Motion blur reduction for liquid crystal displays using a structure controlled filter

Kwintenberg, G.

Award date:
2009

Motion blur reduction for Liquid Crystal Displays

using a structure controlled filter

ing. Geert Kwintenberg
Eindhoven University of Technology,
Den Dolech 2, 5600 MB Eindhoven,
The Netherlands
`g.j.kwintenberg@student.tue.nl`
`geert.kwintenberg@gmail.com`

ir. Frank van Heesch
Philips Research Laboratories Eindhoven,
Prof. Holstlaan 4 (HTC 36), 5656 AA Eindhoven,
The Netherlands
`frank.van.heesch@philips.com`

July 2, 2009

Abstract

To reach optimal dynamic sharpness on liquid crystal displays, motion blur reduction is necessary. Motion blur on liquid crystal displays exists of two blur components: Display blur and camera blur. Display blur is a result of the sample and hold characteristic of the display. Camera blur is caused by objects moving in front of the camera when the shutter is open. From the mathematical analysis of motion blur, straightforward methods based on IIR and FIR filtering can be derived, but are sensitive to noise and create artifacts. Heuristic approaches based on FIR filtering used to reduce noise and artifacts, show large computational complexity.

Content adaptive filtering is proposed by means of a structure controlled filter, which can reduce this complexity. Objective and subjective evaluations of this method show a successful reduction of the camera blur. Subjective evaluations learned that improvement is clearly visible for individual frames, although the perceived sharpness improvement is reduced when looking at video material. Pre-correction for display blur is not successful using the structure controlled filtering method, where the filter is trained on display blur simulated training data.

To support hardware efficiency, alternative implementations have been investigated, which are bounded by constraints. The fixed aperture constraint has significant consequences on the LUT size and aperture shape. This alternative implementation, although more efficient in terms of implementation, did not yield significant motion blur reduction. The huge design freedom of choosing a filter kernel and the lack of design guidelines, make the design of such a hardware efficient implementation difficult.

Contents

Chapter 1

Introduction

Over the last years the Cathode Ray Tube (CRT) has disappeared rapidly as Television Display. This disappearance is caused by the advent of Flat Panel Displays. Many types of flat panel displays can be discriminated such as: liquid crystal display (LCD), plasma display panel (PDP), organic light-emitting diode display (OLED), light-emitting diode display (LED). Currently the LCD has the largest share in the consumer market for displays. Mainly caused by the high performance, low price, light weight and thin depth compared to its competitors.

The *motion blur* on a liquid crystal display consists of two blur components. The first component called *display blur* is caused by the characteristics of the display itself. *Camera blur* is the second component due to characteristics of the camera which is used to record the scene. At the introduction of LCDs, the CRT had one major advantage over LCD. This advantage occurred when displaying moving objects. Moving objects are perceived less sharp on an LCD. This is caused by two factors, slow response time of the LC material and the fact that every LCD pixel emits light during the whole frame time. This last property is called the sample and hold effect. Recent developments in hardware and software have removed this advantage. For example developments in LC material and their driving schemes, made sure that the response time is below the frame period and a lot of effort is spend on further improvement of the response time. An important technique to reduce the response time is called Overdrive[Oku93]. This technique compares the luminance of the current pixel with the luminance of the next(temporal) pixel, based on this difference an 'overdrive' value is applied such that the response time is improved. If we would be able to produce an LCD with infinitely fast response time, we would still perceive moving objects blurred due to the sample and hold effect. The sample and hold effect of an LCD is the property that the pixels emit light during the whole frame time. A pixel of a CRT on the contrary emits light for a very short time with respect to the frame time. The human visual system (HVS) has a property called eye tracking. When objects move, the eyes (and head) follow the smooth motion of the object such that the object has a fixed position on the retina. If the HVS tracks a moving object, combined with the sample and hold effect of the LCD, causes pixels to land on different parts of the retina. This produces display blur on the retina because the eye integrates in the temporal domain. Display blur, with infinite fast response time of the LCD, leads to the same blurring as a phenomena called *camera blur.* Camera blur occurs when objects move in front of a camera when its shutter is open. During this shutter time, the object lands on different positions of the photosensitive layer. This causes an amount blur depending on the motion speed and the camera shutter time.

1.1 Problem description

The main goal of the project is to get optimal dynamic sharpness for an LCD. To be successful we have to remove both blur components. To reduce display blur several methods have been proposed: increasing the frame rate[BJP07], inserting black frames between every two input frames[HOP$^+$04], scanning backlight[Fis01] and motion compensated inverse filtering[KV04]. The last method filters along the motion vectors using an inverse filter. The idea behind this method is to pre-compensate the image in the direction of motion, in such a way that the perceived image looks sharp. This requires spatial filtering along the motion vector for every pixel. An implementation of this method showed a high computational complexity and only limited improvement in perceived video quality. Therefore this report describes a method which improves up on earlier work by reducing computational complexity and/or improving the perceived video quality. The method proposed in this report uses structured controlled filtering. A structure controlled filter or also called trained filter, is known for its low computational complexity and ability to improve spatial sharpness of video. Unfortunately this filter does require a large look up table. Since display blur and camera blur are closely related, an camera blur reduction filter is also investigated based on the structured filter approach. We assume that the camera shutter time related to the input video is a known value. In this document we will investigate whether we should put two structured controlled filters in cascade, or combine the two filters and reduce camera- and display blur at the same time.

1.2 Outline

Chapter 2 gives a thorough explanation of motion blur by the use of a mathematical model. From this mathematical model some straightforward motion blur reduction methods are derived. At the end of the chapter methods are discussed which are specific solutions for the reduction of camera blur or display blur. The third chapter gives a explanation of the structure controlled filter. A complete mathematical formulation is used to describe the filter properties. Chapter 4 proposes the method. First the research on classification of the pixel neighborhood is documented, where the method is applied on a simplified case. Later the method is generalized to any case. Since the method generates some artifacts, the reduction of these are described. Finally the chapter ends with some considerations to reduce the complexity, which make hardware implementation possible. The quality of the method, is validated using a perception test in Chapter 5. This test includes 16 candidates evaluating the results. Chapter 6 gives a summary of the tools used in this projects and gives a description of the workflow. This information can give a quick start for future research based on this project. In the final chapter the conclusions are formulated.

Chapter 2

Motion blur

In this chapter motion blur is explained using a mathematical formulation. From this formulation a straight forward motion reduction method is derived. At the end of this chapter specific solutions for camera blur and display blur are described.

2.1 Motion blur analysis

When we compare the luminance of a CRT pixel with an LCD pixel during a single frame time we spot a major difference. A CRT is an impulse type display emitting light only a fraction of the frame time. This is caused by the phosphor of the pixel. It glows for a very short period after a electron beam 'hit'. In contrast, an LCD pixel emits light during the whole frame time. In the first model we assume that an LCD pixel has an infinite fast response i.e. a transition from the lowest luminance level to the highest level or reversed can happen instantaneous. Suppose the HVS tracks a moving object on an LCD screen, which moves with a constant speed $\vec{v}$. This object has an associated displacement vector $\vec{D} = \vec{v}.T$, where $|\vec{D}|$ is the number of pixels the object moves between two consecutive frames. The angle of the vector defines the direction in which the object moves. In Figure 2.1 the x-coordinate of a horizontally moving object is depicted for a set of consecutive frames. In the left graph, the position of the object on the screen shows the sample and hold effect. This sample and hold effect combined with the smooth motion of the eye tracking the object, causes the object movement on the retina shown in the right graph. Between two consecutive frames the object moves proportional to $\vec{D}$ on the retina. This movement on the retina is perceived as blur, because the eye acts as a lowpass filter in the temporal domain.

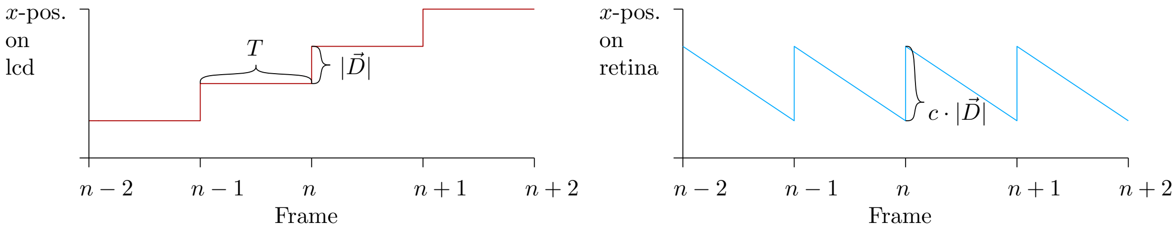


Figure 2.1: The position of a horizontally moving object on an LCD(left). Eyetracking this object causes the object position on the retina depicted on the right. The swift object movement on the retina causes the eye to integrate the light.

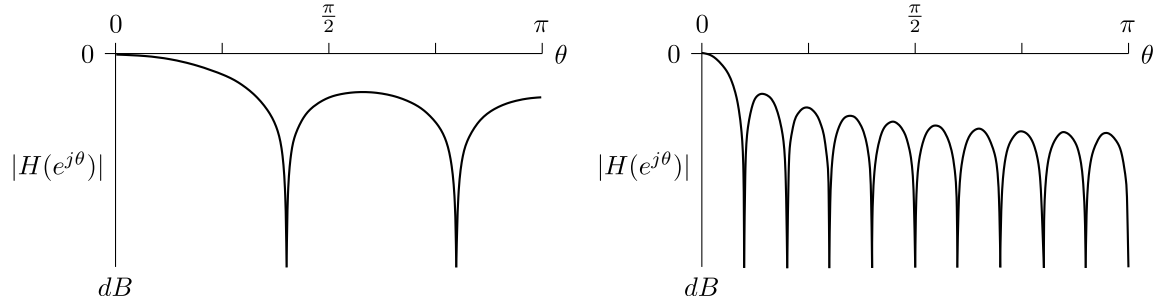
This averaging effect of the eye can be modeled by integrating the image along the displacement vector. Integrating the image can be modeled by an averaging filter. For example an object that moves with the speed of 5 pixels per frame, would yield a $[\frac{1}{5},\frac{1}{5},\frac{1}{5},\frac{1}{5},\frac{1}{5}]$-filter. This model can also be used for camera blur. Suppose we would move the object with the same displacement in front of a camera, this would lead to a similar blurring effect. The difference is that the integration now takes place on the photo sensitive layer of the camera instead of the retina. Note that the camera shutter time must be equal to T. When we calculate the Discrete-Time Fourier Transform of the box rectangular filter using:

$$H\left(e^{j\theta}\right) = \sum_{k=-\infty}^{\infty} h[k]e^{-jk\theta} \tag{2.1}$$

we get:

$$H\left(e^{j\theta}\right) = 1 + e^{-j\theta} + e^{-2j\theta} + e^{-3j\theta} + e^{-4j\theta} \tag{2.2}$$

If we plot the magnitude of the frequency response (Figure 2.2(a)), we can see that this response contains zeros. These zeros make reconstruction difficult, since frequencies associated to these zeros are completely removed. When the object has a higher speed and therefore a larger integration range, the frequency response contains more zeros (Figure 2.2(b)) , which makes reconstruction even harder.



(a) $|H\left(e^{j\theta}\right)|$ of a 5-taps averaging filter

(b) $|H\left(e^{j\theta}\right)|$ of a 20-taps averaging filter

Figure 2.2: Magnitude of the frequency response of the averaging filter.

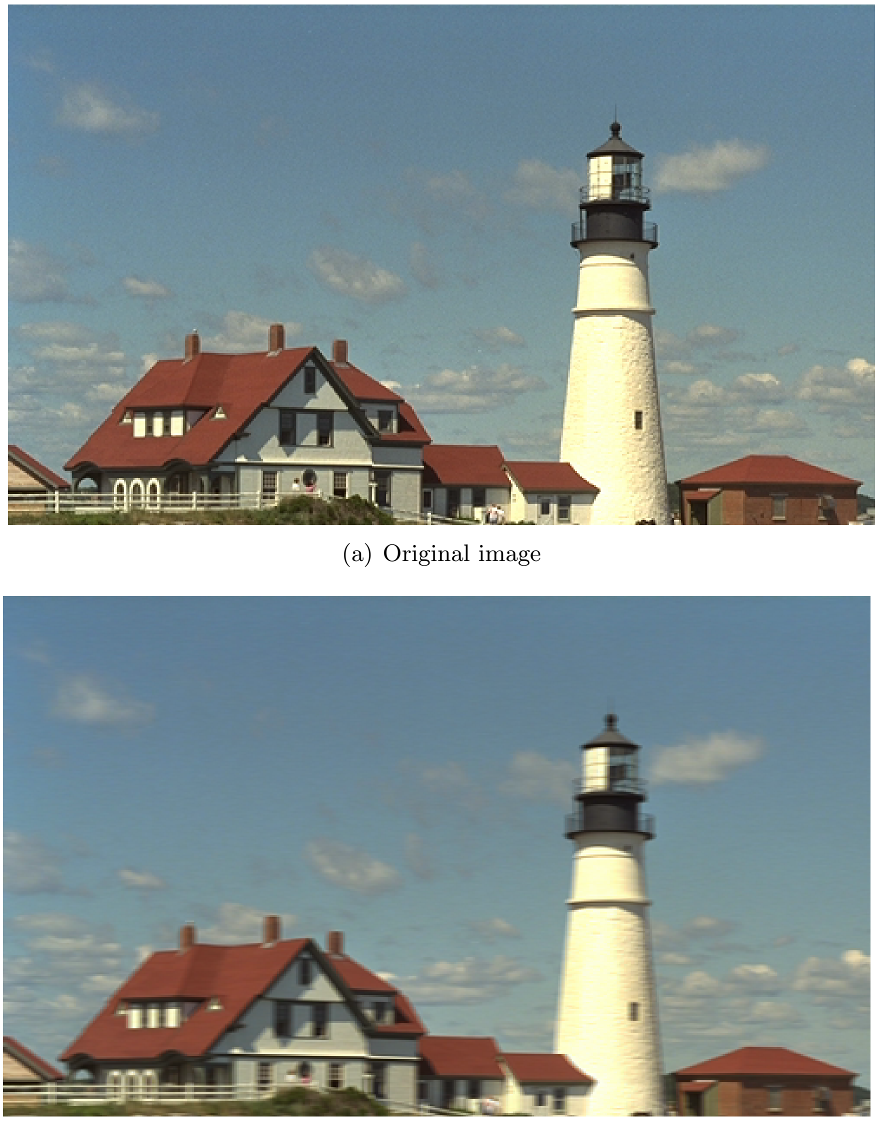
2.2 Motion blur reduction

To demonstrate some straightforward reconstruction methods we limit ourselves to horizontal motion. For a more formal description of a horizontal filter we use the notation from [dH08] The luminance of a display signal is denoted by $F(\vec{x}, n)$ where $\vec{x}$ is a vector $\begin{pmatrix} x \\ y \end{pmatrix}$ to a pixel in image n. The luminance output $F_{oh}(\vec{x}, n)$ of the filter with an impulse response $h_h(k)$ is:

$$F_{oh}(\vec{x}, n) = \sum_{\forall k} h_h(k) \cdot F(\vec{x} + k \cdot \vec{u_x}, n), \sum_{\forall k} h_h(k) = 1 \tag{2.3}$$

where $\vec{u_x} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ is the horizontal unit vector in the pixel grid.

Suppose we would have an image which moves from left to right over the screen with a speed of 5 pixels per frame. According to our model this would induce the same amount of display blur as camera blur under the following conditions. The image would have to move with the same speed in front of a camera when the shutter time is equal to the frame time. In our simplified display blur model this can be simulated by applying a lowpass filter to the image having the following impulse response: $h_h(k) = \frac{1}{5}$ for $0 \leq k \leq 4$, or in simplified notation a $[\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}]$-filter. The simulated motion blur shown in Figure 2.3(b) applied to the original Figure 2.3(a).



(a) Original image

(b) Simulated motion blur

Figure 2.3: Image (b) is the result of the simulation of horizontal motion blur applied on (a). This simulation is performed by a convolution of (a) and a $[\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}]$ kernel.

