Blur Compensation in Predictive Video Coding

by

Sina Firouzi, B.A.Sc.

A thesis submitted to the

Faculty of Graduate and Postdoctoral Affairs

in partial fulfillment of the requirements for the degree of

Master of Applied Science in Electrical and Computer Engineering

Carleton Institute for Electrical and Computer Engineering

Department of Systems and Computer Engineering

Carleton University

Ottawa, Ontario

September, 2012

© Copyright

Sina Firouzi, 2012

The undersigned hereby recommends to the

Faculty of Graduate Studies and Research

acceptance of the thesis

Blur Compensation in Predictive Video Coding

Submitted by **Sina Firouzi, B.A.Sc.**

In partial fulfillment of the requirements for the degree of

Master of Applied Science in Electrical Engineering

Chris Joslin, Thesis Supervisor

Howard M. Schwartz, Chair,
Department of Systems and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering

Department of Systems and Computer Engineering

Carleton University

September, 2012

Abstract

Video compression has received a very large amount of interest over past decades. One of the key aspects of video compression is predictive coding which consists of motion estimation and motion compensation; however, current estimation and compensation techniques are not robust against the artifacts of blurring in video frames. We show that when blurring appears in video frames, the number of non-matched blocks in the block-matching technique, which is essential to today`s video coding standards is decreased by an average of 20.29%, due to the fact that a large number of the blocks which are corrupted by blur degradation can no longer be matched with the non-blurred reference blocks; which decreases the compression ratio of the video as the non-matched blocks need to be intra-coded. We also show that blur degradation interferes with the current motion compensation techniques due to the corruption of the predicted blocks by the blurring artifacts. In this case, we introduce a blur compensation method, this method detects the type of blur degradation (focal or motion blur), the metrics of the degradation, and introduces compensation methods.

Acknowledgements

I would like to thank my dear parents and two brothers who gave me their love and support.

I would like to thank my supervisor, Professor Chris Joslin, who helped me in every step and pointed me in the right direction.

I would also like to thank my friends and coworkers Parnia Farokhian, Chris Taylor and Omar Hesham who generously helped whenever I had a question.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

BDM: Block Distortion Measure

DCT: Discrete Cosine Transform

DS: Diamond Search

DSA: Digital Subtraction Angiography

FFT: Fast Fourier Transform

FPS: Frames per Second

FSS: Four Step Search

HEXBS: Hexagon-Based Search

MAD: Mean Absolute Difference

MSE: Mean Squared Error

PSNR: Peak Signal to Noise Ratio

SAD: Sum of Absolute Differences

TSS: Three Step Search

Chapter 1

Introduction

We can consider a video as a sequence of frames ordered in time. Some typical frame-rates for videos are 29.97fps and 25fps which are the common frame-rates for television and 24fps for cinema which are sufficient frame rates for the human vision. Another characteristic of a video sequence is the resolution of the frames which determines the number of pixels in the frame.

The size of a video sequence can be very large. As an example, a 60 minute long video with the frame rate of 25 fps, resolution of 4096 x 2160 pixels (2K resolution) and 24 bits/pixel color depth would require 19.1 Terabits of storage; also, transmitting this large amount of data would be very slow and inefficient; therefore, it is necessary to compress the video to reduce the bit-rate. Video compression plays a big role in today's modern life and advances in multimedia; without compression, storing and transmitting video sequences would be very difficult.

Years of study on video coding has resulted in many advancements including the evolution from the early video coding standard, MPEG [49] through several iterations, to the more recent and

superior H.264/AVC [29]. Currently, H.264/AVC has improved video coding in both compression efficiency and flexibility of use compared to older standards such as MPEG-2 [20], H.263 [50, 51], etc. The H.264/AVC standard features translational block-based motion compensation, adjustable quantization, zigzag pattern arrangement and the run-length coding. It also features context-based arithmetic coding which is an efficient entropy coding scheme. H.264/AVC was designed based on conventional block-based video coding used in previous standards with some important differences which include [28, 29]:

- Enhanced motion-prediction capability
- Use of small block-size exact-match transform
- Adaptive in-loop deblocking filter
- Enhanced entropy coding methods

The emerging next generation video compression standard, HEVC (high efficiency video coding) is the successor to H.264/AVC. The HEVC standard aims to increase the compression efficiency by 50% compared to H.264/AVC with the same level of perceptual quality [30]. HEVC is also a block-based scheme which allows for larger block sizes which are suitable for smooth areas as well as smaller and more flexible block sizes which are suitable for detailed areas of the frame with high amounts of texture. Similar to H.264, HEVC uses block-based predictive coding but is more flexible in block size modes [26].

In general, most of video compression generally consists of three key components:

- Perceptual coding
- Entropy coding
- Predictive coding

Perceptual coding and Entropy coding are intra-coding techniques while predictive coding is an inter-coding technique. Techniques employed in perceptual coding and entropy coding are generally shared with the techniques used in image compression.

1.1 Perceptual Coding

Perceptual coding is based on removing the elements which have less significance to the observer's perception. For example, our visual system is less sensitive to higher frequencies; therefore, such components can be removed with little impact on the quality which is perceived by human vision. This is a lossy operation, since some information is lost and an exact pixel match of the original frame cannot be obtained from the compressed frame.

The amount of the data removed using any perceptual coding technique would depend on the application. In applications where very little detail is needed, more data can be removed; therefore, there is a compromise between the compression ratio and the perceptual visual quality, for example the sharpness of the image or the resolution.

In a common DCT based image compression scheme, the image is divided into smaller blocks such as 8x8 blocks. Each block is then transformed to frequency domain using the DCT transform. DCT transform is widely employed in image and video compression; unlike the Fourier transform it does not produce an imaginary part which makes it faster than the Fourier transform. Higher frequency components are then removed using the quantization process [25].

1.2 Entropy Coding

Entropy is a measure of uncertainty in a random set of data. Entropy coding is based on assigning shorter codewords to symbols which are more likely to occur. Entropy compression is a lossless operation, meaning that no information is lost and the original values can be reproduced from the compressed bit stream. Two of the well-known entropy coding schemes used in video coding are Huffman coding and arithmetic coding.

After applying the zigzag pattern [27], predefined tables are used to assign binary codewords based on the amplitude of every component and the run-length of zeros preceding it (for AC components). Codewords of the tables are set in a way that shorter codewords are assigned to values which have higher probabilities of occurrence.

1.3 Predictive Coding in Video Compression

By using the perceptual coding and entropy coding, video frames are compressed independently and can be considered no different to images in this case; however, in a video sequence, the contents of frames are not necessarily independent and in many cases, the contents of frames (especially consecutive frames) are dependent; therefore, videos can be further compressed by taking advantage of these dependencies between frames through the process of predictive coding. A common predictive coding [17] scheme consists of two units : motion estimation and motion compensation [1].

Motion estimation basically determines the matching elements between the frames. For example, these elements can be the pixel values in a block. Figure 1.1 shows an example of the similarities between two frames in a video sequence; four matching blocks between the

frames are displayed which have similar pixel values. Motion estimation and compensation can be performed on both previous and future frames. The transfer functions, which transfer these elements from one frame to another are also determined during the motion estimation process. In the simple case of a translational transfer function between the two blocks, the transfer function is referred to as a motion vector. A motion vector is the displacement in the x and y axes. The H.264/AVC and HEVC are block-based and employ the motion vectors between the blocks.

Figure 1.1: Matching blocks of two frames in a video sequence are shown with red squares[1].

By using the motion compensation process, instead of storing or transmitting the redundant data for every frame, the data is stored/transmitted only once (reference block). The matching data (predicted block) is then reconstructed in other frames with the knowledge of the transfer function (commonly motion vector). Commonly, a copy of the decoder is used in the encoder side which is used to calculate the residual error between the predicted bock and the reconstructed block. The residual error is stored or transmitted to the receiver after

[1] Images taken from "2012" the movie, 2009, Centropolis Entertainment (as Centropolis), Columbia Pictures (presents), The Mark Gordon Company, Farewell Productions, Sony Pictures Home Entertainment.

being perceptual and entropy coded; therefore, the information transmitted to the receiver regarding predictive coding consists of:

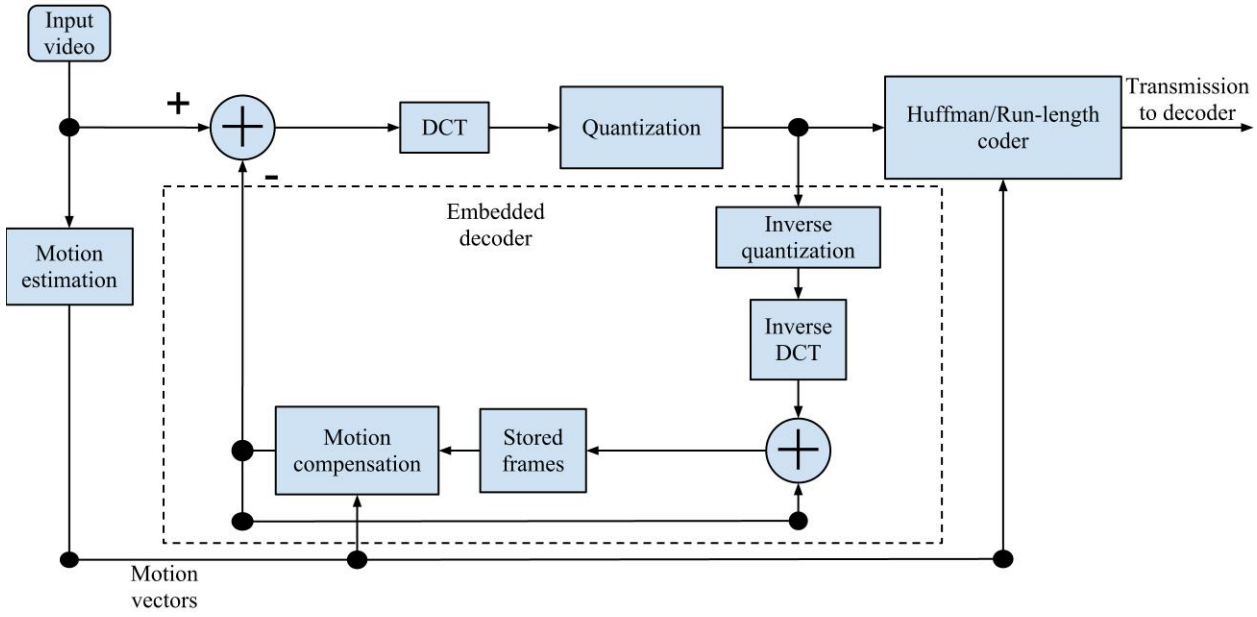
- Reference blocks
- Transfer functions
- Residual errors

On the decoder side, after the predicted blocks are reconstructed from the reference blocks, the residual errors are added to obtain the matching block. The important key in predictive coding is to identify well matching blocks in a way that the residual error is small. If the residual error is large, predictive coding would be inefficient and ineffective since the amount of data needed to encode the residual error would be very large. The encoder/decoder system employed in MPEG video standard is shown in Figure 1.2.

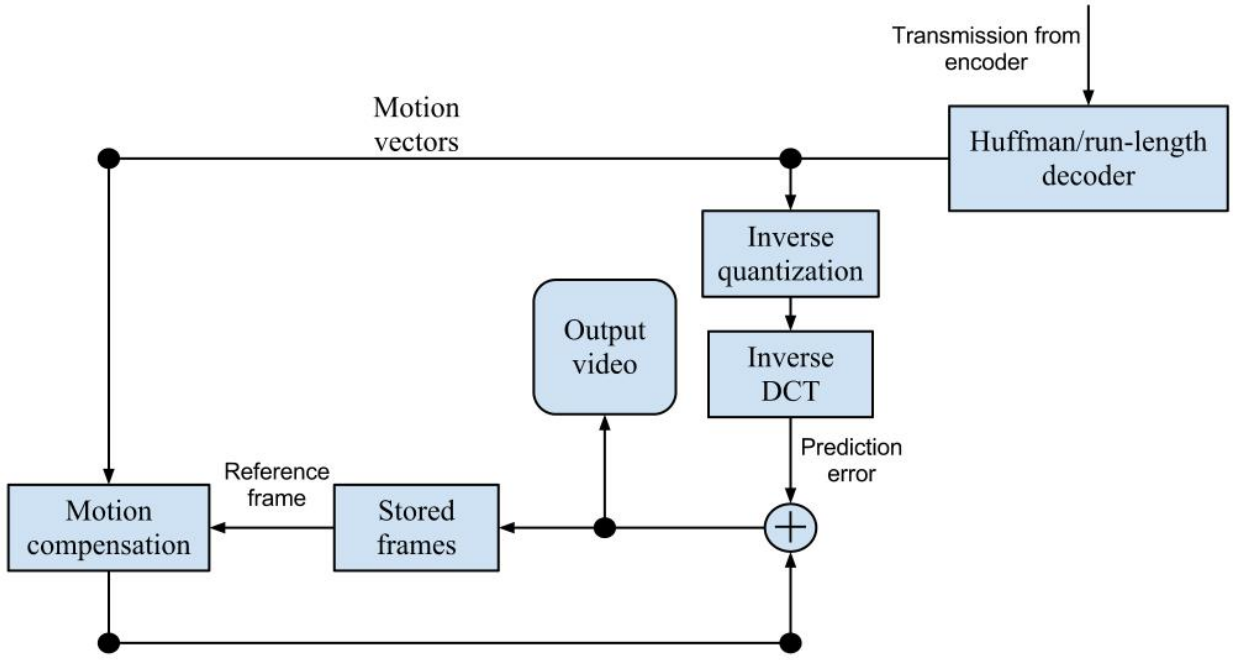
1.4 Effects of Blur Artifacts on Video Compression

Commonly, blur is seen as an undesirable artifact or corruption in imaging and many researches deal with the removal of the blurring effect (deblurring); however, in general, blur is a natural effect which also exists in the human vision system [31]. In animations and computer generated videos, blur is artificially added to enhance the visual experience.

As explained in the previous sections, predictive coding which consists of motion estimation and motion compensation is essential to today's video compression technology; however, presence of blur in a video sequence tremendously interferes with both motion estimation and motion compensation.



(a)



(b)

Figure 1.2: Encoder/decoder system used in the MPEG video standard.

(a) Encoder, (b) Decoder.

Commonly In literature, the performances of motion estimation algorithms are not evaluated for scenarios in which one of the frames/blocks is non-blurred and the other one is blurred, or they are both blurred but with different blurring extents. We discovered that the block-matching technique [32], which is widely used in current video standards such as H.264, is not resilient against blur degradation in the explained scenarios. We applied blur to frames of video sequences and the number of matched blocks between the blurred frames and the non-blurred frames was decreased by an average of 20.29%. We also discovered that the phase correlation technique which is very resilient to noise is fragile in the presence of blur degradation; however, the more apparent and severe effect of blur is observed in motion compensation. Even in the case that motion estimation performs well and matching blocks are detected, motion compensation would be greatly affected. When the camera or the objects inside the frame start to move, non-blurred blocks (before the motion starts) are changed into blurred blocks due to the motion; therefore, the blurred blocks in the first few frames no longer match the non-blurred reference blocks until the motion becomes steady and consecutive frames are blurred in the same way. This is also true when the motion stops and blurred blocks are changed into non-blurred blocks; also when objects go into and out of focus, blocks in consecutive frames no longer match. This increases the residual error between the reference block and the blurred block since the PSNR between them is very low. Such low PSNRs are not satisfactory for motion compensation since the bit-rate required to store or transmit the residual effect is increased to the point that predictive compensation is ineffective; therefore, current video compression techniques are much less effective in the scenarios affected by blur degradation. An example of this effect is shown in Figure 1.3. The predicted block is corrupted and its pixel values no longer match the pixel values of the reference block.

(a) (b)

Figure 1.3: Effect of blur degradation on motion compensation.[2]

(a) Reference frame and reference block, (b) Predicted block corrupted by blurring.

1.5 Motivation and Problem Description

The main goal of this thesis is to present a solution for the inefficiency of current video compression systems in the described scenarios of blur degradation by introducing a novel blur compensation technique. The idea is to manipulate the reference block which is not blurred (or blurred with smaller extent) in a way that would make it similar to the predicted block which is degraded by blur artifacts. This blur compensation process would decrease the residual error between the blurred block and the manipulated reference block and therefore, the bit-rate would be decreased.

[2] Images taken from "2012" the movie, 2009, Centropolis Entertainment (as Centropolis), Columbia Pictures (presents), The Mark Gordon Company, Farewell Productions, Sony Pictures Home Entertainment.

Compensation of blur raises several challenges. There are various types of blur with diverse characteristics, such as blur corruptions caused by motion and blurs caused by objects being out of the focus of the camera. Furthermore, there are several types of motion which include rotational, translational, scaling and zooming which result in diverse effects of motion blur.

This thesis involves four major components:

1. Detection of areas affected by blur, commonly referred to in the literature as blur detection. This also involves determining the type of blur. For example motion blur against out-of-focus blur also referred to as focal blur.
2. Identifying the metrics of blur usually referred to as blur identification. For example, in a case of linear and translational motion blur, the goal of blur identification is to determine the angle and length of the blur. Blur identification is mainly considered as a step to provide information for image de-blurring. In this research however, this information is used for blur generation.
3. Blur generation or reconstruction, also referred to as blur simulation. Blur generation is vital to compensating the blur. Research in blur generation is mainly done in the field of 3D graphics. This thesis required the investigation of artificially generated blur compared against non-artificial blur.
4. Encoding the supplementary information acquired from previous steps and analyzing the results.

In this research, the goal was not only improving the compression bit-rate, but to thoroughly investigate the effects of blur in video sequences. The findings of this research would be valuable in various applications including image registration, image deblurring and blur generation.

