96 $\sigma_{inp} = 6$	34.87 $\sigma_{inp} = 6$	36.19 $\sigma_{inp} = 8$	35.62 $\sigma_{inp} = 8$	36.61 $\sigma_{psd} = 4$	36.50	**36.80**
Stem	34.74 $\sigma_{inp} = 8$	34.66 $\sigma_{inp} = 8$	34.22 $\sigma_{inp} = 6$	33.72 $\sigma_{inp} = 6$	36.14 $\sigma_{inp} = 8$	35.75 $\sigma_{inp} = 8$	36.50 $\sigma_{psd} = 4$	36.22	**36.59**
Tennis	33.33 $\sigma_{inp} = 6$	33.46 $\sigma_{inp} = 6$	34.78 $\sigma_{inp} = 6$	33.95 $\sigma_{inp} = 6$	36.23 $\sigma_{inp} = 6$	35.98 $\sigma_{inp} = 6$	33.67 $\sigma_{psd} = 80$	36.76	**36.93**
Average MSE based	35.17	35.20	35.32	34.79	36.55	36.28	36.83	37.08	**37.44**

Table 6.5: Spatially correlated noise (profile-2): Average error in PSNR (dB) for 150 frames. The input of all methods are tuned to give the highest PSNR. For ours the inputs are fixed ($\sigma_p = 6$, $\hat{\gamma}_0 = 1.6$, $\hat{\gamma}_1 = 1$).

	DDID [23]	BM3D [20]	MHMCF [17]	STGSM [12]	VBM3D [8]	VBM4D [15]	RF3D [61]	Ours R=2	Ours R=5
Bus	33.21 $\sigma_{inp} = 6$	33.23 $\sigma_{inp} = 6$	34.19 $\sigma_{inp} = 6$	32.57 $\sigma_{inp} = 4$	33.52 $\sigma_{inp} = 6$	33.26 $\sigma_{inp} = 6$	33.75 $\sigma_{psd} = 2$	35.35	**35.64**
Flower	33.85 $\sigma_{inp} = 8$	33.59 $\sigma_{inp} = 8$	35.09 $\sigma_{inp} = 6$	33.38 $\sigma_{inp} = 6$	35.26 $\sigma_{inp} = 10$	34.93 $\sigma_{inp} = 10$	**37.24** $\sigma_{psd} = 2$	35.90	36.18
Foreman	35.34 $\sigma_{inp} = 10$	34.80 $\sigma_{inp} = 10$	35.72 $\sigma_{inp} = 6$	33.66 $\sigma_{inp} = 10$	36.42 $\sigma_{inp} = 12$	36.37 $\sigma_{inp} = 10$	35.73 $\sigma_{psd} = 2$	37.32	**37.58**
News	35.82 $\sigma_{inp} = 10$	35.47 $\sigma_{inp} = 10$	37.88 $\sigma_{inp} = 6$	34.65 $\sigma_{inp} = 10$	40.11 $\sigma_{inp} = 12$	39.25 $\sigma_{inp} = 12$	33.80 $\sigma_{psd} = 2$	39.81	**40.20**
Parkjoy	34.02 $\sigma_{inp} = 8$	33.80 $\sigma_{inp} = 8$	34.84 $\sigma_{inp} = 6$	32.80 $\sigma_{inp} = 4$	33.60 $\sigma_{inp} = 8$	33.53 $\sigma_{inp} = 8$	34.11 $\sigma_{psd} = 10$	34.84	**35.01**
Rushhour	35.97 $\sigma_{inp} = 10$	35.53 $\sigma_{inp} = 10$	35.84 $\sigma_{inp} = 6$	34.78 $\sigma_{inp} = 10$	37.66 $\sigma_{inp} = 12$	37.46 $\sigma_{inp} = 12$	34.34 $\sigma_{psd} = 2$	38.10	**38.38**
Soccer	33.07 $\sigma_{inp} = 6$	33.27 $\sigma_{inp} = 6$	35.29 $\sigma_{inp} = 6$	32.92 $\sigma_{inp} = 6$	34.80 $\sigma_{inp} = 8$	34.84 $\sigma_{inp} = 8$	33.87 $\sigma_{psd} = 2$	36.78	**36.96**
Stefan	34.09 $\sigma_{inp} = 8$	33.91 $\sigma_{inp} = 8$	34.80 $\sigma_{inp} = 6$	32.87 $\sigma_{inp} = 4$	34.83 $\sigma_{inp} = 10$	34.41 $\sigma_{inp} = 8$	33.86 $\sigma_{psd} = 2$	35.84	**36.01**
Stem	33.54 $\sigma_{inp} = 8$	33.37 $\sigma_{inp} = 8$	34.09 $\sigma_{inp} = 6$	32.59 $\sigma_{inp} = 4$	34.76 $\sigma_{inp} = 10$	34.48 $\sigma_{inp} = 8$	35.60 $\sigma_{psd} = 2$	35.53	**35.79**
Tennis	32.80 $\sigma_{inp} = 4$	33.01 $\sigma_{inp} = 4$	34.63 $\sigma_{inp} = 6$	32.81 $\sigma_{inp} = 6$	34.41 $\sigma_{inp} = 6$	34.29 $\sigma_{inp} = 6$	32.35 $\sigma_{psd} = 80$	35.86	**36.26**
Average MSE based	34.04	33.91	35.13	33.24	35.18	34.97	34.29	36.32	**36.58**

Table 6.6: Fixed input parameters: Average error in PSNR (dB) for all videos under SCN.

	DDID [23]	BM3D [20]	MHMCF [17]	STGSM [12]	VBM3D [8]	VBM4D [15]	RF3D [61]	Ours R=2	Ours R=5
SCN profile-1	34.99 $\sigma_{inp} = 8$	35.13 $\sigma_{inp} = 6$	35.32 $\sigma_{inp} = 6$	34.76 $\sigma_{inp} = 6$	36.35 $\sigma_{inp} = 8$	36.10 $\sigma_{inp} = 8$	35.89 $\sigma_{psd} = 4$	37.08	**37.44**
SCN profile-2	33.75 $\sigma_{inp} = 8$	33.69 $\sigma_{inp} = 8$	35.13 $\sigma_{inp} = 6$	32.95 $\sigma_{inp} = 4$	34.89 $\sigma_{inp} = 10$	34.88 $\sigma_{inp} = 10$	33.02 $\sigma_{psd} = 2$	36.32	**36.54**

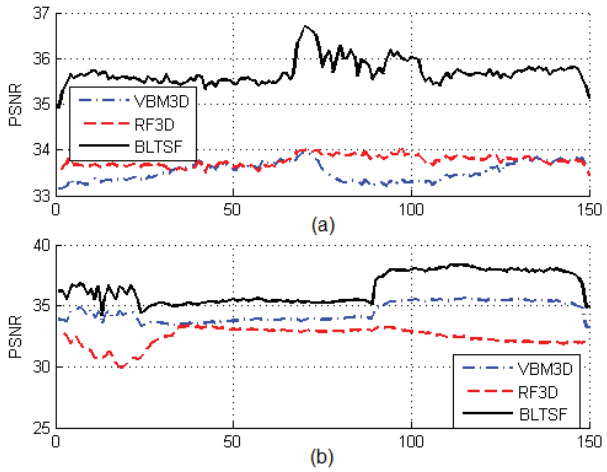

Figure 6.10: Output PSNR of three denoisers under SCN (Table 6.5) for two videos (a) *Bus* and (b) *Tennis*.

changed the input parameters of other algorithms to reach the highest visual quality based on noise-blur trade-off. Figures 6.12 and 6.13 show two examples of denoising for the leading methods of Table 6.4 and 6.5. Normally, MHMCF introduces blocking artifacts and leaves noise since the motion is often not accurately estimated. Block matching methods VBM3D VBM4D, and RF3D tend to leave more LF noise compared to BLTSF and blur the textures under signal-dependent noise such as in the roof and trees in the Figure 6.13.

6.5.4 Effect of motion estimation

We have analyzed the two proposed ideas to solve two problems: propagation of local minima error and blockiness. To address the former, we proposed a homography-based motion correction and for the latter we proposed two-band motion estimation and back-signal subtraction. We have analyzed the effect of homography-based motion correction and we observed as the resolution of the video (and thus the number of levels in the Gaussian pyramid) increases this effect becomes more significant since the error in one block at lowest resolution causes error in large part of frame

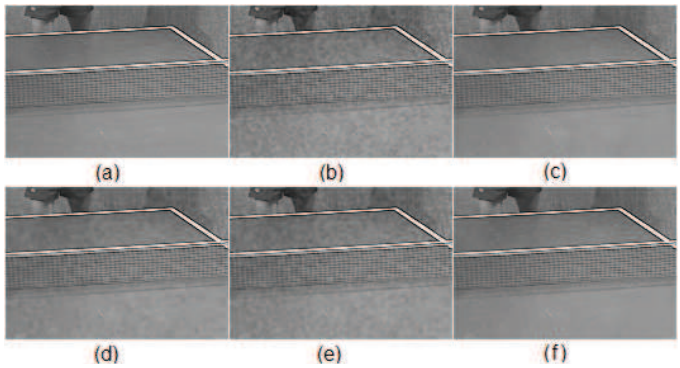

Figure 6.11: Visual result of synthetic SCN. (a) original frame, (b) noisy frame 32.6dB, (c) MHMCF 37.15dB (d) VBM3D 35.46dB, (e) RF3D 32.89dB, and (f) proposed BLTSF 37.94dB.

in the highest resolution. To test this we selected a noise-free 1920×1080 video *Sunflower* and we estimated the motion between two consecutive frames using proposed motion estimation with and without homography-based motion correction. We computed the difference between reference and motion compensated frame and measured the PSNR with 6.14 showing the results. Homography-based motion correction significantly improved the PSNR.

To reduce the blocking artifacts, instead of a block-based, we proposed a two-band motion compensation. Figure 6.15 compares the result of block-based and two-band motion compensation approach for motion compensation of original frame of *Foreman* that is degraded with processed noise (WGN filtered by G-Blur). It shows noticeable less blocking artifacts in the two-band method.

The goal of *back-signal* subtraction is mainly to reduce the blocking artifacts created by block-based motion estimation which leads to a more natural denoising output. It also improves the temporal processing when there is shift in the mean of signal (e.g., flickering). Theoretically, signal decomposition leads to loss of data due to the fact that when the signal is separated into two parts, the characteristic of each part is different compared to original signal. We have managed to minimize the information loss by applying strong edge-stopping filter to extract the back-signal. Thus, the back-signal subtraction improves the visual results especially when the motion is complex and the frame contains powerful LF signal (gradients of shades). Figure 6.16 shows an example of *back-signal* and *fore-signal* and how the back-signal subtraction reduces the blocking artifacts.

Our experiments show that in most cases by using *back-signal* subtraction the performance

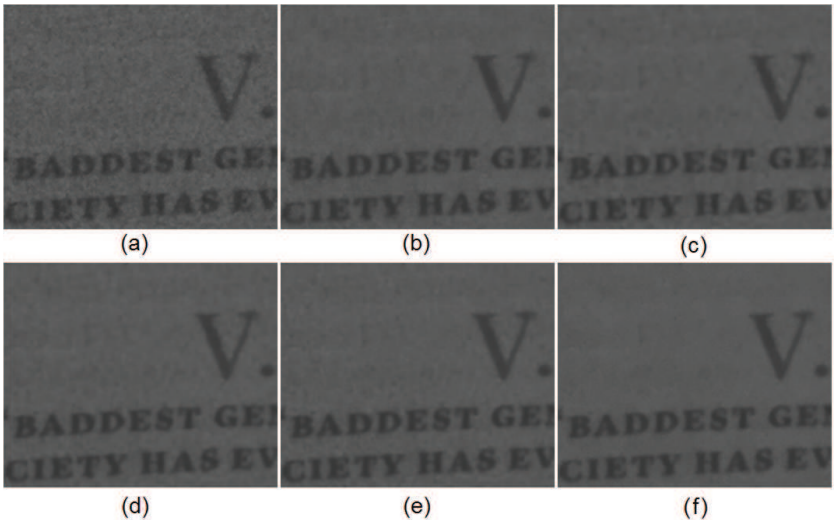

Figure 6.12: Denoising results of real processed noise (a), using (b) MHMCF ($\sigma_i = 3$), (c) VBM3D ($\sigma_i = 7$), (d) VBM4D ($\sigma_i = 7$), (e) RF3D ($\sigma_{psd} = 1.45$), and (f) BLTSF ($\sigma_p = 7, \hat{\gamma}_0 = 1.6, \hat{\gamma}_1 = 1.1$). Input parameters of all methods are set to obtain best visual results, i.e., trade-off between noise and blur. For ours, noise is automatically estimated. LF noise is better removed by BLTSF.

of temporal filter increases objectively and subjectively. It significantly decreases the blocking artifacts. It increases also the effectiveness of temporal filtering which lead to less spatial filtering and thus, less spatial domain artifacts (e.g., ringing and posterization). We have tested the effect of *back-signal* subtraction using synthetic noisy videos under 25dB WGN and we compared the results. Figure 6.17 shows the its effect on the PSNR for the output of temporal filter G_t. By excluding LF signal from the process the temporal filter becomes more operative and performance improves. Another important use case of *back-signal* subtraction is when the mean of signal fluctuates over the time. This take places when the luminance source flickers or the white balancing system is not stable. The white balancing system in the cameras is normally sensitive to the overall brightness of the scene. When the overall brightness changes, although the scene change is small, a different white balancing function will be applied compared to the previous frame. This changes the global mean of the frame. In these cases *back-signal* subtraction helps by excluding the LF and mean shift from from processing. To test the flickering effect, we have added a fixed value 5 to every other noisy frame in the previous experiments. Figure 6.18 shows the effect of *back-signal* on the PSNR for the output of temporal filter G_t.

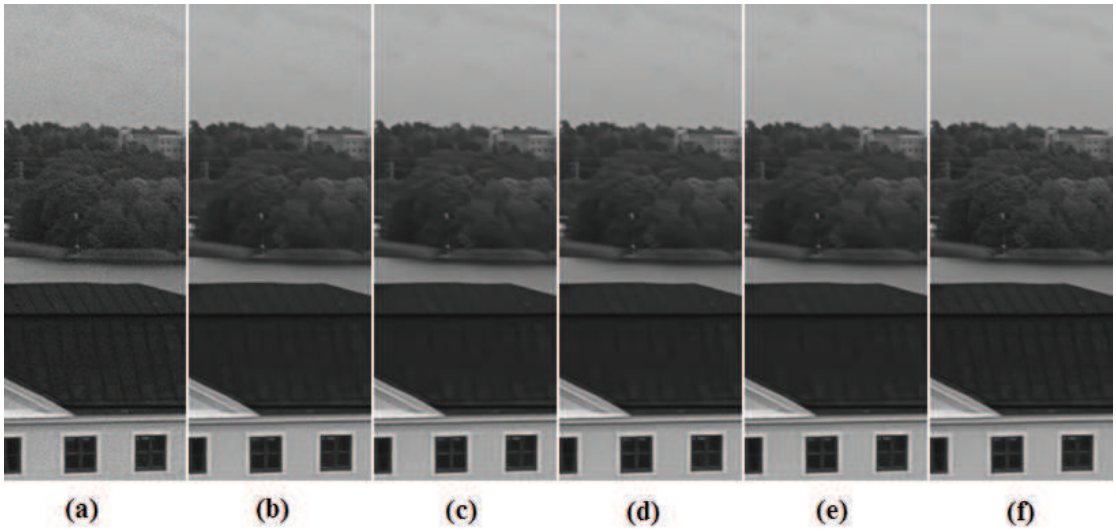

Figure 6.13: Denoising results of real camera noise (a), using (b) MHMCF, (c) VBM3D, (d) VBM4D, (e) RF3D, and (f) proposed BLTSF. Input parameters of all methods are set to obtain the best visual results, i.e., noise-blur trade-off. For ours, noise is automatically estimated. BLTSF is able to remove noise in the sky and keep the textures of roof and trees.