The first straightforward reconstruction method would be to isolate the original image from the equation describing the linear filtering:

$$F_{oh}(\vec{x}, n) = \frac{1}{5} \cdot \sum_{k=0}^{4} F(\vec{x} + k \cdot \vec{u_x}, n) \tag{2.4}$$

Rewriting 2.4 we get:

$$F_{oh}(\vec{x}, n) = \frac{1}{5} \cdot F(\vec{x}, n) + \frac{1}{5} \cdot \sum_{k=1}^{4} F(\vec{x} + k \cdot \vec{u_x}, n) \tag{2.5}$$

After isolating $F(\vec{x}, n)$ from 2.5 we get:

$$F(\vec{x}, n) = 5 \cdot F_{oh}(\vec{x}, n) - \sum_{k=1}^{4} F(\vec{x} + k \cdot \vec{u_x}, n) \tag{2.6}$$

Equation 2.6 describes a recursive filter, since output values are used as feedback. From this equation we can define a so called Infinite Impulse Response (IIR) filter. A block diagram of this filter can be seen in Figure 2.4. From this filter we can compute the frequency response:

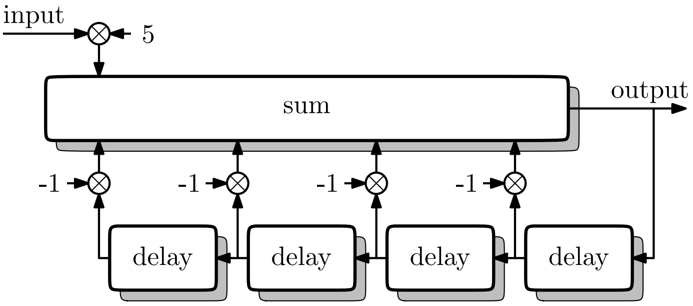


Figure 2.4: Block scheme of an 5-taps Infinite Impulse Response Filter. The delay component stores a value at the input at time n, and outputs this value at $n+1$. IIR filters have an impulse response that is non-zero over and infinite length of time.

$$H\left(e^{j\theta}\right) = \frac{5}{-1 - e^{-j\theta} - e^{-2j\theta} - e^{-3j\theta} - e^{-4j\theta}} \tag{2.7}$$

When plotting the magnitude $|H\left(e^{j\theta}\right)|$ (Figure 2.5) we can see that this filter is exactly the inverse of the frequency response of the blur filter (Figure 2.2(a)). When the IIR filter is applied to the blurred image we do not get the original image (Figure 2.6). The image is perceived sharper than the blurred image, but it contains artifacts. These artifacts are caused by quantization. After the blur filter operation, the values are quantized to 8-bit values. This quantization noise causes small errors when reconstructing the image by means of an IIR filter. Since the filter in the current configuration is not stable, these errors will spread throughout the whole image. A stable filter could only be build by this approach, when the transfer function of the blur filter does not contains zeros. In general the filter coefficients are quantized as well, hence disturbing perfect reconstruction.

The second solution would be to use a Finite Impulse Response(FIR) filter. These filters have a finite impulse response and are therefore stable by definition. In Figure 2.7, the block diagram of such a filter is depicted.

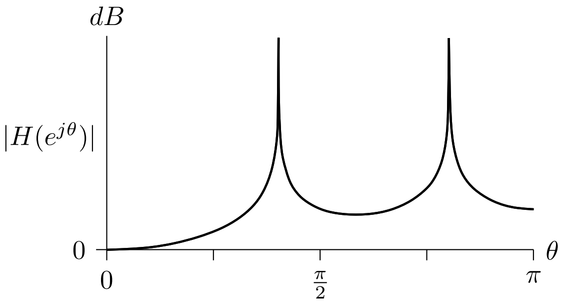


Figure 2.5: The magnitude of the frequency response of the IIR filter. This response is the ideal inverse of a 5-taps averaging filter. Note that for frequencies $2\pi/5$ and $4\pi/5$ the amplification is infinite.

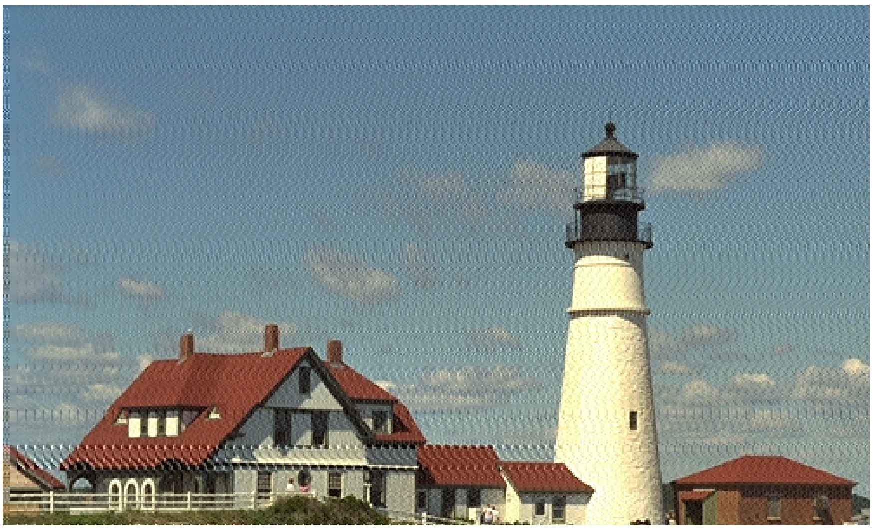


Figure 2.6: Image reconstructed by means of an IIR filter. The visible artifacts are caused by amplification of quantization noise. Quantization noise arises when the pixel values are quantized by the blur simulator.

The goal is to design a FIR filter in such a way that its frequency response is a good approximation for the frequency response of the IIR filter. The FIR filter has to be a approximation because the ideal frequency response requires infinite amplification. First we sample the magnitude of the frequency response $|H\left(e^{j\theta}\right)|$ of the ideal filter. The sampling is done using L sample points spread equally in the frequency range from 0 until 2π. We call this sampled frequency response $|H[k]|$. To these samples we add a linear phase, so we get:

$$H[k] = \left|H\left(e^{j\frac{2\pi k}{L}}\right)\right| \cdot e^{-j\frac{L-1}{L}\pi k} \tag{2.8}$$

From $H[k]$ we can derive a linear phase L taps FIR filter, by $h[n] = IDFT(H[k])$, where IDFT is the Inverse Discrete Fourier Transform defined as:

$$h[n] = \frac{1}{N}\sum_{k=0}^{N-1} H[k]e^{j\frac{2\pi kn}{N}} \tag{2.9}$$

The resulting filter is an approximation of the sampled filter, its frequency response is exact for the sample points. Hence increasing the number of samples gives a better approximation,

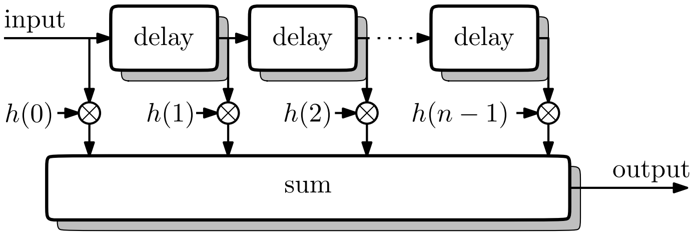


Figure 2.7: A Finite Impulse Response filter with n stages. Each stage i has an independent delay and amplification gain $h(i-1)$. In general FIR-filters are less efficient compared to IIR-filters, demand more computation but are stable by definition and are easy to implement.

but increases the computational complexity. As an experiment we sampled the frequency response using 41 sample points. This results in a FIR filter where the impulse response contains 41 taps. The result of this approximation for the filter described by Equation 2.4 can be seen in Figure 2.8. If we apply this filter to the blurred image (Figure 2.3(b)) it will

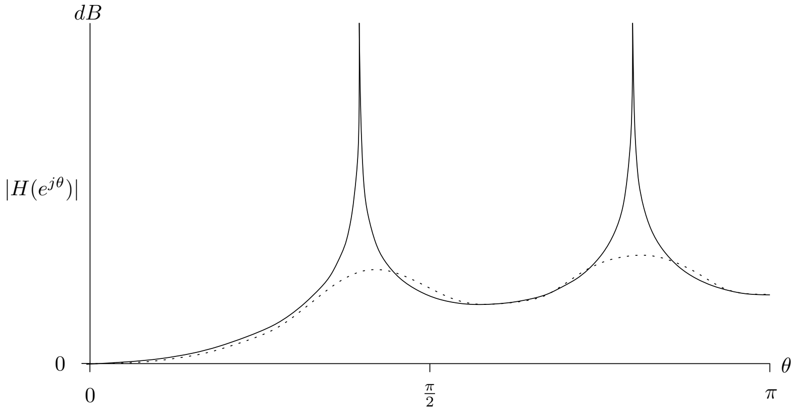


Figure 2.8: The solid line depicts the ideal frequency characteristic. The dotted line is the resulting frequency characteristic of a 41 taps FIR filter. As can be seen, infinite amplification makes accurate approximation difficult.

result in Figure 2.9, we observe some visible artifacts. These are ringing artifacts caused by the coarse approximation of the high amplifications in the frequency response. Furthermore IIR and FIR filtering have the drawback, since they both amplify the high frequencies, they are enhancing noise. Therefore these methods cannot be applied directly for picture quality enhancement.

The mathematical analysis for display and camera blur are similar, but their reduction methods are not. Current methods in the literature for solving these problems are different for both blur sources, therefore we discuss them separately. These methods are different because display blur is a phenomena that occurs on the retina, and camera blur occurs on the photo sensitive layer in the camera.

2.2.1 Camera blur reduction

The camera blur reduction is similar to the restoration of motion blurred photos. Photos can suffer from motion blur if there is movement when the shutter is open. This motion blur reduction is a double blind deconvolution problem. Since the signal and the distortion to this

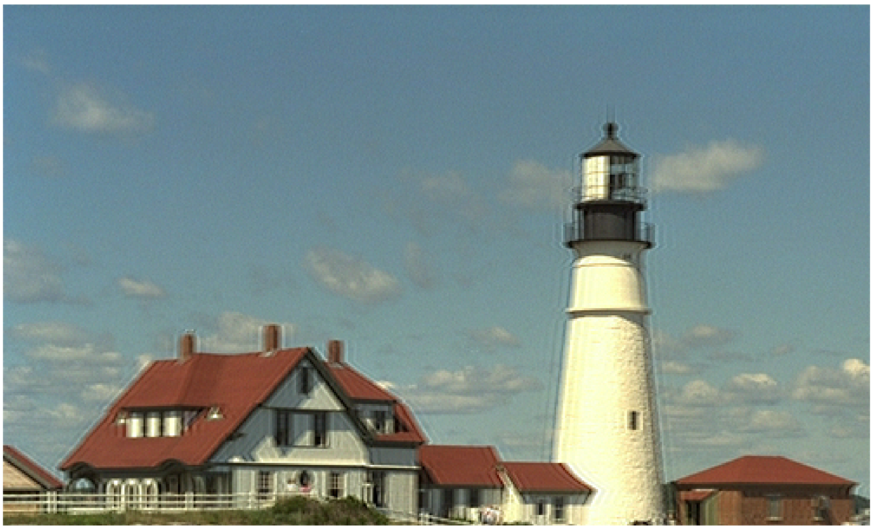


Figure 2.9: Image reconstructed by means of an 41-taps FIR filter. The artifacts around the sharp edges are ringing artifacts. These are caused by the coarse approximation of the high amplification area in the ideal frequency characteristic.

signal is unknown. As an example, it is not possible to discriminate between a blurred detail and a low frequent detail. The restoration consists of a two step procedure. First the point spread function (PSF) is estimated. For estimation of the PSF two parameters need to be derived from the photo, namely the length and the angle. With this information the restoration algorithm can apply inverse filtering according to this PSF. Such an algorithm which combines both functions is called *Blind Deconvolution.* Blind deconvolution is prior art in the area of digital photography and astronomy. A well know algorithm is the RichardsonLucy deconvolution[Ric72], which uses and iterative procedure for restoring the image. In our case estimation of the PSF is not necessary, since we have a sequence of images instead of a single image. From a sequence of images, a motion estimation can be executed. This motion estimation algorithm returns the motion speeds and angles of the objects in the sequence. These motion vectors can be translated easily into a PSF for every object. Hence PSF estimation for video simplifies to estimation of the shutter time and the motion. Since inverse filtering also enhances the noise, Norbert Wiener found a solution for this problem in the 1940s. The *wiener filter* provides an 'optimal' tradeoff between signal and noise enhancement. Unfortunately a model for noise and signal is necessary when using this filter. For a TV signal such a model does not exist.

2.2.2 Display blur reduction

Several methods have been proposed to reduce display blur. They can be categorized in three groups.

1. Improvement of the response time
2. Reduction of the hold time
3. Inverse filtering

The first category contains methods which reduce the response time of the liquid crystals. These methods like Overdrive[Oku93] have significantly reduced the response time, as far as below the frame time. Reducing the response time any further would not significantly improve the image quality, since the motion blur is mainly caused by the hold time of the display.

The second category tries to reduce the hold time. The first method is called *Motion Compensated Upconversion*[BJP07]. This method performs an upconversion of the frame rate. The motion compensation is very important, since simply repetition of the frames would not reduce the hold time. To explain the method we use the same Figure(2.10) as before. Again the graphs show the x-coordinate of a horizontally moving object. The colored lines belong to the upconverted sequence, whereas the grey line belongs to the original. Because of the motion compensated upconversion, the object integration distance is decreased by a factor two. Therefore the display blur is also reduced by a factor two. As can be seen, this method is very effective, but also expensive in terms of hardware. This because the display must be able to show the sequence at the necessary frame rate. Besides upconversion using motion compensated frames, other upconversion methods have been investigated. These methods insert black[HOP$^+$04], grey or smooth frames. Another method to reduce the hold time is to use a scanning backlight[Fis01]. The backlight used is brighter than traditional backlight, but the pixels are only illuminated during a short portion of the frame time. In this way, the display becomes an impulse type display instead of a sample and hold type, which reduces the display blur. Unfortunately this method does give flicker.

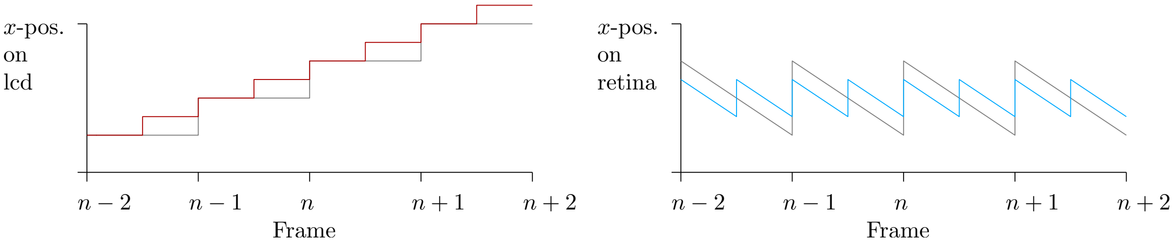


Figure 2.10: On the left can be seen that Motion Compensated Upconversion reduces the hold time compared to the original (grey line). This reduction reduces the length of the quick object shift on the retina. Hence the blur length is reduced.

An example of the third category is *Motion Compensated Inverse Filtering*(MCIF)[KV04] which tries to pre-correct the image. This pre-correction is applied in such a way, that the perceived display blur is reduced. This pre-compensation is based on motion compensated high pass filtering. For instance a simple 3-tap high pass filter is applied and added to the original image. This filter approximates the ideal inverse filter (Like Figure 2.8). The angle and the gain of this filter depends on motion vectors generated by a motion estimator. An abstract block diagram of this method can be seen in Figure 2.11. The 3-tap filter is a trade off to keep computational complexity low enough for real time implementation. But this simplicity also has its downside. The coarse approximation gave artifacts like overshoots. Also the gain factor becomes very high when the motion speed increases, this causes two problems. The high amplification enhances noise. The second problem is the limited dynamic range of the display. The high amplification causes clipping of pixel values. A simple solution for the noise enhancement, it to use a image complexity threshold. In low detailed areas of the image, where the noise is visible, no pre-correction is applied. This is generally no problem, because blurring of low detailed areas can not be perceived very well.

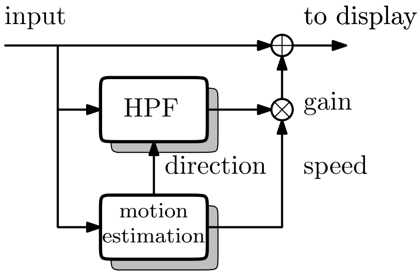


Figure 2.11: Block diagram of Motion Compensated Inverse Filtering. Vectors resulting from motion estimation of the input frames, control the high pass filter and the gain. The HPF output is added to the input frame, which result in a pre-corrected output frame. This pre-correction reduces the display blur visible by the human eye.

Chapter 3

Structure Controlled Filters

A structure controlled filter is also called a *trained filter*. A structure controlled filter consists of two parts. A training processes and a processes when the filter is applied. The training process can be performed off-line. The training will be explained using the block diagram(Figure 3.1). To train the filter we use a large set of images, that represent the preferred output quality. These images are distorted such that it matches the distortions we like to remove. In our case, the images are distorted by blur using an averaging filter, in cascade with additive noise that models the transmission noise. The training process classifies every pixel according to the structure of a certain aperture around this pixel. Pixels that fall in the same class are assigned a filter that minimizes the squared error compared to the 'original' pixel value from the training set. Finally for every classcode the optimal filter coefficients are stored in a look up table.

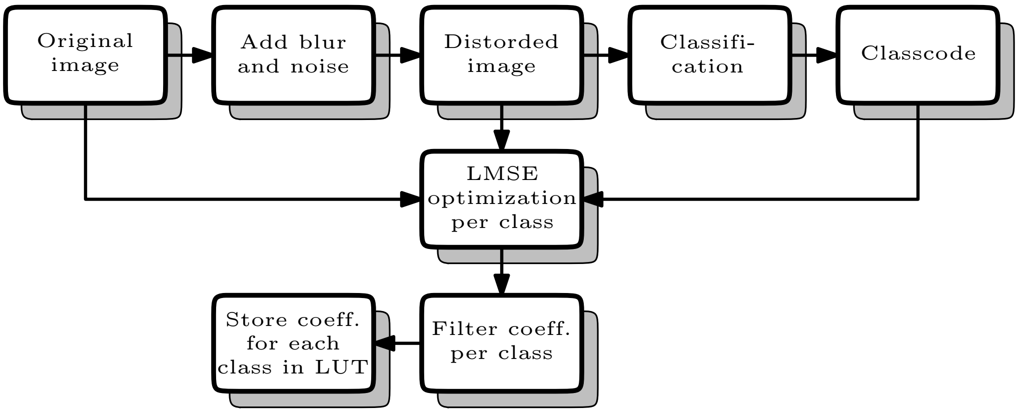


Figure 3.1: Block diagram of the training process of the structure controlled filter. Each pixel in the distorted image is classified depending on local structure. Every class is assigned coefficients, based on least means square error optimization between the distorted pixel apertures belonging to the class and their original values.