1.6 Contributions of This Research

Different contributions are proposed in this research in order to achieve the desired objectives:

a. **A novel blur compensation scheme.** We developed on a distinctive idea of blur compensation which consists of identifying the characteristics of the blur in the blurred blocks and manipulating the non-blurred blocks to make them similar to the blurred blocks. The goal is to reduce the residual error between the blurred and non-blurred blocks in order to make motion compensation more efficient in the presence of blur.

b. **Phase correlation based motion estimation in blurred frames.** Phase correlation is a strong tool in motion estimation; however, it is not resilient to the effects of blur on frames. We modified the phase correlation scheme in order to make it suitable for motion estimation in the presence of blur.

c. **Investigation of blur generation and current blurring models used in literature in comparison to natural blur with application in video coding.** Blur models are mainly used with the purpose of removing blur as a corruptive effect in imagery. They are also used to recreate blur as a visual effect; however, the similarity of natural blur and artificial blur in terms of actual pixels values and PSNR has not been widely investigated. The nature of this research required us to investigate this matter. We worked on ways to improve the PSNR between natural blur and artificial blur.

d. **Improvement in the accuracy of the estimation of the angle of motion blur from a single image.** Blur identification deals with identifying the blur metrics. We made an improvement in techniques used to obtain the angle of motion blur from a single image.

e. **A novel blur-type detection technique.** There are two basic types of blur; motion blur and out-of-focus blur. These two types have distinguishable effects on the frame in the spectrum domain. We developed a DCT based blur type detection technique which identifies an out-of-focus blurred frame/block from a motion blurred frame/block.

1.7 Organization of the Thesis

The present document is organized as follows: Chapter 2 provides a review on current motion estimation techniques. Chapter 3 discusses various aspects of blur analysis in the literature. Chapter 4 introduces a novel blur compensation algorithm. Chapter 5 presents the experimental results of the proposed algorithms. Chapter 7 concludes the document and discusses potential directions for future research.

Chapter 2

Literature Review: Motion Estimation

Motion estimation is the estimation of the displacement between the elements of two or more frames. This displacement can be reflected as a motion vector. Following is a review on motion estimation techniques.

There are various approaches to estimation of the motion between frames. These methods can be classified in different ways; Following is a general classification of main motion estimation and optical flow methods (optical flow is generally used to track the motion of objects in space with application in machine vision):

- Feature/Region matching methods
    - Block matching
- Frequency based methods
    - Phase correlation
- Spatiotemporal gradient based methods used in optical flow

Block matching is one of the well-known and popular methods of motion estimation and is vital to today's video coding technologies. Hundreds of block matching algorithms have been proposed during the past two decades. The good performance in terms of compression ratio and simplicity of block matching has made it more suitable for adaptation in video compression standards [32].

Full search block matching is a robust and widely used method for motion estimation but it is very computationally expensive; therefore, various methods have been proposed for more efficient block matching including various search patterns and the hierarchical block matching which is based on the multi-resolution representation of the frame.

Another popular technique for motion estimation is phase correlation. Phase correlation offers robust and computationally less expensive motion estimation compared to full search block matching. Phase correlation can be used for both local and global motion estimation.

One of the advantages of phase correlation over block matching is that phase correlation can be performed on the whole frame at once and describe the motion in a single motion vector, although it can be used for local motion estimation as well. This makes phase correlation very suitable for global motion estimation and also local motion estimation. Global motion estimation is essentially the estimation of the camera movement independently from the motion of objects inside the frame. In case of global motion estimation, phase correlation is more resilient to local motions compared to regular block matching. This is because in phase correlation the global motion can be reflected as the largest correlation pick.

Great resistance to noise is also one of the big advantages of phase correlation. This makes phase correlation a very suitable technique for noisy frames which is common in satellite and medical images.

Feature/Region matching techniques are the most popular motion estimation techniques in video coding. Among these techniques, block matching and phase correlation are employed in this project. Following is a review on these methods.

2.1 Block Matching Motion Estimation

Block matching is one of the block-based motion estimation techniques. In this technique, the frame is divided into blocks, and then each block in the present frame (reference block) is compared against blocks located in past and/or future frames inside a search area (target blocks). The target block which is the best match and its displacement from the reference block are identified and used in motion compensation [4]. The process of block matching is shown in Figure 2.1.

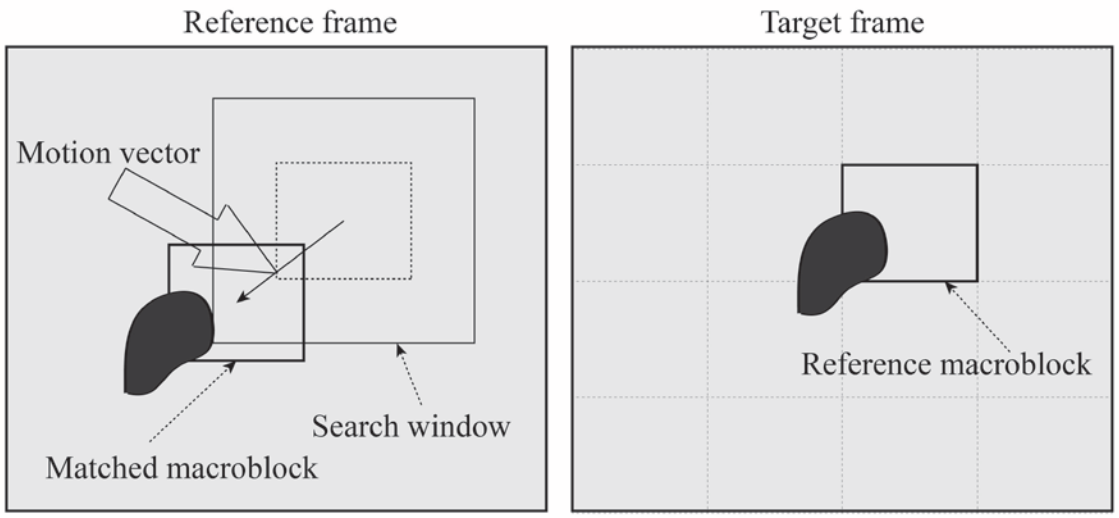


Figure 2.1: Principle of block matching.

Similarity between the target blocks and reference blocks are assessed using a block distortion measure (BDM) [5]. Different methods are used to measure the BDM. Among them are:

- Sum of absolute differences (SAD)
- Mean Squared Error (MSE)
- Mean absolute difference (MAD)
- Sum of squared errors
- Sum of absolute transformed differences

SAD, MAD and MSE are among the most popular and widely used methods in block matching.

MSE is defined as:

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [I(i,j) - K(i,j)]^2 \tag{2.1}$$

The number of pixels in the x and y dimensions are shown by m and n respectively. $I(i,j)$ and $K(i,j)$ are the pixels values of the two frames/blocks, located in the location of (i,j).

MAD is defined as:

$$MAD = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} |I(i,j) - K(i,j)| \tag{2.2}$$

In which $|I(i,j) - K(i,j)|$ is the absolute value of $I(i,j) - K(i,j)$.

A widely used measure of the similarity between the blocks is peak signal to noise ratio (PSNR) which is calculated as:

$$PSNR = 10 * \log_{10}\left(\frac{MAX_I^2}{MSE}\right) \quad (2.3)$$

The maximum pixel value of the frame is shown by MAX_I.

Full search (FS) is the most simple block matching algorithm. In this method, a rectangular search area around the target block is assumed, and the target block is compared against every candidate block located inside the search area. Larger search area might result in better matching blocks but it would also increase the computation load.

Although the FS achieves the best results compared to other search patterns of block matching in terms of PSNR and compression ratio, the algorithm is very computationally expensive since all the candidate blocks inside the search area are compared against the reference block. The followings are some of the search patterns that have been proposed to lower the computational cost of FS:

- Three step search (TSS)
- Four Step Search (FSS)
- Diamond Search (DS)
- Hexagon-based Search Algorithm (HEXBS)

These algorithms increase the speed of block matching by analyzing the BDM a smaller number of candidate blocks. In other words these algorithms compromise between the BDM

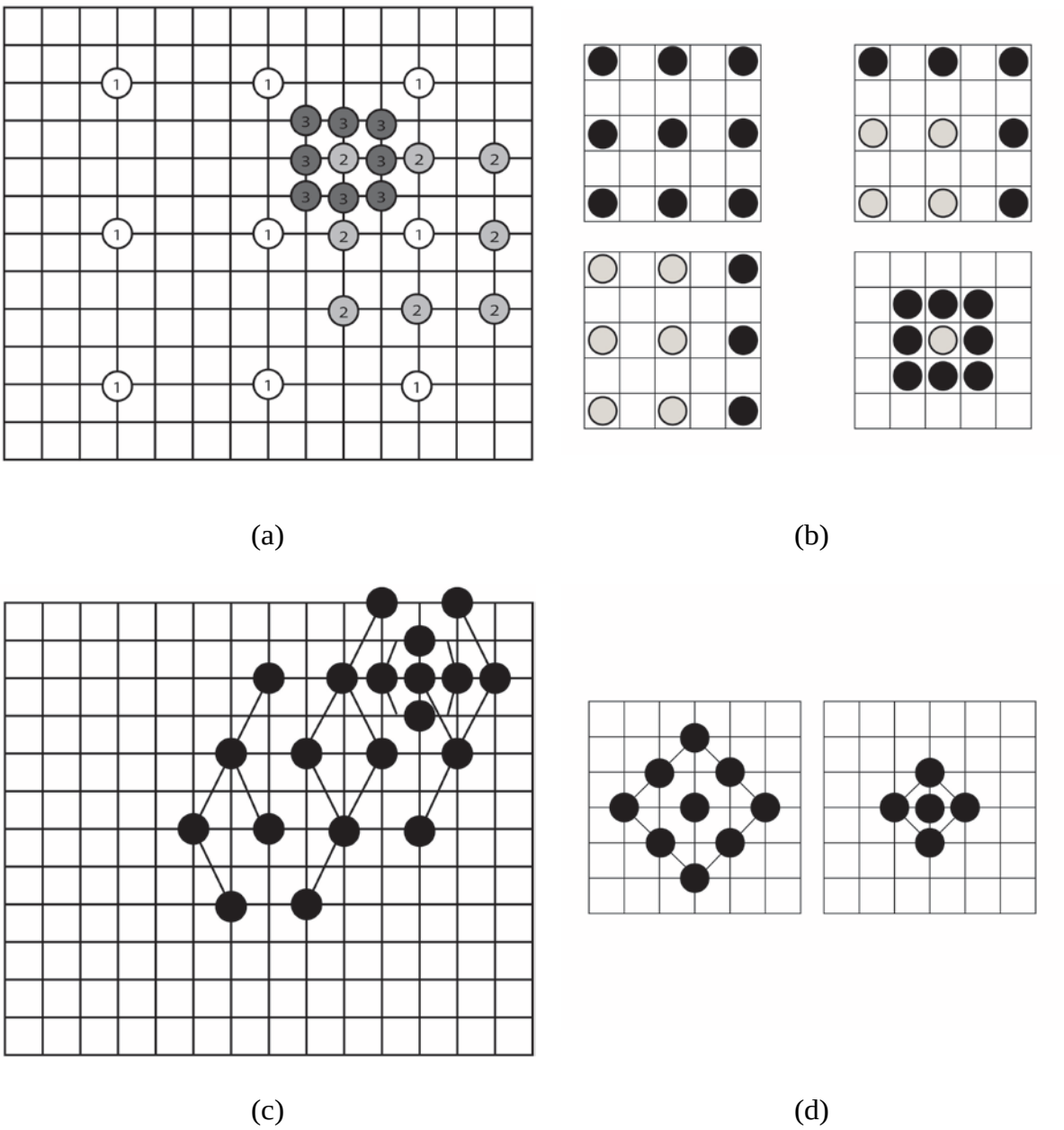


Figure 2.2: Block matching search algorithms.

(a) Three step search pattern, (b) Four step search pattern, (c) Hexagon based search pattern, (d) Diamond search pattern.

and computational cost. These search patterns are shown in Figure 2.2. For example in the diamond search, candidate blocks form a diamond shape.

In this thesis, FS block matching is employed in order to assess the performance of the proposed blur compensation method and the performance of block matching in blurred scenarios in terms of the number of matched blocks.

2.2 Motion Estimation by Phase Correlation

Phase correlation is based on the shift property of the Fourier transform. If a function is shifted in space domain, this shift appears as an exponential function in the spectrum domain:

$$f_{k+1}(x, y) = f_k(x - d_x, y - d_y) \tag{2.4}$$

And:

$$F_k(u, v) = F_{k+1}(u, v).\exp[j2\pi(ud_x + vd_y)] \tag{2.5}$$

When f_k is the k'th frame in the sequence and f_{k+1} is the following frame. $F_k(u, v)$ is the Fourier transform of the function in the spectrum domain.

The continuous 2D Fourier transform is defined as:

$$F(u, v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x, y)\exp(-j2\pi\,(ux + vy))dxdy \tag{2.6}$$

When u and v are the coordinates in the spectral domain. m and n indicate the size of the input image.

The discrete 2D Fourier transform on the scope of a finite and rectangular image is defined as:

$$F[u,v] = \sum_{x=0}^{m-1} \sum_{y=0}^{n-1} f[x,y] \exp\left(-j2\pi(ux+vy)\right) \tag{2.7}$$

The number of pixels in the x and y directions are shown by m and n respectively.

The cross correlation between two functions is defined as:

$$c_{k,k+1} = f_{k+1}(x,y) * f_k(-x,-y) \tag{2.8}$$

When $*$ is the convolution operator. This cross correlation can be taken into the spectrum domain using the Fourier transform as follows:

$$C_{k,k+1} = F_{k+1}(u,v) {F_k}^*(u,v) \tag{2.9}$$

When ${F_k}^*(u,v)$ is the complex conjugate of $F_k(u,v)$, defined as:

$$\left(A.e^{j\theta}\right)^* = A.e^{-j\theta} \tag{2.10}$$

To eliminate the effects of the luminance variations, the cross correlation is normalized as follows:

$$C_{k,k+1}(norm) = \frac{F_{k+1}(u,v)F_k{}^*(u,v)}{|F_{k+1}(u,v)F_k{}^*(u,v)|} \tag{2.11}$$

When $|F_{k+1}(u,v)F_k{}^*(u,v)|$ shows the magnitude and $C_{k,k+1}(norm)$ is the normalized form of $C_{k,k+1}$. The above equation gives the phase of the cross correlation. Now assuming that the second frame is the shifted frame, we would have:

$$C_{k,k+1}(norm) = \frac{F_{k+1}(u,v)F_{k+1}(u,v)\exp\left(j2\pi\left(ud_x + vd_y\right)\right)}{\left|F_{k+1}(u,v)F_{k+1}(u,v)\exp\left(j2\pi\left(ud_x + vd_y\right)\right)\right|} \tag{2.12}$$

But:

$$\left|\exp\left(j2\pi\left(ud_x + vd_y\right)\right)\right| = 1 \tag{2.13}$$

And:

$$|F_{k+1}(u,v)F_{k+1}(u,v)| = F_{k+1}(u,v)F_{k+1}(u,v) \tag{2.14}$$

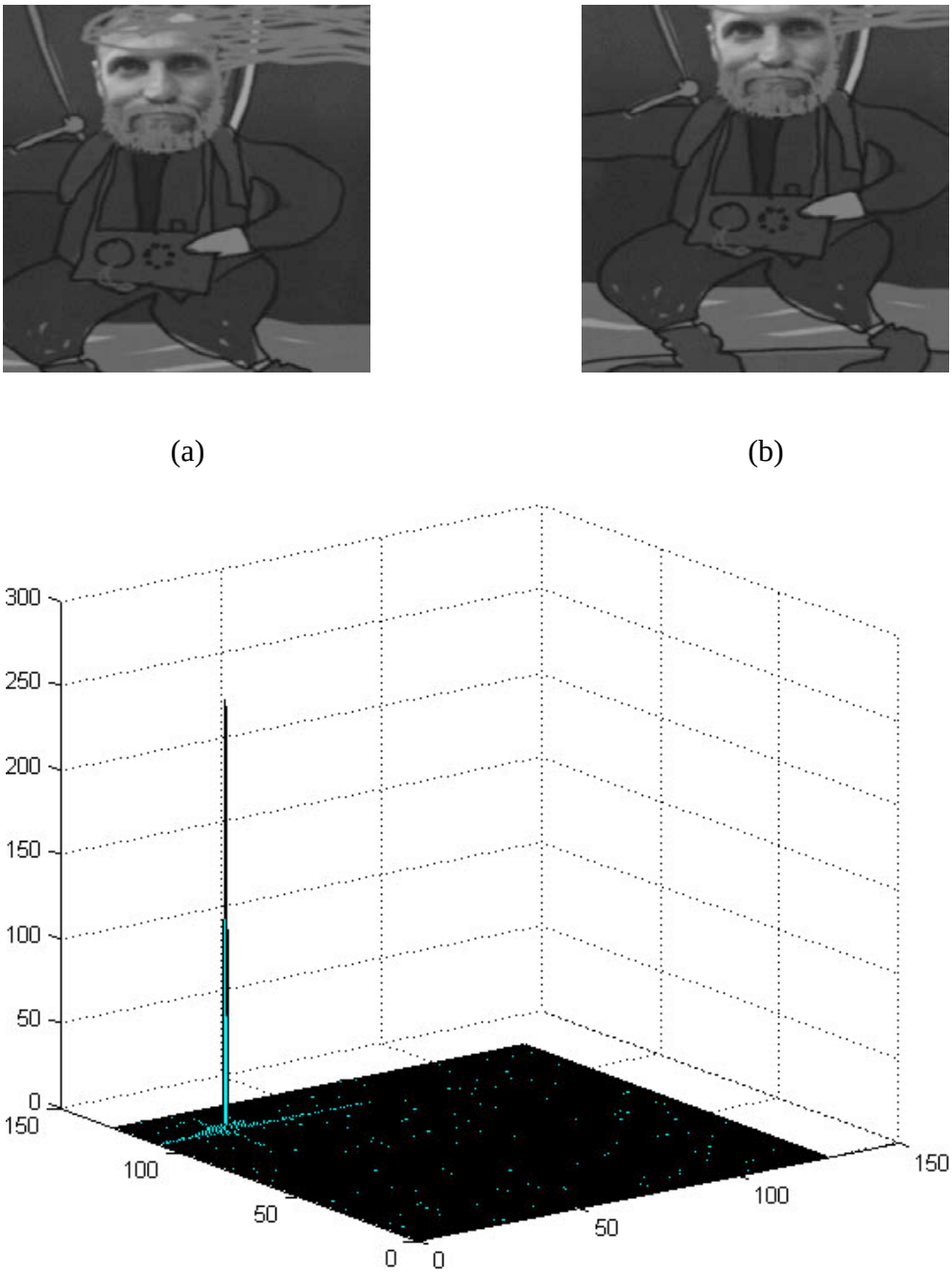


Figure 2.3: Basic Phase correlation.[3]

(a) Reference frame, (b) shifted frame, (c) Peak shows the displacement in the x and y direction.