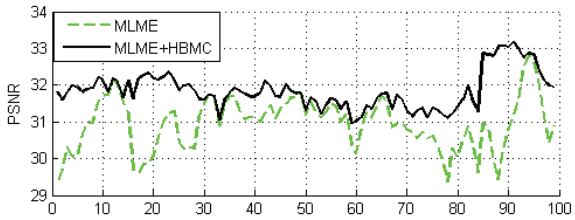

Figure 6.14: Effect of homography-based motion correction (HBMC) in multi-layer motion estimation (MLME). We calculated the difference (in PSNR) between the current frame and the motion compensated subsequent frame for the non-noisy *sunflower* video sequence.

6.5.5 Effect of spatial filter

Temporal information has an important role in our proposed noise removal method especially when the majority of MVs are accurate. This makes using a complex spatial filter less justified especially when the noise is processed, i.e., already spatially correlated. We have analyzed the effect of using a high-performance complex spatial filter by using a significantly more complex DDID [23]. We compared the performance of proposed and DDID spatial filters. Verified by Figure 6.19 in all tests DDID improves the performance, however, it is significantly (150 times) slower.

Figure 6.15: Two-band motion-compensation for a noisy *Foreman* frame. (a) block-based motion-compensation. (b) proposed motion-compensation. Blocking artifacts are significantly reduced in (b).

6.5.6 Temporal filter applied to WGN

We have tested the proposed BLAB with recursive structure in comparison with two other state-of-the-art methods MHMCF and CICIF. We have added WGN with STD of 15 to 10 videos (dataset Figure 5.14) and created noisy frames. For all methods, we have set the block size $W_r = 16$ and temporal window $R = 2$. Figure 6.20 compares the PSNR results of denoised outputs for 10 videos sequences (dataset of Figure 5.14) averages over 150 frames. In all videos proposed methods provides higher PSNR compared to CICIF. It also provides higher PSNR compared to MHMCF except for *Tennis* video. In average proposed method provides 1.1dB and 0.6dB higher PSNR compared to MHMCF and CICIF. Figure 6.21 also shows the PSNR curve for sequences *Foreman* and *Stefan* which the proposed provides higher PSNR in average. Figure 6.22 also shows the visual result for 20^{th} frame of *Foreman* video. Using two-band motion compensation and back-signal subtraction, the blocking artifacts is eliminated. Also by a reliable error detection, proposed method better preserves the details.

6.5.7 Spatial filter applied to WGN

To test the performance of proposed spatial filter under WGN, we have added zero-mean WGN with STD = 14.3 (PSNR = 25dB) to the ground-truth frames with Table 6.7 showing the MSE comparison between bilateral filter [21] and non-local mean [118], DDID_1 (only first iteration), DDID complete [23], BM3D [20] and our algorithm. We used the first frame of 10 videos in this experiment. Results show the proposed filter is reliable under WGN considering that according to our simulations the proposed method is 910 times faster than DDID, 210 times faster than DDID_1, and 83 times faster than BM3D.

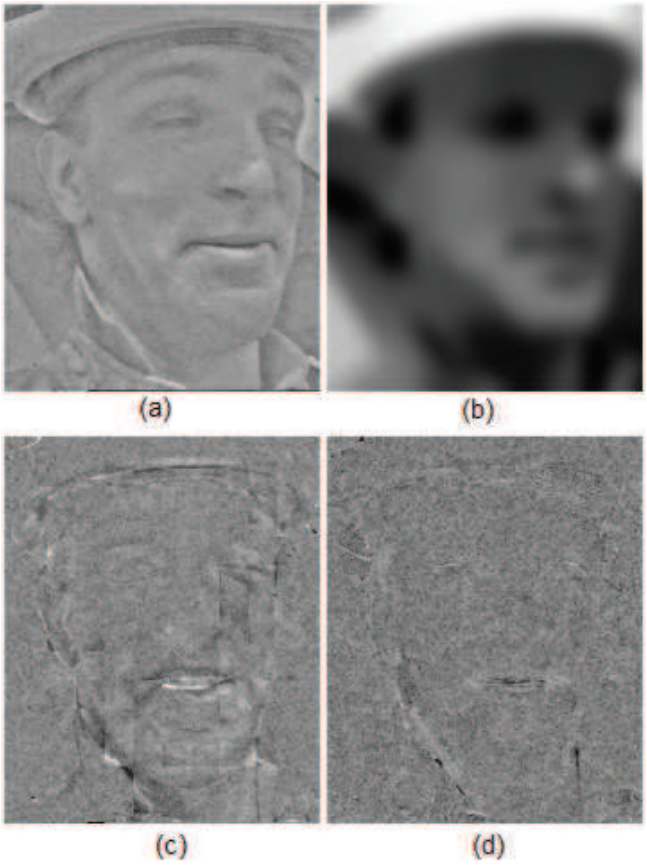

Figure 6.16: Example of *back-signal* subtraction. The noisy frame is decomposed into (a) *fore-signal* and (b) *back-signal*. (c) and (d) are the temporal difference when the original noisy frame and *fore-signal* are motion compensated. Temporal artifacts is significantly reduced in (d).

6.5.8 Implementation issues

We have implemented the proposed framework on both CPU and GPU platforms using C++ and OpenCL programming languages and we have run our method under MATLAB platform. Using fast filtering operations, the proposed method is highly parallelizable and we gained a significant acceleration employing GPU. We used an Intel i7 3.07GHz CPU and Nvidia GTX 970 GPU. For a CIF video (352×288) the per frame processing time of BLTSF is 190ms under CPU and 21ms under GPU. For other methods we used their MATLAB implementations that may not be speed-wise optimized but the timings are 71 second for DDID, 1.29 second for BM3D, 240ms for MHMCF, 14.32 second for STGSM, 290ms for VBM3D, 290ms for VBM3D, 2.16 second for VBM4D, and 898ms for RF3D.

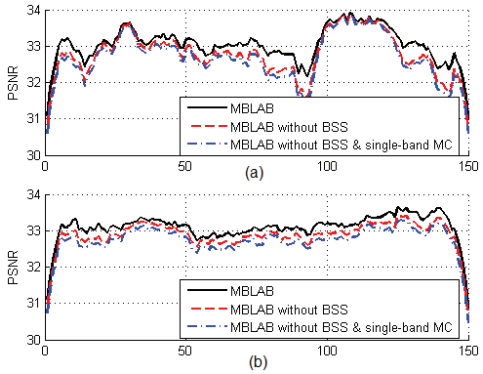

Figure 6.17: The effect of back-signal subtraction and two-band motion compensation on the performance of temporal filter. For the (a) *Foreman* proposed leads to 0.18dB and 0.4dB higher PSNR in average compared without *back-signal* subtraction (BSS) and single-band motion compensation. For the (b) *Rushhour* proposed leads to 0.21dB and 0.35dB higher PSNR in average.

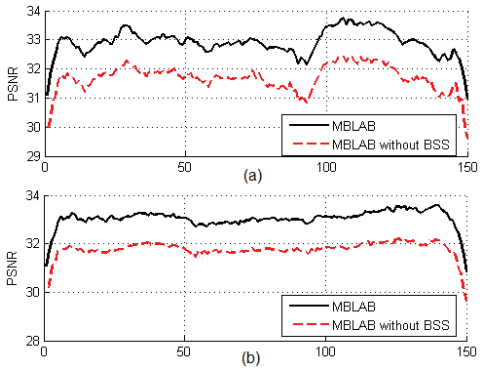

Figure 6.18: The effect of BSS on the performance of temporal filter when the mean of signal changes over the time (flickering). For the (a) *Foreman* and (b) *Rushhour back-signal* subtraction (BSS) leads to 1.2dB and 1.3dB higher PSNR in average.

6.6 Conclusion

We have developed a time-space video denoising method that employs a fast block-matching motion-estimation method, yet yields to competitive results compared to the state-of-the-art methods. We propose a two-band motion compensation, smooth weight calculation and band-limited filtering to address blocking created by motion estimation. It uses a two-scale motion error error detection and adjusts the the noise level when it is overestimated which leads to less motion blur effects compared to relevant methods. In addition, our A modular design provides the feature to adjust the performance-speed point by changing certain components and parameters, such as motion-estimation, temporal radius, and spatial filter. We have benefited from the parallelizable structure of

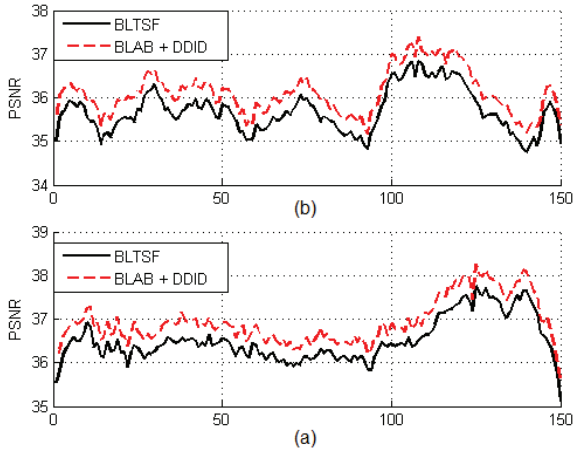

Figure 6.19: Effect of using a costly spatial filter in BLTSF. For the (a) *Foreman* and (b) *Rush-hour* using DDID [23] leads to 0.41dB and 0.42dB higher PSNR in average. Synthetic 25dB WGN is used (Table 6.3).

Table 6.7: MSE comparison under WGN.

	Bilateral filter	Non-local mean	$DDID_1$	DDID	BM3D	Ours
Akiyo	79.70	20.54	21.99	13.28	12.65	17.81
Bus	118.09	74.26	72.87	58.03	55.91	59.61
Coastguard	112.20	66.29	58.84	50.40	49.22	56.33
Flower	118.68	108.91	87.52	69.77	67.85	72.27
Foreman	87.97	28.05	30.91	20.87	20.32	26.82
Hall	91.13	30.49	33.12	21.82	19.62	25.12
News	91.26	34.41	36.59	24.92	23.00	27.97
Sean	94.73	43.21	43.55	31.90	29.87	35.00
Stefan	122.79	65.02	67.57	51.20	47.50	52.30
Tennis	111.72	104.10	96.32	91.51	96.82	89.49
Average	102.83	57.53	54.93	43.37	42.27	46.27

our method and we accelerated that using GPU. We also presented a method to integrate temporal filter to any spatial filter by considering the level of residual noise at each pixel. We presented a fast but effective spatial filter to be used as a back-up for temporal filter to remove residual noise left after temporal filtering.

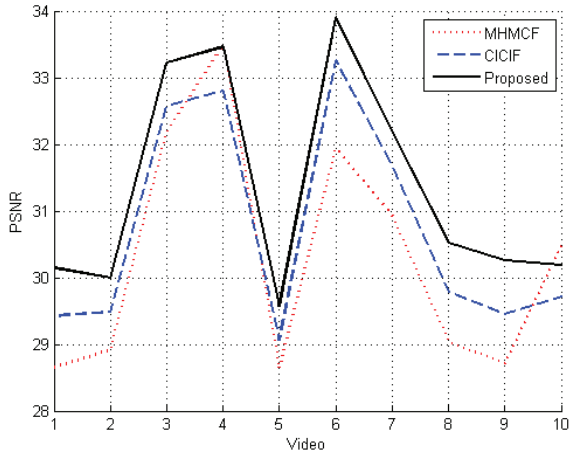

Figure 6.20: PSNR results averaged over 150 frames for 10 video sequences (dataset Figure 5.14) under WGN with STD of 15. The average PSNR for all videos are 30.0dB for MHMCF, 30.4dB for CIFIC, and 31.1dB for proposed method.

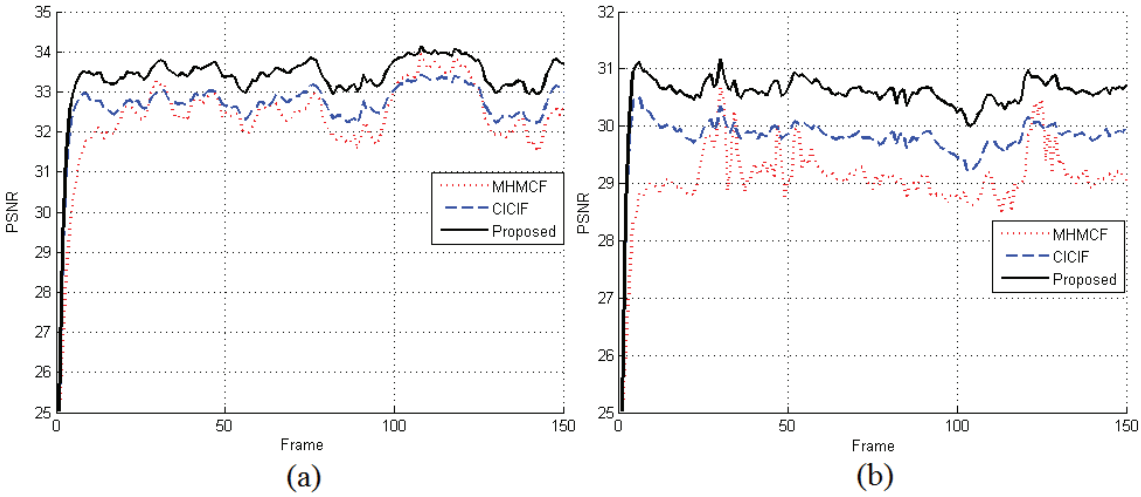

Figure 6.21: PSNR result for video sequence (a) *Foreman* (with average PSNR of 32.1dB for MHMCF, 32.5dB for CIFIC and 33.2dB for proposed), and (b) *Stefan* (with average PSNR of 29.0dB for MHMCF, 29.7dB for CIFIC and 30.5 for proposed)

.