To better understand the least mean square optimization, we define the aperture around the distorted pixels as the 2-dimensional array $F_{D,c}(i,j)$, and the original pixels as $F_{O,c}(j)$.

Both arrays belong to the same classcode c. The filtered pixels $F_{F,c}(j)$ can be defined as:

$$F_{F,c}(j) = \sum_{i=1}^{n} w_c(i) F_{D,c}(i,j) \tag{3.1}$$

Where $w_c(i)$ for $1 \leq i \leq n$ are desired square error optimal filter coefficients. The aperture contains n pixels, hence we need equal number of filter coefficients. With the index j we can address all m different pixel apertures which belong to the same class c. The summed square error between the filtered pixels and the original pixels can be defined as:

$$(3.2)$$

$$\begin{aligned} e^2 &= \sum_{j=1}^{m} (F_{O,c}(j) - F_{F,c}(j))^2 \\ &= \sum_{j=1}^{m} \left(F_{O,c}(j) - \sum_{i=1}^{n} w_c(i) F_{D,c}(i,j) \right)^2 \qquad (3.3) \end{aligned}$$

To minimize the squared error e^2, the first derivative of e^2 to $w_c(k)$ for $1 \leq k \leq n$ should be zero.

$$\frac{\partial e^2}{\partial w_c(k)} = \sum_{j=1}^{m} 2F_{D,c}(k,j) \cdot \left(F_{O,c}(j) - \sum_{i=1}^{n} w_c(i) F_{D,c}(i,j) \right) = 0 \tag{3.4}$$

The optimal coefficients $w_c(i)$ can be found by Gaussian Elimination applied on Equation 3.4. Which can be seen below:

$$\begin{bmatrix} w_c(1) \\ w_c(2) \\ \dots \\ w_c(n) \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{m} F_{D,c}(1,j)F_{D,c}(1,j) & \dots & \sum_{j=1}^{m} F_{D,c}(1,j)F_{D,c}(n,j) \\ \sum_{j=1}^{m} F_{D,c}(2,j)F_{D,c}(1,j) & \dots & \sum_{j=1}^{m} F_{D,c}(2,j)F_{D,c}(n,j) \\ \dots & \dots & \dots \\ \sum_{j=1}^{m} F_{D,c}(n,j)F_{D,c}(1,j) & \dots & \sum_{j=1}^{m} F_{D,c}(n,j)F_{D,c}(n,j) \end{bmatrix}^{-1} \times \begin{bmatrix} \sum_{j=1}^{m} F_{D,c}(1,j)F_{O,c}(j) \\ \sum_{j=1}^{m} F_{D,c}(2,j)F_{O,c}(j) \\ \dots \\ \sum_{j=1}^{m} F_{D,c}(n,j)F_{O,c}(j) \end{bmatrix} \tag{3.5}$$

The resulting optimal coefficients are then stored in the LUT. The block diagram of the process when we apply the structure controlled filter to distorted images can be seen in Figure 3.2. First a pixel is classified using exactly the same classifier as the one used in the training process. From this classification we get a classcode. From this classcode we can find the corresponding coefficients in the LUT. Using these coefficients the filter operation is applied. In case the table is not completely filled during the training process, an all-pass filter acts as a failsafe. So the training process automatically fills empty table entries with an all-pass filter:

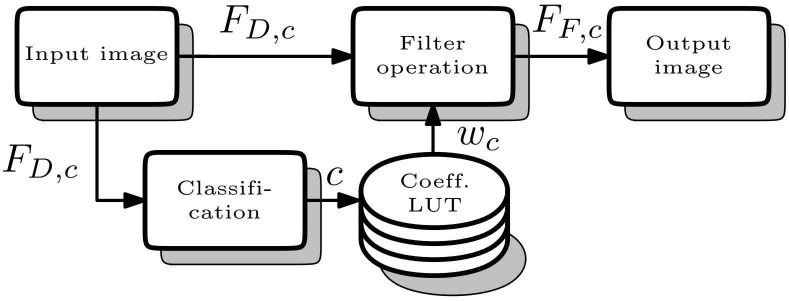


Figure 3.2: Block diagram of the filtering process of the structure controlled filter. Each pixel in the input image is classified depending on local structure. According to the resulting classnumber the filter coefficients are taken from the LUT. Using these coefficients the pixel is filtered.

$$w_c(i) = \begin{cases} 1 & \text{if } \ i = \lceil \frac{n}{2} \rceil, \\ 0 & \text{else} \end{cases} \tag{3.6}$$

Where we assume that the center pixel has position $i = \lceil \frac{n}{2} \rceil$ in the aperture. An empty table entry could be caused by a couple of reasons. The size of the training set might not be sufficient, or not representative to the data on which the filter is applied. The matrix on which the gaussian elimination is applied can be singular, which can not be solved. It is important to keep the LUT of reasonable size, because an enormous amount of training data might be necessary to fill the LUT. To estimate whether the LUT is sufficiently filled, one might perform experiments to see whether the result converges to a certain quality when increasing the training set. The size of the LUT depends on the number of classes. For example when using a structure based classifier, the number of classes might be exponential to the number of pixels used in the classifier. In classifier design lies a tradeoff between classifier quality and LUT size. In the design of a classifier lies an enormous freedom of choice, for example the shape of the structure and how structure is encoded to a class.

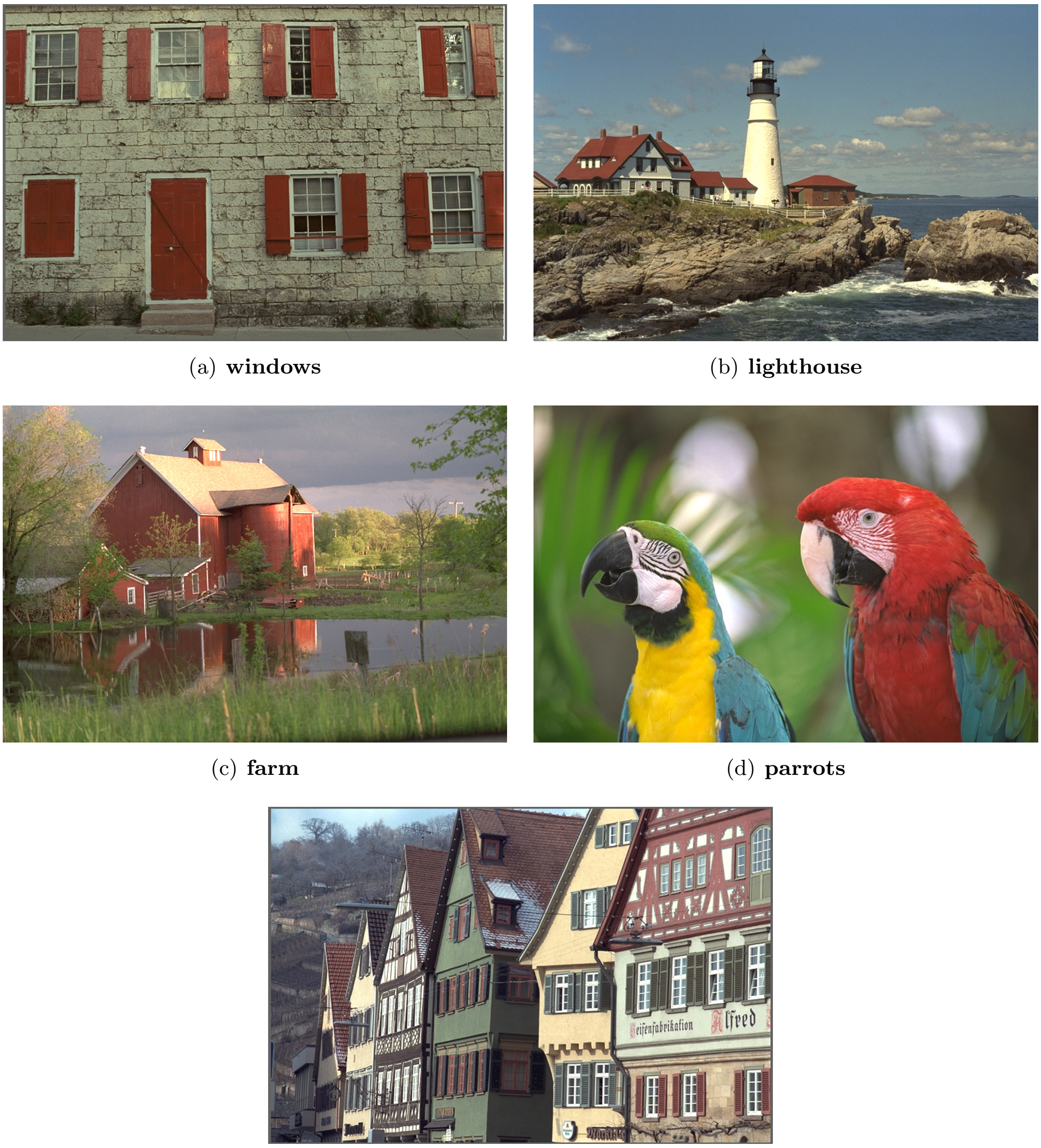
Chapter 4

Proposed Method

The method proposed here uses the Structure Controlled Filter for the reduction of motion blur. MCIF suffers from artifact caused by the coarse approximation of the ideal filter. These artifacts were suppressed by means heuristics. Implementing these heuristics made the method computational expensive. Using a structure controlled filter instead of a 3-taps filter used in Motion Compensated Inverse Filtering might give a better result. Structure controlled filtering brings adaptivity to noise, edges and texture. Since motion blur consists of two components, camera- and display blur, we investigate them separately. First, the camera blur reduction is investigated. Secondly, we try to pre-correct the sequences to reduce the display blur. To design the *Motion Compensated Structure Controlled Filter* we used the following strategy: In the first section we design a filter which can solve a 1-dimensional case of camera blur with a fixed blur length. In the second section the result of section 1 is generalized to any motion speed and direction. In this section the generalized filter is applied on real video data and analyzed. The third section describes the artifacts generated by the method and proposes some artifact reduction solutions. In the fourth section the generalized filter is applied to pre-correct sequences. The last section describes a hardware efficient version of the Motion Compensated Structure Controlled Filter.

4.1 Motion blur characterization

The classification is essential to the structure controlled filter method. A accurate classification, classifies pixels such that filtering pixels belonging to the same class, require the same filter coefficients to get the desired result. Unfortunately there is a trade-off between the classification quality and the LUT size. An accurate classification always results in very large amount of classes. In this section we investigate different classification methods and apply them to a fixed amount of camera blur. For the first experiments the structure controlled filter is being trained by a set of 25 degraded images. The resolution of the images is 1600x1200 pixels, where every pixel is used for the training. This training set is degraded by a horizontal, 5-pixel averaging filter to simulate camera blur. After this blurring some gaussian additive noise is added. The filter is then applied on a set of five degraded benchmark images. These images have the same degradation as the training data. An important remark is that the benchmark images are not part of the training data, because this gives a result which is too optimistic. As an objective measurement to evaluate the classification, the mean square error of pixel luminance values is used. As a reference we use the mean square error between the



(a) **windows**

(b) **lighthouse**

(c) **farm**

(d) **parrots**

(e) **houses**

Figure 4.1: The five images used to benchmark the classification. The benchmark set contains several common image structures like, high detailed textures, smooth details which are out of focus, sky, and water. This because the algorithm should perform well on average for any kind of image structure.

benchmark images (Figure 4.1) and the degraded benchmark images. Subjective assessment has also been performed and is discussed later in this chapter.

4.1.1 Adaptive Dynamic Range Coding

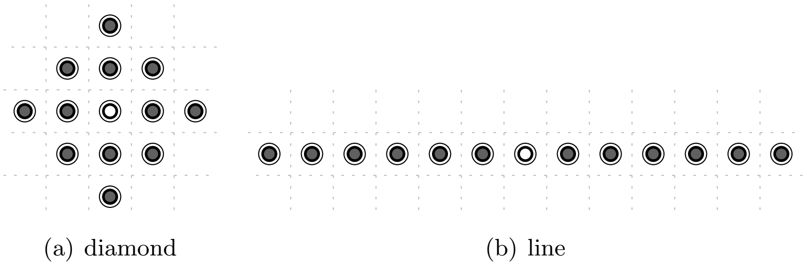
In the first approach Adaptive Dynamic Range Coding (ADRC) is used to classify the pixels. This method is described in [KK95]. This classification tries to capture the local image structure, by encoding whether the luminance of a pixel in the aperture lies above or below the aperture average. For an aperture which contains the pixels $1 \leq x_i \leq n$ we define:

$$C_i = \begin{cases} 1 & \text{if } \ x_i > av, \\ 0 & \text{else.} \end{cases} \tag{4.1}$$

Where the average luminance value av is defined as:

$$av = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4.2}$$

When the class bits C_i are placed in a fixed order, this results in a binary representation of the class number. With this method the number of classes is exponential in the number of pixels used for classification. This limits the aperture size in practice. The next design decision is to select an aperture shape. As a first shape we try the diamond shaped 13-points aperture (Figure 4.2) from [KK95], which will result in $2^{13} = 8192$ classes. This diamond is centered around the pixel which needs to be filtered. Using this aperture shape, the filter is trained using the training data. Afterwards the structure controlled filter uses this aperture to classify the pixels and applies the convolution using the same aperture shape and the coefficients stored in the LUT. From this results we compute the mean square error. See chapter 3 for more detailed information. As can be seen in Table 4.1, using this diamond aperture decreases the mean square error results compared to the reference. Classifying the structure along the degradation might give improvement. This because the blur is caused by a one-dimensional function, we should use a one-dimensional aperture to catch the structure of the blur. Therefore we use a line aperture parallel to the motion vector. Using this aperture shape, the mean square error(Table 4.1) was reduced significantly for all benchmark images compared to the diamond shaped aperture. The advantage of the classification according


(a) diamond

(b) line

Figure 4.2: Two aperture shapes used to classify and filter the pixels from the distorted input image. The line aperture performs better at characterizing the structure of the degradation than the diamond, if positioned parallel to the motion vector. The diamond gives a better characterization of the local structure.

Image	**reference**	**diamond**	**line**	**lmse**
windows	173.786	144.994	96.293	101.394
lighthouse	97.881	79.929	52.771	56.213
farm	97.233	82.720	58.283	61.616
parrots	36.346	29.830	22.933	24.339
houses	478.271	401.524	248.679	264.241
total	176.703	147.800	95.792	101.560

Table 4.1: The first column shows the mean square error results, of the degraded images without any enhancement applied. The second and the third column show the results of the structure controlled filter applied using a diamond and a line aperture. The last column depicts results from an LMSE-filter applied on the benchmark set.

to the local structure becomes clear when we compare the line aperture based structure controlled filter, with a Least Mean Square Error (LMSE) filter using the same line aperture. This filter does not discriminate between structures, so every pixel ends up in the same class. Hence the optimal coefficients are computed which minimize mean square error for all the pixels. From the results in the table it is clear that the structure controlled filter outperforms the LMSE-filter. If we evaluate the mse results in the table we can conclude that the aperture shape has a lot of influence on the mse scores. Since the design space for such an aperture shape is very large, we can only evaluate a limited number by hand. Therefore an automated full search of a part of the design space was executed. The pseudo-code of this algorithm can be written as:

```
Algorithm fullsearch($T_o, T_d, B_o, B_d$)
Input: $T_o$: Set of original Training data, $T_d$: Set of distorted Training data
Input: $B_o$: Set of original Benchmark data, $B_d$: Set of distorted Benchmark data
1.  for every aperture shape $s$ in the search area
2.    then train the filter using $T_o$ and $T_d$ with aperture $s$
3.         $B_r \leftarrow$ apply the filter to $B_d$ with aperture $s$
4.         calculate mse between $B_o$ and $B_r$
```

The algorithm starts by selecting an aperture shape which lies within the search area. The search area is defined as a 19-pixel line, to investigate the possibilities in the direction of the blur. To reduce the number of possible apertures within the search area only symmetrical shapes around the center pixel were taken into account. The selected aperture is used to train the structure controlled filter with a 25-image training set. The second step, is to apply the structure controlled filter on the benchmark images. From the results the mean square error is calculated. The algorithm repeats itself until all aperture shapes are processed. In earlier attempts the filter was trained on the benchmark data only, and applied on the benchmark data. This was done for efficiency reasons, because training the filter on a large number of images is time consuming. The assumption made, is that a good classifier which gives a low mse score for this set, gives a good mse score for any image. This assumption turned out to be invalid. Using this method it became clear that aperture was optimized only for the benchmark images only and not for any arbitrary image. Therefore the filter needs to be trained on a large training set. To find a good tradeoff between the aperture size and the mean square error both are plotted in Figure 4.3. The actual mean square error results and the corresponding aperture shape can be seen in Appendix A. The full search was able to