[3] Images taken from "2012" the movie, 2009, Centropolis Entertainment (as Centropolis), Columbia Pictures (presents), The Mark Gordon Company, Farewell Productions, Sony Pictures Home Entertainment.

Then we would have:

$$C_{k,k+1}(norm) = \varphi(C_{k,k+1}) = \exp\left(-j2\pi\left(ud_x + vd_y\right)\right) \tag{2.15}$$

Now using the inverse Fourier transform we would have:

$$c_{k,k+1}(x, y) = \delta\left(x - d_x, y - d_y\right) \tag{2.16}$$

This Dirac delta function appears as a peak. Location of this peak corresponds to the displacement. Figure 2.3 shows two example frames and the peak resulted from phase correlation which shows the motion between the two frames.

Phase correlation can be used both globally and locally. In the case of local phase correlation, phase correlation is performed on patches of the frames. The smaller the size of the patch gets, the less accurate the result of the phase correlation would become. This is because there are fewer samples and less information in the spectral domain to calculate the accurate motion. One way to improve the phase correlation in smaller patches is to pad the frame with zeros to increase the size of the frame in the spectral domain. We have evaluated phase correlation using various sizes and based on our data we used the minimum size of 64x64 pixels for the phase correlation operations in our research.

Phase correlation is mainly used to detect translational motion; however, it can also be modified to detect rotational movements and scale changes [22].

If the frame is mapped into polar coordinates, rotation would appear as a translational displacement.

$$f_{k+1}(\rho, \theta) = f_k(\rho, \theta - d_\theta) \tag{2.17}$$

And:

$$F_k(u, v) = F_{k+1}(u, v).\exp[j2\pi(u + vd_\theta)] \tag{2.18}$$

Finally we would have:

$$c_{k,k+1}(x, y) = \delta(x, y - d_\theta) \tag{2.19}$$

When ρ and θ are the polar coordinates. As a result, by simply mapping the frame into the polar coordinate, performing the phase correlation would determine the change in θ as if it represents a translational displacement.

Phase correlation can also be modified to detect the changes in scale by mapping the frame into the logarithmic coordinate. Suppose that f_{k+1} is the scaled version of f_k, and the scaling factors in the x and y coordinates are (A, B).

$$f_{k+1}(x, y) = f_k(Ax, By) \tag{2.20}$$

Then in the Fourier domain we would have:

$$F_{k+1}(u,v) = \frac{1}{|AB|} F_k\left(\frac{u}{A}, \frac{v}{B}\right) \tag{2.21}$$

But according to the properties of logarithm we have:

$$\log(a * b) = \log(a) + \log(b) \tag{2.22}$$

As a result, in a logarithmic coordinate we would have:

$$f_{k+1}(\log(x), \log(y)) = f_k(\log(A) + \log(x), \log(B) + \log(y)) \tag{2.23}$$

This is now the matter of finding the translational displacement. Finally we would have:

$$c_{k,k+1}(x,y) = \delta(x + \log(A), y + \log(B)) \tag{2.24}$$

We see that the scaling factors can be obtained from the location of the peak.

Chapter 3

Literature Review: Blur Analysis

Research on the subject of blur analysis generally consists of the topics of de-blurring [35, 36, 37 and 38], blur simulation [22, 33, 34, and 7], blur detection [11, 39 and 40] and blur identification [41, 42 and 43]. The most common topic of blur analysis is image de-blurring, while blur detection and blur identification are also mainly used for the purpose of de-blurring. This chapter is a review on some aspects of these topics in blur analysis.

3.1 Motion Blur

Motion blur is considered as a cause of image degradation. Motion blur is caused by the relative motion between the camera and the captured scene which results in the reduction of the image sharpness [6]. Motion blur is formed when in the time in which the camera shutter is open, the camera is not still relative to the scene being captured and light is integrated in the direction of motion. An example of an image degraded by motion blur is shown in Figure 3.1.

The captured blur would differ depending on many variables including the camera sensor, shutter speed and the camera lens. Longer shutter speed would result in more degradation and lower shutter speed would result in smaller degradation of image. In the ideal case of a shutter with an infinite shutter speed, no amount of motion blur would occur.

(a) (b)

Figure 3.1: Motion blur degradation.

(a) Sharp image, (b) Image degraded by motion blur.

In animations, motion blur is purposely added to improve the visual experience [23, 33, and 34]. This is due to the fact that the human vision is sensitive to blurring and added blur makes animations more similar to real life scenarios. Lack of blur in animations causes unpleasant perceptual effects such as double objects referred to as the doubling effect [7].

In general blur degradation is modeled as:

$$g(x,y) = d(x,y) * f(x,y) + n(x,y) \tag{3.1}$$

When $f(x,y)$ is the original image, $g(x,y)$ is the blurred image, $d(x,y)$ is the blur PSF or blur kernel and $n(x,y)$ is the noise which corrupts the image during blur degradation.

In literature, a translational, linear and space-invariant motion blur [8] is modeled as:

$$d(x,y;l,\theta) = \begin{cases} \frac{1}{l} & \sqrt{x^2+y^2} \leq \frac{l}{2} \quad and \ \frac{x}{y} = -\tan(\theta) \\ 0 & otherwise \end{cases} \tag{3.2}$$

When l is the length of the blur and θ is the angle. The discrete function is estimated as follows for the case of $\theta = 0$.

$d[m,n;l,\theta = 0]$

$$= \begin{cases} \frac{1}{l} & m = 0, |n| \leq floor\left(\frac{l-1}{2}\right) \\ \frac{1}{2l}\left((l-1) - 2 * floor\left(\frac{l-1}{2}\right)\right) & m = 0, |n| = floor\left(\frac{l-1}{2}\right) \\ 0 & otherwise \end{cases} \tag{3.3}$$

For motion blur in other angles, the filter is rotated to the specific angle by applying the following rotation matrix.

$$A_\theta = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \tag{3.4}$$

The blur model is also referred to as the point spread function (PSF), since in a blurred image instead of a single intensity point, the point is recorded by the camera as a spread-out intensity pattern; however, in reality, motion blur cannot be modeled using the simple PSF shown in Equation (3.2) since motion blur in its nature is not space-invariant, meaning that, the image is not affected by the motion blur the same way in every point in space. Modeling a space-variant motion blur is still a largely unsolved problem; furthermore, motion blur can be non-linear, rotational or due to scale change. How these more complex types of motion blur affect this project shall be discussed in future chapters.

In this project, the effect of space-variant blur was disregarded due to the local nature our system. As a result, determining a global model for the whole frame was unnecessary at this stage; also, the focus was mostly on linear motion and other types of blur are discussed as directions for future research.

3.1.1 Effects of Camera Shutter on Blur Formation

Shutter is a device which controls the exposure of the camera film or electronic sensor to light. The shutter speed determines the exposure time between the time the shutter is open until it's closed. It is common to confuse shutter speed and frame rate when speaking of motion blur formation. Frame rate is defined as the number of frames per second; however as described before, shutter speed determines for what amount of time each frame is exposed to light from the time the shutter is opened to the time it is closed. Frame rate and

shutter speed of a video camera can be changed independently from one another. Figure 3.2 shows how a video with a constant frame rate can be recorded by different shutter speeds [13].

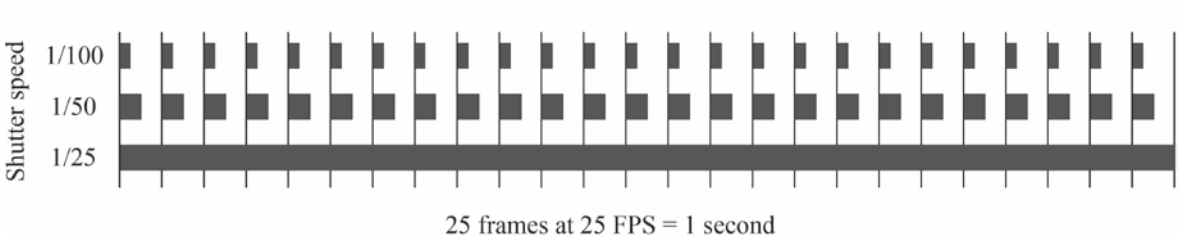


Figure 3.2: Demonstration of various shutter speeds with constant frame rate.

This is significant in the study of motion blur formation since it shows that two similar cameras with equal frame rate and different shutter speeds would record different amounts of the motion blue affect. In other words, the extent of the motion blur depends on the shutter speed and not the frame rate. Also frames with the same distance of motion can have different amount of blurring depending on the shutter speed; therefore, knowledge of the motion vector is not sufficient to estimate the blur length.

3.2 Out-of-focus Blur

If the camera lens is not properly focused, any single intensity point would project into a larger circular area and the image would be degraded [9]; this is referred to as out-of-focus blur or focal blur. An example of an image degraded by out-of-focus blur is shown in Figure

3.3. The degradation would depend on several elements including the aperture size of the camera lens, the aperture shape, and the distance between camera and the object.

(a) (b)

Figure 3.3: A sharp image degraded by out-of-focus blur.

(a) Sharp image, (b) Image degraded by out-of-focus blur.

If the considered wavelengths are relatively small compared to the degree to which the image is defocused, the PSF can be modeled by a uniform intensity distribution [8]:

$$d(x,y;r) = \begin{cases} \frac{1}{\pi r^2} & \sqrt{x^2+y^2} \leq r^2 \\ 0 & otherwise \end{cases} \tag{3.5}$$

When r is the radius of the PSF. The discrete estimation would be:

$$d[m,n;r] = \begin{cases} \frac{1}{k} & \sqrt{m^2+n^2} \leq r^2 \\ 0 & otherwise \end{cases} \tag{3.6}$$

This function is also referred to as the Pillbox function which is shown in Figure 3.4. Since during the blurring process no energy is observed or generated, k is calculated so that:

$$\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} d[m,n] = 1 \tag{3.7}$$

M and N are the number of samples in the m and n directions respectively.

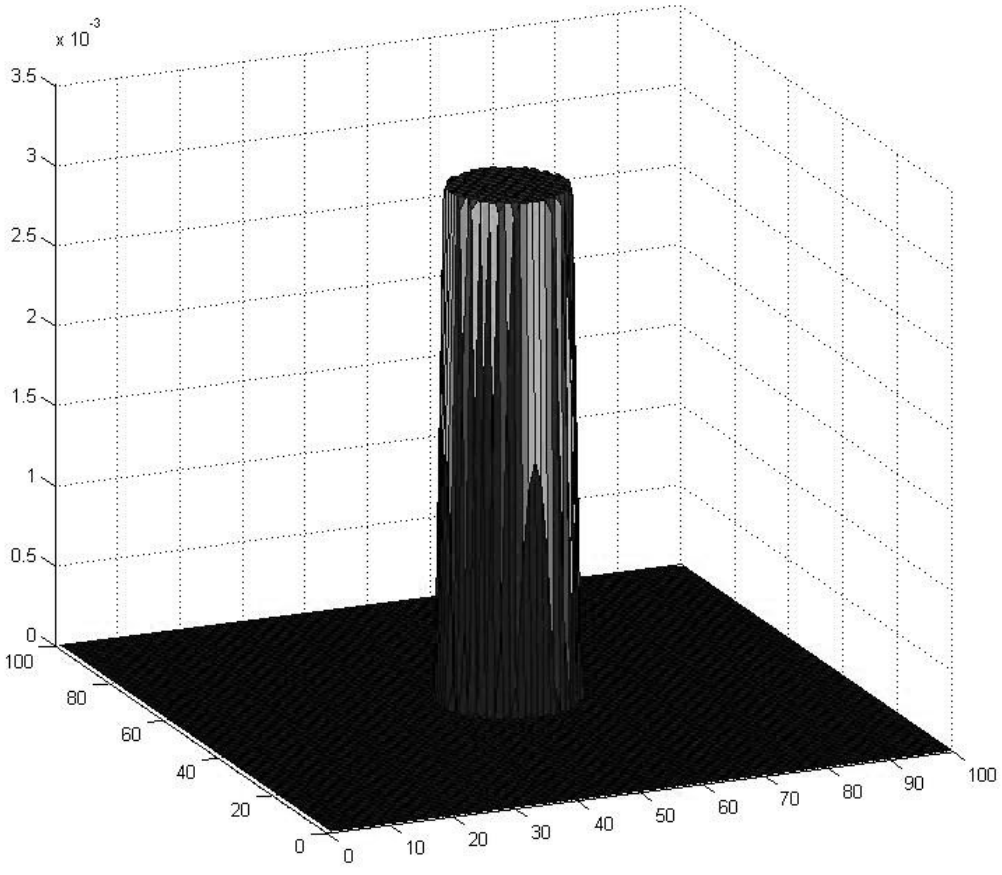


Figure 3.4: The Pillbox model.

Another popular out-of-focus blur model is the Gaussian function which is shown in Figure 3.5 and is modeled as:

$$d(x, y; r) = k * \exp\left(-\left(\frac{x^2 + y^2}{2r^2}\right)\right) \tag{3.8}$$

With the same premise described for the Equation (3.7), k is calculated so that:

$$\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} d(x, y)\ dx\, dy = 1 \tag{3.9}$$

Similarly, for the case of a discrete image we have:

$$d[m, n; r] = k * \exp\left(-\left(\frac{m^2 + n^2}{2r^2}\right)\right) \tag{3.10}$$

k is calculated based on the same premises explained before.

Similar to motion blur, in reality, out-of-focus blur is not space-invariant. How this fact would affect blur compensation shall be explained in future chapters. Figure 3.6 shows an image blurred by the Pillbox and Out-of-focus models.

3.3 Detection of Blur Degradation

Detection of blur includes detection of the presence of blur degradation, and detection of the blur type including the motion blur, out-of-focus blur, rotational blur, and non-linear blur.

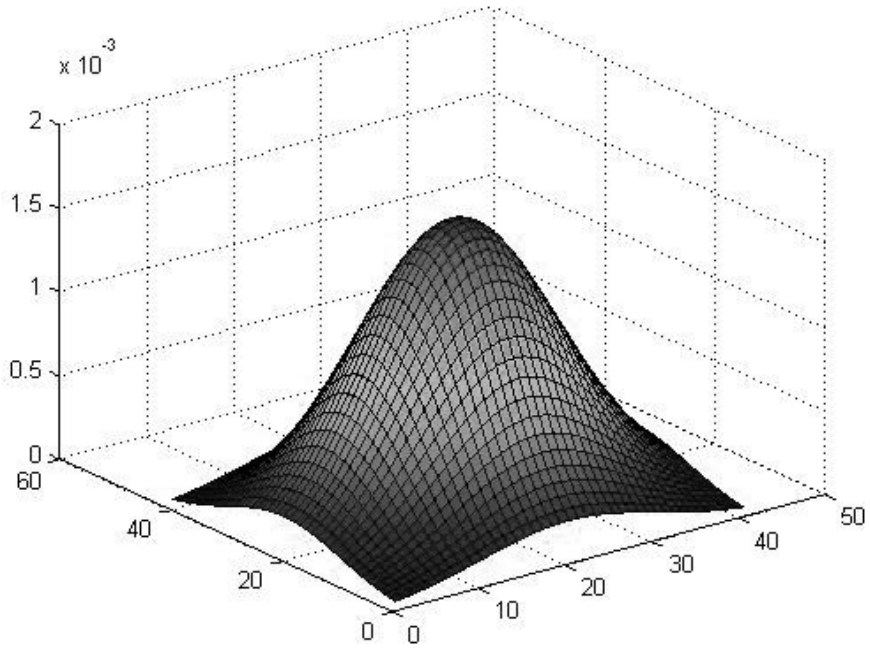


Figure 3.5: The Gaussian model for blur PSF

Motion blur and out-of-focus blur are the two main types of blur; also, it is important to detect the smooth areas of the frame. This is due to the reason that these areas are less affected by the blur degradation and do not reflect the characteristics of the blur degradation as well as the areas with high textures; therefore, using these areas for blur analysis is not reliable and results in erroneous data. Figure 3.7 shows the process of blur detection.

3.3.1 Separating the Blurred Blocks from Smooth Blocks

First challenge in detecting the presence of blur is to distinguish between a smooth image/block and a blurred image/block. In general, blur acts similar to a low pass filter by

removing the high frequency components and results in lower frequency components with larger amplitudes.

(a) (b)

Figure 3.6: An image is degraded by the Gaussian model and the Pillbox model of blur PSF. The two models are very similar perceptually.

(a) Degraded by the Pillbox model, (b) Degraded by the Gaussian model.

This makes the properties of a blurred image very similar to the properties of a smooth image; therefore, differentiating between the two is still a largely unsolved and unaddressed problem in blur analysis.

We first encountered the problem during our experiments on the identification of motion blur parameters which would be described in future sections. Attempting to identify the blur parameters in smooth areas would result in erroneous data.