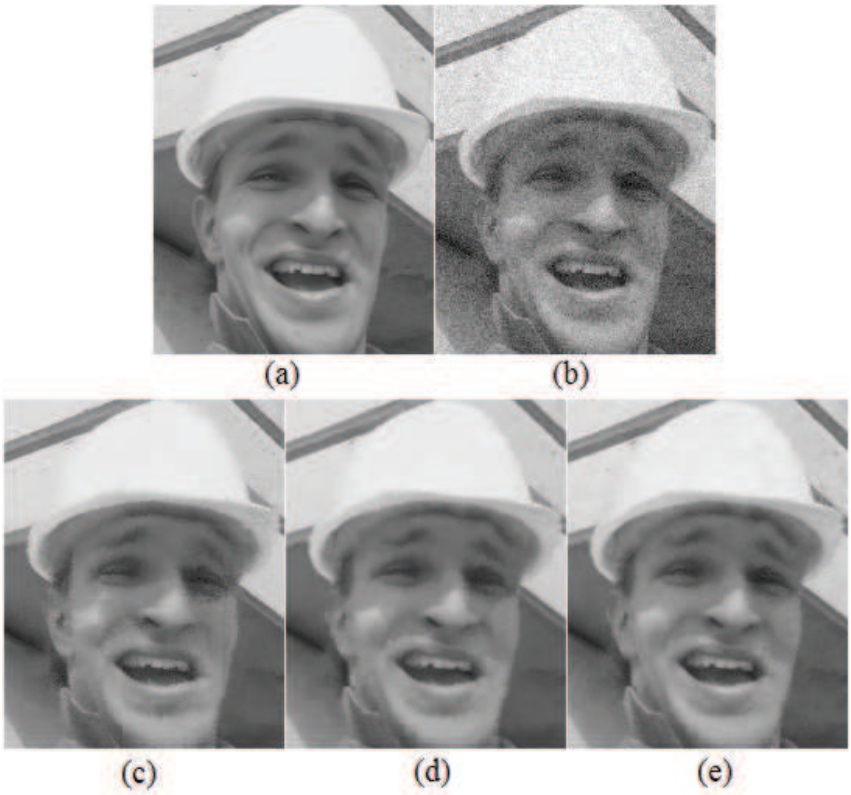

Figure 6.22: Visual results for *Foreman* video. (a) Original (b) Noisy (c) MHMCF with PSNR 31.9dB, (d) CICIF with PSNR 32.4dB, and proposed with PSNR of 33.2dB.

Chapter 7

No-Reference Image Quality Assessment

7.1 Introduction

In this chapter, we present a no-reference image quality assessment, NR-IQA method based on estimation of local image structure using orientation dominancy and patch sparsity. We propose a fast method to find the dominant orientation of image patches, which is used to decompose them into singular values. Combining singular values with the sparsity of the patch in the transform domain, we measure the possible image content and noise of the patches and of the whole image. To measure the effect of noise accurately, our method takes both low and high textured patches into account. Before analyzing the patches, we apply a shrinkage in transform domain to increase the contrast of genuine image structure. We assume noise can be real noise (SDSCN) as discussed in the chapter 2. Our objective and subjective results confirm the correspondence between the measured quality and the ground-truth. We show that the proposed method rivals related NR-IQA approaches.

Dissimilar to other methods that use SVD of the local gradient to detect image content our method 1) applies a Fourier shrinkage prior to orientation detection, 2) takes the Fourier sparsity of the patches into account, and 3) addresses the blur by considering the absolute power of image signal.

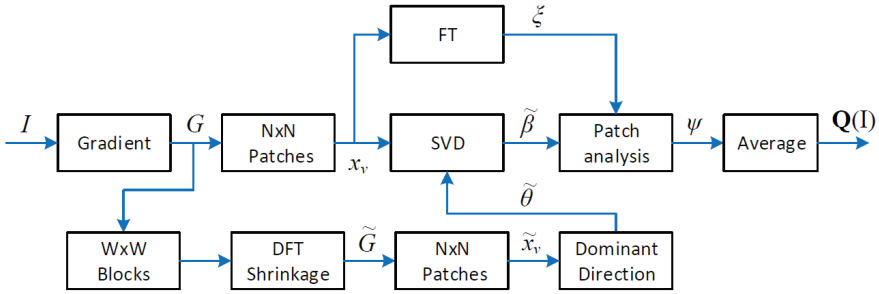

Figure 7.1: Block diagram of the proposed algorithm.

7.2 Overview of proposed quality assessment

The proposed method consists of three main steps. In the first step we compute the image gradient and apply a local Fourier shrinkage on the gradient image to generate a better approximation of the image signal. For speed consideration, we divide the gradient image into $W_q \times W_q$ overlapping blocks and we use two dimensional DFT to apply the shrinkage. In the second step, we divide the shrunk gradient image into non-overlapping patches of $N_q \times N_q$ and compute the dominant gradient orientation for each patch using the SVD analysis. In the final step, we divide the original gradient image into non-overlapping patches and we compute both SVD and DFT sparsity of each patch. We combine the sparsity information to measure the local signal noise power and finally image quality. The proposed NR-IQA method can be summarized as in Algorithm 4, and as in the block diagram of Fig. 7.1.

Algorithm 4 Proposed SDQI

1: **Compute** the complex gradient map $\mathbf{G}$ from the input image I using (87).
2: **Divide** the gradient $\mathbf{G}$ into overlapping blocks of $W_q \times W_q$ and apply a Fourier shrinkage via (93).
3: **Divide** the shrunk gradient $\tilde{\mathbf{G}}$ into non-overlapping patches of $N_q \times N_q$ and compute the dominant direction $\tilde{\theta}$ for each patch via (97).
4: **Divide** the gradient $\mathbf{G}$ into non-overlapping patches of $N_q \times N_q$ and compute the local sparsities using (90) (92).
5: **Calculate** quality value for each patch using (99).
6: **Output** the QI by averaging the local values via (102).

7.3 Proposed method

7.3.1 Proposed image content model

Assuming a transform such as SVD decorrelates a signal into orthogonal coefficients; generally most of the signal energy is represented in few coefficients creating a sparse representation of the image [119]. We employ this feature to differentiate between signal and noise to detect the true image content. Since the nature of the noise is random, it is unlikely to represent the noise in sparse form especially when the noise is represented in all orthonormal coefficients (e.g., white noise). Thus, the signal is more likely to be image content when it can be represented in a sparser form. We use SVD and DFT to maximize the chance of detecting content by benefiting the advantages of both. SVD is useful to detect single orientation signal, but cannot detect multi-orientation signals. On the other hand, DFT is beneficial in finding multi-orientation signals, but mistakes spatially correlated noise with signal.

High-frequency image components carries the edge information and the goal of a denoiser is to preserve them while removing the noise. We use the image gradient to extract edge information. For an image of interest I the gradients $\mathbf{G}_x$, $\mathbf{G}_y$, and the complex gradient image $\mathbf{G}$ can be calculated as

$$\mathbf{G}_x = (\mathbf{H}_s^T \mathbf{H}_d) * I,\ \mathbf{G}_y = (\mathbf{H}_d^T \mathbf{H}_s) * I,\ \mathbf{G} = \mathbf{G}_x + \jmath \mathbf{G}_y, \tag{87}$$

where $\jmath = \sqrt{-1}$, $\mathbf{H}_d = [-1\ 0\ 1]$, and $*$ denotes the two-dimensional convolution. Examples for $\mathbf{H}_s$ are $\mathbf{H}_s = \frac{1}{2}[0\ 1\ 0]$ or the Sobel operator $\mathbf{H}_s = \frac{1}{8}[1\ 2\ 1]$. Gradient orientations of the pixels on the edges are similar, whereas on the noisy pixels are random. The similarity of gradient orientations can be utilized to distinguish edge pixels and hence image content. We utilize SVD and DFT to find this similarity as follows.

Sparse DFT

Let $\dot{\mathbf{x}}$ be a patch (block) of the gradient image $\mathbf{G}$ of size $N_q \times N_q$ which rearranged into a column vector with size of $N_q^2 \times 1$; thus $\dot{\mathbf{x}}$ can be represented as a linear combination of orthonormal DFT

dictionary (basis) matrices as

$$\dot{\mathbf{x}} = \mathbf{D}\alpha, \tag{88}$$

where $\mathbf{D}$ is a $N_q^2 \times N_q^2$ DFT dictionary and α is a $N_q^2 \times 1$ vector of complex numbers. The sparsity of DFT means that most of the power of α is concentrated in few elements. If we sort the α according to their magnitude from high to low to create $\hat{\alpha}$, we define ι as the set of numbers that meets,

$$\iota = \left\{ n \mid n \in \mathbb{N}\ ,\ \frac{\sum_{k=1}^{n} |\hat{\alpha}_k|^2}{\sum_{k=1}^{N_q^2} |\hat{\alpha}_k|^2} \geq \boldsymbol{T}_\delta^q \right\}, \tag{89}$$

where $0 \leq \boldsymbol{T}_\delta^q \leq 1$ is a constant, $\hat{\alpha}_k$ means the k^{th} elements of $\hat{\alpha}$ and thus $\sum_{k=1}^{N^2} |\hat{\alpha}_k|^2$ is the whole energy of the patch. We define ι_{min} as the minimum number that contains $\boldsymbol{T}_\delta^q$ energy of the signal, $\iota_{min} = \min(\iota)$. For instance, when $\boldsymbol{T}_\delta^q = 0.9$, at least 90% of the energy of α is assigned to the ι_{min} element. We then define the inverse sparsity degree of DFT as follows,

$$\xi^{-1} = \frac{\iota_{min} \cdot \boldsymbol{T}_\delta^q \sum_{k=1}^{N^2} |\hat{\alpha}_k|^2}{N_q^2 \sum_{k=1}^{\iota_{min}} |\hat{\alpha}_k|^2}, \tag{90}$$

The more sparse α is, the higher the ξ becomes. Fig. 7.2(b) shows an example of ξ for the image *Barbara* with $\boldsymbol{T}_\delta^q = 0.75$.

Sparse SVD

Consider x a patch of gradient image $\mathbf{G}$ with size $N_q \times N_q$ is separated into real and imaginary part forming a $N_q^2 \times 2$ matrix $\dot{\mathbf{x}}_v = [\dot{\mathbf{x}}_r\ \dot{\mathbf{x}}_i]$. $\dot{\mathbf{x}}_v$ can be decomposed into two singular values s_1 and s_2 [120, 121] as

$$\dot{\mathbf{x}}_v = \begin{bmatrix} U_r & U_i \end{bmatrix} \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}, \tag{91}$$

where $[U_r\ U_i]$ is an orthonormal matrix, meaning $U_r U_i^T = 0$, $s_1 \geq s_2 \geq 0$, and θ is a constant that represents the dominant orientation of $\mathbf{G}$. When the signal energy is concentrated in one direction $s_1 \gg s_2$, *orientation dominancy* or *SVD sparsity* happens. Consider the SVD sparsity factor β as

$$\beta = \frac{s_1}{s_2} \geq 1. \tag{92}$$

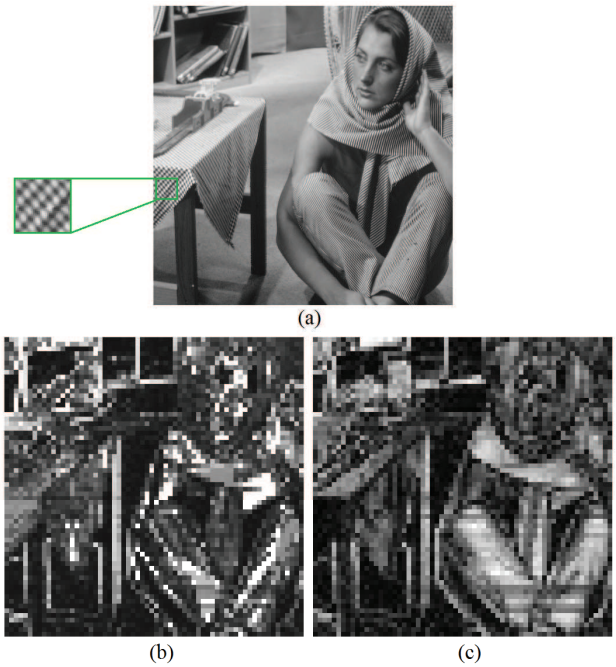

Figure 7.2: Sparsity of gradients in 8×8 patches for the image *Barbara* (a). (b) shows the DFT sparsity ξ with $\boldsymbol{T}_{\delta}^{q} = 0.75$ and (c) shows SVD sparsity β. Brighter patches are sparser. SVD sparsity cannot detect multi-orientation textures while DFT can.

The more the pixels of x are aligned in a single direction the higher the β becomes. On the other hand, if x contains pixels with random directions, β becomes small. Fig. 7.2(c) shows an example of β for the image *Barbara*. SVD sparsity locates single direction edges accurately but not image content with multi-direction repeated textures, as highlighted in Fig. 7.2(a), while DFT can.

7.3.2 Fourier shrinkage

The objective of this step is to increase the contrast of edge signals by suppressing the noisy Fourier coefficients. Assuming we divide the gradient image $\mathbf{G}$ into overlapping blocks of $W_q\times W_q$ and the DFT coefficients α are computed, a shrinkage procedure suppresses the noisy α and increases the contrast between signal and noise. We use the term *patch* to indicate an $N_q\times N_q$ image region and *block* for an $W_q\times W_q$ one.

We assume α coefficients with a relative small magnitude are more likely to be noise, thus they should be suppressed. We use the median of $|\alpha|$ as a reference point. Let W_q be an even number and $\alpha_{med} = \frac{1}{2}(|\hat{\alpha}[\frac{1}{2}W_q^2 - 1]| + |\hat{\alpha}[\frac{1}{2}W_q^2]|)$ is the median of $|\alpha|$. In order to suppress small DFT

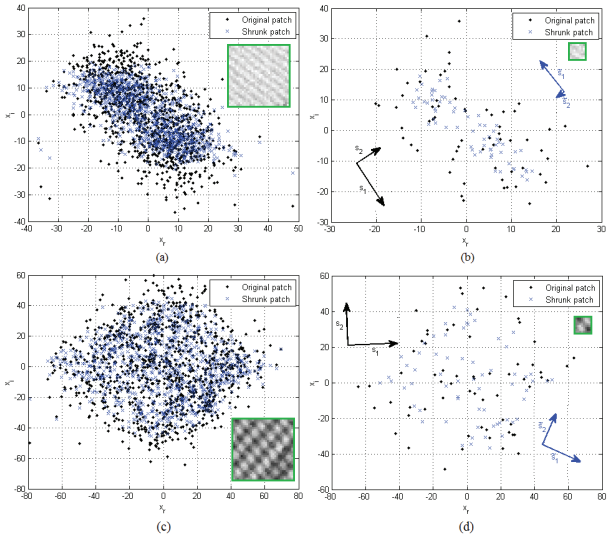

Figure 7.3: Shrinkage with $W = 16$ changes the dominant orientation (angle of arrows). For a sample 32×32 part of *Barbara* (a) and (c), the shrinkage is applied which changed the dominant orientations of 8×8 patches (b) and (d).

coefficients we propose modifying each DFT coefficient as

$$\tilde{\alpha} = \alpha \cdot \exp\left(-\frac{\boldsymbol{c}_{\alpha}^{q} \cdot \alpha_{med}^{2}}{|\alpha|^{2}}\right), \tag{93}$$

where $\boldsymbol{c}_{\alpha}^{q}$ is a constant. (93) suppresses (or shrinks) the small coefficient relative to α_{med}. If α_{med} is small, i.e., α is sparse, the shrinkage has no effect. For each block, first α is computed and suppressed to $\tilde{\alpha}$ and then an inverse DFT is applied to obtain the modified (or the shrunk) gradient map in pixel domain. Since we are using sliding windows with overlapping, the results of the individual blocks are averaged to create the whole shrunk gradient map $\tilde{\mathbf{G}}$. Let O_q be the size of overlapping in pixels; $\tilde{\mathbf{G}}$ at each position is calculated from the average of $W_q^2/(W_q - O_q)^2$ blocks. For example, when $O_q = \frac{W_q}{2}$ the average of 4 blocks is required to calculate each pixels. Our idea is to set $W_q > N_q$ so a more global shrinkage on a larger block affects a local small patch. Fig. 7.3 shows how a global gradient shrinkage affects the dominant orientation, especially for smaller patches. Since the noise is suppressed, dominant orientation can be estimated more accurately after shrinkage.