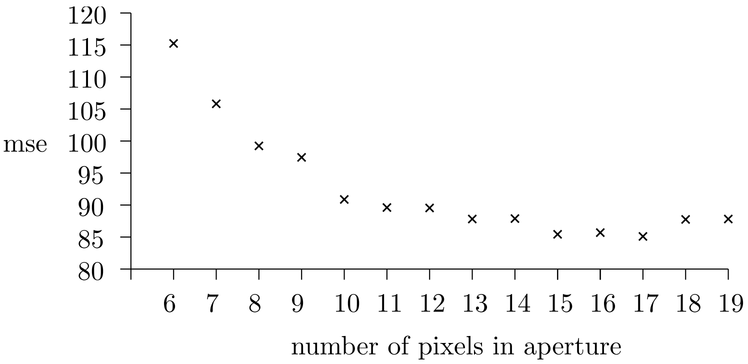


Figure 4.3: This graph shows the mean square error optimal apertures for each aperture size. All symmetrical possible apertures within a 19-pixel line are included in the full search. The aperture consisting of an even number of pixels show a slightly worse result compared to odd size apertures. This is probably caused by the center pixel not part of the aperture shape.

find aperture shapes, which reduced the mse significantly compared to the line aperture. To see whether more mse reduction is possible, the search area was expanded. The search area was expanded in the direction orthogonal to the motion vector. These pixels might encode additional valuable information of the neighborhood. This new search area is depicted in Figure 4.4. To keep the coefficient table within practical size, the number of pixels in each

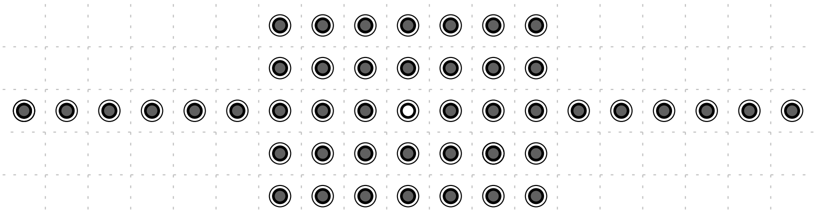


Figure 4.4: The expanded search area used in the full search, to exploit the fact that pixels orthogonal to the motion vector might encode useful information, to further reduce the means square error of degraded images.

aperture is smaller or equal to 19. To keep the number of possible apertures within a practical limit only symmetrical shapes are used. These shapes are symmetric horizontal and vertical to the center pixel. To investigate the tradeoff between the aperture size and the mean square error both are plotted in Figure 4.5. If we compare (Table 4.2) the results from the extended search area with the 19 pixel line search area, we can see that there is improvement for apertures which are larger than 11 pixels. When we look at the corresponding shapes in Appendix B, it can be seen that these shapes use information from the direction orthogonal to the motion vector. From this can conclude that using information from the structure orthogonal to the motion can improve the result. Probably it is possible the find a better aperture shape by enlarging the search area, but the number of different aperture shapes is exponential in the search area size, and therefore are impractical with a full search algorithm. To loosen to constraint of the search area size, some more advanced algorithm could be used. It might be useful to apply *Simulated Annealing*[KGV83] for searching large a area. Simulated annealing is a generic, heuristic optimization algorithm used to find an approximation of the

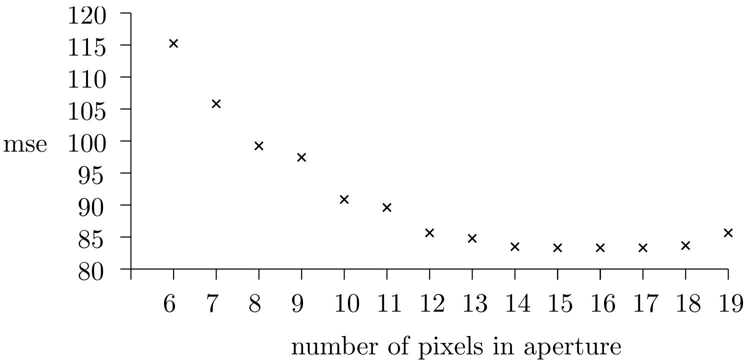


Figure 4.5: This graph shows the mean square error optimal apertures for each aperture size. All horizontally and vertically symmetrical possible apertures within the extended search area are included in the full search.

global minimum of a certain function in a large search area. Since the mean square error is an objective metric that might not in line with the human perception, to illustrate this two images are depicted. Figure 4.6(a) is the result of **houses** enhanced with a 13-pixel line aperture and Figure 4.6(b) is the result of **houses** enhanced by the 13-pixel optimal aperture. The mean square error the former figure is 14% higher, but comparing the figures visually, there is not much difference. From this search we can conclude that the optimal aperture shape further reduces the mean square error, although the perceptive image quality improvement can be limited.

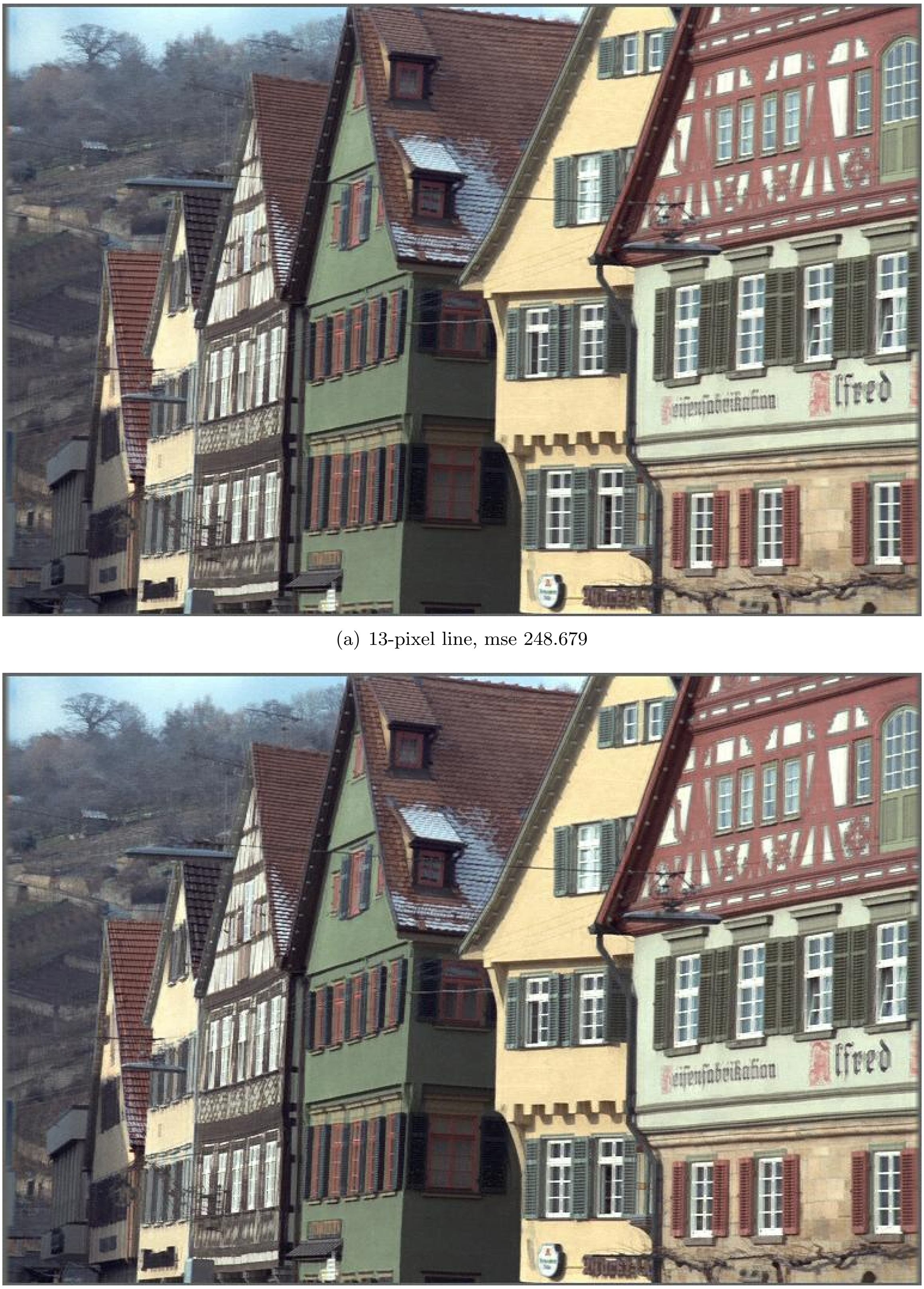
search area	**6**	**7**	**8**	**9**	**10**	**11**	**12**
line	115.123	105.669	99.548	97.392	91.375	89.623	89.561
extended	115.123	105.669	99.548	97.392	91.375	89.623	85.529
	13	**14**	**15**	**16**	**17**	**18**	**19**
line	88.041	88.148	86.733	87.484	86.290	88.107	88.228
extended	84.851	83.861	83.390	83.347	83.404	83.999	85.850

Table 4.2: Mean square error comparison of the 19-pixel line and the extended search area. Up to 11-pixel apertures the optimal apertures are exactly the same. For larger apertures, orthogonal pixels further reduce the mse score.

4.1.2 3 Level Adaptive Dynamic Range Coding

This classification method also called *Local Ternary Pattern* uses three levels to classify each pixel in the aperture. The levels are defined by whether a pixel lies within a certain threshold from the average of the aperture, or above or below this window. This method might give a better classification since it can distinguish between noise and texture. The shortcoming of ADRC is that small deviations are classified the same as very large deviations from the aperture average. For an aperture which contains the pixels $1 \leq x_i \leq n$ we define:

$$C_i = \begin{cases} +1 & \text{if } x_i > av + th, \\ -1 & \text{if } x_i < av - th, \\ 0 & \text{else.} \end{cases} \tag{4.3}$$



(a) 13-pixel line, mse 248.679

(b) 13-pixel optimum, mse 217.789

Figure 4.6: Benchmark image **houses** enhanced by structure controlled filtering, using a 13-pixel line and a 13-pixel optimal aperture. In mse there is a 14% reduction, but there is almost no visible difference.

Where the average luminance value av is defined as:

$$av = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4.4}$$

When the class bits C_i are placed in a fixed order, this results in a ternary representation of the class number. Now the table size scales exponential with a base three. Hence we can only allow very small aperture sizes. A practical aperture size would be 11 pixels. The resulting table has $3^{11} = 177,147$ entries. Since this table is quite large, the training set was increased to 145 images with 1600x1200 resolution. For this classification there are two variables. First how to choose the aperture shape and the second is how to choose the threshold th. Searching this design space is not practically feasible. Therefore we eliminate one variable by choosing an 11-pixel optimal ADRC aperture shape(Appendix A, Figure (f)). We assume here that this aperture will perform well for this method. Since the threshold can have at most 255 values, the mean square optimal value can be calculated easily. From this analysis we can conclude that the threshold $th = 5$ is the optimum for the chosen aperture shape in terms of means square error. The results are depicted in Figure 4.7. If we compare (Table 4.3) this method to ADRC with the same aperture, we see that there is only a small reduction in mse. Whereas the look up table is almost 100 times larger. When we look at the perceptive image

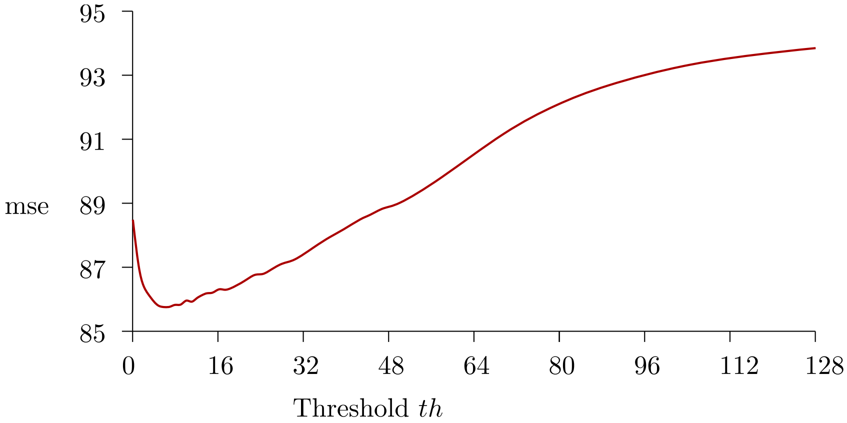


Figure 4.7: The mean square error results of the three level ADRC as a function of the threshold. This might give information on how to choose the threshold value. Threshold $th = 5$ gives the lowest mse score.

Image	**3L-ADRC**	**ADRC**
windows	90.588	91.334
lighthouse	47.492	49.658
farm	55.230	57.112
parrots	19.058	23.492
houses	221.888	224.380
total	86.851	89.195

Table 4.3: Comparison between structure controlled filtering by means of 3L-ADRC and ADRC classification. The mse scores of the 3L-ADRC are reduced, at the cost of the LUT being 100 times larger compared to ADRC

quality we can see that the noise is suppressed. Especially in flat areas for example the sky. This is because the classification is now able the distinguish noise in flat areas from detailed textures, that differ in contrast.

4.1.3 Complexity metric

The shortcoming of ADRC mentioned in 4.1.2 is that for example a flat area with some noise might be classified in the same class as a detailed texture. As a result noise in flat areas might be enhanced. To prevent this we should also classify the complexity of the pixels in the neighborhood, because this discriminates noise from textures. In the paper [SZdH08] four classification methods are presented to fulfill this task:

For a region which contains the pixels $1 \leq x_i \leq n$ we define:

- *Dynamic Range*

  $$DR = \max(x_1, x_2, \ldots, x_n) - \min(x_1, x_2, \ldots, x_n) \tag{4.5}$$

  The dynamic range is simply the difference between the pixel with rank 1 and rank n.

- *Local Entropy*

  $$H = -\sum_{i=1}^{N} P_R(i) \log_2 P_R(i) \tag{4.6}$$

  Variable $P_R(i)$ denotes the probability density function, which can be computed as: $P_R(i) = H_R(i)/N$, where $H_R(i)$ denotes the histogram, which can be obtained by counting how many times a certain luminance value occurs in the image. The luminance range is divided into regions called bins. Variable i indicates the bin index in the histogram, N is the total number of bins and R is a local region around the central pixel of which the entropy is calculated. A region which has a high complexity has a distributed histogram leading to a high entropy value. If the region is flat like blue sky, the histogram contains a few peaks, leading to a low entropy value.

- *Mean Absolute Difference*

  $$MAG = \frac{1}{n-1} \sum_{i=2}^{n} |x_1 - x_i| \tag{4.7}$$

  Where x_1 is the luminance of the center pixel.

- *Standard Deviation*

  $$STD = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - av)^2} \tag{4.8}$$

  Where av is defined in Equation 4.4.

To find a suitable complexity measure we make a small comparison between the four possibilities. In this comparison one bit is reserved for complexity classification in combination with an 13-pixel ADRC optimal aperture. With this complexity bit, we can classify whether a pixel lies above of below a complexity threshold. In Figure 4.8 we can see the mean square error results for a range of threshold values for the four different methods. Notice that there are different scales used on the x-axis. For the complexity metrics: DR, MAG and STD we used the aperture also used in the classification and the filtering. For the Local Entropy this is not very suitable since the number of pixels is not sufficient. Hence we enlarged the window to a 7 x 7 square for this complexity metric. From this graph we conclude that all these methods have very similar mse characteristics. Also when comparing the results of these methods

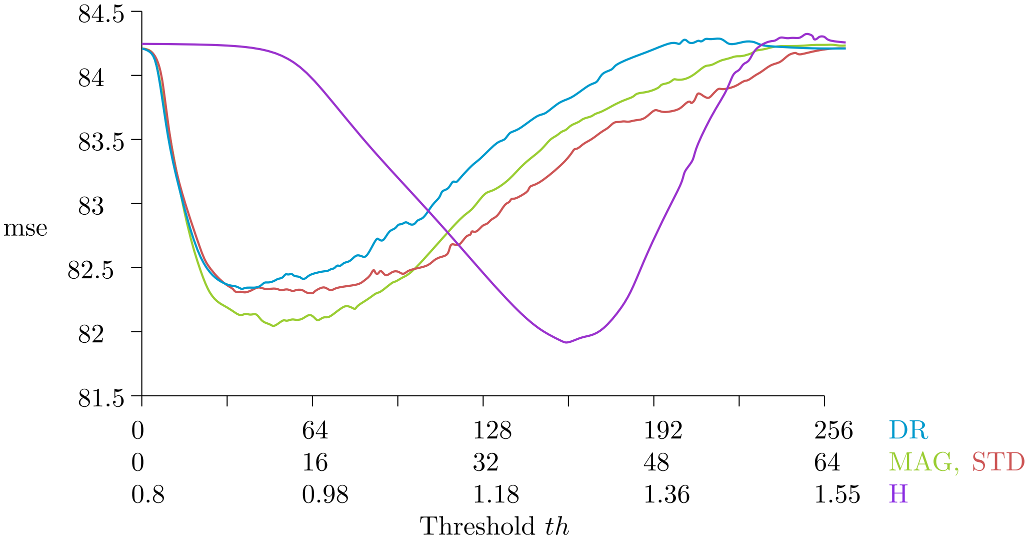
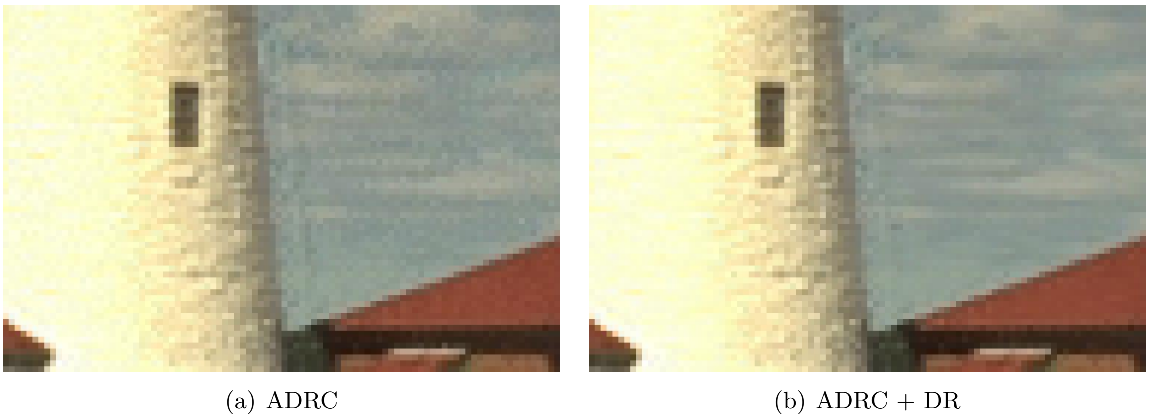


Figure 4.8: The mean square error results of several complexity metrics combined with ADRC as a function of the threshold. Note that there are three different threshold scales. It can be seen that choosing the right threshold is important for the complexity metric to be successful.