Figure 3.7: Process of blur detection

The problem is also reported in [10] which resulted in erroneous data in their experiments with blur type detection, a solution was proposed in [11] which is an algorithm based on the standard deviation of the image. Standard deviation is calculated as:

$$\sigma = \sqrt{E[(X-\mu)^2]} = \sqrt{E[X^2] - (E[x])^2} \tag{3.11}$$

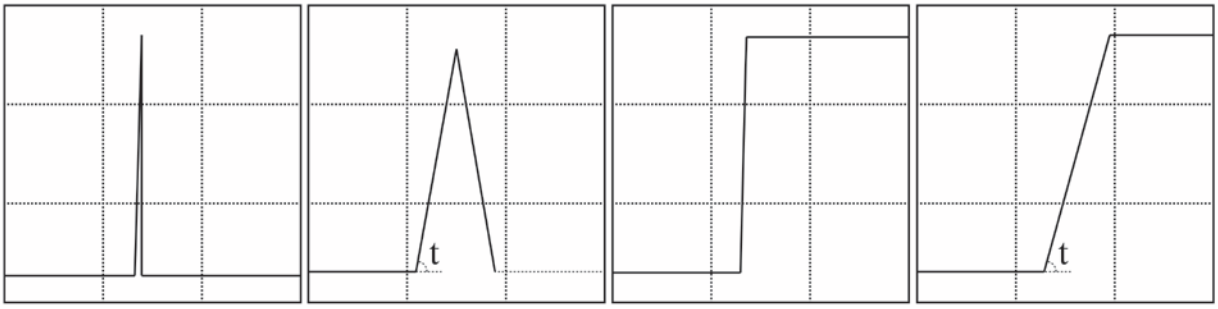
When X is a random variable with mean value of μ.

$$E[X] = \mu \tag{3.12}$$

It is proposed that smooth images have lower values of standard deviation; therefore, images with standard deviations higher than a threshold are marked as sharp and textured while images with values lower than the threshold are marked as smooth images; however, it is assumed that blurred images would result in larger standard deviation values compared to smooth images. This is not a strong argument since blurring behaves similar to a low pass filter and reduces the deviation in the image; therefore, high amounts of blur degradation would decrease the standard deviation value to the point that it is undistinguishable from the standard deviation of a smooth image.

3.3.2 Separating Blurred Blocks from Non-blurred Blocks

Many blur detection algorithms are based on the fact that blurring reduces the sharpness of the edges in an image. A wavelet based algorithm to detect blurred images is proposed in [46]. The proposed algorithm uses the Harr wavelet transform to decompose an image into hierarchical levels with different scales. A wavelet pattern with one level is shown in Figure 3.9. The wavelet transform can be used to measure the type and sharpness of the edges. Edges identifiable by wavelet transform are shown in Figure 3.8. It is proposed that blurred images would have a large number of edges with Gstep-Structure and Roof-structure and a small number of edges with Dirac-structure and Astep-structure; therefore, the blurriness of the image is measured based on the ratio of Gstep-structure and Roof-structure edges to the Astep-structure and Dirac-structure. The extent of blur was measured based on the slope of the Gstep-structure and Roof-structure presented in Figure 3.8 as t.



(a) (b) (c) (d)

Figure 3.8: The four types of edges in an image

(a) Dirac-structure, (b) Roof-structure, (c) Astep-structure, (d) Gstep-structure

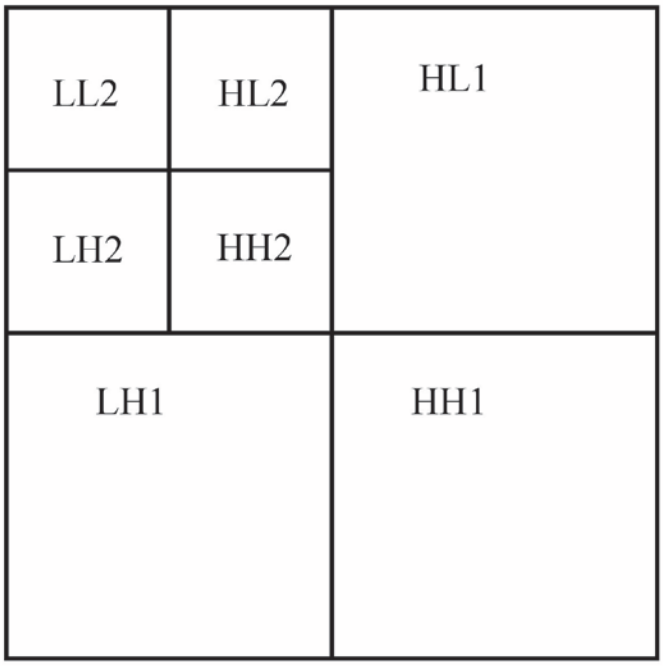


Figure 3.9: The decomposition process used in wavelet. HH represents finer detail and LL represents coarser detail.

In [12] an edge-based blur detection method based on standard deviation in spatial domain is proposed. In this algorithm, the standard deviation of each pixel and its neighboring pixels is calculated, resulting in a standard deviation map (as we refer to), and areas with low values of standard deviation are marked as motion blurred areas. Since blur acts similar to a low frequency filter, it would make the edges smoother than what they should be without the blur degradation; therefore, blurred areas would have lower deviations in spatial pixel values. Standard deviation maps of two images are shown in Figure 3.10. The blurred image is degraded by natural out-of-focus blur and it is seen that it results in lower values in the standard deviation map compared to the sharp image. The total sum of values in the standard deviation map is calculated and

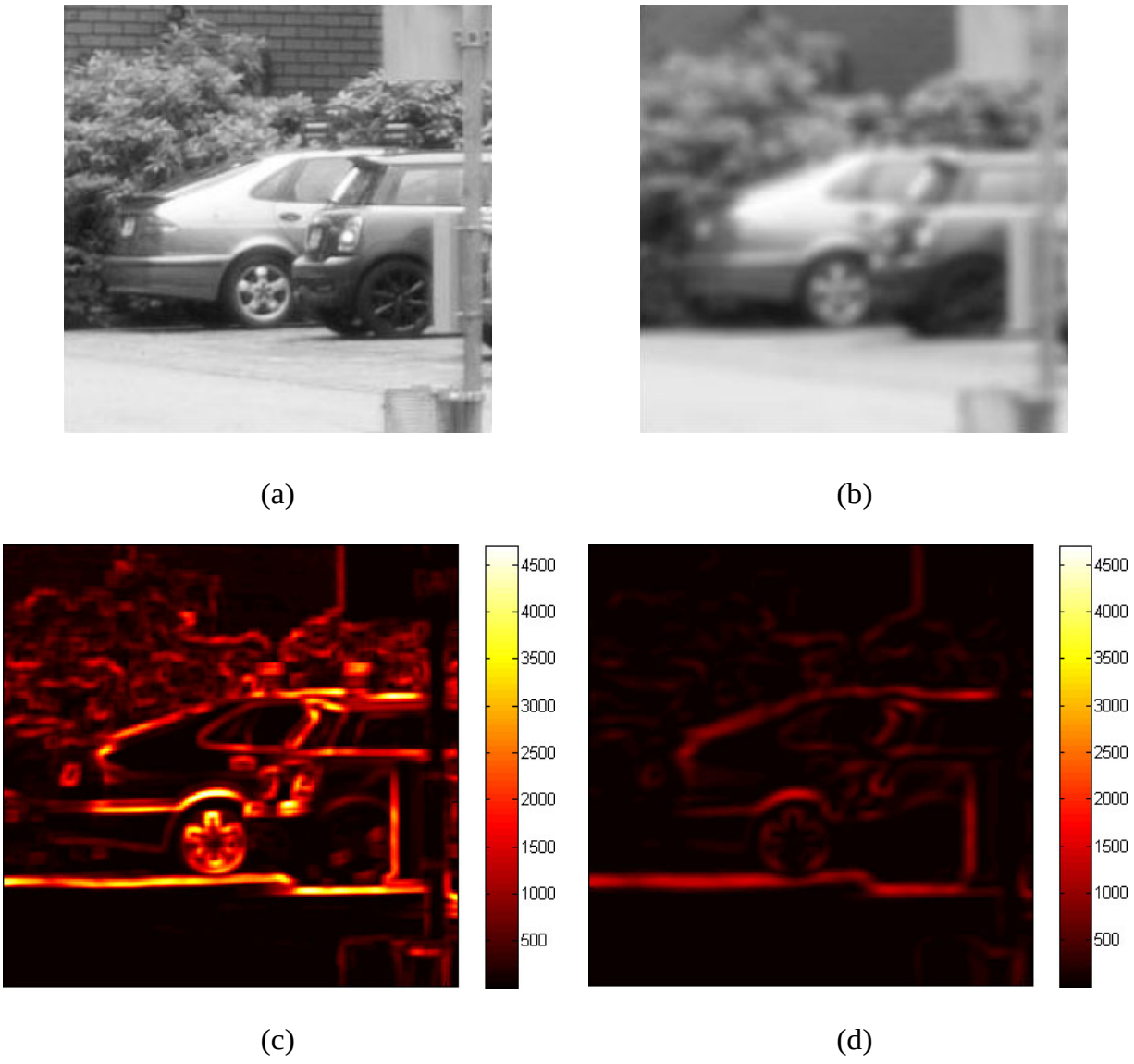


Figure 3.10: Two images and their corresponding variance maps.

(a) Sharp image, (b) Image degraded by natural out-of-focus blur, (c) Variance map of the sharp image, (d) Variance map of the blurred image

compared against a threshold; however, smooth images would result in a small value similar to blurred images; therefore, in this algorithm, to eliminate the smooth areas from blur detection the original, non-blurred image is needed, since different images have different values of standard

deviation even without any blur degradation. This fact makes the algorithm impractical for common applications; also, motion blur is not distinguished from out-of-focus blur and the algorithm is proposed to detect only motion blur.

In [47] it is proposed that blurring an already blurred image would result in smaller variations of neighboring pixels compared to larger variations resulting from blurring a sharp image; therefore, blur is applied to the image and the change in the variation of neighboring pixels is used to measure a no-reference blur metric; however, our experiments show that applying the same blur to various non-blurred images would result in different patterns in pixel variations depending on the content of the image. Figure 3.11 shows some examples of these patterns; therefore, the algorithm is not suitable to determine the blur metric or detect the smooth areas of the image.

A DCT based blur detection algorithm is proposed by [11]. DCT transform is similar to the discrete Fourier transform, and it is a very popular transform in image and video compression. There are several DCT variations; the most common variation is the DCT-II, which is defined as:

$$C(u,v) = \alpha(u)\alpha(v)\sum_{m=0}^{M-1}\sum_{n=0}^{N-1} f(m,n)\cos\left[\frac{\pi(2m+1)}{2M}\right] * \cos\left[\frac{\pi(2n+1)}{2N}\right] \quad (3.13)$$

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{M}} & u = 0 \\ \sqrt{\frac{2}{M}} & u \neq 0 \end{cases} ; \quad \alpha(v) = \begin{cases} \sqrt{\frac{1}{M}} & v = 0 \\ \sqrt{\frac{2}{M}} & v \neq 0 \end{cases}$$

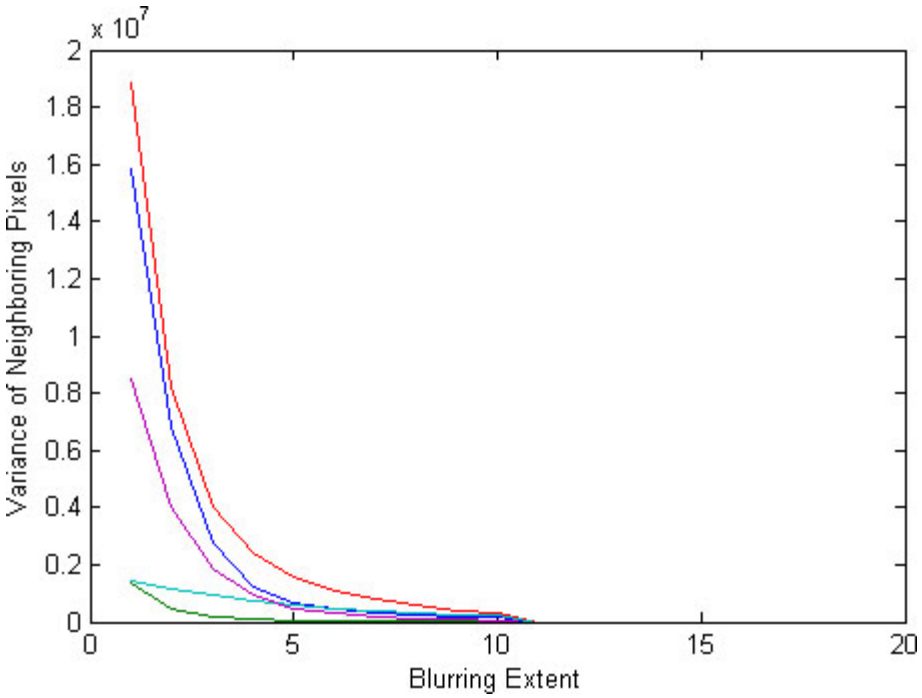
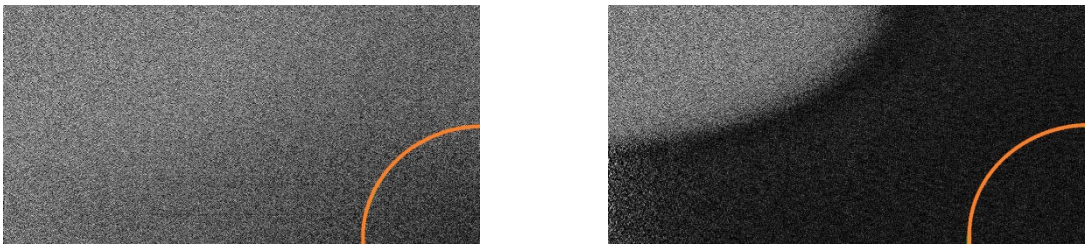
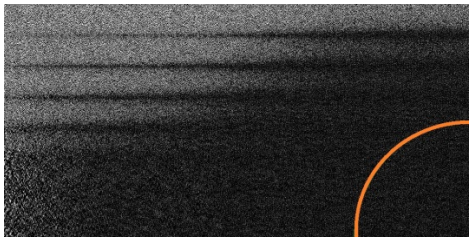


Figure 3.11: Patterns of the change in variance for patches of the same images degraded by various extents of out-of-focus blur degradation.

In general, blur degradation removes the high frequency components; As a result, a blurred image would have low frequency components with high amplitudes and the high frequency components would have lower amplitudes; therefore, by compared the values of high frequency components against a threshold, images/blocks resulting in low values of high frequency components can be marked as blurred; however, different types of blur have not been taken into account and all images with low high-frequency values are considered to be motion blurred. As seen in Figure 3.12, both cases of out-of-focus and motion blur would result in high frequency components with small amplitudes; therefore, examining the amplitudes of high frequency components of an image would not be enough to distinguish between motion and out-of-focus blur.



(a) (b)



(c)

Figure 3.12: DCT spectrums of three images

(a) non-blurred, sharp image, (b) Image degraded by out-of-focus blur, (c) Image degraded by horizontal motion blur

3.3.3 Detection of the Type of the Blur Degradation

The biggest challenge in blur detection is the detection of the type of the blur degradation. This step of blur detection is very often neglected in the literature and still is a largely unsolved problem. Not differentiating between out-of-focus and motion blur is a common mistake in the literature. In [11] it was proposed that the blur type can be determined using the autocorrelation function [24]. In image processing, autocorrelation is a correlation between the image and the shifted version of the image. It was proposed that in the case of motion blur, correlation would be stronger when the image is shifted in the direction of the

blur; whereas in the case of out-of-focus blur, the values would be similar in every direction due to the circular pattern of the blur degradation.

3.4 A Novel Algorithm for the Detection of the Blur Type

During this research, we developed a new algorithm for blur-type detection which distinguishes motion blur from out-of-focus blur.

First, the image is transformed to the DCT domain. In the DCT domain, out-of-focus blur can be distinguished from motion blur by their DCT values. As seen in Figure 3.13, out-of-focus blur and motion blur produce recognizable shapes in the DCT domain. In the DCT domain, out-of-focus blur would result in a circular shape around the origin (top left), which is caused by the large values of low frequency components, as motion blur would appear as parallel lines in the direction of the blur.

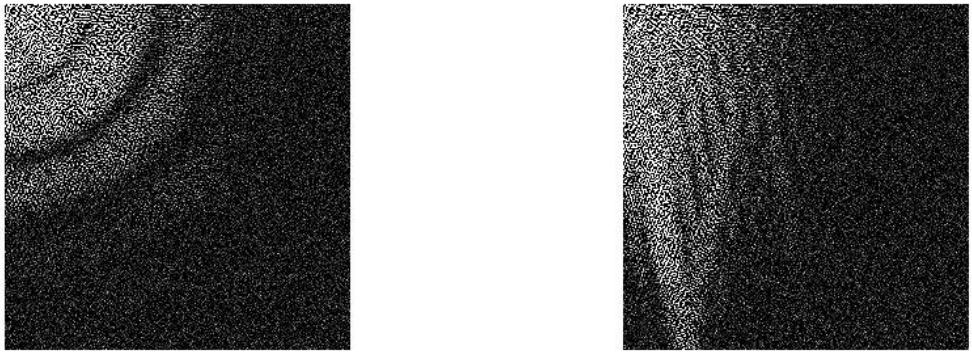
First, The DCT transform of the image and the absolute values of the DCT values are calculated; all following operations are performed on the absolute values. We then applied a low pass filter to the DCT image to reduce the deviation of DCT components. We used a low-pass Hamming filter for this purpose. The 2D Hamming function with a circular region of support is defined as:

$$f[m,n] = 0.54 + 0.46\cos\frac{2\pi\sqrt{m^2+n^2}}{r^2} \tag{3.14}$$

The region of support is represented as:

$$R = \{\, [m,n] \mid m^2 + n^2 \leq r^2 \,\} \tag{3.15}$$

Here r represents the radius of the area. Figure 3.14 shows the low-pass filtered samples.



(a) (b)

Figure 3.13: DCT spectrums of an image degraded by out-of-focus and motion blur.

(a) Out-of-focus, (b) motion.

To isolate the overall shape of the spectrum image we used a threshold to transform the low pass-filtered DCT image into a binary image. We set the threshold as:

$$thrsh = \frac{1}{M * N} \sum_{1}^{M-1} \sum_{1}^{N-1} G[u,v] \tag{3.16}$$

$$G[u,v] = \begin{cases} 1, & G[u,v] \geq thrsh \\ 0, & G[u,v] < thtsh \end{cases} \tag{3.17}$$

The low-pass filtered DCT image is represented by $G[u, v]$. This threshold transforms the DCT image to the binary domain by removing the components with low values and setting the larger components which represent the overall shape of the blur degradation in the spectrum domain to one. Figure 3.15 shows the binary images resulting from motion and out-of-focus blur degradations.