7.3.3 Dominant orientation

When the pixels are locally aligned in a single orientation, there is a high chance of image content presence. To exploit this property, we divide the shrunk gradient map $\tilde{\mathbf{G}}$ into non-overlapping patches of $N_q \times N_q$ and for each patch we form $\tilde{\mathbf{x}}_v = [\tilde{\mathbf{x}}_r \ \tilde{\mathbf{x}}_i]$ by rearranging the real and imaginary part of patch values to form a $N_q^2 \times 2$ matrix. The dominant orientation of the shrunk patch $\tilde{\theta}$ is computed to meet (91). Generally, algorithms for computing singular values are related to eigenvalue computing of symmetric matrices. The QR algorithm [122] reduces rectangular matrix to *bidiagonal* using a *Householder* reduction. Although these iterative matrix computations give accurate results [123], their complexity makes them hard to implement. To compute the dominant orientation $\tilde{\theta}$, instead of iterative approaches, we propose a simpler solution. To meet the condition $U_r U_i^T = 0$ or equally $\tilde{s}_1 U_r \cdot \tilde{s}_2 U_i^T = 0$ in (91) we should have,

$$\sum_{k=1}^{N^2} (\tilde{\mathbf{x}}_r[k]\cos\tilde{\theta} + \tilde{\mathbf{x}}_i[k]\sin\tilde{\theta})(-\tilde{\mathbf{x}}_r[k]\sin\tilde{\theta} + \tilde{\mathbf{x}}_i[k]\cos\tilde{\theta}) = 0, \tag{94}$$

where $\tilde{\mathbf{x}}_r[k]$ and $\tilde{\mathbf{x}}_i[k]$ are the k^{th} element of $\tilde{\mathbf{x}}_r$ and $\tilde{\mathbf{x}}_i$, respectively; it follows,

$$\cos\tilde{\theta}\sin\tilde{\theta} \sum_{k=1}^{N^2} (\tilde{\mathbf{x}}_i^2[k] - \tilde{\mathbf{x}}_r^2[k]) = (\cos^2\tilde{\theta} - \sin^2\tilde{\theta}) \sum_{k=1}^{N^2} (\tilde{\mathbf{x}}_r[k]\tilde{\mathbf{x}}_i[k]). \tag{95}$$

With $cos(2\tilde{\theta}) = (\cos^2\tilde{\theta} - \sin^2\tilde{\theta})$ and $sin(2\tilde{\theta}) = \frac{1}{2}(\cos\tilde{\theta}\sin\tilde{\theta})$,

$$\tan(2\tilde{\theta}) = \frac{2\sum_{k=1}^{N^2} (\tilde{\mathbf{x}}_r[k]\tilde{\mathbf{x}}_i[k])}{\sum_{k=1}^{N^2} (\tilde{\mathbf{x}}_i^2[k] - \tilde{\mathbf{x}}_r^2[k])}, \tag{96}$$

so, the dominant orientation $\tilde{\theta}$ in $\tilde{\mathbf{x}}_v$ can be computed as,

$$\tilde{\theta} = \frac{1}{2}\tan^{-1}\left(\frac{2\sum_{k=1}^{N^2} (\tilde{\mathbf{x}}_r[k]\tilde{\mathbf{x}}_i[k])}{\sum_{k=1}^{N^2} (\tilde{\mathbf{x}}_i^2[k] - \tilde{\mathbf{x}}_r^2[k])}\right), \tag{97}$$

which is simpler to implement and faster compared to general iterative SVD computations. We use dominant orientation $\tilde{\theta}$ to compute SVD sparsity in (98).

7.3.4 Patch sparsity analysis

We developed a solution using two criteria; orientation dominancy (SVD sparsity) and DFT sparsity; we used them to analyze the likelihood of image signal presence in a patch. To exploit orientation dominancy, we divide the original gradient map $\mathbf{G}$ into patches of $N_q \times N_q$ and we use $\tilde{\theta}$ estimated from shrunk gradient map $\tilde{\mathbf{G}}$ to compute the orientation dominancy of $\mathbf{G}$. Assuming $\tilde{\mathbf{x}}_v$ is a $N_q^2 \times 2$ of a patch of shrunk gradient $\tilde{\mathbf{G}}$ at a certain location and $\dot{\mathbf{x}}_v$ is the corresponding patch of $\mathbf{G}$ at the same location; First, we compute the dominant orientation of shrunk patch $\tilde{\theta}$ according to (97) for $\tilde{\mathbf{x}}_v$ and then we use it to calculate the modified singular values of $\dot{\mathbf{x}}_v$, i.e., $\tilde{s}_1$ and $\tilde{s}_2$ as

$$\begin{aligned} \tilde{s}_1 &= \sqrt{\sum_{k=1}^{N^2} \left| \dot{\mathbf{x}}_r[k] cos\tilde{\theta} + \dot{\mathbf{x}}_i[k] sin\tilde{\theta} \right|^2} \\ \tilde{s}_2 &= \sqrt{\sum_{k=1}^{N^2} \left| \dot{\mathbf{x}}_i[k] cos\tilde{\theta} - \dot{\mathbf{x}}_r[k] sin\tilde{\theta} \right|^2}. \end{aligned} \tag{98}$$

$\tilde{s}_1$ and $\tilde{s}_2$ are different from s_1 and s_2 the singular values of $\dot{\mathbf{x}}_v$ (see Fig. 7.3). In (98) we use $\tilde{\theta}$ the dominant orientation of shrunk patch instead of θ the dominant orientation of $\dot{\mathbf{x}}_v$. $\tilde{s}_1$ is the energy of signal along the $\tilde{\theta}$ and $\tilde{s}_2$ is the energy of along the perpendicular direction $(\pi/2 - \tilde{\theta})$. Fig. 7.4 illustrates how these energies are computed according to (98). Using (92), we define the sparsity of SVD after shrinkage as $\tilde{\beta} = \frac{\tilde{s}_1}{\tilde{s}_2}$. When the singular values are sparse, i.e., $\tilde{s}_1 \gg \tilde{s}_2$ (or $\tilde{\beta} \gg 1$), the probability of image content presence is higher. Theoretically $\beta \geq 1$; however, it is not guaranteed that $\tilde{\beta} \geq 1$. $\tilde{\beta} \leq 1$ implies that the probability that the patch contains image signal is low. We propose a likelihood function that maps this property to the local quality of the patch as

$$\psi = \frac{(\tilde{\beta} - 1 - \epsilon)}{\tilde{\beta} + \tilde{\beta}_0}, \tag{99}$$

where $\epsilon \geq 0$ and $\tilde{\beta}_0 \geq 0$ are computed in (100) and (101) based on $\tilde{s}_1$ and ξ. ψ is a value indicating the relative quality of the patch. In case $\tilde{\beta} \leq 1$, ψ becomes negative, implying that the patch contains no useful signal. In related work [36], patches with small signal to noise ratio are rejected and the effect of noise in noisy patches is not considered. Our idea is that ψ can be negative to highlight the impact of noise in overall image quality. We can consider $|\psi|$ as a probability that shows signal

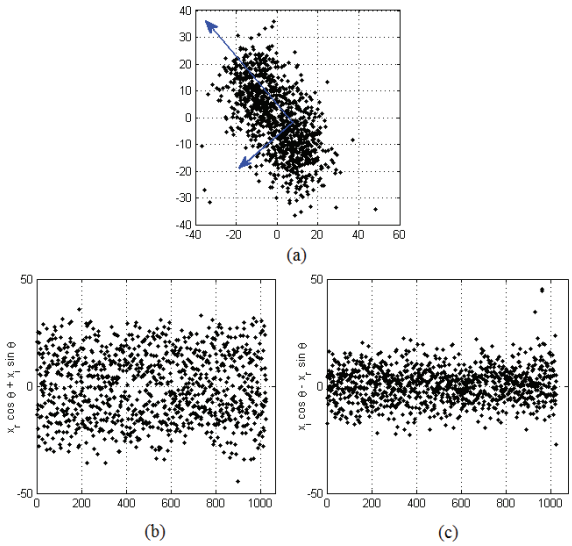

Figure 7.4: Example of computing the singular values for (a) a gradient patch. Samples are rotated to be aligned in the dominant (b) and perpendicular (c) orientation. The energy $s_1 = 505.8$ along the dominant orientation is greater than the energy $s_2 = 267.9$ along the perpendicular orientation.

presence in the case of $\psi > 0$ and noise existence in case of $\psi < 0$. Therefore, the effect of noise is more considered in our algorithm. We propose to use the DFT sparsity of $\dot{\mathbf{x}}_v$, i.e., ξ to compute the ϵ,

$$\epsilon = \max(\xi^{-1} - \xi_{max}^{-1}, 0), \tag{100}$$

where ξ_{max} is a constant representing a relatively large value for ξ. When $\xi \geq \xi_{max}$, i.e., the DFT is very sparse, $\epsilon = 0$. On the other hand, when ξ is relatively small, ϵ becomes non zero. In this case when $\tilde{\beta}$ is also relatively small ($\tilde{\beta} < \epsilon$), ψ becomes negative. In fact, ϵ is an adjustment to increase the reliability by taking the DFT sparsity into account. Fig. 7.5 shows ψ with $\tilde{\beta}_0 = 0$ for different values of ϵ. In (99), only relative values of decomposed signal, i.e., ratios of high power to low-power coefficients are considered. In a weak-textured patch, it is possible that the absolute values $\tilde{s}_1$ and $\tilde{s}_2$ are small but their relative value $\tilde{\beta}$ is large. In order to detect blur and compression artifacts, weak-textured patches should be addressed by ranking lower the smaller s_1. We define $\tilde{\beta}_0$ to adjust the quality of patch as

$$\tilde{\beta}_0 = \frac{c_\beta^2}{c_\beta^2 + \tilde{s}_1^2}, \tag{101}$$

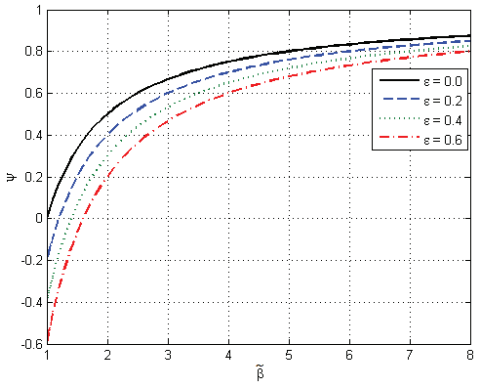

Figure 7.5: Quality of a patch ψ according to the SVD sparsity $\tilde{\beta}$ and DFT based adjustment ϵ.

where $\boldsymbol{c}_{\beta}^{q}$ is a constant. Smaller value of $\tilde{s}_1$ compared to $\boldsymbol{c}_{\beta}^{q}$ leads to larger value of $\tilde{\beta}_0$ and thus smaller ψ. Increasing $\boldsymbol{c}_{\beta}^{q}$ makes ψ more sensitive to blur, but decreases the sensitivity to noise. Thus, to define $\boldsymbol{c}_{\beta}^{q}$, a suitable trade-off between noise and blur should be considered.(see section 7.4).

7.3.5 Quality index

In (99) we have defined ψ as a measure to quantify the probability of the signal presence in each patch. Assuming the directional energy $\tilde{s}_1$ is the signal of interest, we consider its expected value as a measure for genuine image content. The expected energy of signal is computed by multiplying $\tilde{s}_1$ by its presence probability ψ. By aggregating all of genuine energies (i.e., expected values), we compute the overall genuine energy for the entire image to quantify the quality of the image $\mathbf{Q}(I)$, assuming the input image contains K patches,

$$\mathbf{Q}(I) = \frac{1}{K} \sum_{k=1}^{K} \tilde{s}_1[k]\psi[k], \tag{102}$$

where $\tilde{s}_1[k]$ and $\psi[k]$ are the $\tilde{s}_1$ and ψ of the k^{th} patch. Negative value of ψ implies presence of noise without any signal. Thus, when $\psi < 0$, $|\psi|$ is the probability of noise presence with no genuine signal which leads to negative $\tilde{s}_1[k]\psi[k]$. In theory, $\mathbf{Q}(I)$ can be negative which means the power of noise is more than signal. In practice, only relative result of $\mathbf{Q}(I)$ is informative, therefore the sign of $\mathbf{Q}(I)$ is important and a negative QI shows lower quality than a positive QI.

7.4 Experimental results

To evaluate the performance of the proposed SDQI method, seven state-of-the-art approaches (BRISQUE [42], CPBD [39], JNB [38], LPC [40], S3 [41], BIQI [37], and MetricQ [36]) have been compared objectively and subjectively. All analyses are performed on the gray-level image, however, there is no restriction for performing the algorithm on other channels.

7.4.1 Method parameters

We have run extensive simulations to set the algorithm's parameters: N_q SVD patch size, W_q shrinkage block size, and O_q shrinkage overlapping size, $\boldsymbol{c}_{\alpha}^{q}$ of (93), $\boldsymbol{T}_{\delta}^{q}$ of (89), ξ_{max} of (100), and $\boldsymbol{c}_{\beta}^{q}$ of (101). N_q should be small enough to contain a distinct orientation. Since the proposed algorithm detects one dominant orientation a large patch may contain many different orientations which cannot be accurately detected. However, for an accurate estimation of orientation, sufficient number of pixels are required and a very small patch does not satisfy this condition. Considering that the DFT operation is faster when N_q is power of 2, our experiments show that $N_q = 8$ is optimal for the performance. We set the shrinkage window size $W_q = 2N_q$ to process the image details more globally before analyzing the patch. We set $O_q = \frac{W_q}{2}$ and our experiments show that by increasing the overlapping size (e.g., $O_q = \frac{3W_q}{4}$) the performance slightly improves; however, it does not justify the computational complexity (e.g., 4x). We have analyzed the effect of Fourier shrinkage on the performance of the algorithm by altering $\boldsymbol{c}_{\alpha}^{q}$ to change the shrinkage strength. For this, we used the denoising methods BM3D [20] and bilateral filter [21] and we changed the noise removal force, i.e., input standard deviation of noise σ_n, and measured the output QI. Fig. 7.6 shows the QI of the proposed method with changing both σ_n and $\boldsymbol{c}_{\alpha}^{q}$. By increasing $\boldsymbol{c}_{\alpha}^{q}$, the ability of detecting noise increases by providing lower QI for noisy image ($\sigma_n = 0$); however, the capability of detecting blur decreases since the QI peak shifts towards higher σ_n. Thus, we set $\boldsymbol{c}_{\alpha}^{q} = 4$ as a balanced trade-off to detect both noise and blur. By conducting extensive simulations we determined other algorithm's parameters $\boldsymbol{T}_{\delta}^{q} = 0.75$, $\xi_{max} = 8$, and $\boldsymbol{c}_{\beta}^{q} = 20$ that give the highest correspondence with ground-truth quality metrics PSNR and MSSIM.