(a) ADRC

(b) ADRC + DR

Figure 4.9: Structure controlled filter with and without a dynamic range included in the classification. The noise in the flat areas, such a the rooftop and sky, are suppressed.

visually, there is almost no difference. Since there is no difference, we pick the method which is computationally least expensive. This method is Dynamic Range, because this method does not require divisions or multiplications. In Figure 4.9(b) we see a restored image using a Dynamic Range threshold of 32, whereas in Figure 4.9(a) an image is depicted where no complexity metric is applied. We can see that the noise in flat areas is reduced significantly.

The addition of DR coding to the classification doubles the LUT. This can be prevented by assigning all low complexity pixels to the same class instead of 2^{13} classes. Hence we use the same mean square error optimal filter coefficients for all low complexity pixels. This method is successful because visibility of the blur in low complexity areas is reduced. Therefore these coefficients reduce the noise only in these areas. Using this method the LUT increased by only one entry. A comparison between these two methods is shown in Figure 4.10. From this figure we can see that we can reduce the LUT by this method, with only a small mse difference. Although choosing the threshold value is more critical. For these circumstances the optimal value is $th = 22$.

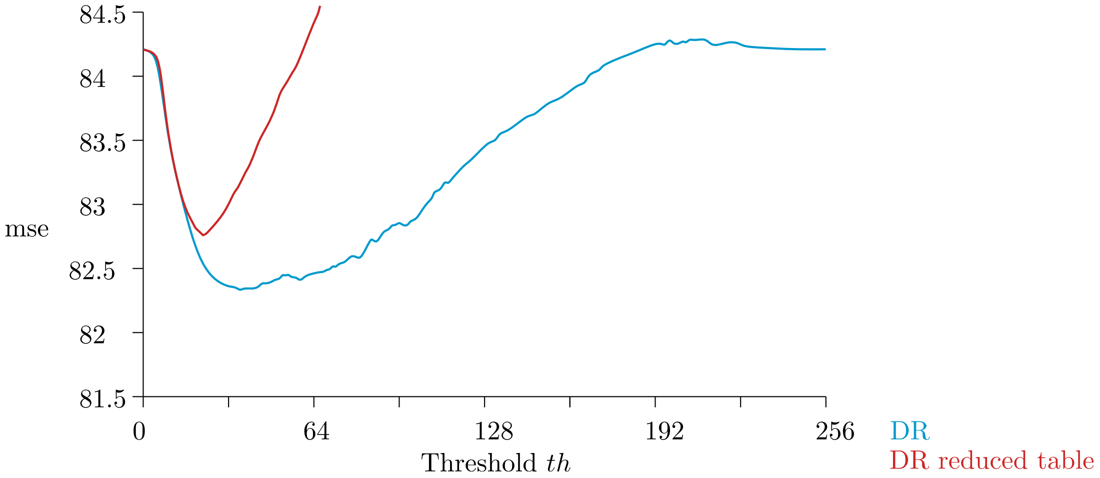


Figure 4.10: Reducing the LUT size while using a complexity metric, is done by using a LMSE filter for all classes which are below the complexity threshold. Note that choosing the threshold value is much more critical by this method.

Instead of using a threshold, one could also use k bits for the Complexity classification. Therefore it is possible to define 2^k Complexity levels. Unfortunately this leads to an increase of LUT size by 2^k. Some brief experiments using 4 levels showed limited improvements in mse. Therefore this has not been further investigated.

4.1.4 Determining training set size

It is not obvious in advance what the required size of the training data is. Therefore a small experiment is conducted to investigate the optimal training set size. At first the filter is trained using only one image. This filter is applied on the benchmark set using a 13-pixel optimal dynamic range aperture. The result is compared by the original benchmark set using mean square error. In the next iteration, a new image is added to the training set. This is repeated until the training set contains 145 images. In Figure 4.11 the mse for all training

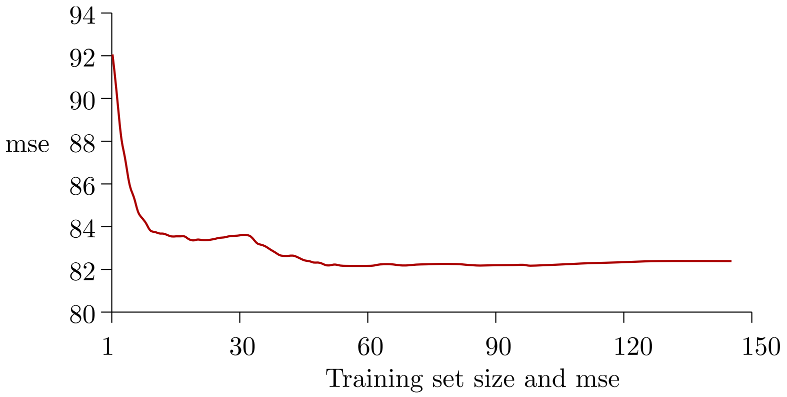


Figure 4.11: Mean square error results of the benchmark set as a function of the number of images in the training set. At some point there are sufficient images in the training set, adding even more does not increase the mean square error.

set sizes is depicted. It can be concluded that at least 60 images are necessary for a sufficient training. We can also see that adding even more images to the training set does not increase the mean square error. We assume that the benchmark set represents arbitrary video material. Under this assumption the structure controlled filter is trained sufficiently, for arbitrary video material.

4.2 Extension to 2D motion blur

In the previous section the structure controlled filter was optimized for camera blur with a fixed speed and a fixed direction. In real life this is not very useful, since there is motion in all directions and all speeds. For the generalization of the simple case we need to estimate the motion in the image sequence. For this the *3-Dimensional Recursive Search* motion estimator[dH93] has been used. 3DRS is a high quality true motion estimator that has been successfully used in consumer products for motion judder reduction. The estimator is able to estimate the true motion of an object, compared to for example a full search block matcher. This quality is important, because the quality of the method immediately depends on the quality of the motion vectors. This because wrong vectors leads to invalid filter operations to corresponding pixels. Suppose the motion speed would be fixed, then the problem could be solved by simply rotating the aperture. When rotating this aperture it is necessary to use interpolation when fetching the pixel values. In the experiments bilinear interpolation is used. To generalize the method to different blur lengths several possibilities are compared. In order to make a comparison, the mean square error metric is used again. The benchmark set is degraded using $5, 10, 15, 20, 25$ and 30 pixel blur length in horizontal direction. Afterwards Gaussian noise is added using a standard deviation (σ) of 5.

4.2.1 Aperture scaling

The first method scales the aperture corresponding to the blur length. Note that the blur length depends on two factors namely, the motion speed and the camera shutter time. Since the aperture has a two-dimensional shape, only scaling is applied in the direction parallel to the blur angle. The filter is trained using data which has been degraded using a fixed blur length. In the first experiment the filter is trained on a blur length of 5. When the algorithm is applied to camera blur where the blur length is larger than 5, the aperture needs up-scaling. When the algorithm is applied to camera blur where the blur length is smaller than 5, the aperture needs down-scaling. The scaling is linear, for example when the blur length is 10, the aperture is scaled using a factor 2. Bilinear Interpolation is used to fetch pixels when using a scaled aperture. The results of this method can be seen in Figure 4.12. In this figure the mse of the degraded benchmark images is depicted(Reference) as well. The aperture used here is a 17 pixel optimal ADRC aperture. For a complexity metric, a Dynamic Range threshold of 32 is used. It can be concluded that there is improvement for all blur lengths. A portion of the improved **houses** figure for blur lengths $5, 10, 15, 20, 25$ and 30 can be seen in Appendix C. As can be expected, image reconstruction becomes more challenging for increase of blur length. This is caused by the number of zeros in the magnitude of the frequency response of the blur kernel. Image details at these frequencies are completely removed, hence they can not be restored.

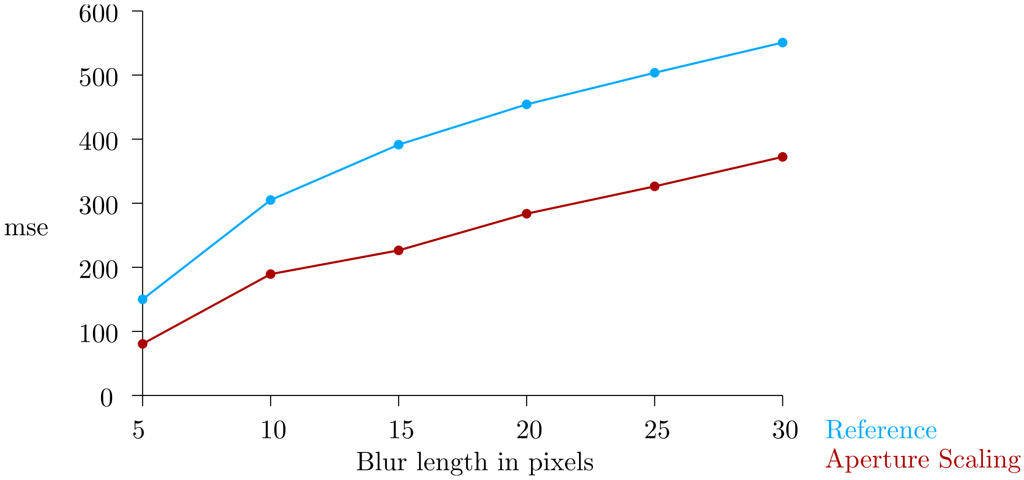


Figure 4.12: Mean square error results of the benchmark set as a function of the blur length. For all blur lengths the mean square error is reduced. The values have been calculated for the dots on the line only.

The above scaling method is based on the training of a training set degraded by 5-pixel blur. It might be possible that this “basis” is not optimal. Training on higher-pixel blur using a scaled aperture might give better results. There is a tradeoff when selecting another training basis. If the filter is trained on a small blur length, the signal to noise ratio is higher than training on a large blur length. But training on a higher blur length gives a more accurate measure of the blur. To find the optimal basis a comparison(Table 4.4) is made based on the mean square error. From the results we can see that on average a low blur length basis performs best for low blur lengths and a high blur length basis performs best for high blur lengths. Instead of scaling it is also possible to add the blur length to the classification. This requires a set of tables, where each table belongs to a certain speed window. If for every speed window, an optimal aperture is used, this method could outperform the scaling method. Although this method would be expensive in terms of LUT size.

	Blur	**5**	**10**	**15**	**20**	**25**	**30**
Basis	5	79.943	191.561	226.485	288.797	327.337	371.215
	10	90.638	224.192	235.602	311.963	338.555	393.159
	15	84.120	192.681	217.565	281.922	318.494	362.555
	20	86.870	204.976	221.448	291.210	322.629	371.154
	25	86.670	194.869	219.465	283.378	319.714	363.453
	30	88.237	202.854	221.775	289.154	321.886	368.703

Table 4.4: Mean square error comparison under two parameters. The vertical parameter is the blur length on which the filter is trained. Horizontally, the blur length of the degraded input material on which the filter is applied.

4.2.2 Extension to motion blur in video

With the identification of the classification method and its extension to arbitrary motion vectors, we apply the method on real video data to reduce the camera blur. For the classification we use a 17 pixel optimal aperture, and a dynamic range threshold of 32. Furthermore the filter is trained on 5 pixel blur degraded data. The filtering process of the system can be described using the following block diagram(Figure 4.13). Note that we could only use video material from which the camera shutter time parameter is known. If this parameter is unknown, which is the typical case in TV reception, the parameter needs to be estimated. The first sequence(Figure 4.14(a))is from the movie Monsters, Inc. This sequence contains a global vertical pan. Therefore the blur angle is directed upwards. We can see that the restored sequence(Figure 4.14(b)) contains significantly less camera blur, as the edges of the digits for example are much sharper. Sequences 4.14(c) and 4.14(e) are recorded with a known shutter speed. Both sequences are shot using a shutter speed of 315 degrees. This high speed combined with a horizontal camera pan, gives long blur lengths. We can see that the edges of the color chart, contain blur. The readability of the blurred text in Figure 4.14(e) is very low. After restoration we can see that both sequences have been improved in terms of sharpness. For example, the edges of the color chart are improved. Also the text has become more readable. Note that the restoration is executed on the luminance only. The chrominance channels are not processed.

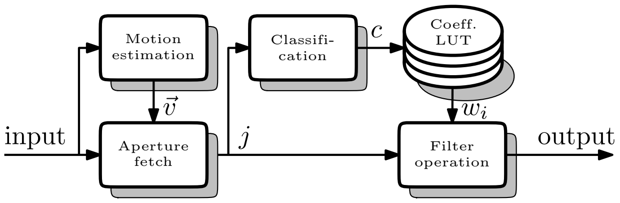
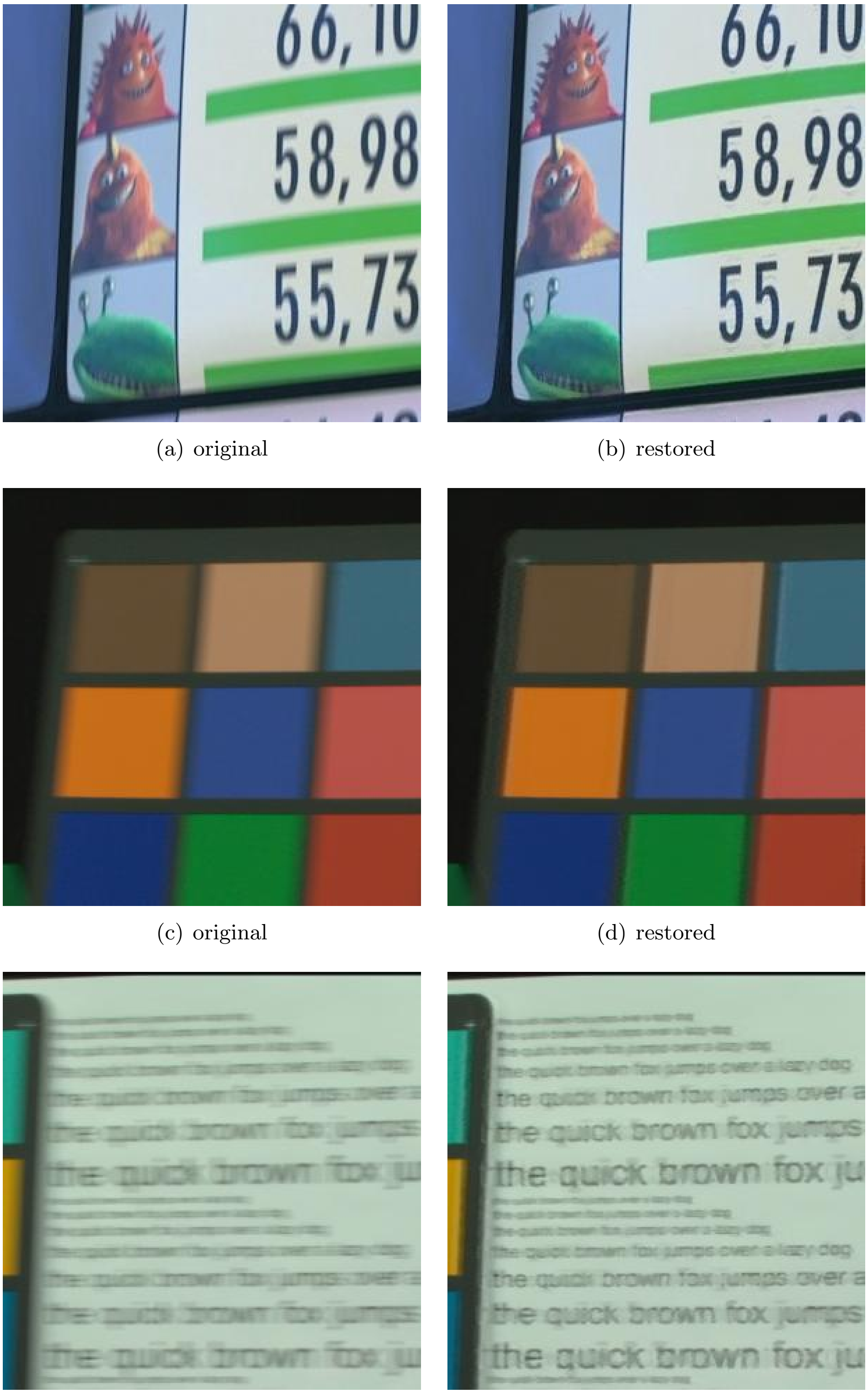


Figure 4.13: Block diagram of Motion Compensated Structure Controlled Filtering. The aperture fetch depends on the angle and length of the motion vector. According to class c resulting from classification of aperture j, coefficients w_i are taken from the LUT. These coefficients are used to apply the filter operation on aperture j.