(a) (b)

Figure 3.14: Low pass filtered DCT spectrums of an image degraded by out-of-focus and motion blur.

(a) Out-of-focus, (b) motion.

Finally a test is performed to decide whether or not, the shape of image in spectrum domain is similar to a circular shape. This was done by calculating the variance of the distance between the origin and the points which represent the borders of the shapes which are shown in Figure 3.16. Considering the fact that for an ideal circle, the variance would be zero; the resulting variance is then compared against a threshold. If the variance is smaller than the

threshold, the blur degradation would be marked as out-of-focus blur degradation and if the variance is larger than the threshold it would be marked as a motion blur.

(a) (b)

Figure 3.15: Low pass filtered DCT spectrums of an image degraded by out-of-focus and motion blur is transferred to binary domain using a suitable threshold.

(a) Out-of-focus, (b) motion.

It is necessary to perform blur detection prior to this step. This is due to the fact that this algorithm should be applied only on blurred images; also, smooth areas in the image would decrease the accuracy of the algorithm; therefore, smooth areas should be detected and excluded prior to this step.

3.5 Identification of the Parameters of Motion Blur Based on a Single Image

Commonly, blur identification is performed on single, independent images due to the application of deblurring in photography, and is a well-studied topic in blur analysis.

(a) (b)

Figure 3.16: Edges of the binary DCT spectrums of an image degraded by out-of-focus and motion blur. Variance is used to measure the similarity of the edges to a circular shape.

(a) Out-of-focus, (b) motion.

3.5.1 Blur Identification Using Steerable Filters and the Cepstral Transform

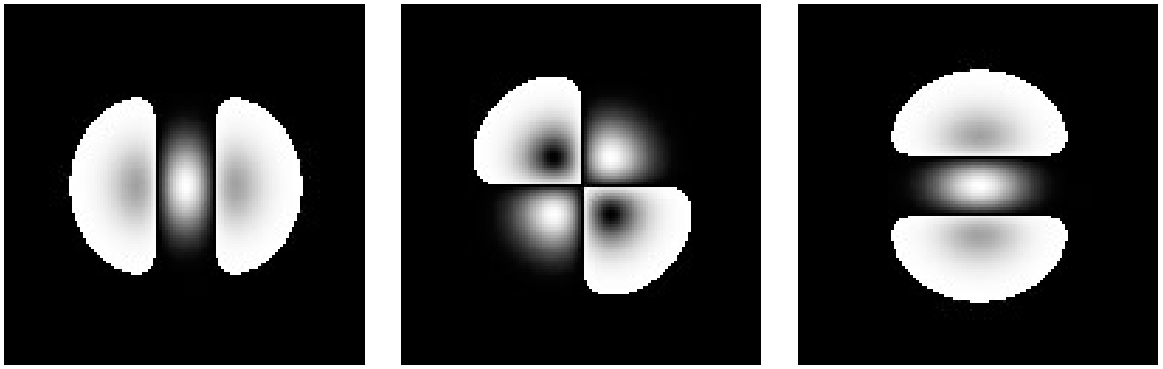
A popular algorithm to identify the blur parameters from a single image was proposed in [14]. Following is a review on the algorithm.

3.5.1.1 Identification of the Blur Angle

As discussed before, in spectral domain, motion blur would appear as a ripple and parallel lines in the direction of the motion blur as highlighted in Figure 3.18.

The angle of the parallel lines represents the angle of the blur. This fact is the basis of many blur identification algorithms. In this case, steerable filters are used to find the angle of the motion blur. Steerable filters were introduced by Freeman in [15] which are mainly used in

edge detection. In this algorithm, the second derivatives of the Gaussian filter which are shown in Figure 3.17 are used to find the direction of the ripple.



(a) (b) (c)

(d)

Figure 3.17: Steerable filters.

(a) G_{2a}, (b) G_{2b}, (c) G_{2c}, (d) G_2^{θ}.

The high efficiency of using the steerable filters makes them suitable for blur angle identification even in real-time scenarios.

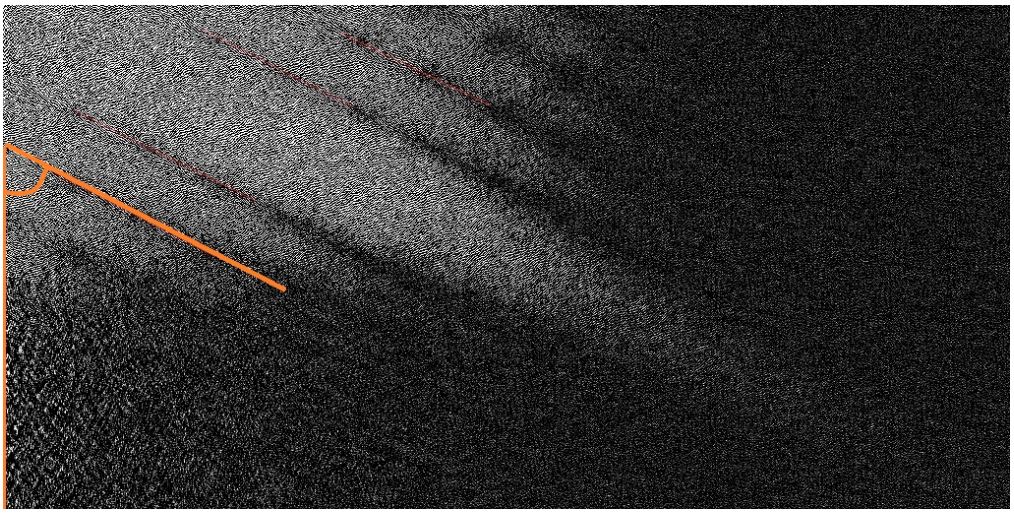


Figure 3.18: DCT spectrum of a motion blurred image. The added lines demonstrate the angle of the motion blur degradation.

The response to the steerable filters in any angle is then calculated as follows [15]:

$$RG_2^{\theta} = K_a(\theta).RG_{2a} + K_b(\theta).RG_{2b} + K_c(\theta).RG_{2c} \tag{3.18}$$

$$G_{2a} = 0.9213(2x^2 - 1)\exp\left(-(x^2 + y^2)\right) \tag{3.19}$$

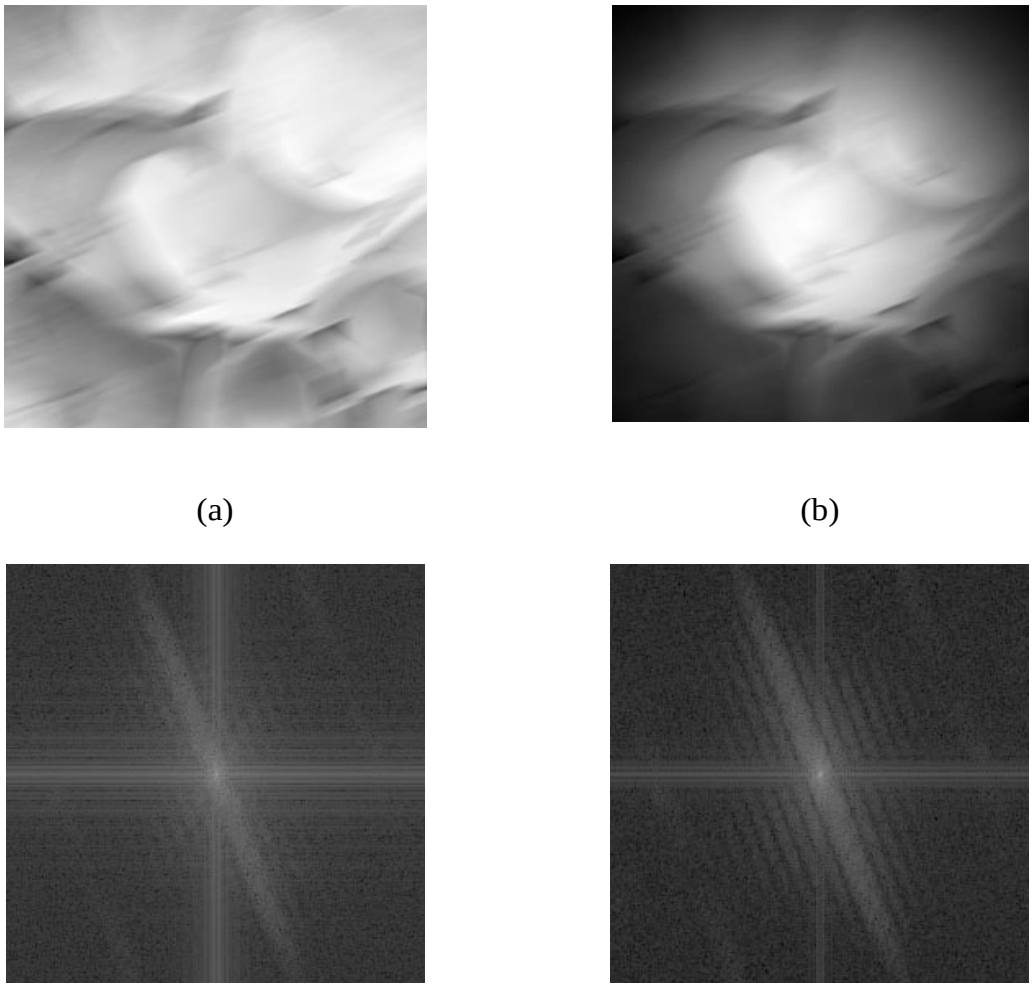
$$G_{2b} = 1.843xy\exp\left(-(x^2 + y^2)\right) \tag{3.20}$$

$$G_{2c} = 0.9213xy\exp\left(-(x^2 + y^2)\right) \tag{3.21}$$

$$K_a(\theta) = \cos^2(\theta) \tag{3.22}$$

$$K_b(\theta) = -2\cos(\theta)\sin(\theta) \tag{3.23}$$

$$K_c(\theta) = \sin^2(\theta) \tag{3.24}$$



(a) (b)

(c) (d)

Figure 3.19: Effect of masking on power spectrum

(a) image patch, (b) masked image patch, (c) Fourier spectrum of the patch, (d) Fourier spectrum of the masked patch

RG_2^{θ} is the response of the second derivative Gaussian rotated by θ degrees. First, response to components G_{2a}, G_{2b} and G_{2c} is calculated and then the response to RG_2^{θ} at any angle is calculated using Equation (3.18). This procedure means that finding the response of the power spectrum image to the second derivative Gaussian in every angle is very

computationally inexpensive. This algorithm can be used globally or locally on small patches. First a patch from the image is selected. A mask is used to reduce the ringing effect when selecting a patch. Here, ringing effect is caused by the sudden change of values in the borders when taking a small patch of the image. Several masks can be used to reduce this effect including the Gaussian window and the Hamming window. Masking is shown in Figure 3.19.

The masked image is then zero-padded to give a more optically detailed image in the frequency domain. This operation increases the sampling rate of the Fourier transform. After calculating the Fourier transform of the masked and zero-padded patch, the power spectrum is calculated.

$$F(u,v) = R\{F(u,v)\} + jI\{F(u,v)\} = |F(u,v)|\exp\big(j\varphi(u,v)\big) \tag{3.25}$$

$$|F(u,v)| = \sqrt{R^2(u,v) + I^2(u,v)} \tag{3.26}$$

$$\varphi(u,v) = \tan^{-1}\left(\frac{I(u,v)}{R(u,v)}\right) \tag{3.27}$$

$|F(u,v)|$ is the Fourier spectrum and $\varphi(u,v)$ is the phase of the Fourier transform. The power spectrum is calculated as:

$$P(u,v) = |F(u,v)|^2 \tag{3.28}$$

Steerable filters are applied to the power spectrum of the patch; the angle resulting in the response with the highest value is selected as the blur angle. The blur angle is within the

range of $0^{\circ}$to 180°, or similarly $-90^{\circ}$to $+90^{\circ}$. This is due to the fact that due to the nature of motion blur and relative motion, a blur with the angle of θ is the same as a blur with the angle of $\theta + 180$.

3.5.1.2 Identification of the Blur Length

After determining the blur angle, the next step is to determine the blur length along the angle of the blur. Motion blur would cause a ripple in the direction of motion. This ripple is very similar to a Sinc function [45]. Figure 3.20 shows a plot of the Sinc function. Without loss of generality, assuming that the motion blur is one dimensional, the motion blur in continuous domain would be defined as:

$$d(x;l) = \begin{cases} \frac{1}{l} & -\frac{L}{2} \leq x \leq \frac{L}{2} \\ 0 & otherwise \end{cases} \tag{3.29}$$

By applying the Fourier transform we would have:

$$D(\omega) = 2\sin\frac{\left(\frac{\omega\pi L}{2}\right)}{\omega\pi L} \tag{3.30}$$

This is a Sinc function in the frequency domain. Sinc function is defined as:

$$sinc(x) = \frac{\sin(x)}{x} \tag{3.31}$$

In the discrete domain we would have:

$$H[\omega] = \sin\frac{\left(\frac{L\omega\pi}{N}\right)}{L\sin\frac{\omega\pi}{N}}, \qquad 0 \leq \omega \leq N-1 \tag{3.32}$$

N is the size of the signal. In order to determine the value of L, we need to solve the following equation:

$$H[\omega] = \sin\frac{\left(\frac{L\omega\pi}{N}\right)}{L\sin\frac{\omega\pi}{N}} = 0, \qquad \sin\left(\frac{L\omega\pi}{N}\right) = 0 \tag{3.33}$$

We would have:

$$\omega = \frac{kN}{L}, \quad k > 0 \tag{3.34}$$

Considering that L is the distance between two successive zeros we would have:

$$L = \frac{N}{d} \tag{3.35}$$

When d is the distance between two successive dark lines in the power spectrum.

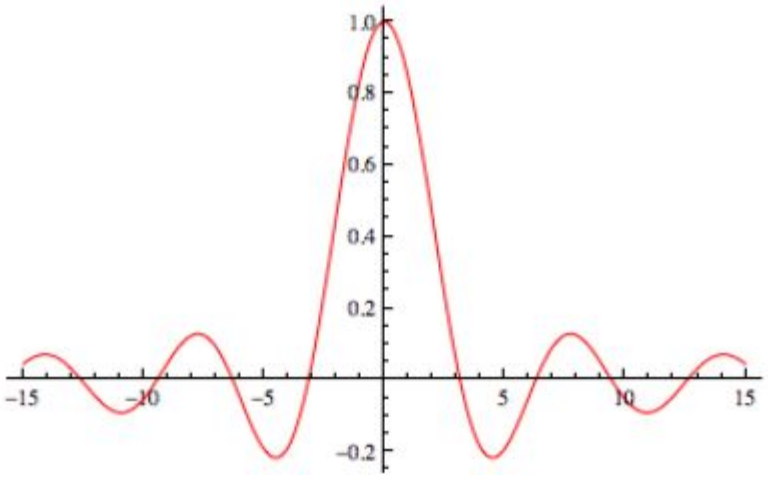


Figure 3.20: The Sinc function

By applying the Cepstrum to this ripple, the blur length which appears as the period of the signal, would appear as a large negative peak. Estimating the blur length using Cepstrum is a well-studied method.

There are different definitions of Cepstrum based on the application it is used for. The common application of Cepstrum is in audio processing, for example in voice identification and pitch detection. The Cepstrum used here is defined as:

$$Cep\{f(x)\} = F^{-1}\{\log\big(F(\omega)\big)\} \tag{3.36}$$

When $F^{-1}\{\log(F(\omega))\}$ is the inverse Fourier of $\log\big(F(\omega)\big)$. First, the 2D power spectrum image is collapsed to a 1D signal along the angle of the motion blur. To collapse the image to 1D along a desirable angle, every pixel of the image is projected to the line that passes through origin along the desirable angle [14]. By applying the Cepstrum transform to the 1D signal, the blur length would appear as a negative peak in the Cepstrum domain. Figure 3.21

shows the collapsed power spectrum and the Cepstrum transform. The image was blurred with the angle of 60 degrees and the length of 15 pixels. In this case, the blur length is accurately estimated by the negative peak in the Cepstrum.

This algorithm works best in images with sufficient amount of detail and high textures; also, when used locally, with smaller patch sizes, the performance changes drastically when different patches of the image are selected. The algorithm is highly unreliable for patches of size 64x64 and the result from patches of sizes 128x128 and 256x256 can be erroneous. This is due to the nature of the algorithm and the amount of detail in the spectrum domain which is needed for the algorithm to work. The algorithm is also erroneous for very small lengths of motion blur. This is due to the fact that a small length of blur causes a wider ripple in the frequency domain; this makes it hard for the steerable filters to detect the accurate angle.