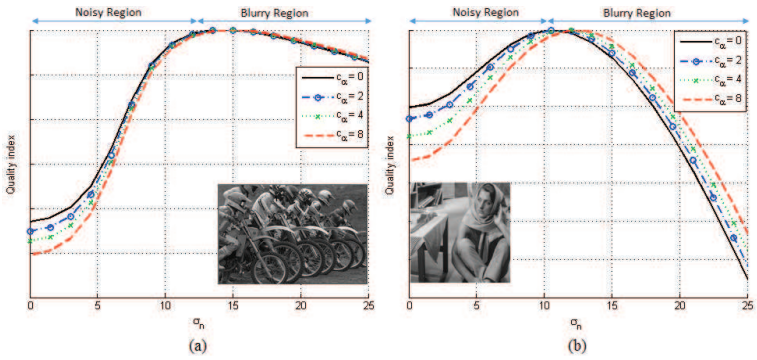

Figure 7.6: The effect of Fourier shrinkage on the performance of the proposed algorithm. The noisy images with added $\sigma_a = 10$ (28dB) are denoised by BM3D [20] (a) and (b) bilateral filter [21] using different input σ_n. Increasing $\boldsymbol{c}_\alpha^q$ increases the capability of detecting noise, but shifts the maximum QI to a blurrier result, i.e., higher σ_n. $\boldsymbol{c}_\alpha^q = 0$ means no shrinkage.

7.4.2 Optimizing denoising parameters using NR-IQA

In image or video denoising, the goal is to remove noise without degrading the sharpness (i.e., introducing blur). Thus, finding an optimal point between noise and blur is the key to achieve the highest quality. If the key denoising parameter $\mathbf{p}$ is not well selected, the output will be degraded with either noise or blur. A NR-IQA which detects the genuine image content, such as edges of physical objects, local sharpness, and textures, can be used to select such key parameter. Assuming I is the observed noisy image and a filtering process outputs the filtered image $I_{\mathbf{p}}$ using the input parameter $\mathbf{p}$. As proposed by [36] by changing $\mathbf{p}$ and measuring the output quality using a NR-IQA, the denoiser output can be optimized. In order to evaluate the performance of NR-IQA, we consider that the ground-truth quality metric such as PSNR or MSSIM is available and denoted by $\Phi(\cdot)$, which measures the quality of the NR-IQA based denoiser output. Assuming $\mathbf{QI}(I_{\mathbf{p}})$ is the quality index measured by a NR-IQA, the NR-IQA based denoiser output leads to the highest $\mathbf{QI}(I_{\mathbf{p}})$ as

$$I_{\mathbf{opt}} = \underset{I_{\mathbf{p}}}{\operatorname{argmax}} \left[\mathbf{QI}(I_{\mathbf{p}})\right], \quad \Phi_{gtm} = \Phi(I_{\mathbf{opt}}, I_{\mathbf{ref}}), \tag{103}$$

where $I_{\mathbf{opt}}$ is the output of NR-IQA based denoiser at highest QI, $I_{\mathbf{ref}}$ is the reference image, and Φ_{gtm} is its quality according to the reference and considered ground-truth. Due to imperfection of NR-IQA, the output quality may deviate from maximum achievable quality. Fig. 7.7 is an example of computing Φ_{gtm} for NR-IQA methods BIQI, MetricQ, and SDQI using *Peppers* contaminated

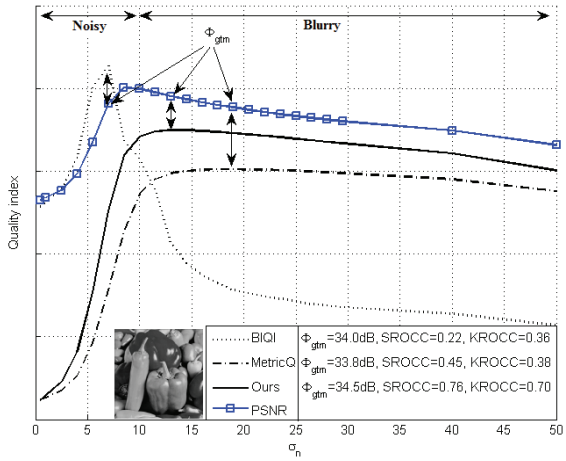

Figure 7.7: Evaluation of NR-IQA methods in selecting the denoiser BM3D parameter for *Peppers* corrupted with WGN 28dB ($\sigma_a = 10$) considering PSNR as the ground-truth quality metric.

with WGN with $\sigma_a = 10$ (PSNR = 28dB) and denoised using BM3D. Considering PSNR as the ground-truth metric, the maximum of $\mathbf{QI}(I_{\mathbf{p}})$ does not coincide perfectly with maximum quality (here, PSNR) for none of three methods. However, the methods that reach higher quality is more desirable to be used as a denoising parameter selector. Fig. 7.8 shows a block diagram for optimizing the parameters of denoising (NR-IQA based denoiser). We use Φ_{gtm} to evaluate the NR-IQA according to ground-truth metric (see Tables 7.3, 7.5 and 7.7).

In Fig. 7.8 only the maximum value of QI is taken into account, however, the behaviour of all QIs with respect to ground-truth should also be examined. We consider two well-known correlation factors, Spearman and Kendall rank order correlation coefficient (SROCC and KROCC) to evaluate the performance of NR-IQA. We change the denoiser parameter and compute QI of denoiser output. We then compute SROCC and KROCC using the set (one value for each parameter) of QI and the corresponding ground-truth. Since SROCC, KROCC, and Φ_{gtm} do not necessarily match (compare Table 7.2 and 7.3) we consider all of them for our evaluations. When the reference is available and the degradation is noise or blur, many studies show a high correspondence between subjective and objective measures such as PSNR and MSSIM (see [124]). Depending on the availability of the reference image, in our simulations we have used subjective metric mean opinion scores (MOS), and objective metrics, PSNR and MSSIM as the ground-truth.

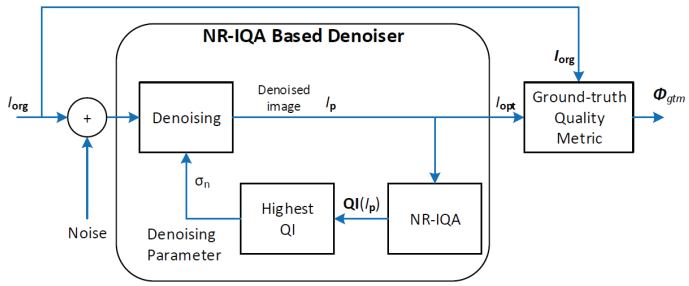

Figure 7.8: Optimizing the parameters of a denoiser using a NR-IQA method and finding the quality of output based on the ground-truth quality metric.

Table 7.1: Correlation factors between MOS and considered NR-IQA methods for denoising of real images.

	BRISQUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	MetricQ [36]	Ours
SROCC	0.36	0.27	0.30	0.59	0.30	0.50	0.66	**0.93**
KROCC	0.32	0.23	0.26	0.55	0.26	0.46	0.58	**0.87**

7.4.3 Real noise

To analyze the performance of the NR-IQA methods under real noise we have selected 25 real noisy images and video frames (see Fig. 7.9). We applied BM3D [20] on each image and created sets of denoised images with different levels of denoising. For video frames where the temporal data was available we used VBM3D [8]. We have conducted a human subject study and asked the participants in a pairwise comparison to vote for the image with the better visual quality. A total of 26 participants between the age of 20 and 55 participated in our experiments and we have obtained the MOS for all images and calculated the SROCC and KROCC for the data set. Table 7.1 shows the SROCC and KROCC result of each NR-QIA method. Our methods shows higher correlation with MOS. In CPBD, JNB, and S3 noise is less taken into account and for all cases the noisier images shows higher QI compared to denoised ones. BIQI and BRISQUE also tend to select noisier images over the denoised ones. One the other hand, MetricQ and LPC tend to select blurrier images with destructed details. Fig. 7.10 shows part of original and denoised images by different levels from our real noisy dataset. BIQI and BRISQUE select the noisiest (left column) as the highest quality. MetricQ selects the blurriest (third column) as the highest quality and ours selects the ones with highest MOS.

Figure 7.9: Samples of our image dataset corrupted with real noise.

7.4.4 Synthetic noise

We have evaluated the performance the NR-IQA methods by analyzing the behavior of each method in finding the best balance between noise and blur in image denoising under synthetic WGN and synthetic processed noise. We use both correlation factor (SROCC and KROCC) and Φ_{gtm} (quality at maximum QI) to objectively evaluate the performance of NR-IQA methods (see Fig. 7.8). We consider two well-known reference-based quality metrics PSNR and MSSIM [125] as the ground-truth quality metric. We have considered the TID2013 [124] database as the ground-truth image and added synthetic noise, then, we have varied the main parameter of the denoiser (here σ_n and σ_{RF3D}). We have computed the correlation between ground-truth quality metric and computed QI. We have considered two high-performance denoising methods BM3D and DDID [23] in these experiments. We also considered the quality of NR-IQA output in these experiments. The PSNR and MSSIM values at the maximum QI are measured and compared. The NR-IQA method that gives the highest Φ_{gtm}, is more suitable to be used in a NR-IQA based denoiser design.

Synthetic WGN

In this experiment we added zero-mean WGN to the ground-truth images from TID2013. The noisy images were generated by adding WGN with standard deviation σ_a of 10 (PSNR = 28dB). For all synthetic noise tests using TID2013, we consider the standard deviation of noise σ_n as the main parameter of the denoiser and we have varied that using 15 different levels of denoising from relatively small (which leads to noisy results) to large values (which leading to blurry results). Table 7.2 compares the SROCC and KROCC between different NR-IQA methods. In case of BM3D as

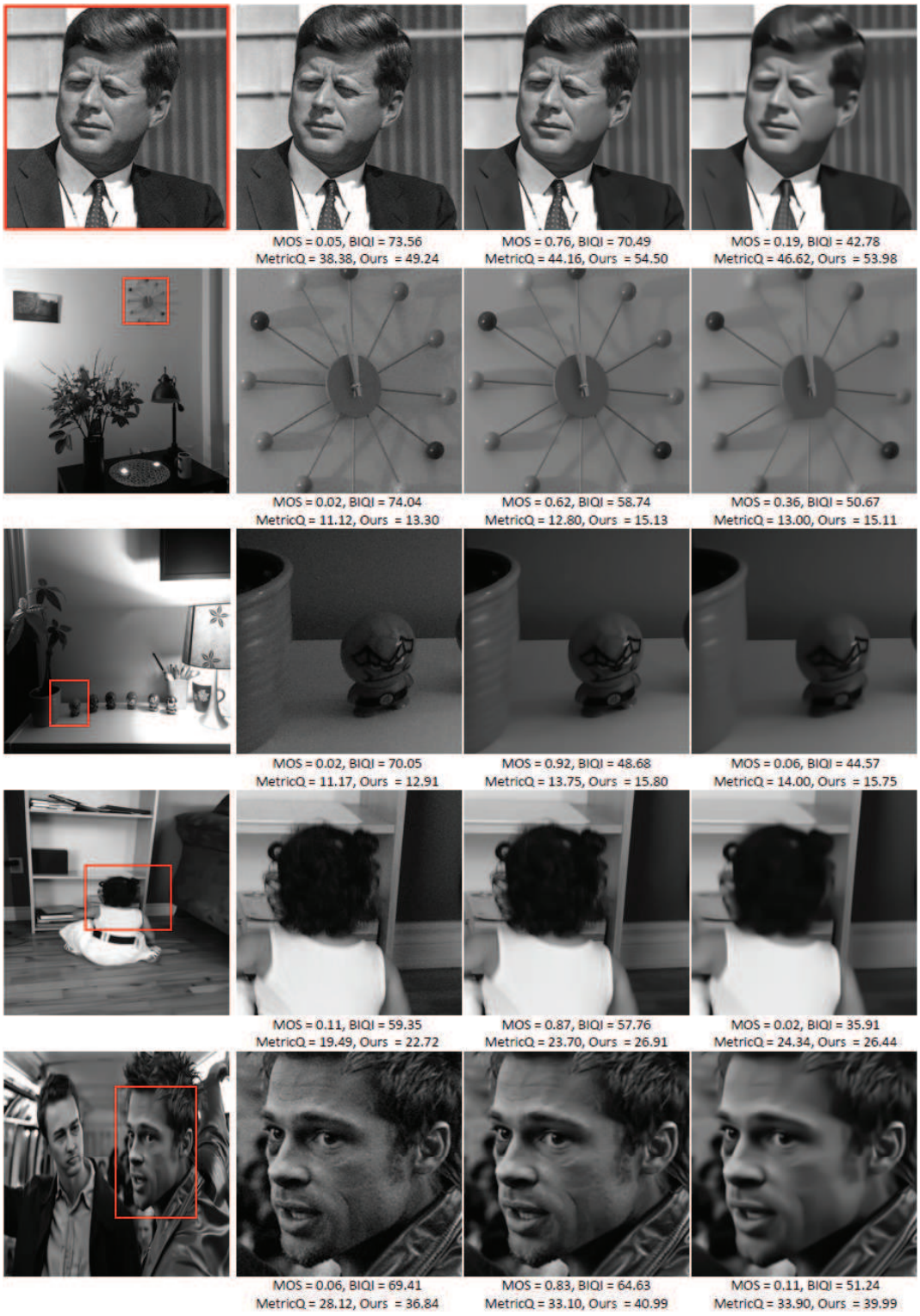

Figure 7.10: Original, part of original, and two denoised outputs using BM3D with two different levels. SDQI selects the output with the highest MOS.

Table 7.2: WGN: Correlation factor for TID2013 database.

	BM3D								DDID							
Correlation with PSNR																
	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours
SROCC	**0.78**	-0.28	-0.28	0.47	-0.28	0.58	0.50	0.57	0.75	-0.33	-0.33	0.42	-0.33	0.60	0.73	**0.76**
KROCC	**0.66**	-0.17	-0.18	0.38	-0.17	0.51	0.43	0.50	0.65	-0.22	-0.22	0.31	-0.23	0.51	0.65	**0.69**
Correlation with MSSIM																
	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours
SROCC	**0.73**	-0.35	-0.35	0.49	-0.35	0.51	0.54	0.61	0.72	-0.37	-0.37	0.42	-0.37	0.55	0.73	**0.76**
KROCC	**0.64**	-0.19	-0.20	0.39	-0.19	0.47	0.46	0.53	0.63	-0.23	-0.23	0.31	-0.24	0.49	0.65	**0.69**

Table 7.3: WGN: Quality of NR-IQA based denoiser, Φ_{gtm}, averaged over images of TID2013 database.

	BM3D								DDID							
	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours
PSNR	**33.47**	28.24	28.24	31.47	28.22	32.93	32.06	32.46	**33.35**	28.26	28.26	30.68	28.23	33.11	32.96	33.06
MSSIM	**0.90**	0.69	0.69	0.84	0.68	0.89	0.86	0.87	**0.90**	0.69	0.69	0.81	0.68	0.89	0.88	0.88

the denoiser, BRISQUE clearly outperforms other methods followed by BIQI and proposed SDQI. In case of DDID as the denoiser, proposed method outperforms other methods followed by MetricQ and BRISQUE. CPBD, JNB, and S3 are more sensitive to blur and less to noise which yield negative correlations. Table 7.3 compares the Φ_{gtm} for NR-IQA methods averaged over TID2013 database. BRISQUE achieves slightly higher PSNR and MSSIM as proposed method. The performance of BRISQUE and BIQI is relatively higher when the noise is white, however, according to Table 7.1 it degrades when the noise is non-white. It is worth noting that we have also tested the performance of NR-IQA methods under signal-dependent noise. In this case the variance of noise is a function of image intensity. We selected a "close to reality" noise level function, i.e., variance of noise at each intensity. We computed the average Φ_{gtm} and the results show QI values relatively similar to Table 7.3 for all eight methods.