66,10
58,98
55,73
66,10
58,98
55,73
the quick brown fox jumps over a lazy dog
the quick brown fox jumps over a
the quick brown fox jumps
the quick brown fox ju

(a) original

(b) restored

(c) original

(d) restored

(e) original

(f) restored

Figure 4.14: Camera blur reduction applied on three videos. On the left frames of the original videos and on the right frames of the enhanced video. Note that the reduction is only applied on the luminance component.

4.3 Artifact reduction

When the method is applied on more complex sequences, that have locally different motions or complexity, artifacts arise. Since these artifacts clearly visible, some measures must be taken to reduce these artifacts. The artifacts can be categorized in four groups:

1. Noisy edges caused by transitions between high and low complexity areas.
2. Vector field inconsistency, caused by low complexity areas.
3. Blocking artifacts because of low vector field resolution.
4. Halos which occur at object boundaries.
5. Distortions at image boundaries.

The first artifact can be seen in Figure 4.9(b). It can be seen that on the right side of the tower there is a noisy edge. This noisy edge arises because, the pixels in this area are classified as part of a complex image part. They are classified as such, because the aperture also includes some pixels from the tower itself. Therefore this is a high complexity area. To reduce these noisy edges we could use a weighted dynamic range, where the pixels close to the center pixel get a higher weight. As such we define a weighted dynamic range, containing the pixels $1 \leq x_i \leq n$:

$$y_i = (av - x_i)w_i \tag{4.9}$$

Where w_i is the weight assigned to the according pixel in the aperture.

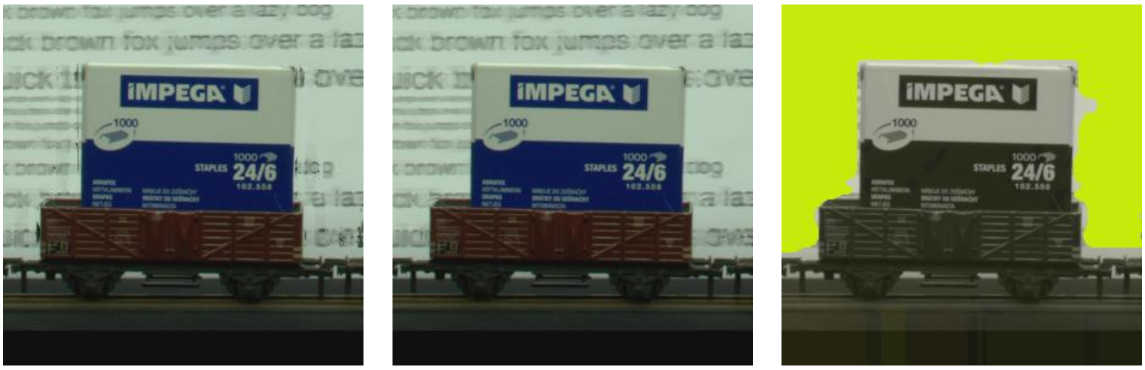
$$WDR = \max(y_1, y_2, \ldots, y_n) - \min(y_1, y_2, \ldots, y_n) \tag{4.10}$$

Where av is defined in Equation 4.4. Experiments with this dynamic range weighting showed that removal of noise is not possible by this method. But the visibility of the edge is reduced, since the noise gradually decreases further away from high complexity object edges.

Second source of artifacts are generated by the motion estimation. The vector field might be inconsistent in low detailed image portions. Because the blocks which are being matched are similar. Fortunately this is not a big problem, since blur in low detailed image portions is not visible. Hence, it not necessary to correct for it.

The third problem is caused by the fact that the 3-DRS motion estimator is a block based motion estimator. This means that every block of pixels is assigned the same motion vector. At the contours of moving objects, these blocks become visible in the restored sequences. To deal with this problem block erosion is applied on the vector field.

Another problem are the edges of moving objects. In Figure 4.15(a), the camera is following the train, therefore the background contains camera blur and the train is sharp. The artifacts we are trying to reduce are the 'halo' artifacts around the contours of the train cargo. If we look at Figure 4.15(c), we can see the motion vectors of the background. These motion vectors are indicated by the yellow labeling. The halo artifacts because of the filter operation that is applied on the pixels just on the boundary of the vector field. The aperture of these pixels, is partly located on the foreground object. Hence pixels from the foreground



(a) original (b) reduced (c) vector field

Figure 4.15: The center image shows the result of Halo reduction applied on the left image. On the right we can see the associated vector field. Halos occur when applying filter operations at the edges of a vector field.

are used in the convolution. To reduce this effect the motion vectors according to the pixels in the aperture should be consistent. The consistency check, is performed by measuring the deviation from the vector associated to the center pixel. The solution used to correct is straightforward. When one or more motion vectors are not consistent, the center pixel luminance value is taken instead of the luminance value(s) from the pixel itself. The result of this operation can be seen in Figure 4.15(b). More advanced reduction algorithms are possible, for example extrapolation, but this is not the main focus of this research.

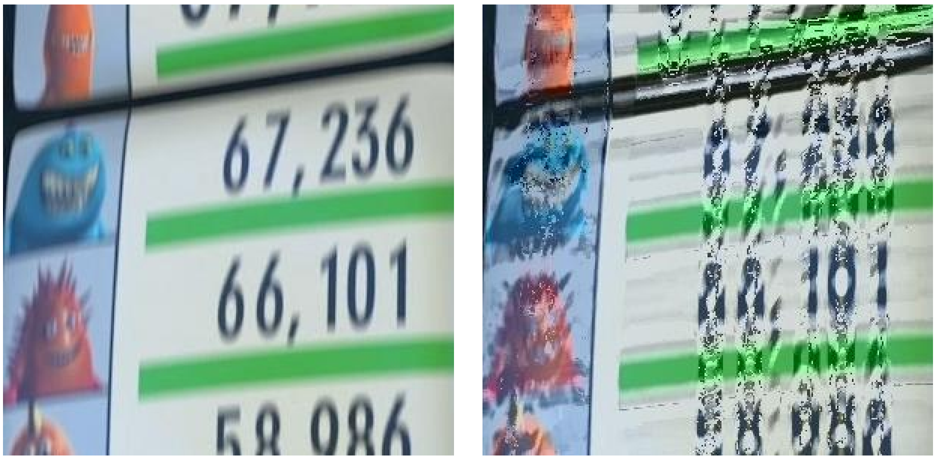
The last kind of artifact appears at the image boundaries, where the aperture might be partly located outside the image boundary. The straightforward solution used to solve the fourth kind of artifact could also be applied here.

4.4 Pre-correction for display blur

In this section it is investigated whether a structure controlled filter can pre-correct a sequence in such a way, that the perceived sequence looks sharp when shown on a sample and hold display. When training this filter, there are two options.

- Degrade training data by a display blur simulator, and use the original data as the reference data.
- Pre-correct the training data by a (computationally expensive) algorithm and use this data as the reference, and use the original data as the degraded set.

We chose to use the first option, since it possible to design a good display blur simulator. The second option is more comprehensive. Suppose we would be able to pre-correct the training data, then we would have solved the problem already. It could be that this algorithm would be very computationally expensive, then structure controlled filtering might give a faster approximation of the pre-correction. For this display blur simulator there are two basic properties which need to be simulated. The first property is the sample and hold effect of the display. Secondly the slow response time of the LC-material. To test a simple proof of concept, only the first property is modeled. Hence we assume that the LC-material has an infinite fast response. Using this simplified model, we can conclude that display blur causes exactly the same blurring as camera blur with a shutter time of 360 degrees. So we apply the camera blur reduction filter, with the shutter speed parameter set to 360, to original (non-blurred) data. When we look at Figure 4.16(b) we can see that the result is not visually pleasing. When eye tracking the objects in this movie, does not improve the result either. After examination of the classification of the pixels and their corresponding filter coefficients, it was found that the results are caused by the training method. Since the filter is trained on simulated display blur only, classes are sufficiently trained which classify blurred image data. In this image data the high frequencies are removed, because of the blurring. When we apply the pre-correction filter, the input data does contain high frequencies. Therefore high frequency image portions are classified as classes, which have not been trained sufficiently. Insufficiently trained classes, have coefficients which can cause distortions. To verify this, we depicted an overlay which encodes the class count as luminance. Where black means a low class count, and white a high class count. The class count associated to class c is defined as the number of pixels in the training data, which are classified as class c. Figure 4.17(a) shows the class count overlay of the pre-corrected image. Compared with the class count overlay(Figure 4.17(b)) of camera blur reduction of the same video, we can see that the pre-correction uses a huge number of insufficiently classes.

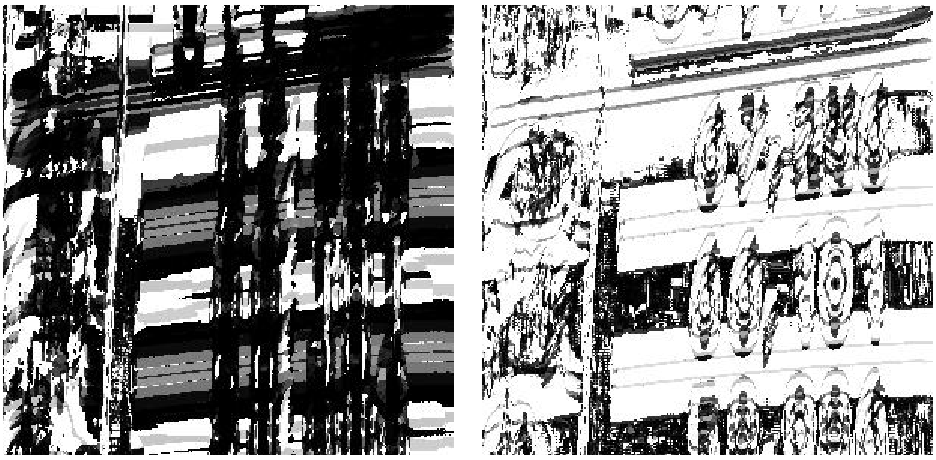


(a) original (b) Pre-corrected

Figure 4.16: On the right, Pre-correction of Monsters, Inc. The distortions are caused by insufficient training of the classes.

4.5 Alternative implementations

Alternative implementations of the Motion Compensated Structure Controlled Filter have been investigated. The benefit of this implementation is that is enables a less complicated hardware implementation. Although it can be expected that the performance of this implementation might be reduced. For such an implementation it is important that the aperture meets the following constraints. The aperture has a fixed shape on a fixed position according to the centerpixel. This has serious implications on the method described in this report.



(a) Class count overlay, Pre-correction

(b) Class count overlay, Camera blur reduction

Figure 4.17: For each pixel, the corresponding class count is encoded as the luminance. The class count associated to class c is defined as the number of pixels in the training data, which are classified as class c.

For example scaling or changing the rotation of the aperture is no longer possible. For this method to be successful we need to find an aperture which performs relatively good for every reasonable blur length in terms of mean square error reduction. It also needs to be suitable for every rotation angle. We propose to use the aperture depicted in Figure 4.18. This aperture might be able to classify and filter different blur lengths.

Since we can not scale or rotate the aperture, we have to use the classification for this problem. Since the number of bits used in the classification is limited, we chose the bit assignment in Table 4.5. The local structure is encoded by ADRC using 17 pixels from the neighborhood. To classify the speed four bits are reserved which can classify each speed into 16 different windows. When we look at the angle only two bits are reserved. Because the aperture is symmetrical in the horizontal and vertical direction, we can already encode four angles. Since the aperture is nearly symmetrical in the two diagonal directions, we can encode eight angles. With the two additional angle bits it is possible to classify four angle regions within 45 degrees. This leads to a 32 level angle adaptive filtering. To benchmark the performance of this method, the benchmark set is used together with the mse metric. The filter is applied on the benchmark set which is degraded by 5,10,15,20,25 and 30 pixel blur.

Property	**Bits**
structure	17
speed	4
angle	2
total	23

Table 4.5: The division of the bits used for classification. The total number of bits is limited by a practical LUT size. Exploiting the symmetry of the aperture, the angle can be classified in 32 different parts.

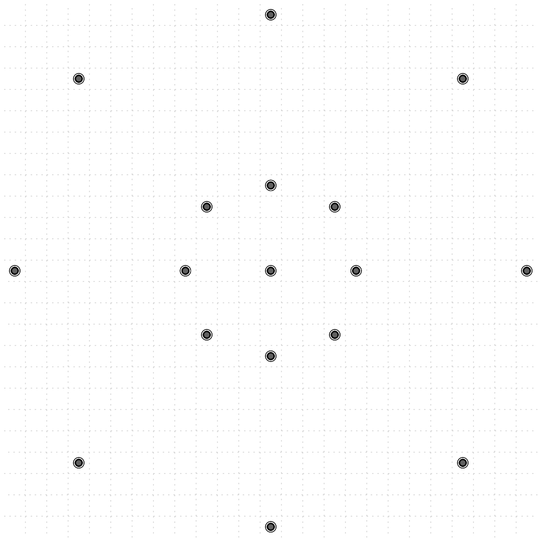


Figure 4.18: Proposal for a fixed aperture shape, which might be invariant to different blur lengths and rotations. The symmetry of the shape is important to reduce the LUT size, since it makes rotations and mirrorings of the coefficients possible.

This test is conducted under a 0,15,30 and a 45 degree angle. The results from Figure 4.19 are compared to the degraded benchmark set (reference) and the best result using aperture scaling, described in Section 4.2.1. The mse scores show only a minor improvement is obtained. The largest improvement occurs at 20 pixel length for the 0 and 45 degree angle. It shows that the aperture dependents on the blur length. When we analyze the effect on the angle of the aperture, we can see that 15 and 30 degree performs worst. This is likely caused by the fact that the pixels in the aperture are misaligned to the blur direction. This alternative implementation, although more efficient in terms of implementation, does not yield significant improvement. The huge design freedom of choosing a filter kernel and the lack of design guidelines, make this approach difficult.

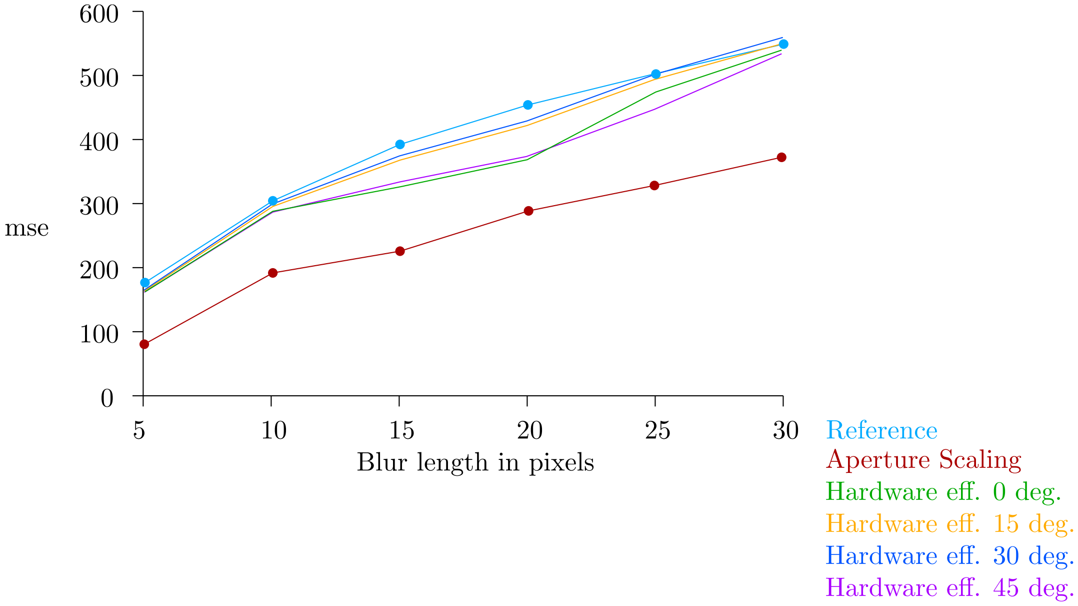


Figure 4.19: The mean square error result of the method. Results are depicted for six different blur lengths. As a reference the mse results of the unprocessed images and the results of the aperture scaling method are shown. The results of the method using the fixed aperture are drawn for four different angles.

Chapter 5

Subjective Perception Test

When we apply the method on real video data(Figure 4.14) we can see that there is improvement. This improvement is clearly visible when we compare individual frames of the original and the processed sequences. Unfortunately this improvement is less visible when we play the video data. The subjective perception test was executed, to see if the visual difference holds while viewing sequences. The visual difference might be reduced, because viewers attention might not be focussed on image portions that contain camera blur. Second problem is the display blur caused by the display used to show the sequences to the subjects. A third issue which might influence the visual difference, is the human visual system tracking the moving objects. The first section describes the experimental setup of the test. The second section describes the results of the test. Finally the conclusions which can be made from the results, and some additional discussion.