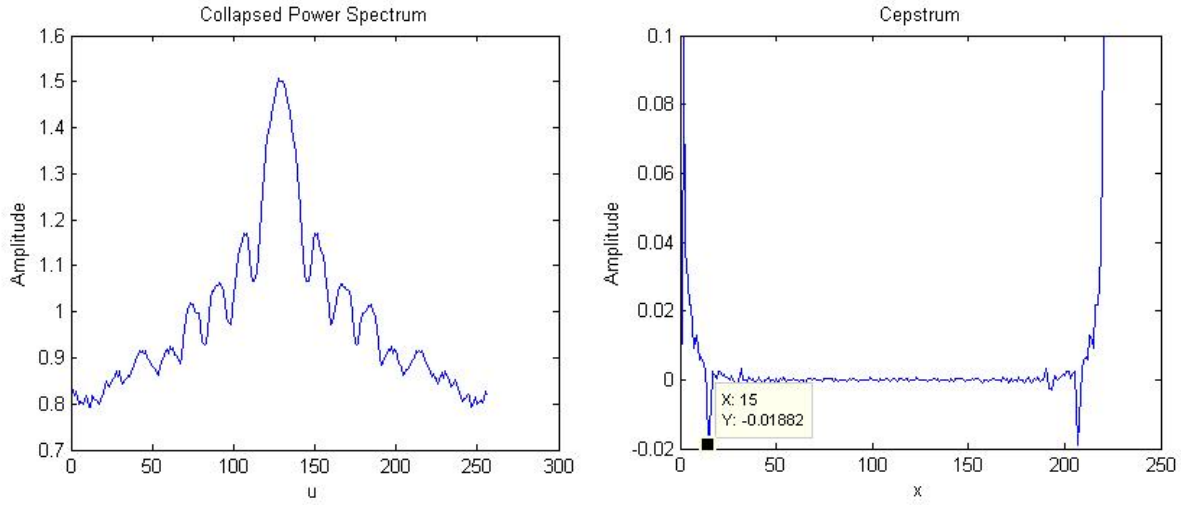


Figure 3.21: 1D power spectrum and the Cepstrum transform of the 1D power spectrum signal

3.5.2 Improvement in the Angle Identification

During this research, we developed an algorithm to improve the estimation of blur angle from a single image. After identifying the initial and possibly erroneous blur angle using the steerable filters or other similar algorithms, the 2D power spectrum image was collapsed along the angles close to the candidate angle. The image was projected to 1D according to the algorithm presented in [48]. The process is shown in Figure 3.22. The pixel $P(x, y)$ is orthogonally projected into the pixel $P_l(d)$ on the line l which has an angle of t° against the y axis. The distance d is calculated using the following equation.

$$d = d_x \cos t + d_y \sin t \tag{3.37}$$

In the discrete domain however, the value of d is not necessarily an integer; therefore, we map every pixel $P(x, y)$ into two pixels of $P_l(\lfloor d \rfloor)$ and $P_l(\lfloor d \rfloor + 1)$; therefore, if the value of d is for example $d = 10.8$, the pixel would contribute 80% of its amplitude to $P_l(\lfloor d \rfloor + 1)$ and 20% of its amplitude to $P_l(\lfloor d \rfloor)$. Furthermore, since we are collapsing a finite 2D signal, a larger number of pixels would be mapped to the corresponding pixels near the center with small d values and fewer pixels would be mapped to a pixel on the line as get far from the center and the value of d becomes larger. To compensate for this, every 1D pixel value is divided by the number of 2D pixels which are mapped into it. Since our original image has real pixel values, the Fourier transform would be symmetrical. This property can be used to achieve a better estimation of the collapsed 1D signal of the image in the Fourier domain. The Fourier transform of a 1D signal with n real values is symmetrical around $\frac{n}{2} + 1$;

therefore, after the collapsing process we make the collapsed 1D signal more similar to a symmetrical signal by averaging according to the following equation [14].

$$P_l\left(\frac{n}{2}+1+i\right) = P_l\left(\frac{n}{2}+1-i\right)$$

$$= \frac{P_l\left(\frac{n}{2}+1+i\right) + P_l\left(\frac{n}{2}+1-i\right)}{2} \qquad 1 \le i \le \frac{n}{2}-1 \qquad (3.38)$$

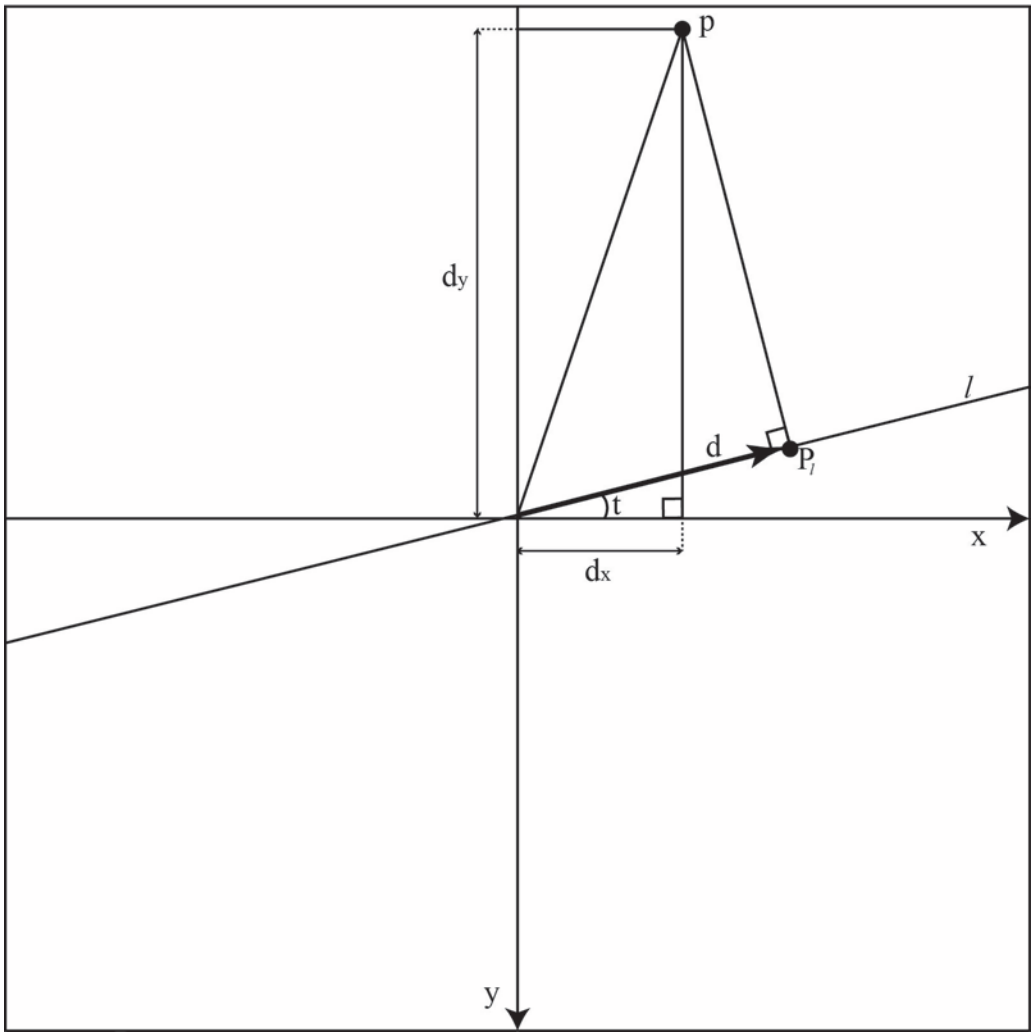


Figure 3.22: Projecting (collapsing) a 2D image to a 1D signal along a line with the angle of t.

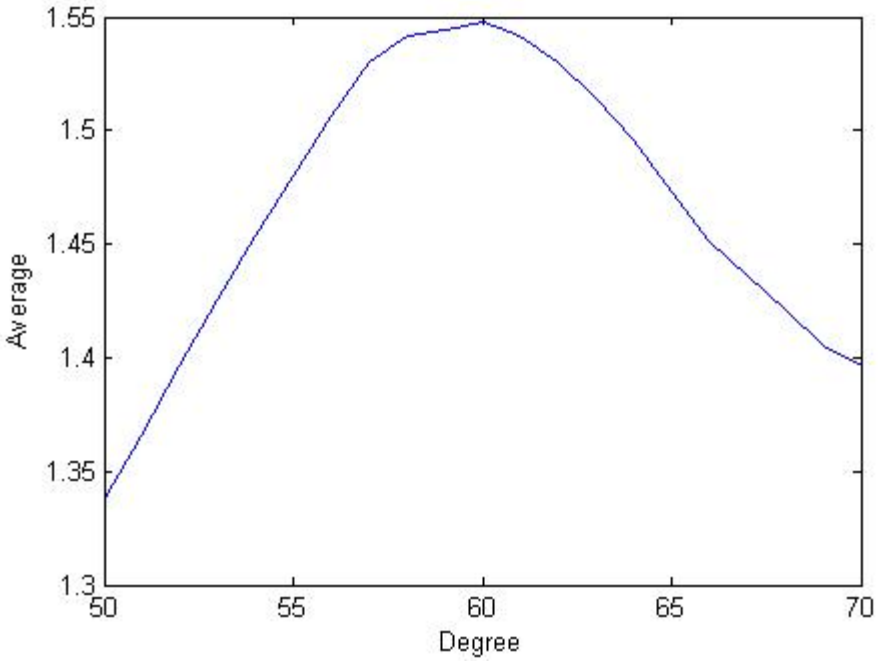


Figure 3.23: Average of neighboring values around the peak in different angles; The angle of blur is $\mathbf{60}^{\circ}$.

It was observed that the signal collapsed along the correct angle results in higher values around the peak. This is due to the fact that the ripple caused by the motion blur is in the angle of the motion and therefore, it has the highest amplitudes in this direction; therefore, after estimating the initial blur angle, neighboring angles are examined to find the angle producing the highest average of the peak value and values around the peak. This angle is selected as the corrected estimated angle. Our experiments show great reduction in the angle estimation error using this algorithm. Figure 3.23 shows the averages of values near the peak for an image blurred with an angle of 60 degrees. The estimation using the steerable filters was 54 degrees and using our algorithm the angle was corrected to 60 degrees. As

shows in the Figure 3.23 the average value is increased when going towards the correct angle of the motion blur.

3.5.3 Blur Identification Based on the Radon Transform

Another popular algorithm to identify the angle of motion blur is based on the Radon transform [16] which is based on the similar premises and is very similar to the steerable filters algorithm. This method is based on the direction of the ripples in the spectrum domain as well. One of the applications of the Radon transform, which is a line fitting algorithm similar to Hough transform, is to find the angle of a line; however, unlike the Hough transform, Radon transform does not need candidate points to operate. Radon transform is defined as:

$$R(\rho,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\delta(\rho - x.\cos\theta - y.\sin\theta)\,dx\,dy \qquad (3.39)$$

Or:

$$R(\rho,\theta) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\rho.\cos\theta - s.\sin\theta, \rho.\sin\theta + s.\cos\theta)\,ds \qquad (3.40)$$

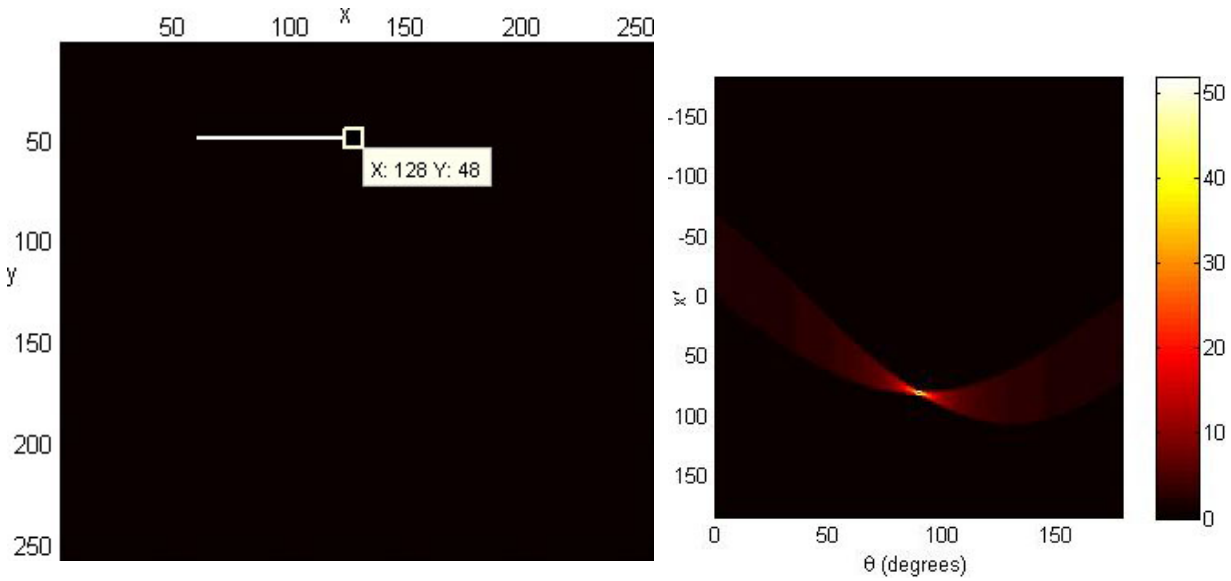
In this method, the angle of the parallel lines representing the blur angle is determined by applying the Radon transform.

First the logarithm of the Fourier spectrum is calculated as follows:

$$\log(1 + |F(u, v)|^2) \tag{3.41}$$

In the Radon transform of the logarithm of Fourier transform, the angle of the parallel lines appears as a strong peak representing the angle of the parallel lines. Figure 3.27 shows the discrete radon transform for all angles between 0° and 180° of an example 256x256 image which is an image of a horizontal straight line. The θ value of the large peak in the Radon domain represents the angle of the line against the y axis which is 90° and the x' value represents the distance between the center of the image located at (128,128) and the line along the y axis [52], which is 80 in this case. In this process, the Radon transform needs to be calculated for all discrete angles between 0° and 180° which results in the high computational cost of the algorithm. Again, after finding the blur angle, the blur length is calculated by using the Cepstrum technique.

Both methods use the same concept in order to estimate the blur parameters, although using the Radon transform would result in more accurate estimation of the blur angle, using steerable filters is far less computationally expensive making it more suitable for video processing.



(a) (b)

Figure 3. 24: Using the Radon transform to estimate the angle of a line.

(a) An image of a straight line, (b) The peak in the Radon domain corresponds to the angle of the line. Here the angle is shown as 90°, which is the angle against the y axis.

3.5.4 Other Methods of Blur Identification

In [44], the angle of the dark lines in the Fourier domain is estimated using the Hough transform which unlike the Radon transform needs candidate points to operate. It is proposed that the gradient vectors of the Fourier spectrum are perpendicular to dark lines. Figure 3.25 shows the Gradient vectors around a dark line. Gradient vector direction of any point is calculated as:

$$\theta_f(u,v) = \tan^{-1}\left(\frac{\sum_{j=v-l}^{v+l}|F(u-l,j)| - \sum_{j=v-l}^{v+l}|F(u+l,j)|}{\sum_{i=u-l}^{u+l}|F(i,v-l)| - \sum_{i=u-l}^{u+l}|F(i,v+l)|}\right) \qquad (3.42)$$

Size of the region surrounding the point (u, v) is $(2l + 1) \times (2l + 1)$. The Hough transform is used with this information to detect the dark lines. After the detection of the dark lines, their angle and the distance between them are calculated as the blur angle and the blur length.

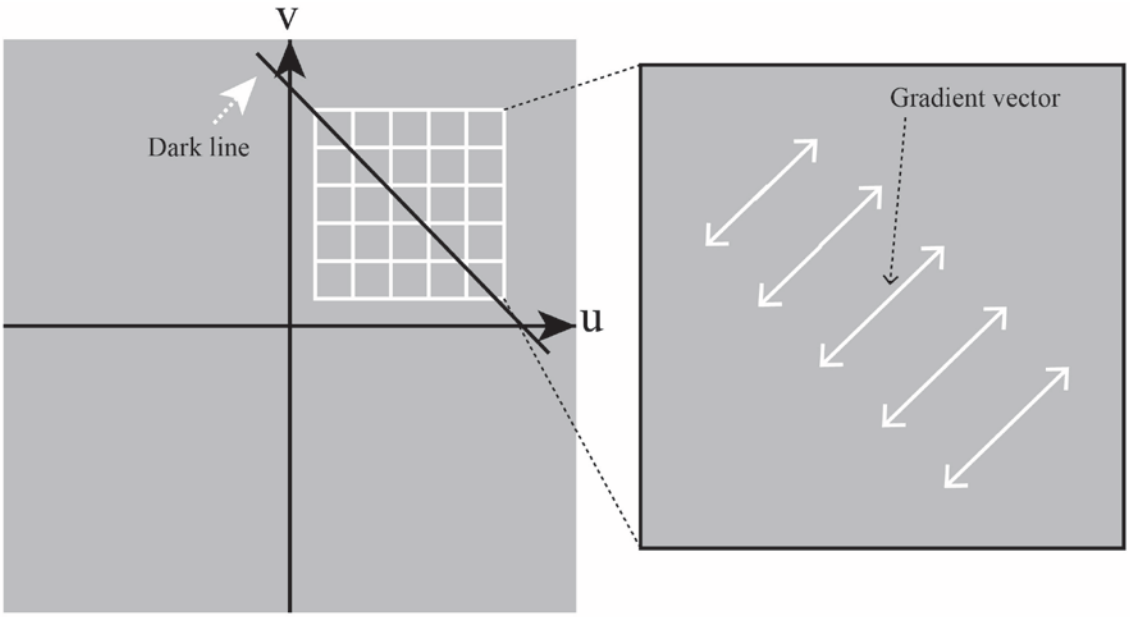


Figure 3.25: Gradient vectors around the parallel dark line in the spectrum domain of a motion blurred image.

3.6 Identification of the Parameters of Out-of-Focus Blur

Similar to motion blur, the parameters of out-of-focus blur can be determined from the Fourier domain of the image/block. Commonly in the literature, the Pillbox model and the Gaussian model are assumed as the models for out-of-focus blur degradation.

In [18] out-of-focus images were artificially created by applying the pillbox blur, and then the diameter of the pillbox was determined by calculating the Cepstrum of the Fourier spectrum and compared to the original diameter.

Calculating the blur diameter or blur length in the case of out-of-focus blur is based on the same premises as of the motion blur. In the case of out-of-focus blur, the 2D power spectrum image can be collapsed to a 1D signal along any direction due to the circular pattern of the ripples caused by the blur degradation. A more computationally expensive but more accurate measure is to collapse the image along all angles between 0° to $360^{\circ}$and average over the collapsed signals to produce the 1D power spectrum signal.

$$P(x) = \frac{1}{360}\sum_{0}^{359} g(x,\theta) \tag{3.43}$$

When $g(x,\theta)$ is the collapsed signal of $f(x,y)$ at the angle θ. $P(x)$ is the averaged power spectrum signal over all angles. By applying the Cepstrum transform to this 1D signal, the diameter would appear as a negative peak.

3.7 Generation of Artificial Blur in an Image

Blur generation and rendering is commonly used in 3D graphics techniques, which are not in the scope of our application. Blur generation is also used in imagery as a special effect to visually simulate a blurred object in an image. This is commonly done by applying the blur PSF to the image.

In this project however, blur generation is used for a different purpose. The aim of this project is to simulate blur in a way that the BDM between the real blur and the simulated blur is small. Smaller BDM means that the difference in energy and the amount of data we need to transmit as the residual error is smaller.