We have considered the case that the adjustable parameter of the denoiser is not the standard deviation of the noise. Fig. 7.11 shows the PSNR result of NR-IQA based denoiser using RF3D [61] as the video denoiser. Input parameter of RF3D is the power spectral density (PSD) which is defined by a 2D Gaussian lowpass filter with different sigma σ_{RF3D}. Proposed method can better, i.e., leads to outputs with the higher PSNR, select the parameter of RF3D σ_{RF3D} compared to MetricQ.

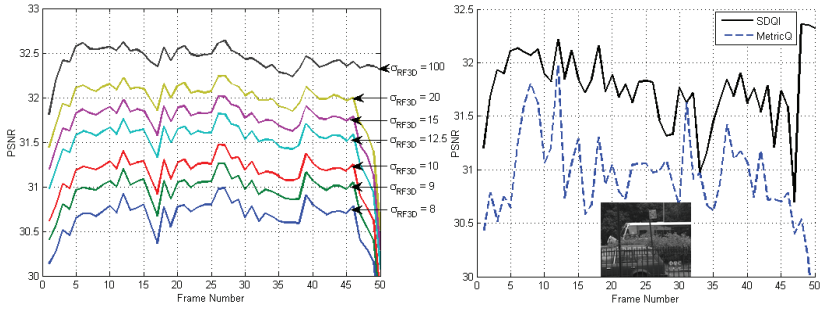

Figure 7.11: Selecting the parameter of *RF3D* [61] video denoiser. (a) PSNR with different parameters for the 25dB *Bus* video. (b) PSNR of denoiser output when σ_{RF3D} is selected using MetricQ and proposed method. Since noise is white, higher σ_{RF3D} (flat PSD) leads to higher quality.

Spatially correlated noise

Camera noise usually becomes manipulated due to processing such as filtering, lossy compression, or demosaicing. Thus, in order to evaluate our method under this real conditions, we assume that the noise is spatially correlated (similar to noise after demosaicing, upscaling, or filtering) and we generated noisy images by adding filtered WGN to the ground-truth images from TID2013. Noisy images were denoised using BM3D and DDID with 15 levels of denoising. We used 5×5 Gaussian filter with sigma of 0.6 and $\sigma_a = 20$. Table 7.4 compares the SROCC and KROCC between selected NR-IQA methods using PSNR and MSSIM as the ground-truth. For both BM3D and DDID, proposed SDQI outperforms other methods followed by MetricQ and LPC. Table 7.4 results corresponds with real noise results in Table 7.1. The performance of BRISQUE and BIQI degrades as the noise deviates from whiteness and in some cases yield negative correlations. Similar to WGN, CPBD, JNB, and S3 give negative correlations. Table 7.5 compares the average of Φ_{gtm} for considered NR-IQA. Our method achieves more accurate results followed by BIQI in case BM3D denoiser and MetricQ in case DDID denoiser. Comparing Table 7.4 and Table 7.2 gives an idea about the sensitivity of methods to the high-frequency components of the noise. Performance of BIQI and BRISQUE significantly decreases in this situation while SDQI shows stable performance. Fig. 7.12 shows visual quality comparison, applying BM3D where the filter parameter σ_n is selected using BIQI, MetricQ, LPC and proposed. BIQI leads to noisy results, however, MetricQ and LPC yield blurry results which correspond to the results in Table 7.1.

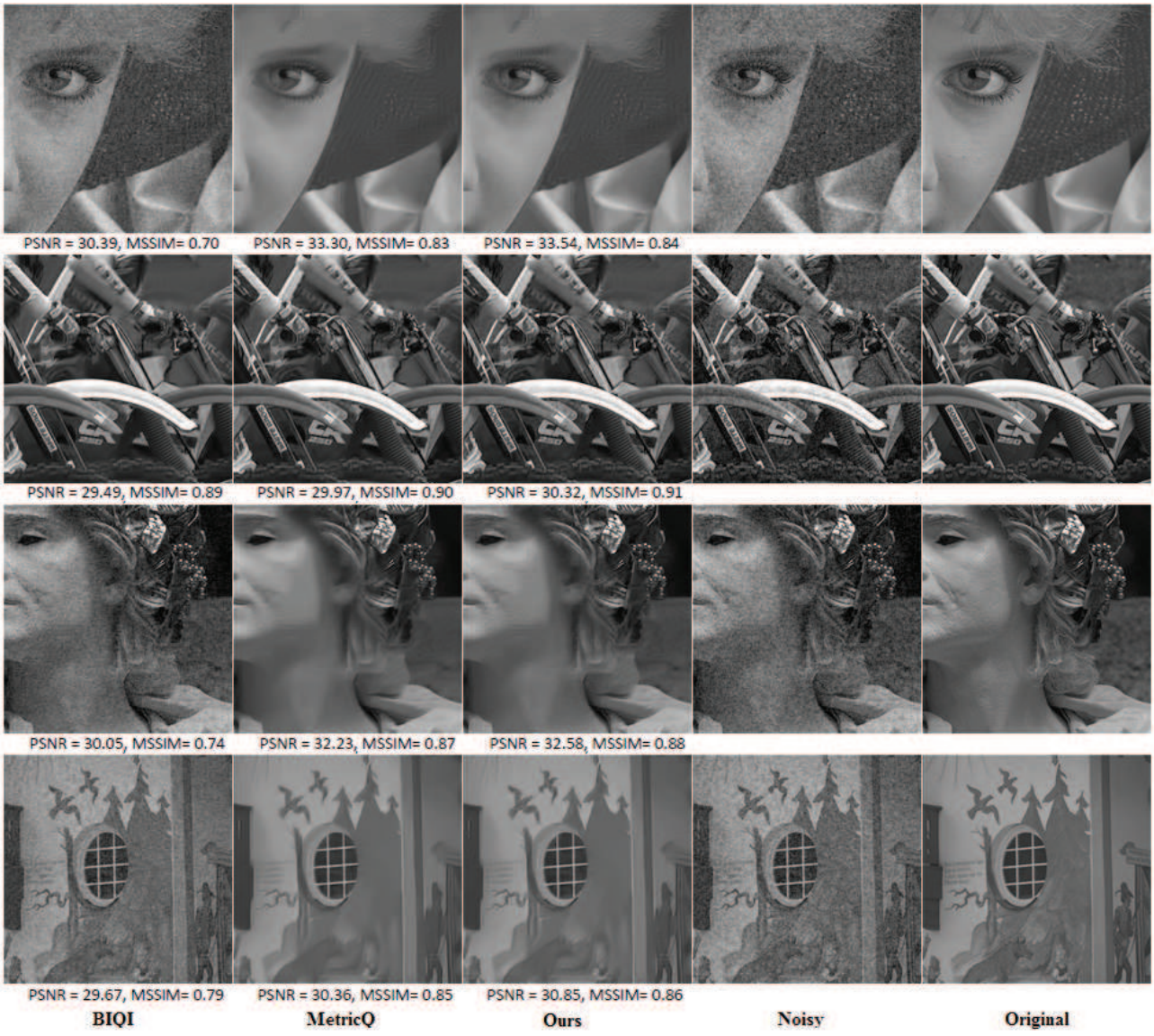

Figure 7.12: Visual comparison: selecting BM3D parameter for denoising images from TID2013 corrupted with spatially correlated noise.

Table 7.4: Spatially correlated noise: Correlation factor for TID2013 database.

	BM3D								DDID							
Correlation with PSNR																
	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours
SROCC	-0.01	-0.24	-0.26	0.55	-0.27	0.49	0.53	**0.63**	-0.25	-0.29	-0.33	0.51	-0.34	0.43	0.73	**0.76**
KROCC	0.10	-0.20	-0.21	0.48	-0.24	0.41	0.47	**0.57**	-0.23	-0.26	-0.27	0.45	-0.31	0.38	0.66	**0.70**
Correlation with MSSIM																
	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours
SROCC	-0.15	-0.37	-0.40	0.52	-0.41	0.35	0.63	**0.70**	-0.32	-0.39	-0.42	0.48	-0.43	0.33	0.73	**0.75**
KROCC	0.01	-0.29	-0.31	0.46	-0.33	0.33	0.56	**0.65**	-0.28	-0.32	-0.34	0.42	-0.37	0.32	**0.67**	**0.67**

Table 7.5: Spatially correlated noise: Quality of NR-IQA based denoiser, Φ_{gtm}, averaged over images of TID2013 database.

	BM3D								DDID							
	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours
PSNR	29.89	28.60	28.41	30.69	28.21	31.16	30.77	**31.23**	28.88	28.63	28.40	29.95	28.21	30.99	31.57	**31.65**
MSSIM	0.78	0.72	0.71	0.83	0.70	0.84	0.84	**0.85**	0.74	0.72	0.71	0.80	0.70	0.83	**0.86**	**0.86**

Lossy compressed noise

Images are often lossy compressed. Thus, we repeated the above experiments by applying a lossy compression on the noisy images. Noisy images were generated by adding WGN with $\sigma_a = 10$ to TID2013 database, then we compressed them using standard JPEG with quality factor (QF) of 75, finally we denoised them using BM3D and DDID with 15 levels of denoising. Table 7.6 compares the SROCC and KROCC between selected NR-IQA methods using PSNR and MSSIM as the ground-truth. For both BM3D and DDID, proposed SDQI outperforms other methods followed by MetricQ and LPC. Similar to spatially correlated noise, the performance of BRISQUE and BIQI degrades as the noise becomes lossy compressed and CPBD, JNB, and S3 give negative correlations. Table 7.5 compares the average of Φ_{gtm} for considered NR-IQA methods using PSNR and MSSIM as the ground-truth. The proposed SDQI is able to select a more accurate σ_n compared to other methods, suggesting it being more suitable for denoising applications.

7.4.5 General quality assessment

We have tested our algorithm, independent of denoising in general degradation conditions. We have added different types of distortion to TID2013 database. We have examined WGN with two levels ($\sigma_a = 5, 10$), spatially correlated noise (SCN) with two levels ($\sigma_a = 10, 20$), WGN that is lossy

Table 7.6: Lossy compressed noise: Correlation factor for TID2013 database.

	BM3D								DDID							
Correlation with PSNR																
	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours
SROCC	0.30	-0.39	-0.39	0.50	-0.39	0.33	0.58	**0.65**	0.32	-0.44	-0.44	0.38	-0.44	0.34	0.79	**0.82**
KROCC	0.30	-0.27	-0.28	0.42	-0.28	0.31	0.49	**0.57**	0.26	-0.34	-0.33	0.31	-0.34	0.32	0.71	**0.75**
Correlation with MSSIM																
	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours
SROCC	0.27	-0.41	-0.42	0.49	-0.41	0.29	0.59	**0.64**	0.32	-0.44	-0.44	0.37	-0.44	0.32	0.76	**0.78**
KROCC	0.30	-0.27	-0.27	0.40	-0.27	0.30	0.49	**0.56**	0.27	-0.31	-0.30	0.29	-0.31	0.32	0.68	**0.71**

Table 7.7: Lossy compressed noise: Quality of NR-IQA based denoiser, Φ_{gtm}, averaged over images of TID2013 database.

	BM3D								DDID							
	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours
PSNR	31.64	27.91	27.92	31.01	27.90	31.70	31.66	**31.92**	31.73	27.91	27.95	30.23	27.90	31.85	32.39	**32.42**
MSSIM	**0.86**	0.69	0.69	0.83	0.69	**0.86**	0.85	**0.86**	0.86	0.69	0.69	0.79	0.69	0.86	**0.88**	**0.88**

compressed (i.e., WGN + JPEG), SCN that is lossy compressed (i.e., SCN + JPEG), Gaussian blur with sigma of 1, and impulse noise with occurrence probability of 0.5%. Table 7.8 compares the SROCC considering PSNR as the ground-truth and shows the average MSE. Note that KROCC results are similar to SROCC results. The proposed method is the most successful in all distortion types.

Table 7.8: SROCC values for NR-IQA distortions added to TID2013 database using the PSNR as the ground-truth.

	BRIS-QUE [42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q [36]	Ours
WGN_5	**1.00**	-1.00	-1.00	0.58	-1.00	0.67	**1.00**	**1.00**
WGN_{10}	**1.00**	-1.00	-1.00	0.58	-1.00	**1.00**	**1.00**	**1.00**
SCN_{10}	-0.33	-0.75	-0.67	0.75	-0.92	0.42	**1.00**	**1.00**
SCN_{20}	-0.50	-0.83	-0.92	0.83	-1.00	0.92	**1.00**	**1.00**
WGN_7+JPEG	0.75	0.08	0.33	**0.92**	-0.83	**0.92**	-0.25	**0.92**
SCN_5+JPEG	0.67	0.17	0.50	0.83	-0.92	0.50	-0.17	**0.92**
Gaussian blur	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Impulse noise	**1.00**	-0.83	0.33	0.75	-1.00	0.08	0.25	**1.00**

Table 7.9: Elapsed time in seconds to process a full HD image.

BRIS-QUE[42]	CPBD [39]	JNB [38]	LPC [40]	S3 [41]	BIQI [37]	Metric-Q[36]	Ours	Ours GPU
0.67	4.47	7.29	6.94	188.78	1.03	9.31	0.75	0.018

7.4.6 Implementation issues

The source codes of algorithms [36–42] were obtained from the author's websites. In BIQI [37] and BRISQUE [42] the maximum and minimum quality happens at 0 and 100 respectively. To obtain the consistency with other methods which give higher QI for higher quality, we subtracted the QI of these methods from 100. We implemented our method using *MATLAB* (MEX). We have tested the acceleration factor of proposed SVD computation approach by comparing it to standard MATLAB *'svd'* function. For the patch size of $N_q = 32$ for instance, it shows a speedup by a factor of 1.9. Our approach saves, for example, 0.2 seconds for calculating the SVD of a full HD image with resolution of 1920×1080 with patch size of 8. We took the advantage of block-based operation to utilize the ability of parallel processing and accelerated our implementation using capability of GPU in parallel processing. We used OpenCL programming language to implement our method. Table 7.9 compared the speed of different methods to compute the QI of full HD image (1920×1080). For all tests we used Intel 3.07 GHz, i7 CPU and NVIDIA GTX 970 GPU. Table 7.9 cannot fully reflect the speed of algorithms since the implementations of the different algorithms may not be optimized. However, computation time gives an idea about the lower bound of the speed.