5.1 Experimental setup

In this experiment subjects were shown six different sequences, where the original and the camera blur reduced sequence were shown on two displays side by side, based on a left/right random permutation. An individual frame of these six sequences can be seen in Appendix D.The subjects were asked to judge which sequence was perceived sharper. They were able to choose between three options; Left display, Right display or No difference. For this experiment the algorithm used a 17 pixel optimal aperture, using a dynamic range threshold set to 32. Furthermore, measures were taken to reduce the artifacts described in Section 4.3. The training process of the structure controlled filter was based on a 145-image set. The displays used in this experiment have a 100Hz refresh rate. To reduce the display blur, picture rate conversion was done by a motion compensated upconverter. These displays also consisted of a scanning backlight with an effective duty cycle of 40% of the frame time. A small summary of the sequences can be seen in Table 5.1

The sequences are either film or video. The film sequences were shot at 25 progressive frames per second, therefore it needs a factor 4.0 upconversion. The video sequences were first deinterlaced to obtain 50Hz progressive sequences. Motion compensated upconversion with a factor 2.0 yielded 100 Hz sequences. For all sequences Overdrive was applied to make sure correct luminance levels were obtained at the end of each frame period.

Seq.	V/F	Upconv.	Shut.	Playtime
1	Film	4.0	180°	5.5 sec.
2	Film	4.0	180°	3.1 sec.
3	Film	4.0	180°	4.0 sec.
4	Film	4.0	288°	11.7 sec.
5	Video	2.0	270°	5.4 sec.
6	Video	2.0	315°	4.9 sec.

Table 5.1: The sequences used in the subjective perception test. Depending on whether the material is film or video a different upconversion is necessary. From this material the camera shutter angle is known, this a crucial variable for estimating the blur length.

5.2 Results

The experiment was conducted on 16 subjects which lead to the result in Table 5.2. In this table each cell consists the number preferences which the participants gave to that cell. The green color depicts the enhanced sequence, and the red cell depicts the original sequence. To make a distinction between the participants which might be expert or non-expert viewers, for each participant a score is calculated. The score is based on the answers the person gave. A preference for the processed sequence adds +2 to the score, a preference for the original sequence adds -2 to the score and a ‘no difference’ answer adds -1 to the score. The individual scores can be seen in Table 5.3(a). The distinction is made using a threshold, where the expert viewers have a score which is higher or equal to 2. In this way we get eight experts and eight non-expert. From this classification we can derive separate results for both groups. These will be discussed in the next section.

Seq.	Left	No diff.	Right
1	🟥 1	1	🟩 14
2	🟥 4	4	🟩 8
3	🟩 4	7	🟥 5
4	🟥 7	4	🟩 5
5	🟩 7	4	🟥 5
6	🟩 8	4	🟥 4

Table 5.2: Results of the subjective perception test. The green colored cells represent on which screen the enhanced sequences were shown. The red cells represent on which screen the original sequences were shown.

Person	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Score	6	4	-5	-6	2	6	1	9	6	0	-3	-3	-3	-7	3	6

Table 5.3: For each of the subjects an individual score was assigned. This score could be a metric to separate the experts from the non-experts. Initially the score starts at 0. For each preference for the original sequence a -2 is added to the score, for each preference for the processed sequence answer +2 is added to the score and for a ’no difference’ answer -1 is added to the score.

(a)

Seq.	Left	No diff.	Right
1	0	0	8
2	0	1	7
3	1	6	1
4	1	4	3
5	6	2	0
6	6	1	1

(b)

Seq.	Left	No diff.	Right
1	1	1	6
2	4	3	1
3	3	1	4
4	6	0	2
5	1	2	5
6	2	3	3

Table 5.4: Subjective perception test results for expert(a) and non-expert(b) viewers. The discrimination is based on the individual score. Subjects which are assigned a score greater or equal to two are considered experts.

5.3 Conclusions

When we look at Table 5.2 we notice that the camera blur reduction of the first sequence is perceived sharper by almost all subjects. In this sequence the camera was panning across a detailed area, hence the viewers attention was probably focussed on the moving details and was able to see the enhancement. The second sequence was perceived sharper by twice as many subjects as the original sequence was preferred. Also some subjects found it hard to see any difference. This is probably caused by the camera following a moving object. Therefore the object has a fixed position on the screen and the background is moving. Likely most viewers have been focussed on the static object instead of the moving background, making the difference difficult to see. This effect was much stronger in sequences three and four where the camera followed a much larger object. From the results we can see that there no significant perceived sharpness difference between the improved and the original sequence. Sequence five and six are both sequences where the camera follows a moving train. These sequences differ in shutter speed, whereas the fifth and the sixth sequence have a 270° and a 315° shutter speed respectively. In these sequences the moving background contains highly detailed areas, like text and colored checkerboard patterns. Also the moving train was not located in the center of the sequence. This might be the reason why both sequences have a slightly better preference for the improved sequence.

We conclude from the results that the perceived sharpness also depends on the expertise of the viewer and the sequence we look at (Table 5.4). From the expert viewer results, it is clear that determining the sharpness of sequence one, two, five and six gave a clearly visible sharpness difference. But sequence three and four get a high 'no difference' vote, caused by the low detailed moving background, and the big static foreground.

5.4 Discussion

There were some visible artifacts in the enhanced sequences. Future work might concentrate on how these artifacts influence the perception of sharpness. It might also be interesting how the results would be, if the participants were given some additional information prior to the test. This information could be directions where to look for. Comments from the participants in this experiment learned that sharpness comparison was difficult. This because simultaneous comparison of moving objects is difficult, since it is only possible to focus at one detail at the time. After focussing on a detail, at the other display this detail might be gone or changed in scale, rotation or location. A solution might be to use shorter sequences. Shorter sequences would decrease the delay between the repetition of the detail, this makes comparison more easy. Another option might be to use a split screen where one half displays the original sequence and the other half the improved sequence. Although this method has other disadvantages. Since the content is split, it is not possible to compare exactly the same moving details. Also the transition line between original and processed content might be visible. When using split screen, visibility might influence the object tracking.

Chapter 6

Workflow and Tooling

This chapter describes the hardware and software tools used in this project. All these software tools are developed within Philips Research. A short summary of the software and hardware is described in the first section. The second section describes the complete workflow of the structure controlled filtering method in detail. All the steps and the tools used in these steps are described.

6.1 Tools

- *PFSPD*
  Philips File Standard for Pictorial Data(PFSPD) is a file format which has been used throughout the project. PFSPD has been developed to get efficient and easy file access. This is a file format for uncompressed sequences and supports progressive and interlaced video files. Several color formats are supported such as: YUV 4:4:4, 4:2:2, 4:2:0 and RGB 4:4:4. Pixels can be encoded as 8, 10, 12, 14 & 16 bit per pixel. Furthermore additional components can be added easily. For example adding x and y components of a motion vector field.

- *PV*
  Pfspd View(PV) can display images from a pfspd sequence using Xwindows or MS windows. It provides an easy to use graphical interface to view individual images from a pfspd sequence. Also support several options to influence how the images are displayed.

- *PP*
  Pfspd Player(PP) plays images from a pfspd sequence on the MS windows platform in real-time. Through the command line options it is possible to specify many options. For example display multiple files using horizontal or vertical splits.

- *PTS*
  Pfspd Tool Shop(PTS) provides a variety of tools for creating, manipulating and analyzing PFSPD files. The tool can create many test patterns or synthetic motion sequences. For manipulation there are options like: crop, scale, affine, (de)interlace, and rotation. For image analysis it includes: histogram, statistics, file comparison and error metrics.

- *VIDPROC*
  VIDeo PROCessing software, is a program that is designed to help understand the

video processing algorithms in [dH08]. The software enables processing of various sequences to: estimate motion, resolution upconversion, frame-rate conversion, sharpness enhancement, add noise, noise reduction, fourier transform and popular video file format conversions.

- *VIPLIB*
  VIdeo Processing LIBrary(VIPLIB) is a C-library which contains many video post processing algorithms. Like Color conversion, deinterlacing, image enhancement, motion estimation, interpolation, image scaling and noise reduction.

- *DRC*
  DRC is an implementation of the structure controlled filter. This implementation can be found in the development version of VIPLIB.

- *BIGGRID*
  The BiG Grid project (led by partners NCF, Nikhef and NBIC) aims to set up a grid infrastructure for scientific research. This research infrastructure contains compute clusters, data storage, combined with specific middleware and software to enable research which needs intensive computing power or data storage.

- *VISA*
  VIdeo Sequence Availability is a database which contains video sequences and images of the Philips video- and image processing R&D community. This database aims to provide an easily accessible inventory of the available sequence and image source material, as well as the knowledge we have about it. By storing, maintaining, and making the information available in a centralized way, is assured that the sequences, images, knowledge, and experience, are optimally distributed and remain available even after the people who contributed them have left.

6.2 Workflow

This section describes the steps taken to make motion compensated structure controlled filtering possible.

6.2.1 Obtaining the training set

To generate the training set the following steps have been taken: First the images have to be translated into the PFSPD format, this can be done by:

```
pts translate
```

After this operation, it is very convenient to concatenate all the separate files to a sequence. To do the concatenation use:

```
pts cat
```

The resulting PFSPD file is in the RGB 4:4:4 format. Since we apply the camera blur reduction on the luminance only, we need to convert to the YUV 4:2:2 format. This can by done by:

```
pts convert -yuv422
```

To degrade the PFSPD files with the simulated motion blur, we use: `pts filter`

Afterwards gaussian noise is added tot the sequence using VIDPROC.

6.2.2 Preparing input data

We could apply the algorithm on either real input data or synthetic input data. For synthetic input data we could use images which are manually degraded by a motion blur model. This data can be processed as described in the previous subsection, but afterwards motion vectors need to be added to the file. Using the PFSPD library, a simple programs can we written which can fulfill this task. When using real input data, it is very convenient to use sequences from VISA. These sequences are already in the PFSPD format and useful file information is available. To add true motion vectors to the sequence, the 3DRS motion estimator could be used. This estimator is available in VIPLIB.

6.2.3 Algorithm evaluation

There are two ways to benchmark the results. One could either use objective evaluation or a subjective evaluation. For objective evaluation, the mean square error metric could be used. Note that this is only possible when reference material is available. To execute mean square error comparison use:

```
pts cmp -mse
```

The other option is to do subjective evaluation. Displaying single frames PV could be used, or when analyzing the video in real time is necessary you could use PP.

6.2.4 Searching for optimal apertures

The full search described in this report needs a lot of computing power. Fortunately the search can be parallelized, since there are no data dependencies. Each mean square error evaluation for each of the apertures could be processed independently. To exploit this parallelism the full search was executed on BIGGRID. This computing grid is able to process thousands of processes in parallel.

Chapter 7

Conclusions

From the mathematical analysis of motion blur, some straightforward reduction methods can be derived. The first method uses an Infinite Impulse Response(IIR) filter. This method has a frequency characteristic which is the exact inverse of the motion blur model. Unfortunately this IIR-filter is very sensitive to noise. The second method uses a Finite Impulse Response(FIR) filter. For this filter, the frequency characteristic is an approximation of the ideal inverse blurring frequency characteristic. Although this method is less sensitive to quantization noise because of stability, it suffers from ringing effects caused by the coarse approximation of the frequency characteristic.

Motion Compensated Inverse Filtering(MCIF) uses FIR filtering to reduce camera blur and display blur. Heuristics applied to reduce artifacts caused by frequency characteristic approximation make the method computationally expensive. Motion Compensated Structure Controlled Filtering can reduce the camera blur without the computational complex heuristics for artifact removal used in MCIF. The performance of a motion compensated structure controlled filter depends mainly on the classification.

In classifier design there is a lot of design freedom. Aperture shapes, structure encoding and complexity metrics can improve the mean square error and the subjective sharpness. Objective and subjective evaluations of this method show a successful reduction of the camera blur. Subjective evaluations learned that improvement is immediately visible for individual frames, although the perceived sharpness improvement is reduced when looking at video material. This is caused by viewers attention on image portions which do not contain camera blur, some remaining display blur on the displays used to show the videos and difficult comparison because of the movement in the details.

Pre-correction for display blur is not successful by the structure controlled filtering method, where the filter is trained on display blur simulated training data. Since the filter is trained on simulated display blur only, classes are sufficiently trained which classify blurred image data. In this image data the high frequencies are removed, because of the blurring. When we apply the pre-correction filter, the input data does contain high frequencies. Therefore high frequency image portions are classified as classes, which have not been trained sufficiently. Insufficiently trained classes, have coefficients which can cause distortions.

To support hardware efficiency, alternative implementations have been investigated. The fixed aperture constraint has significant consequences on the LUT size and aperture shape. This alternative implementation, although more efficient in terms of implementation, did not yield significant improvement. The huge design freedom of choosing a filter kernel and the lack of design guidelines, make this approach difficult.

Further research could concentrate on further improvement of the classification, since a good classifier is key in this method. For example the aperture search could be executed on a larger search area. A more advanced implementation of the artifact reduction algorithm, could remove the visible artifacts. This would alienate the question whether artifacts influence the sharpness perception, during a subjective test.

Acknowledgements

Many thanks go to colleagues at Philips Research. Especially to Frank van Heesch who guided me through this master thesis by intensive coaching. The many interesting discussions and technical support were of great help. Also many suggestions on how to improve the quality of this report were very much appreciated. Many thanks goes to Gerard de Haan who created the opportunity for this graduation project. Also the interest Gerard showed, lead to valuable input. The author would also like to thank Wesley Yarde for his research on the estimation of the shutter speed parameter, which is key to successful camera blur reduction. Finally, thanks to all the colleagues who were willing to participate in the subjective perception test.

Bibliography

[BJP07] Erwin B. Bellers, Johan G.W.M. Janssen, and Maurice Penners. Motion compensated frame rate conversion for motion blur reduction. *Digest of SID*, 38:1454–1457, 2007.

[dH93] G. de Haan. True motion estimation with 3-d recursive search block-matching. *IEEE Transactions on Circuits and Systems for Video Technology*, 3:368–388, 1993.

[dH08] G de Haan. *Digital Video, Post Processing.* Royal Philips Electronics, Eindhoven, july 2008 edition, 2008.

[Fis01] N. Fisekovic. Improved motion picture of lcds using a scanning backlight. *IDW'01*, pages 1637–1640, 2001.

[HOP$^+$04] Sunkwan Hong, Jae-Ho Oh, Po-Yun Park, Tea-Sung Kim, and S.S. Kim. Enhancement of motion image quality in lcd. *Digest of SID*, 35:1353–1355, 2004.

[KGV83] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, May 1983.

[KK95] T. Kondo and K. Kawaguchi. Adaptive dynamic range encoding method and apparatus. *US-patent 5,444,487*, 1995.

[KV04] M. A. Klompenhouwer and L. J. Velthoven. Motion blur reduction for liquid crystal displays: Motion compensated inverse filtering. In *Proc. SPIE-IS&T Electronic Imaging, SPIE*, volume 5308, 2004.

[Oku93] H. Okumura. A new low image-lag drive method for large-size lcdtvs. *Journal of the SID*, 1(3):335–339, 1993.

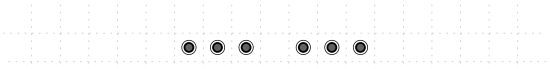
[Ric72] Hadley W. Richardson. Bayesian-based iterative method of image restoration. *Journal of the Optical Society of America*, 62(1):55–59, January 1972.

[SZdH08] Ling Shao, Hui Zhang, and G de Haan. An overview and performance evaluation of classification-based least squares trained filters. *IEEE Transactions on Image Processing*, 17:1772–1782, October 2008.

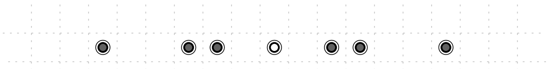
Appendix A

Full search results

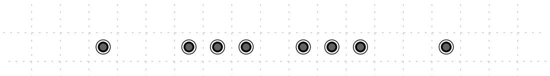
Each figure shows the optimal aperture shape for a fixed number of pixels in the aperture. Also the resulting mean square error for the total benchmark set is depicted.



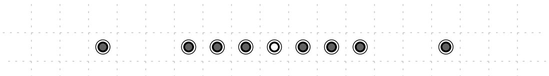
(a) 6 pixel shape, mse 115.123



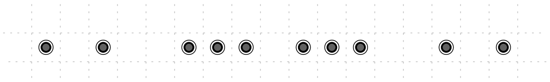
(b) 7 pixel shape, mse 105.699



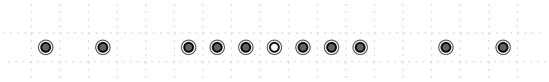
(c) 8 pixel shape, mse 99.548



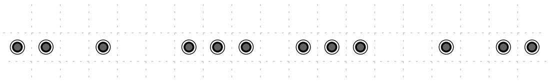
(d) 9 pixel shape, mse 97.392



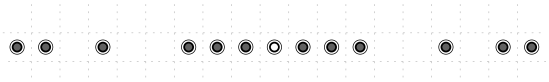
(e) 10 pixel shape, mse 91.375



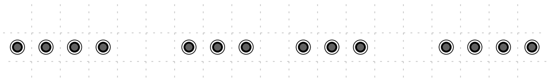
(f) 11 pixel shape, mse 89.623



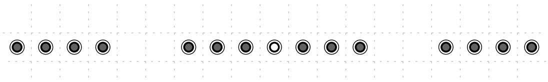
(g) 12 pixel shape, mse 89.561



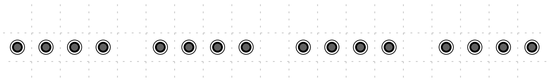
(h) 13 pixel shape, mse 88.041



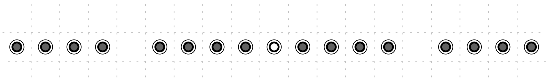
(i) 14 pixel shape, mse 88.148



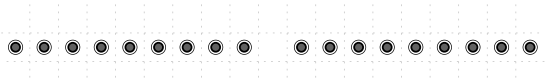
(j) 15 pixel shape, mse 86.733



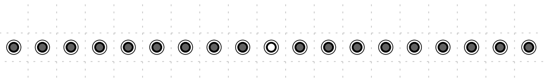
(k) 16 pixel shape, mse 87.484



(l) 17 pixel shape, mse 86.290



(m) 18 pixel shape, mse 88.107

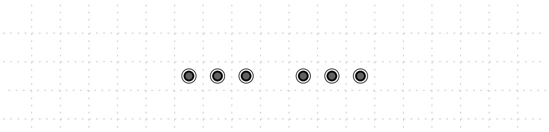


(n) 19 pixel shape, mse 88.228

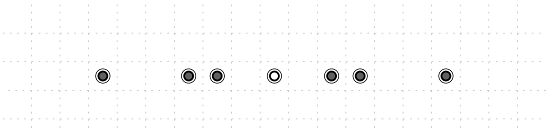
Appendix B

Full search results 2

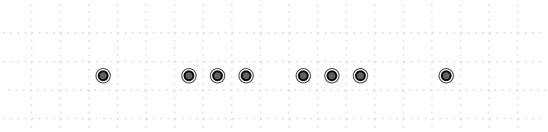
Each figure shows the optimal aperture shape for a fixed number of pixels in the aperture. Also the resulting mean square error for the total benchmark set is depicted.