In other common applications, the BDM between the real blur and the simulated blur is not of great importance, and therefore, this aspect of simulated blur, even with currently used PSFs, is not thoroughly investigated.

During this research we artificially blurred several sharp images and compared them to the naturally blurred versions of the same images. We used MSE as the comparison criteria, and analyzed the results which would be discussed in future chapters.

As mentioned before, the commonly used models for out-of-focus blur are Pillbox model and Gaussian model, and motion blur is modeled according to Equation (3.2)

3.8 Blur Compensation in Video Coding

Although the subject of blur analysis and its many different aspects have been widely discussed in the literature, the effects of blur on video coding have been widely ignored. The researches in the field of blur are mainly focused on applications such as image restoration and generation of blur in animations and 3D graphics; therefore, very little research has been done on the subject. One notable attempt to compensate blur was done in [2]; however, this was done by blind application of simple blurring filters, without identifying the blurred areas, blur type and blur parameters. This greatly limits the achievable level of PSNR and bit-rate reduction. Blur generation was done by iterating over simple averaging filters.

$$b_{a4} = \frac{1}{16} g(4,4) \tag{3.44}$$

$$b_{a8} = \frac{1}{64} g(8,8) \tag{3.45}$$

$$b_{a16} = \frac{1}{256} g(16,16) \tag{3.46}$$

$$b_{m\_4\_0} = \frac{1}{4} g(1,4) \tag{3.47}$$

$$b_{m\_4\_90} = \frac{1}{4} g(4,1) \tag{3.48}$$

$$b_{m\_6\_0} = \frac{1}{6} g(1,6) \tag{3.49}$$

$$b_{a4} = \frac{1}{6} g(6,1) \tag{3.50}$$

When $g(m,n)$ is a matrix of ones with the size $[m,n]$. $b_{a4} = \frac{1}{16} g(4,4)$, $b_{a4} = \frac{1}{16} g(4,4)$ and $b_{a4} = \frac{1}{16} g(4,4)$ are used to model out-of-focus blur and the rest are used to model motion blur. Our tests show that using the out-of-focus filters above are not sufficient to appropriately simulate out-of-focus blur with a high PSNR compared to the natural blur. Furthermore motion blur models used only include horizontal and vertical motion blur with two blur lengths for each orientation. Considering the large number of possible variations of blur angle and blur length, this method would not be beneficial in the majority of scenarios; therefore, a more systematic blur compensation scheme is required to achieve higher bit-rate reductions.

Chapter 4

A Novel Approach to Blur Compensation

In this chapter, a novel approach to blur compensation is introduced. Using this method, first the properties of the blur degradation in blurred blocks are estimated. Based on this knowledge of the blur, the non-blurred blocks are artificially blurred to reduce the negative effects of blur degradation on motion compensation in video streams and improve the ratio of video compression.

4.1 Inter Coding the Otherwise Intra-Coded Blocks

The main motivation for starting this project was to reduce the negative effect of blur degradation on motion compensated video streams. As discussed previously, blocks with similar properties are inter-coded based on the past or future blocks; however, when blur degradation appears, these properties are distorted. For example, as seen in Figure 1.3, the properties of the target block no longer matches the properties of the reference block. This is

caused by the fact that although the target block is captured from the same source as the reference block, its pixels are degraded by blur.

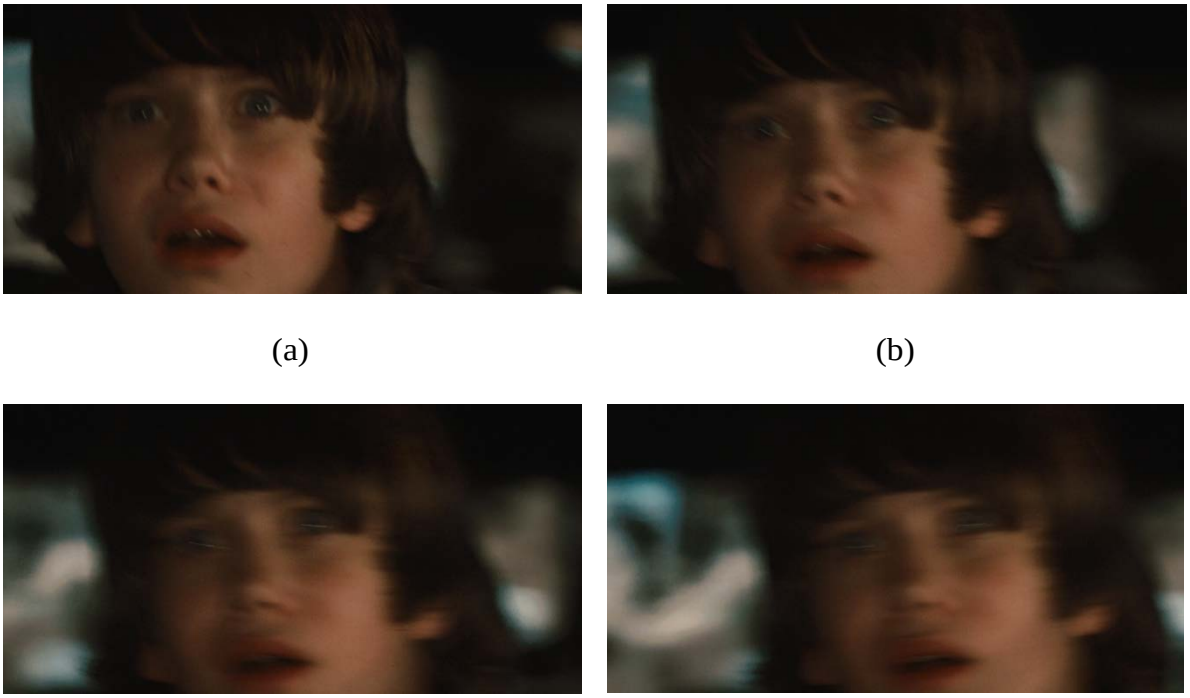
Degradation of the target block interferes with both motion estimation and motion compensation in transition from non-blurred frames/blocks to blurred frames/blocks, or similarly, from blurred frames/blocks to non-blurred frames/blocks. Commonly in the literature, performances of the motion estimation methods are not evaluated for their resilience to blur degradation in the described scenarios. During this project we evaluated two very popular motion estimation methods; block matching which is widely used in today's video coding standards, and phase correlation. In both cases, blur degradation extremely interfered with motion estimation. Small amounts of motion blur degradation resulted in large areas with unmatched blocks in block matching, and phase correlation is very erroneous in blurred frames. Figure 4.1 shows four frames from a video sequence and Figure 4.2 shows their corresponding matched blocks. Blank areas in Figure 4.2 show the areas that were not matched with a motion vector. As seen in the sequence, with higher blur degradation in the third and fourth frame, the number of matched blocks is reduced.

To evaluate the performance of motion compensation, we assumed that motion estimation is performed accurately and the motion vector is acquired; motion compensation is then by using the motion vectors. As we expected, motion compensation resulted in very small PSNRs and large residual errors, rendering the motion compensation ineffective.

We divided the project into four stages:

- Detecting the areas degraded by blur
- Identifying the blur parameters
- Regenerating the blur based on our knowledge of the blur parameters

- Encoding the residual error and the blurring parameters



(a) (b)

(c) (d)

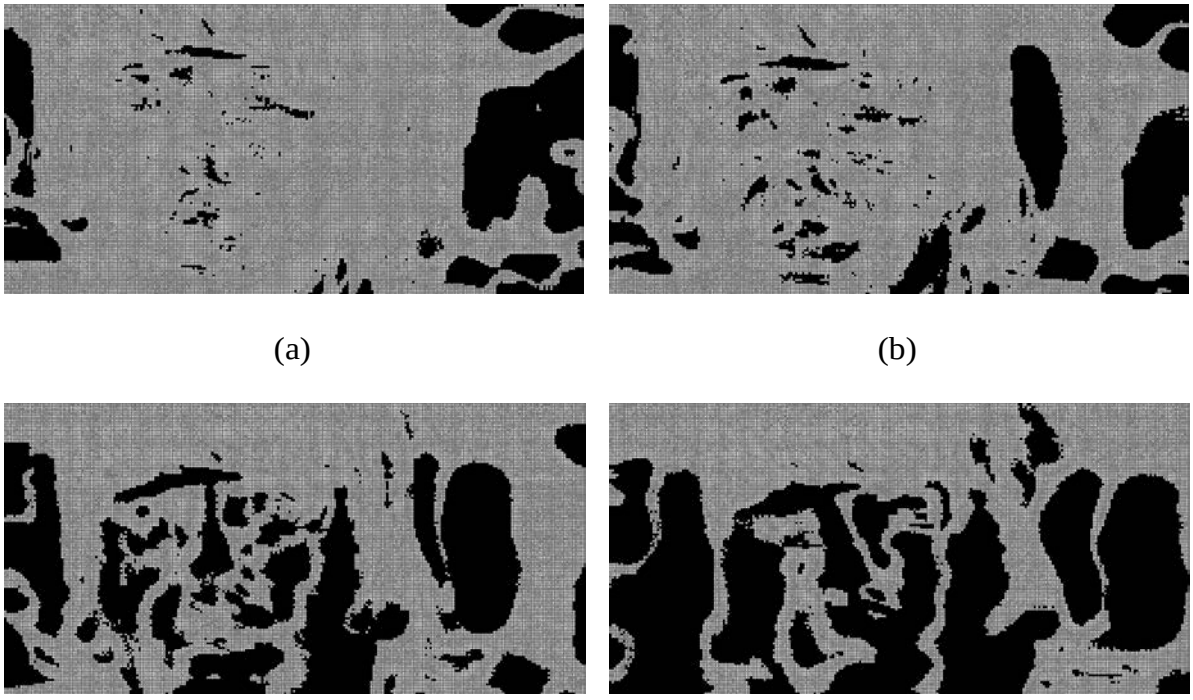
Figure 4.1: Four frames in a video sequence and their matched-blocks against the previous frame.[4]

(a) Frame k, (b) Frame k+1, (c) Frame k+2, (d) Frame K+3.

First the frame should be analyzed for blur degradation, but to make the algorithm faster and more efficient, we based our work on identifying the blocks which are not matched using block matching and analyzing these blocks for blur degradation. Next step is to estimate the parameters and properties of the blurred blocks. This information is used for blur

[4] Images taken from "2012" the movie, 2009, Centropolis Entertainment (as Centropolis), Columbia Pictures (presents), The Mark Gordon Company, Farewell Productions, Sony Pictures Home Entertainment.

regeneration and compensation. In the final step, the information about blur compensation is encoded and transmitted to the receiver. These steps would be discussed in future sections.



(a) (b)

(c) (d)

Figure 4. 2: matched-blocks of four frames against the previous frame.

(a) Matched blocks of k, k-1, (b) Matched blocks of k+1, k, (c) Matched blocks of k+2, k+1, (d) Matched blocks of K+3, k+2

4.2 Detection of the Areas Degraded by Blur

Although including the already inter-coded blocks in blur compensation would still improve the PSNR, in order to make the algorithm faster, we first identify areas with intra-coded blocks and exclude the inter-coded blocks from further operations.

As it would be discussed in later sections, we perform our algorithm on macroblocks of sizes 64x64, 128x128, 256x256, 512x512, 1024x1024 and larger. Sizes are calculated as:

$$m, n = 2^k \tag{4.1}$$

We would discuss in later sections that using macroblocks of sizes smaller than 64x64 is not suitable for blur operation.

First we perform full search block matching motion estimation. In the next step, we scan the frame for macroblocks with large number of intra-coded blocks. To do this we calculate the ratio of intra-coded blocks to the number of inter-coded blocks in every macroblock. We compare the ratio against an arbitrary threshold. If the ratio is larger than the threshold, we mark the macroblock as a candidate for possible blur degradation and perform further blur analysis on it. Otherwise, the macroblock is marked as inter-coded and is excluded from further operations. By choosing a smaller threshold, more positives would occur and the algorithm would take longer to complete; also, there would be more false positives, as some of the macroblocks are not degraded by blur.

4.3 Motion Estimation in Blurred Frames

After identifying the candidate macroblocks with possible blur degradation and performing blur detection, we need to find matching macroblocks in past/future macroblocks and estimate the motion. As discussed before, regular motion estimation methods are not proven to perform well in case of blur degradation. Investigation of the performance of block

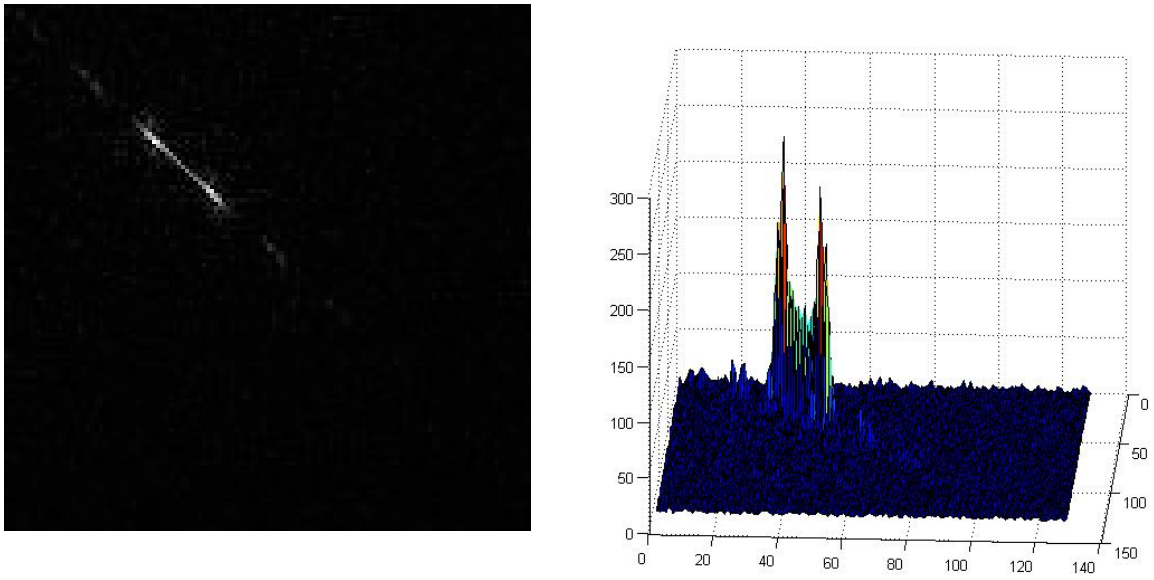
matching and phase correlation revealed their poor performance in the case of blur degradation. We developed a novel algorithm to modify the classic phase correlation for usage in blurred frames.

In [21] an algorithm was proposed to modify phase correlation for blurred frames with application of image registration used in digital subtraction angiography (DSA), the algorithm is based on the following equation:

$$C_{k,k+1}(norm) = \left(\frac{F_{k+1}(u,v) {F_k}^*(u,v)}{|F_{k+1}(u,v) {F_k}^*(u,v)|} \right)^2 \tag{4.2}$$

This phase correlation modification is designed to be a blur-invariant phase correlation in the case of linear and symmetrical blur; however, this modification makes phase correlation less resilient to noise; also, in the case of non-artificial blur, the algorithm fails to perform well. This is probably because of the effect of the squaring shown in Equation (4.2) and the fact that frames were assumed to be noiseless in this algorithm. This is despite the strong resilience of the classic phase correlation to added noise.

To overcome these shortcomings, we proposed a new algorithm. Figure 4.3 shows the two large peaks generated by phase correlation in when one of the frames is blurred. As seen in Figure 4.2, motion blur causes a ripple and instead of one large peak, there would be two peaks in the direction of the motion blur. By examining the location of the two peaks in several cases we discovered that motion blur shifts the peak resulting from phase correlation with the same amount in both directions resulting in two peaks; therefore, the displacement is calculated as:



(a) (b)

Figure 4. 3: Two peaks appearing in the phase correlation of frames motion blurred frames.

(a) 2D view, (b) 3D view.

$$c_{k,k+1}(x,y) = \delta(x - d_{x1}, y - d_{y1}) + \delta(x - d_{x2}, y - d_{y2}) \tag{4.3}$$

$$dx = \frac{d_{x1} + d_{x2}}{2} \tag{4.4}$$

$$dy = \frac{d_{y1} + d_{y2}}{2} \tag{4.5}$$

Similar to regular phase correlation in absence of blur, motion of objects inside the frame results in appearance of independent peaks; therefore, in the case that a number of large peaks appear with amplitudes close to the amplitude of the largest dominant peak, all the

large peaks are picked as candidates for the displacement values; however, using phase correlation on small macroblocks would reduce the effect of local motions and additional peaks. Since the algorithm is used on variable-sized macroblocks, it is valid for both global motion, which is the motion of camera, and for local motion which is the motion of objects inside the frame.

Resilience to noise was examined by adding noise to the frames and the algorithm proved to be highly resilient to noise; also, several non-artificial cases of blur were examined from a film sequence and the performance of the algorithm was evaluated. The results are presented in Chapter 5.

As mentioned before, we avoided using macroblocks of sizes smaller than 64x64. This was due to the observation that by using smaller sizes, there is very little information in the spectrum domain and the result would be erroneous. We would use the same argument in later sections to justify our usage of macroblock of sizes larger than 64x64 in various blur analysis operations.

Our proposed algorithm is also resilient to out-of-focus blur, meaning that in the presence of out-of-focus blur, we would still obtain an accurate estimation of motion. This is important since in the case of out-of-focus blur, we need to know the motion vector between the out-of-focus macroblock and the non-blurred macroblock for the blur compensation process.

4.4 Estimation of Blur Parameters

In this chapter we explain the methods we used in this project in order to estimate the parameters of motion blur and out-of-focus blur.

4.4.1 Motion Blur

As explained in previous sections, our tests showed that when identifying the blur parameters from a single frame in the spectral domain, the size of the block is very important. Our tests showed that in blocks smaller than 128x128, the results can be unsatisfactory erroneous and unreliable; also, these methods are based on a single frame and they do not take advantage of the dependencies of frames in a video sequence. As a result, although these methods show promise for many applications and can be further improved (as we improved the angle estimation), we decided to leave them for future work and take a different direction for this project. Using our method, we also take advantage of the dependencies of frame sequences instead of using a single image.