7.5 Conclusion

In this thesis, we presented a new no-reference image quality assessment approach based on single value decomposition and Fourier transform which are used to estimate the dominant orientation and coefficient sparsity. We propose a fast and easy to implement method to calculate the SVD avoiding recursive operations. Based on SVD analysis and Fourier sparsity, we measure local image structure, noise, and blur, and integrate them to compute overall quality. Our method takes both noisy and structured patches into account, therefore it measures the effect of noise more precisely compared to state-of-the-art methods. To reach more accurate results, especially under heavy noise, we use

a Fourier shrinkage to increase the contrast of image structure before analyzing the patches. We have performed ample simulation using synthetic and real data to validate the performance of the proposed method. We used white Gaussian and processed noise in our simulations to support our claims. The proposed approach is fast and able to provide a more reliable estimation of image quality under different degradations compared to state-of-the-art NR-IQA approaches.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

Noise is present in video signal captured from different sources. Even modern high-quality capturing devices introduce noise of different type. Noise estimation, reduction, and quality assessment methods typically assume video noise is white Gaussian. This thesis bridges the gap between the relatively well studied white Gaussian noise and the more complicated white signal-dependent and non-white processed types. This thesis comprises novel approaches to estimate and reduce noise of different sources and provide a solution to assess the image quality without accessing the reference frame.

We proposed a noise estimation technique that widens noise assumptions based on the classification of intensities (or color) and on the extraction of weights using statistical noise property and homogeneous regions in the images. The classification of intensities into connected clusters of homogeneous patches allowed us to well approximate the noise level function. We estimated the degree of processed versus white noise as a ratio of low to high frequency energies in the input image. Another important feature of our technique is its use of both the input noisy image and its downscaled version. This allowed better differentiation of noise and structure and fast processing. We have shown that the developed visual noise estimation technique robustly handles different type of visual noise: white Gaussian, white Poissonian-Gaussian, and processed (non-white) that are visible in real-world video signals. Our simulation results showed the superiority of the proposed

technique both in accuracy and speed. For the real-world experiment, simulation results have been tested for very challenging sequences. Simulation results in this thesis are given for the gray-level format of test video sequences. However, we have tested our method on color sequences and it also outperforms related work. The main strength point of proposed technique compared to rival methods is utilizing the connectivity of patches and temporal data.

Recent WGN filters are powerful. In order for them to remove real noise, we proposed a system which enables a WGN filter to handle real noise by addressing the signal-dependency and spatial correlation using noise equalization in intensity and frequency domain. Our simulation results show that under real noise the quality of WGN filters significantly improves when they are used in our framework.

We have also presented a time-space video denoising method that employs a fast block-matching motion-estimation method and corrects the results employing homography creation. In order to address blocking artifacts, we propose an interpolation of block-level error, *back-signal* subtraction, and two-band motion compensation. We combines two levels of temporal error detection and adjusts the noise level when it is overestimated which leads to less motion blur effects compared to relevant methods. In addition, our modular framework provides the feature to adjust the performance-speed point by changing certain components and parameters, such as motion-estimation, temporal radius, and spatial filter. We have benefited from the parallelelizable structure of our method and we accelerated that using GPU. Our solution is fast yet yields to competitive results compared to the state-of-the-art methods. We show that the proposed system in the recursive framework is also efficient and rivals relevant methods.

Our final contribution in this thesis was a new no-reference image quality assessment approach based on single value decomposition and Fourier transform which are used to estimate the dominant orientation and coefficient sparsity. We proposed a fast and easy to implement method to calculate the SVD avoiding recursive operations. Based on SVD analysis and Fourier sparsity, we measure local image structure, noise, and blur, and integrate them to compute overall quality. Our method takes both noisy and structured patches into account, therefore it measures the effect of noise more precisely compared to state-of-the-art methods. To reach more accurate results, especially under heavy noise, we use a Fourier shrinkage to increase the contrast of image structure before analyzing

the patches. We have performed ample simulation using synthetic and real data to validate the performance of the proposed method. We used white Gaussian and processed noise in our simulations to support our claims. The proposed approach is fast and able to provide a more reliable estimation of image quality under different degradations compared to state-of-the-art NR-IQA approaches.

8.2 Future work

Noise estimation method uses an outlier removal based on one reference patch which is selected based on variances. Two patches may have similar variances but different means. Since the computations can be accomplished in different precisions (e.g., single or double floating point) the reference block may become different with different mean which affects the selected cluster. For future work we propose using a more efficient outlier removal to make the cluster selection more stable.

Our proposed noise reduction compromised the quality in chroma channels to gain speed. In special cases that chroma channels have most of information the quality is not satisfactory. We propose to have an option to handle such rare cases to perform all processing of luminance channel on the chroma as well. Our homography based motion correction can be improved by adding the camera motion model to homography frame work. Algorithms such as RANSAC [126] based on selected features can compute homography efficiently utilizing singular value of motion vector matrix. Although these methods are computationally complex, they increase the quality specially when the motion model is not complex (burst images). Another idea for future work can be utilizing a method to sharpen straight lines. The problem with classic sharpening using highpass filters is that it increases noise. However, we propose to sharpen the image for straight lines only which can be detected through our proposed algorithm for dominant orientation detection in our NR-IQA method. For future work, we also propose an algorithm to combine recursive and symmetric temporal filter ideas to benefit from advantage of each. When the temporal information of previous frames is not adequate (e.g., scene change or beginning of sequence) we can use forward noisy frame otherwise we use backward denoised frames.

Our proposed NR-IQA may be suboptimal when there is not enough patches with dominant

directions and random shaped structure image exist. One solution can be using homogeneity information to guide the NR-IQA. In this case, the level of noise does not have to be accurate since proposed NR-IQA can work independent of noise level estimation. We also propose using temporal and homogeneity data to guide the proposed NR-IQA. Temporal data can guide the NR-IQA to better differentiate between noise and image structure.

Bibliography

[1] R. Szeliski, *Computer vision: algorithms and applications*. Springer, 2010.

[2] Y. Tsin, V. Ramesh, and T. Kanade, "Statistical calibration of CCD imaging process," in *Computer Vision ICCV, IEEE Int. Conf. on*, vol. 1, pp. 480–487, IEEE, 2001.

[3] G. Healey and R. Kondepudy, "Radiometric CCD camera calibration and noise estimation," *Pattern Analysis and Machine Intelligence, IEEE Trans. on*, vol. 16, pp. 267–276, Mar 1994.

[4] A. Bovik, *Handbook of Image and Video Processing*. Communications, Networking and Multimedia, Elsevier Science, 2010.

[5] A. Bovik, *The Essential Guide to Image Processing*. Elsevier Science, 2009.

[6] C. Liu, R. Szeliski, S. Kang, C. Zitnick, and W. Freeman, "Automatic estimation and removal of noise from a single image," *Pattern Analysis and Machine Intelligence, IEEE Trans. on*, vol. 30, no. 2, pp. 299–314, 2008.

[7] E. Dubois and S. Sabri, "Noise reduction in image sequences using motion-compensated temporal filtering," *Communications, IEEE Transactions on*, vol. 32, pp. 826–831, Jul 1984.

[8] K. Dabov, A. Foi, and K. Egiazarian, "Video denoising by sparse 3d transform-domain collaborative filtering," in *Proc. 15th European Signal Processing Conf.*, vol. 1, p. 7, 2007.

[9] S. Yu, M. Ahmad, and M. Swamy, "Video denoising using motion compensated 3-D wavelet transform with integrated recursive temporal filtering," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 20, pp. 780–791, June 2010.

[10] J. Yang, Y. Wang, W. Xu, and Q. Dai, "Image and video denoising using adaptive dual-tree discrete wavelet packets," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 19, no. 5, pp. 642–655, 2009.

[11] E. Balster, Y. Zheng, and R. Ewing, "Combined spatial and temporal domain wavelet shrinkage algorithm for video denoising," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 16, no. 2, pp. 220–230, 2006.

[12] G. Varghese and Z. Wang, "Video denoising based on a spatiotemporal gaussian scale mixture model," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 20, pp. 1032–1040, July 2010.

[13] V. Zlokolica, A. Pizurica, and W. Philips, "Wavelet-domain video denoising based on reliability measures," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 16, pp. 993–1007, Aug 2006.

[14] F. Jin, P. Fieguth, and L. Winger, "Wavelet video denoising with regularized multiresolution motion estimation," *EURASIP Journal on Advances in Signal Processing*, vol. 2006, 2006.

[15] M. Maggioni, G. Boracchi, A. Foi, and K. Egiazarian, "Video denoising, deblocking, and enhancement through separable 4-D nonlocal spatiotemporal transforms," *Image Processing, IEEE Trans. on*, vol. 21, pp. 3952–3966, Sept 2012.

[16] F. Luisier, T. Blu, and M. Unser, "Sure-let for orthonormal wavelet-domain video denoising," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 20, pp. 913–919, June 2010.

[17] L. Guo, O. Au, M. Ma, and Z. Liang, "Temporal video denoising based on multihypothesis motion compensation," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 17, pp. 1423–1429, Oct 2007.

[18] J. Dai, O. Au, C. Pang, and F. Zou, "Color video denoising based on combined interframe and intercolor prediction," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 23, pp. 128–141, Jan 2013.

[19] S. Rahman, M. Ahmad, and M. Swamy, "Video denoising based on inter-frame statistical modeling of wavelet coefficients," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 17, pp. 187–198, Feb 2007.

[20] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *Image Processing, IEEE Trans. on*, vol. 16, no. 8, pp. 2080–2095, 2007.

[21] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Computer Vision, Int. Conf. on*, pp. 839–846, Jan 1998.

[22] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *Pattern Analysis and Machine Intelligence, IEEE Trans. on*, vol. 12, no. 7, pp. 629–639, 1990.

[23] C. Knaus and M. Zwicker, "Progressive image denoising," *Image Processing, IEEE Trans. on*, vol. 23, pp. 3114–3125, July 2014.

[24] A. Amer and E. Dubois, "Fast and reliable structure-oriented video noise estimation," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 15, no. 1, pp. 113–118, 2005.

[25] T.-A. Nguyen and M.-C. Hong, "Filtering-based noise estimation for denoising the image degraded by Gaussian noise," in *Advances in Image and Video Technology*, pp. 157–167, Springer, 2012.

[26] D.-H. Shin, R.-H. Park, S. Yang, and J.-H. Jung, "Block-based noise estimation using adaptive Gaussian filtering," *Consumer Electronics, IEEE Trans. on*, vol. 51, no. 1, pp. 218–226, 2005.

[27] D. Donoho, , and J. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, no. 3, pp. 425–455, 1994.

[28] M. Hashemi and S. Beheshti, "Adaptive noise variance estimation in Bayes-Shrink," *Signal Processing Letters, IEEE*, vol. 17, no. 1, pp. 12–15, 2010.

[29] S.-C. Tai and S.-M. Yang, "A fast method for image noise estimation using Laplacian operator and adaptive edge detection," in *Communications, Control and Signal Processing ISCCSP, Int. Symposium on*, pp. 1077–1081, 2008.

[30] P. Fu, Q. Sun, Z. Ji, and Q. Chen, "A new method for noise estimation in single-band remote sensing images," in *Fuzzy Systems and Knowledge Discovery, Int. Conf. on*, pp. 1664–1668, May 2012.

[31] M. Ghazal and A. Amer, "Homogeneity localization using particle filters with application to noise estimation," *Image Processing, IEEE Trans. on*, vol. 20, no. 7, pp. 1788–1796, 2011.

[32] J. Tian and L. Chen, "Image noise estimation using a variation-adaptive evolutionary approach," *Signal Processing Letters, IEEE*, vol. 19, no. 7, pp. 395–398, 2012.

[33] S.-M. Yang and S.-C. Tai, "Fast and reliable image-noise estimation using a hybrid approach," *Journal of Electronic Imaging*, vol. 19, no. 3, pp. 033007–033007, 2010.

[34] S. Pyatykh, J. Hesser, and L. Zheng, "Image noise level estimation by principal component analysis," *Image Processing, IEEE Trans. on*, vol. 22, no. 2, pp. 687–699, 2013.

[35] X. Liu, M. Tanaka, and M. Okutomi, "Noise level estimation using weak textured patches of a single noisy image," in *Image Processing (ICIP), IEEE Int. Conf. on*, pp. 665–668, 2012.

[36] X. Zhu and P. Milanfar, "Automatic parameter selection for denoising algorithms using a no-reference measure of image content," *Image Processing, IEEE Trans. on*, vol. 19, no. 12, pp. 3116–3132, 2010.

[37] A. Moorthy and A. Bovik, "A two-step framework for constructing blind image quality indices," *Signal Processing Letters, IEEE*, vol. 17, pp. 513–516, May 2010.

[38] R. Ferzli and L. Karam, "A no-reference objective image sharpness metric based on the notion of just noticeable blur (JNB)," *Image Processing, IEEE Trans. on*, vol. 18, pp. 717–728, Apr. 2009.

[39] N. Narvekar and L. Karam, "A no-reference image blur metric based on the cumulative probability of blur detection (CPBD)," *Image Processing, IEEE Trans. on*, vol. 20, pp. 2678–2683, Sept. 2011.

[40] R. Hassen, Z. Wang, and M. Salama, "Image sharpness assessment based on local phase coherence," *Image Processing, IEEE Trans. on*, vol. 22, pp. 2798–2810, July 2013.

[41] C. Vu, T. Phan, and D. Chandler, "S3: A spectral and spatial measure of local perceived sharpness in natural images," *Image Processing, IEEE Trans. on*, vol. 21, pp. 934–945, March 2012.

[42] A. Mittal, A. Moorthy, and A. Bovik, "No-reference image quality assessment in the spatial domain," *Image Processing, IEEE Trans. on*, vol. 21, pp. 4695–4708, Dec. 2012.

[43] M. Rakhshanfar and M. Amer, "Homogeneity classification for signal-dependent noise estimation in images," in *Image Processing (ICIP), IEEE Int. Conf. on*, pp. 4271–4275, Oct 2014.

[44] M. Rakhshanfar and M. Amer, "Motion blur resistant method for temporal video denoising," in *Image Processing (ICIP), IEEE Int. Conf. on*, pp. 2694–2698, Oct 2014.

[45] M. Rakhshanfar and M. Amer, "No-reference image quality assessment for removal of processed and unprocessed noise," in *Image Processing (ICIP), IEEE Int. Conf. on*, pp. 2179–2183, Sept 2015.

[46] M. Rakhshanfar and M. Amer, "No-reference image quality assessment system and method," *Patent office in Gatineau, Quebec, Canada, Reference no. 62/209,117, Patent Status: In Progress, Filed on*, 2015-08-24.

[47] M. Rakhshanfar and M. Amer, "System and method for the estimation of different types of noise in image and video signals," *Patent office in United States, Reference no. 61/993,469, Patent Status: In Progress, Filed on*, 2014-05-15.

[48] M. Rakhshanfar and M. Amer, "Time-space method and system for the reduction of video noise," *Patent office in United States, Reference no. 61/993,884, Patent Status: In Progress, Filed on*, 2013-05-15.

[49] H. Khalil, R. Rahmat, and W. Mahmoud, "Chapter 15: Estimation of noise in gray-scale and colored images using median absolute deviation (MAD)," in *Geometric Modeling and Imaging, 3rd Int. Conf. on*, pp. 92–97, July 2008.

[50] D. Zoran and Y. Weiss, "Scale invariance and noise in natural images," in *Computer Vision, IEEE 12th Int. Conf. on*, pp. 2209–2216, Sept 2009.

[51] A. Danielyan and A. Foi, "Noise variance estimation in nonlocal transform domain," in *Local and Non-Local Approximation in Image Processing LNLA, Int. Workshop on*, pp. 41–45, IEEE, 2009.