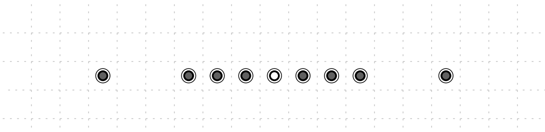
(a) 6 pixel shape, mse 115.123



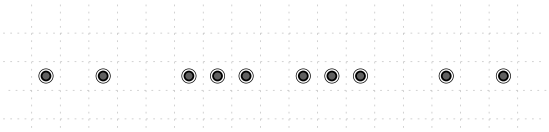
(b) 7 pixel shape, mse 105.699



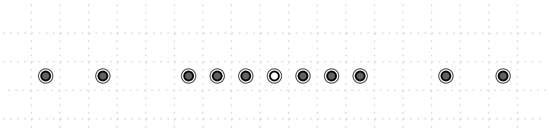
(c) 8 pixel shape, mse 99.548



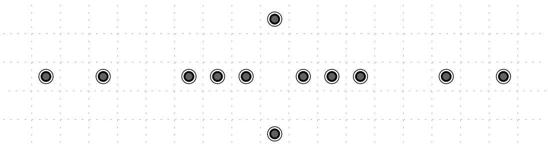
(d) 9 pixel shape, mse 97.392



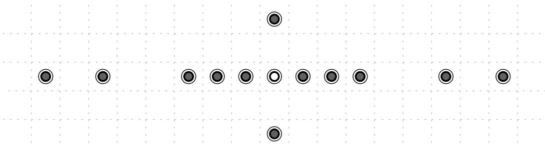
(e) 10 pixel shape, mse 91.375



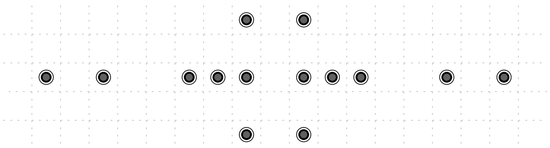
(f) 11 pixel shape, mse 89.623



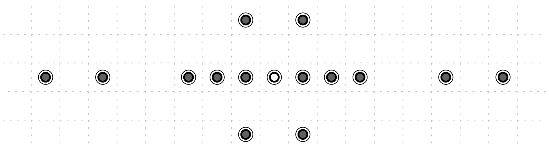
(g) 12 pixel shape, mse 85.529



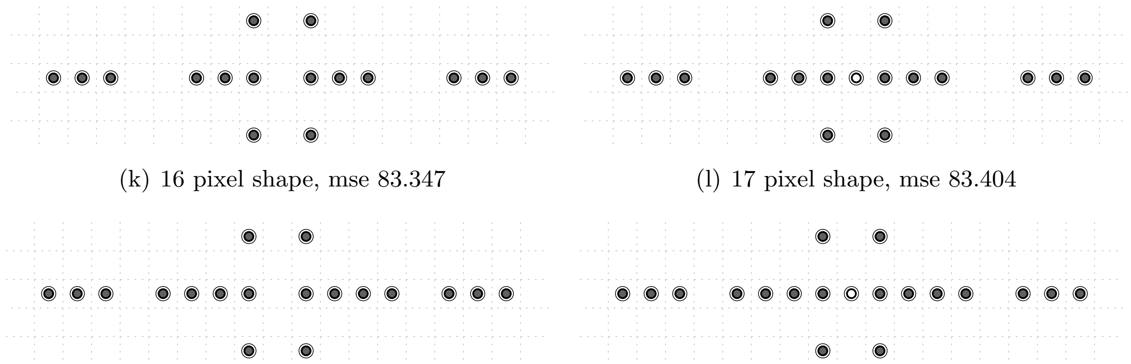
(h) 13 pixel shape, mse 84.851



(i) 14 pixel shape, mse 83.861



(j) 15 pixel shape, mse 83.390



(k) 16 pixel shape, mse 83.347

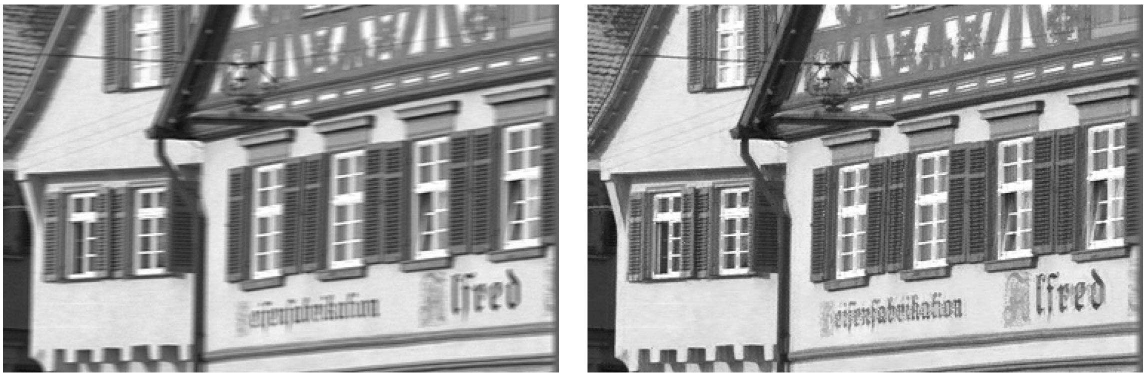
(l) 17 pixel shape, mse 83.404

(m) 18 pixel shape, mse 83.999

(n) 19 pixel shape, mse 85.850

Appendix C

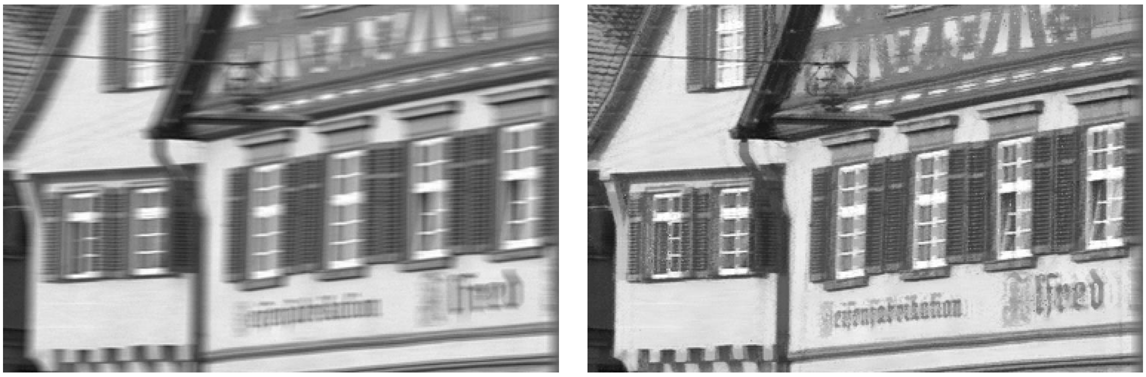
Scaling Benchmark



(a) distorted

(b) restored

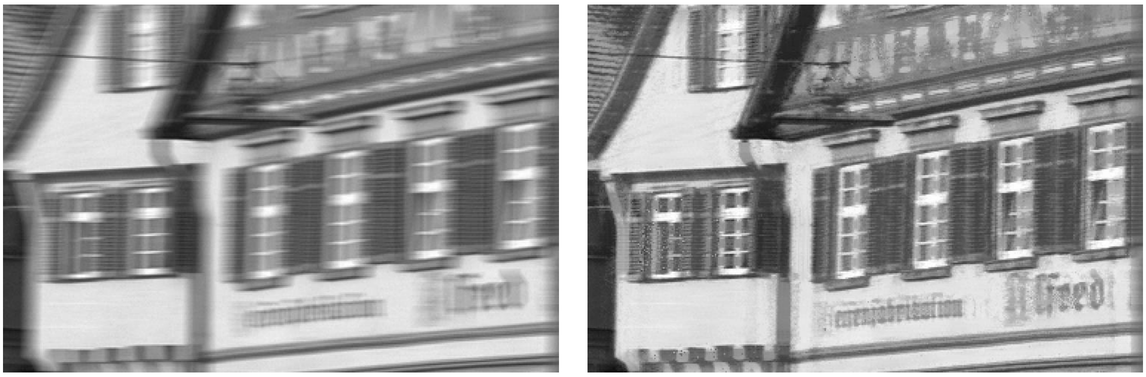
Figure C.1: 5-pixel horizontal blur



(a) distorted

(b) restored

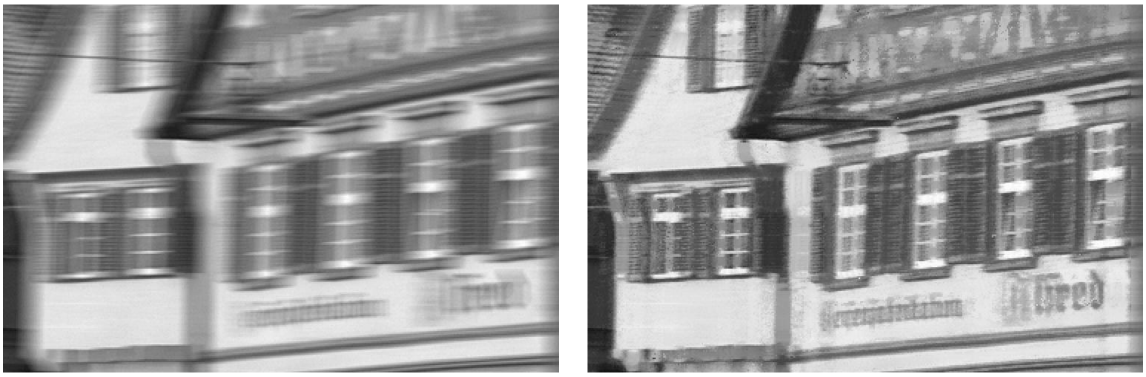
Figure C.2: 10-pixel horizontal blur



(a) distorted

(b) restored

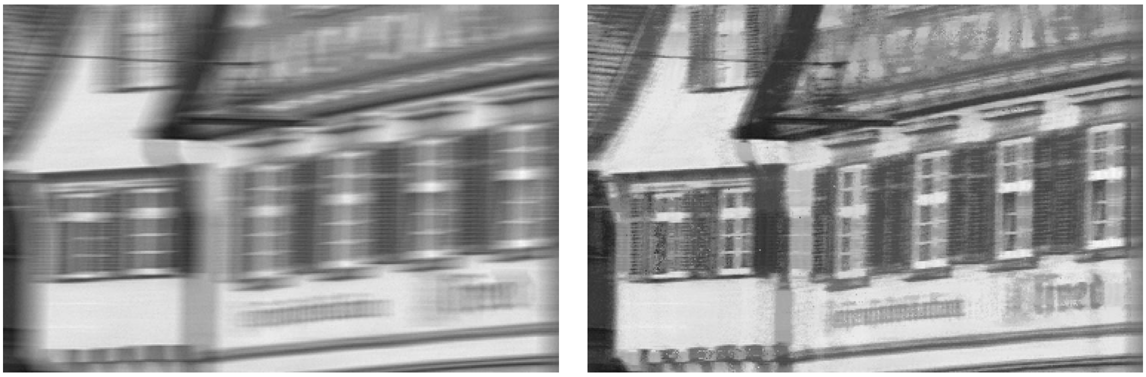
Figure C.3: 15-pixel horizontal blur



(a) distorted

(b) restored

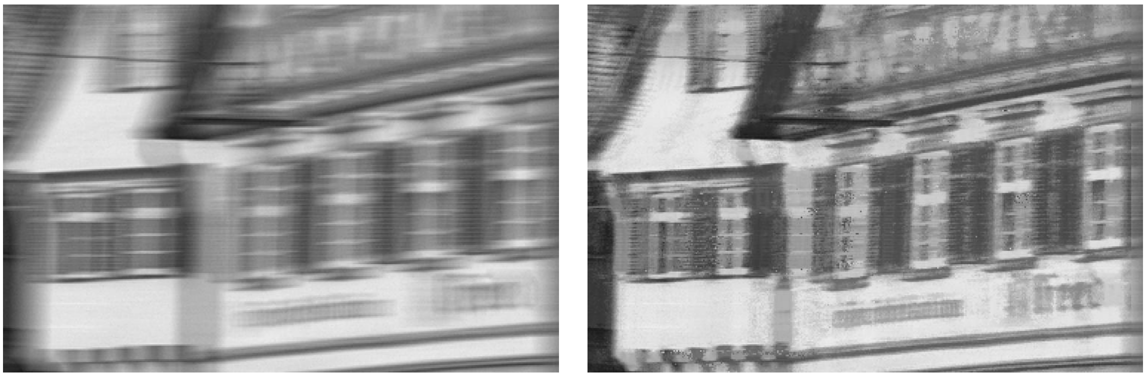
Figure C.4: 20-pixel horizontal blur



(a) distorted

(b) restored

Figure C.5: 25-pixel horizontal blur



(a) distorted

(b) restored

Figure C.6: 30-pixel horizontal blur

Appendix D

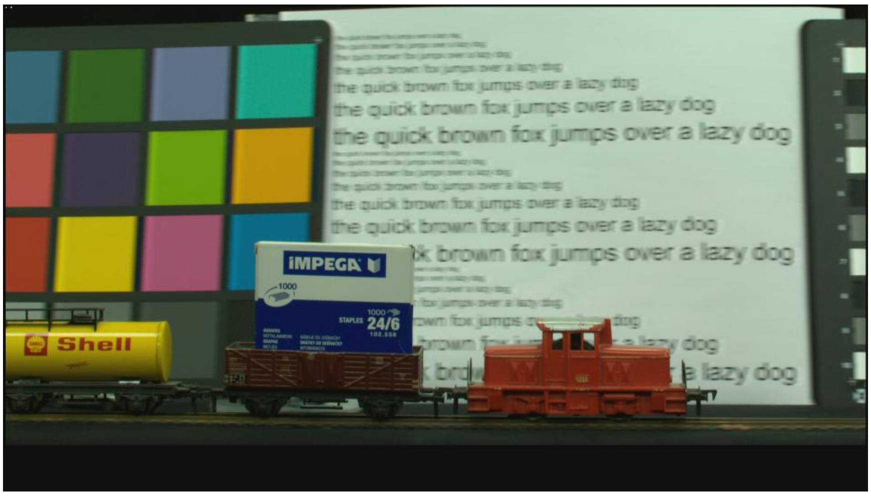
Subjective Perception Test Sequences

(a) sequence 1

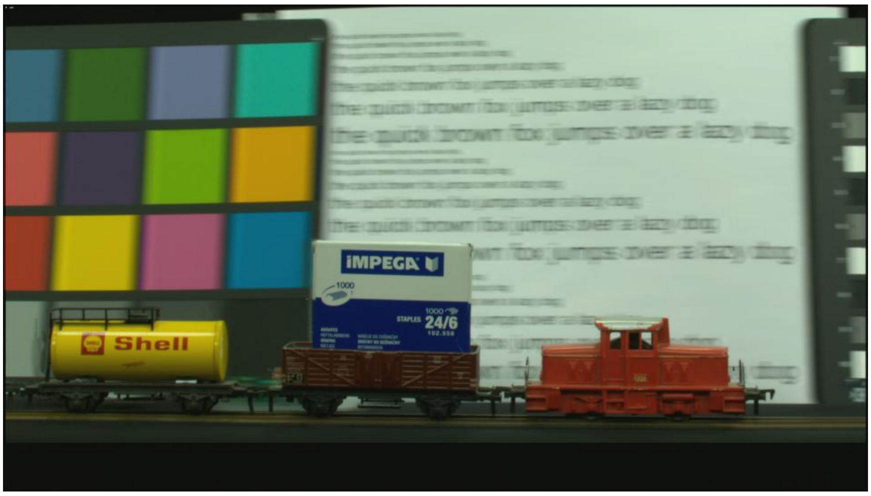
(b) sequence 2

(c) sequence 3

(d) sequence 4



(e) sequence 5



(f) sequence 6

List of Figures

List of Tables