[52] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian, "Practical Poissonian-Gaussian noise modeling and fitting for single-image raw-data," *Image Processing, IEEE Trans. on*, vol. 17, no. 10, pp. 1737–1754, 2008.

[53] A. Foi, "Practical denoising of clipped or overexposed noisy images," in *EUSIPCO, 16th European Signal Processing Conf.*, pp. 1–5, 2008.

[54] A. Jezierska, C. Chaux, J.-C. Pesquet, H. Talbot, and G. Engler, "An EM approach for time-variant Poisson-Gaussian model parameter estimation," *Signal Processing, IEEE Trans. on*, vol. 62, pp. 17–30, Jan 2014.

[55] J. Yang, Z. Wu, and C. Hou, "Estimation of signal-dependent sensor noise via sparse representation of noise level functions," in *Image Processing (ICIP), 19th IEEE Int. Conf. on*, pp. 673–676, Sept 2012.

[56] X. Jin, Z. Xu, and K. Hirakawa, "Noise parameter estimation for Poisson corrupted images using variance stabilization transforms," *Image Processing, IEEE Trans. on*, vol. 23, no. 3, pp. 1329–1339, 2014.

[57] A. Kokaram, D. Kelly, H. Denman, and A. Crawford, "Measuring noise correlation for improved video denoising," in *Image Processing (ICIP), 19th IEEE Int. Conf. on*, pp. 1201–1204, Sept 2012.

[58] M. Colom, M. Lebrun, A. Buades, and J. Morel, "A non-parametric approach for the estimation of intensity-frequency dependent noise," in *Image Processing (ICIP), 21th IEEE Int. Conf. on*, Oct 2014.

[59] H. Tan, F. Tian, Y. Qiu, S. Wang, and J. Zhang, "Multihypothesis recursive video denoising based on separation of motion state," *Image Processing, IET*, vol. 4, pp. 261–268, August 2010.

[60] M. Protter and M. Elad, "Image sequence denoising via sparse and redundant representations," *Image Processing, IEEE Trans. on*, vol. 18, pp. 27–35, Jan 2009.

[61] M. Maggioni, E. Sanchez-Monge, and A. Foi, "Joint removal of random and fixed-pattern noise through spatiotemporal video filtering," *Image Processing, IEEE Trans. on*, vol. 23, pp. 4282–4296, Oct 2014.

[62] D. Zhang, J. Han, J. Kim, and S. Ko, "A gradient saliency based spatio-temporal video noise reduction method for digital tv," *Consumer Electronics, IEEE Trans. on*, vol. 57, pp. 1288–1294, August 2011.

[63] B. Song and K. Chun, "Motion-compensated temporal prefiltering for noise reduction in a video encoder," in *Image Processing, Int. Conf. on*, vol. 2, pp. 1221–1224 Vol.2, Oct 2004.

[64] L. Yan and Q. Yanfeng, "An adaptive temporal filter based on motion compensation for video noise reduction," in *Communication Technology, Int. Conf. on*, pp. 1–4, Nov 2006.

[65] S. Yang and T. Lu, "A practical design flow of noise reduction algorithm for video post processing," *Consumer Electronics, IEEE Trans. on*, vol. 53, pp. 995–1002, Aug 2007.

[66] T. Portz, L. Zhang, and H. Jiang, "High-quality video denoising for motion-based exposure control," in *Computer Vision Workshops (ICCV Workshops), IEEE Int. Conf. on*, pp. 9–16, Nov 2011.

[67] C. Liu and W. Freeman, "A high-quality video denoising algorithm based on reliable motion estimation," in *Computer Vision–ECCV 2010*, pp. 706–719, Springer, 2010.

[68] Z. Liu, L. Yuan, X. Tang, M. Uyttendaele, and J. Sun, "Fast burst images denoising," *ACM Trans. on Graphics (TOG)*, vol. 33, no. 6, p. 232, 2014.

[69] J. Dai, O. Au, W., C. Pang, F. Zou, and X. Wen, "Color video denoising based on adaptive color space conversion," in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE Int. Symposium on*, pp. 2992–2995, May 2010.

[70] S. Reeja and N. Kavya, "Real time video denoising," in *Engineering Education: Innovative Practices and Future Trends (AICERA), IEEE Int. Conf. on*, pp. 1–5, 2012.

[71] M. Black, G. Sapiro, D. Marimont, and D. Heeger, "Robust anisotropic diffusion," *Image Processing, IEEE Trans. on*, vol. 7, no. 3, pp. 421–432, 1998.

[72] D. Tschumperlé, "Fast anisotropic smoothing of multi-valued images using curvature-preserving PDE's," *Int. Journal of Computer Vision*, vol. 68, no. 1, pp. 65–82, 2006.

[73] D. Barash, "Fundamental relationship between bilateral filtering, adaptive smoothing, and the nonlinear diffusion equation," *Pattern Analysis and Machine Intelligence, IEEE Trans. on*, vol. 24, no. 6, pp. 844–847, 2002.

[74] S. Paris and F. Durand, "A fast approximation of the bilateral filter using a signal processing approach," in *Computer Vision Europe Conf. on*, pp. 568–580, Springer, 2006.

[75] E. Bennett and L. McMillan, "Video enhancement using per-pixel virtual exposures," in *ACM Trans. on Graphics (TOG)*, vol. 24, pp. 845–852, ACM, 2005.

[76] S. Awate and R. Whitaker, "Higher-order image statistics for unsupervised, information-theoretic, adaptive, image filtering," in *Computer Vision and Pattern Recognition, IEEE Computer Society Conf. on*, vol. 2, pp. 44–51, 2005.

[77] M. Elad and M. Aharon, "Image denoising via learned dictionaries and sparse representation," in *Computer Vision and Pattern Recognition, IEEE Computer Society Conf. on*, vol. 1, pp. 895–900, 2006.

[78] S. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *Pattern Analysis and Machine Intelligence, IEEE Trans. on*, vol. 11, pp. 674–693, Jul 1989.

[79] D. Field, "Relations between the statistics of natural images and the response properties of cortical cells," *Journal of the Optical Society of America A*, vol. 4, no. 12, pp. 2379–2394, 1987.

[80] J. Rossi, "Digital techniques for reducing television noise," *SMPTE Journal*, vol. 87, no. 3, pp. 134–140, 1978.

[81] E. Simoncelli and E. Adelson, "Noise removal via bayesian wavelet coring," in *Image Processing (ICIP), IEEE Int. Conf. on*, vol. 1, pp. 379–382, 1996.

[82] E. Simoncelli, "Statistical models for images: Compression, restoration and synthesis," in *Signals, Systems & Computers, Conf. Record of the Thirty-First Asilomar Conf. on*, vol. 1, pp. 673–678, IEEE, 1997.

[83] J. Portilla, V. Strela, M. Wainwright, and E. Simoncelli, "Image denoising using scale mixtures of gaussians in the wavelet domain," *Image Processing, IEEE Trans. on*, vol. 12, no. 11, pp. 1338–1351, 2003.

[84] M. Crouse, R. Nowak, and R. Baraniuk, "Wavelet-based statistical signal processing using hidden markov models," *Signal Processing, IEEE Trans. on*, vol. 46, no. 4, pp. 886–902, 1998.

[85] G. Fan and X. Xia, "Image denoising using a local contextual hidden markov model in the wavelet domain," *Signal Processing Letters, IEEE*, vol. 8, no. 5, pp. 125–128, 2001.

[86] A. Bruhn, J. Weickert, and C. Schnörr, "Lucas/kanade meets horn/schunck: Combining local and global optic flow methods," *Int. Journal of Computer Vision*, vol. 61, no. 3, pp. 211–231, 2005.

[87] B. Lucas, T. Kanade, *et al.*, "An iterative image registration technique with an application to stereo vision.," in *Int. Joint Conf. on Artificial Intelligence*, vol. 81, pp. 674–679, 1981.

[88] B. Horn and B. Schunck, "Determining optical flow," in *Technical symposium east*, pp. 319–331, Int. Society for Optics and Photonics, 1981.

[89] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Computer Vision-ECCV 2004*, pp. 25–36, Springer, 2004.

[90] D. Sun, S. Roth, and M. Black, "Secrets of optical flow estimation and their principles," in *Computer Vision and Pattern Recognition (CVPR), IEEE Conf. on*, pp. 2432–2439, IEEE, 2010.

[91] M. Brunig and W. Niehsen, "Fast full-search block matching," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 11, pp. 241–247, Feb 2001.

[92] X. Gao, C. J. Duanmu, and C. Zou, "A multilevel successive elimination algorithm for block matching motion estimation," *Image Processing, IEEE Trans. on*, vol. 9, pp. 501–504, Mar 2000.

[93] W. Li and E. Salari, "Successive elimination algorithm for motion estimation," *Image Processing, IEEE Trans. on*, vol. 4, pp. 105–107, Jan 1995.

[94] T. Toivonen, "Number theoretic transform-based block motion estimation," *Thesis, Afstudeerwerk, Departement Elektrotechniek, Universiteit van Oulu, Finland*, 2002.

[95] Y. Chen, Y. Hung, and C. Fuh, "Fast block matching algorithm based on the winner-update strategy," *Image Processing, IEEE Trans. on*, vol. 10, pp. 1212–1222, Aug 2001.

[96] S. Zhu and K. Ma, "A new diamond search algorithm for fast block-matching motion estimation," *Image Processing, IEEE Trans. on*, vol. 9, pp. 287–290, Feb 2000.

[97] P. Hosur and K. Ma, "Motion vector field adaptive fast motion estimation," in *Second Int. Conf. on Information, Communications and Signal Processing*, pp. 7–10, 1999.

[98] A. Tourapis, "Enhanced predictive zonal search for single and multiple frame motion estimation," in *Electronic Imaging 2002*, pp. 1069–1079, Int. Society for Optics and Photonics, 2002.

[99] T. Li, S. Li, and C. Shen, "A novel configurable motion estimation architecture for high-efficiency mpeg-4/h.264 encoding," in *Design Automation Conf., Proceedings of the ASP-DAC Asia and South Pacific*, vol. 2, pp. 1264–1267 Vol. 2, Jan 2005.

[100] R. Li, B. Zeng, and M.-L. Liou, "A new three-step search algorithm for block motion estimation," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 4, pp. 438–442, Aug 1994.

[101] L. Po and W. Ma, "A novel four-step search algorithm for fast block motion estimation," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 6, pp. 313–317, Jun 1996.

[102] B. Gunyel and A. Alatan, "Multi-resolution motion estimation for motion compensated frame interpolation," in *Image Processing (ICIP), 17th IEEE Int. Conf. on*, pp. 2793–2796, Sept 2010.

[103] G. Gupta and C. Chakrabarti, "Architectures for hierarchical and other block matching algorithms," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 5, pp. 477–489, Dec 1995.

[104] J. Chalidabhongse and C.-C. Kuo, "Fast motion vector estimation using multiresolution-spatio-temporal correlations," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 7, pp. 477–488, Jun 1997.

[105] S. Zafar, Y. Zhang, and B. Jabbari, "Multiscale video representation using multiresolution motion compensation and wavelet decomposition," *Selected Areas in Communications, IEEE Journal on*, vol. 11, pp. 24–35, Jan 1993.

[106] K. M. Nam, J. Kim, R. Park, and Y. Shim, "A fast hierarchical motion vector estimation algorithm using mean pyramid," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 5, pp. 344–351, Aug 1995.

[107] J.-H. Ju, Y. Chen, and S.-Y. Kung, "A fast rate-optimized motion estimation algorithm for low-bit-rate video coding," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 9, pp. 994–1002, Oct 1999.

[108] X. Song, T. Chiang, X. Lee, and Y. Zhang, "New fast binary pyramid motion estimation for mpeg2 and hdtv encoding," *Circuits and Systems for Video Technology, IEEE Trans. on*, vol. 10, pp. 1015–1028, Oct 2000.

[109] Z. Wang, "Applications of objective image quality assessment methods [applications corner]," *Signal Processing Magazine, IEEE*, vol. 28, pp. 137–142, Nov. 2011.

[110] S. Ramani, T. Blu, and M. Unser, "Monte-carlo SURE: A black-box optimization of regularization parameters for general denoising algorithms," *Image Processing, IEEE Trans. on*, vol. 17, pp. 1540–1554, Sept. 2008.

[111] S. Gabarda and G. Cristóbal, "Blind image quality assessment through anisotropy," *Journal of Opt. Soc. Am. A*, vol. 24, pp. B42–B51, Dec. 2007.

[112] E. Cohen and Y. Yitzhaky, "No-reference assessment of blur and noise impacts on image quality," *Signal, image and video processing*, vol. 4, no. 3, pp. 289–302, 2010.

[113] G. Cao, Y. Zhao, and R. Ni, "Edge-based blur metric for tamper detection," *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, no. 1, pp. 20–27, 2010.

[114] C. Feichtenhofer, H. Fassold, and P. Schallauer, "A perceptual image sharpness metric based on local edge gradient analysis," *Signal Processing Letters, IEEE*, vol. 20, pp. 379–382, Apr. 2013.

[115] R. Dash, P. Sa, B. Majhi, *et al.*, "Blur parameter identification using support vector machine," *ACEEE Int. Journal on Control System and Instrumentation*, vol. 3, no. 2, 2012.

[116] M. Chen and A. Bovik, "No-reference image blur assessment using multiscale gradient," in *Quality of Multimedia Experience. Int. Workshop on*, pp. 70–74, July 2009.

[117] G. Wallace, "The jpeg still picture compression standard," *Consumer Electronics, IEEE Trans. on*, vol. 38, pp. xviii–xxxiv, Feb 1992.

[118] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Computer Vision and Pattern Recognition, IEEE Computer Society Conf. on*, vol. 2, pp. 60–65 vol. 2, June 2005.

[119] M. Elad and M. Aharon, "Image denoising via learned dictionaries and sparse representation," in *Computer Vision and Pattern Recognition, IEEE Computer Society Conf. on*, vol. 1, pp. 895–900, June 2006.

[120] X. Feng and P. Milanfar, "Multiscale principal components analysis for image local orientation estimation," in *Signals, Systems and Computers. Conf. Record of the Thirty-Sixth Asilomar Conf. on*, vol. 1, pp. 478–482, Nov. 2002.

[121] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *Image Processing, IEEE Trans. on*, vol. 16, pp. 349–366, Feb. 2007.

[122] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. New York, NY, USA: Cambridge University Press, 3 ed., 2007.

[123] J. Demmel and W. Kahan, "Accurate singular values of bidiagonal matrices," *SIAM J. Sci. Stat. Comput*, vol. 11, pp. 873–912, 1990.

[124] N. Ponomarenko, L. Jin, O. Ieremeiev, V. Lukin, K. Egiazarian, J. Astola, B. Vozel, K. Chehdi, M. Carli, F. Battisti, *et al.*, "Image database TID2013: Peculiarities, results and perspectives," *Signal Processing: Image Communication*, vol. 30, pp. 57–77, 2015.

[125] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *Image Processing, IEEE Trans. on*, vol. 13, pp. 600–612, April 2004.

[126] A. Hast, J. Nysjö, and A. Marchetti, "Optimal RANSAC - towards a repeatable algorithm for finding the optimal set," *Journal of WSCG*, vol. 21, no. 1, pp. 21–30, 2013.