As discussed before, we developed an algorithm to estimate the motion in blurred frames. We use this motion vector to estimate the parameters of motion blur. This method works for linear motion as do the other discussed methods. The angle is simply derived from the motion vector as follows:

$$\theta = \tan^{-1}\left(\frac{d_y}{d_x}\right) \tag{4.6}$$

Finding the length of the blur degradation is more difficult since it depends not only on the length of the motion vector but also on the camera components such as the lens and the shutter speed. We discovered that the distance between the two peaks in phase correlation corresponds to the extent of the motion blur.

$$d \propto l \tag{4.7}$$

Or equivalently we have:

$$d = k * l \tag{4.8}$$

With:

$$k \approx 1 \tag{4.9}$$

When d is the motion length derived from the motion vector, and l is the relative length of the motion blur between the two frames. We examined this relationship on both artificial and real data and observed that the accuracy of k would depend on the amount of added noise. With small amount of noise, in most cases k is very close to 1.

4.4.2 Out-of-Focus Blur

We decided not to use the methods for estimating the out-of-focus blur parameters from a single image, with the same premise explained in the previous section. Instead we use the relation between the frames in the sequence to estimate the blur parameters.

We apply the phase correlation to two frames. Strong peaks would indicate a match with high correlation between blocks. After determining the blur type, we use an iterative method to estimate the blur parameters, which would be discussed in Sections 4.5.2.

4.5 Compensating the Blur

After estimating the blur parameters, next step is to regenerate the blur. Our goal is to achieve a high PSNR between the naturally blurred block and the block with regenerated

blur. This separates our work from common blur generation applications which focus on the visual and perceptual properties of the generated blur.

(a) (b) (c)

Figure 4. 4: Example of out-of-focus blur reconstruction.

(a) A sharp frame, (b) A frame degraded by natural blur captured from the same source, (c) The sharp frame is blurred using the Pillbox model for out-of-focus blur.

We apply blur to the non-blurred blocks which match (according to phase correlation) to the blurred blocks in past/future frames and the residual is encoded and transmitted to the receiver along with the filter values. An example of blur reconstruction using the Pillbox-based model is shown in Figure 4.4. In this example, a sharp macroblock is artificially blurred to resemble the same macroblock when it's naturally blurred in the next frame. The block diagram of our blur compensation system is shown in Figure 4.5.

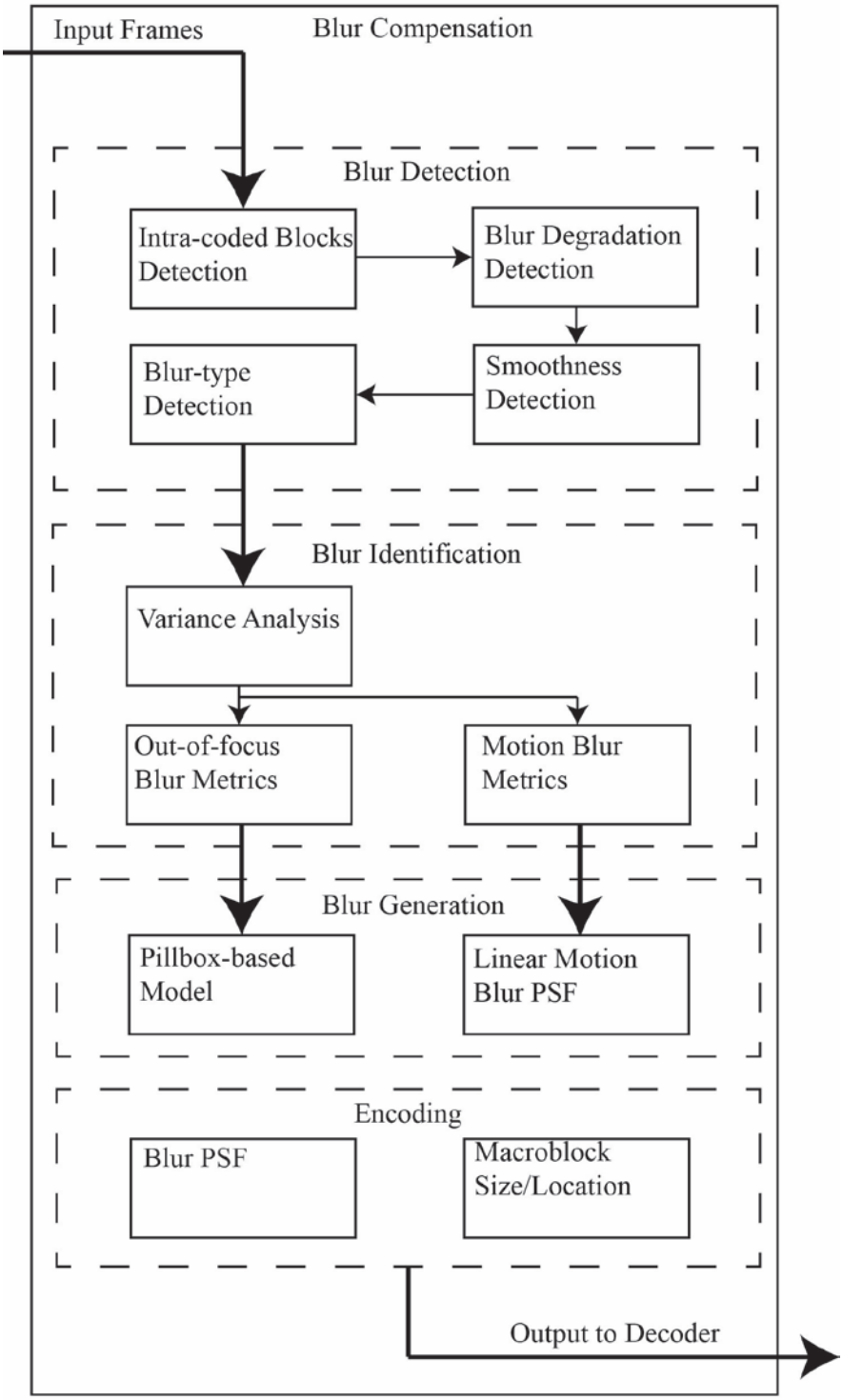


Figure 4. 5: The outline of our blur compensation system.

4.5.1 Motion Blur

In order to regenerate motion blur, we used the motion blur model in Equation (3.3) which is a blurring filter with the angle of 0°. For any other angle; this filter is rotated to the desired angle using Equation (3.4).

This filter is applied to local blocks inside the frame; however, this ideal filter might not result in a satisfactorily high PSNR compared to natural blur. This is partly due to the non-ideal properties of the camera which are not reflected in the existing blur models. For example, we observed that both motion blur and out-of-focus blur result in a blur offset (as we refer to it) in the blurred objects, which is not detectable by phase correlation. This blur offset is dependent to the location of object in the frame. For example, an out-of-focus blurred object in the far right side of the frame would have a relatively large blur offset to the right side compared to an object located near the center of the frame. We refer to this effect as an offset since applying this offset to the artificially blurred macroblock would greatly improve the PSNR. As a result, each macroblock requires a blur offset which is relative to its location inside the frame.

$$O_b(x) \propto x_1 \tag{4.10}$$

$$O_b(y) \propto y_1 \tag{4.11}$$

When $O_b(x)$ is the blur offset value in the x direction and $O_b(y)$ is the blur offset value in the y direction, and x_1 and y_1 are the coordinates of the block. Determining the value for the blur offset is difficult since it is different from a simple translational offset and it is caused

by the lens effects of the camera and nature of blurring. We estimate the best offset value by iterating over a small number of offset values which are estimated based on the location of the block inside the frame. The farther the distance from the frame center, the larger the offset values. We tested this method in frames degraded by real and artificial blurs and analyzed the results which showed improvement in the PSNR values.

4.5.2 Out-of-Focus Blur

To regenerate the out-of-focus blur we took the same approach as the case of motion blur by trying to find the highest PSNR values. Currently two common models are used in the literature to generate out-of-focus blur, the Gaussian model and the pillbox model; however, these models have not been evaluated and compared for the PSNR they produce in relation to real blur. As seen in Figure 3.6, the two models produce almost undistinguishable blurs to our visual perception. But it is not the same case in terms of the PSNR they produce.

To compare these models we used the generalized Gaussian distribution which is the general form of the Gaussian filter. This allowed us to investigate the Gaussian blur with more variables and therefore, more thoroughly.

We tested these models on several frames and concluded that in terms of PSNR, the pillbox model regenerates the out-of-focus blur degradation of a real camera far better than the popular Gaussian model in terms of PSNR. As a result, we used the pillbox model as our basis of generating the out-of-focus blur.

To further improve the PSNR we added the variables k_x and k_y to the basic pillbox model:

$$d[m,n;r] = \begin{cases} \frac{1}{k} & \sqrt{m^2/k_x + n^2/k_y} \leq r^2 \\ 0 & otherwise \end{cases} \quad (4.12)$$

This gives the filter an elliptic shape and helps to better resemble the effect of the lens on out-of-focus blur when the object is farther away from the center. We observed that in some cases, small values of the k_x and ky slightly improve the PSNR value.

As discussed before, the location of the out-of-focus blurred objects in the frame causes a blur offset which reduces the PSNR if not compensated. To overcome this issue we use the same method we used for the case of motion blur to compensate for the blur offset.

The most important value of the pillbox filter in regard of the achievable PSNR value is the radius. We would show that by iterating over a small number of radii, we can find the most suitable radius and acquire the highest PSNR. To do this, we iterate over values between 0.6 and 50. We chose these numbers since for values smaller than 0.6, the blur is almost non-existent and the degradation is negligible; also, a blur with a radius of 50 is extremely degraded and we chose it as the upper limit. To make the process faster we can use the following procedure. We start by calculating the PSNR for a small number of samples between 0.6 and 50 pixels and find the sample which results in the highest PSNR. We then halve the distance between this sample radius and the two neighboring radii; the smaller radius and the larger radius and calculate the sample with the highest PSNR. We continue this procedure until we find the largest PSNR. The reason is that by applying all possible radii between 0.6 and 50, the PSNR values show a pattern similar to a concave pattern and

ideally, the radius resulting in the highest PSNR can be determined using this procedure. However, in the experimental results shown in Chapter 5, we used the simple iteration over the radius values between 1 and 50 with an increment of 1.

4.5.3 Finding the Non-Blurred Macroblock

The next step is to apply the blurring filter to the non-blurred macroblock which matches a blurred macroblock in past/future frames. In order to make sure that we apply the blurring filter to the non-blurred macroblock (or the macroblock with smaller extent of blur) and not the blurred one, we perform a test to identify the macroblock with relatively more severe blur degradation. After finding the matching macroblocks using our phase correlation technique, we use standard deviation to detect the blurred macroblock (the macroblock with a larger extent of blur). We explained before that using standard deviation to detect blur degradation is an imperfect method since different images have different properties and blur degradation affects them differently. We observed the effect of blur degradation on standard deviation in several images. According to our measures, standard deviations of different images follow different patterns depending on the test image. This makes it difficult to determine whether or not the image is degraded by blur; however, in this application, it is practical to use standard deviation. This is due to the fact that we only need to determine which of the matched macroblocks has more blur degradation; we do this by comparing their standard deviation. The macroblock with higher standard deviation is non-blurred or it has less blur degradation in comparison to the macroblock being compared to. The algorithm is as follows:

For each pixel, the standard deviation of the pixel and its neighbors is calculated and located in the same (x, y) location inside an image which we refer to as the standard deviation map. The values are then added together and the total standard deviation of the frame is calculated. This value is compared to the standard deviation value of the frame which was matched using phase correlation. The frame/macroblock with higher standard deviation is marked as non-blurred.

4.6 Encoding for Blur Compensation

We need to transmit the blur data to the decoder in order to regenerate the same blur in the receiver. We do this by adding the encoded filter values of each blur compensated macroblock to the end of the encoded frame.

The information we encode are the type of the blur degradation, the filter values, the macroblock size and the location of the macroblock. This information is encoded using Huffman tables. Using Huffman, codewords with shorter lengths are assigned to values which are expected to have a greater chance of occurrence.

The encoded variables which transmitted to the decoder are as follows:

- Blur filter values
  - Motion blur
    - Length of blur
    - Angle of blur
    - Blur offset (m_x, m_y)
    - k_x and k_y

- Out-of-focus blur
  - Blurring radius
  - Revised magnitude
  - Blur offset (f_x, f_y)
  - k_x and k_y
- Macroblock values
  - Macroblock location
  - Macroblock size

The probability distributions of the mentioned variables are independent from each other; therefore, to encode the values more efficiently, an independent Huffman table is needed for each of the variables. We created the Huffman tables based on our experimental results and estimations. The value ranges which occurred more often during our experiments are assigned shorter code words.

The macroblock size consists of the width (w) and the height (h) variables; therefore, to encode the size more efficiently, instead of encoding the w and the h values, we encoded the w and the $h - w$ values. This was done due to the fact that most often, h and w are equal values; therefore, we only need to encode 0 instead of the h.

The values of k_x and k_y are between 1 and 2 with the intervals of 0.1. To encode these variables, we encoded the values of $10k_x - 10$ and $10k_y - 10$ with the intervals of 1 due to the following:

$$1 \leq k_x \& k_y \leq 2 \tag{4.13}$$

$$0 \leq (10k_x - 10) \,\&\, (10k_y - 10) \leq 10 \tag{4.14}$$

This makes the values suitable for encoding using our tables. The tables we used are shown in Table 4.1 to Table 4.7.

Category	Values	Codeword
0	64	11
1	128	10
2	256	01
3	512	00

Table 4.1: Huffman table for the width of the compensated blocks (w).

Category	Values	Codeword
0	1	1000
1	2,3	1001
2	4,…,7	101
3	8,…,15	11
4	16,…,31	01
5	32,…,63	00

Table 4.2: Huffman table for the radius of out-of-focus blur and the length of motion blur.

Category	Values	Codeword
0	0	1
1	1	01
2	2,3	001
3	4,7	0001
4	8,…,15	00001
5	16,…,20	00000

Table 4.3: Huffman table for $[(k_x, k_y) * 10 - 10]$.

Category	Values	Codeword
0	0	101
1	-1,1	11
2	-3,-2,2,3	01
3	-7,…,-4,4,…,7	1001
4	-15,…,-8,8,…,15	00
5	-31,…,-16,16,…,31	1000

Table 4.4: Huffman table for the blur offset values of motion blur compensated blocks (f_x, f_y).

Category	Values	Codeword
0	0	111
1	-1,1	101
2	-3,-2,2,3	100
3	-7,…,-4,4,…,7	1101
4	-15,…,-8,8,…,15	1100
5	-31,…,-16,16,…,31	0111
6	-63,…,-32,32,…,63	0110
7	-127,…,-64,64,…,127	0101
8	-255,…,-128,128,…255	0100
9	-511,…,-256,256,…,511	0011
10	-1023,…,-512,512,…,1023	0010
11	-2047,…,1024,1024,…,2027	0001
12	-4095,…,-2048,2048,…,4095	0000

Table 4.5: Huffman table for the (x, y) location of the compensated out-of-focus and motion blurred blocks.

Category	Values	Codeword
0	0	10
1	-1,1	00
2	-3,-2,2,3	01
3	-7,…,-4,4,…,7	11

Table 4.6: Huffman table for the blur offset values of motion blur compensated blocks (m_x, m_y).

Category	Values	Codeword
0	0	111
1	-1,1	110
2	-3,-2,2,3	101
3	-7,…,-4,4,…,7	100
4	-15,…,-8,8,…,15	011
5	-31,…,-16,16,…,31	010
6	-63,…,-32,32,…,63	001
7	-89,…,-64,64,…,89	000

Table 4.7: Huffman table for the blur angle of motion blur.

Chapter 5

Experimental Results

In this chapter we present the experimental results of the new blur compensation system and the algorithms developed. Performance of the algorithms would be evaluated by several experiments on various blurred frames. We used two general types of frames: sequences of frames degraded by natural blur, and sequences artificially blurred using blur models. The controlled features of artificial data are suitable to accurately measure the error and analyze the results. In literature, results and conclusions reported based solely on experiments on frames degraded by artificial blur are very common; however, to ensure the performance of the algorithms, we included natural blur in our experiments to evaluate the validity of our system on real data and to experience the variety of potential scenarios in a real video sequence.

To test our algorithms on frames degraded by artificial blur we applied artificial blur to various frames globally. Blur degradations were applied based on common blur models

discussed in previous chapters. We created artificially blurred frames degraded by both motion and out-of-focus blur with various blurring values.

We captured various frames with blur degradation. In addition, some existing video sequences with natural blur were tested. Frames of the video sequence were picked to meet several conditions and various scenarios. Following is a list of the test conditions and scenarios.

- **Texture and edge patterns of the frame.** Blur analysis is highly dependent on the texture in the frame. For example, as explained before, smooth patches of the frames are not suitable for common blur analysis methods and therefore, should be separated.
- **Noise level.** We evaluate our algorithms with various noise levels of the frames. Naturally blurred frames are naturally noisy due to the process of the blur degradation. Furthermore, Gaussian and Uniform noise is added to some of the artificially blurred frames.
- **Size of the frame patches.** As discussed before, size of the frame patch has an impact on the result of our system. Blur analysis on larger patches is more accurate; however, using larger block sizes results in coarser resolution since the frame is divided into larger patches. The sizes in pixels of the patches we used in most our experiments are 512x512, 256x256, 128x128 and 64x64.
- **Characteristics of the blur degradation.** The two general types of blur degradation are out-of-focus blur and motion blur. We examined natural and artificial cases of the out-of-focus blur. Also different types of motion blur degradation such as

translational, scaling, rotational blur and their combinations were analyzed; also, various extents of the blur degradation were included in the experiments.

5.1 Generation of Artificial Data

To generate the artificial data we selected various non-blurred frames based on their characteristics such as texture, edges and smooth areas. For motion blur, various displacements were applied to frames and motion blur was added according to Equation (3.3). For some cases, uniform and Gaussian noise were added to evaluate the resilience of algorithms to noise. In case of out-of-focus blur, blur was generated according to the pillbox model. We chose the pillbox over Gaussian because based on the premises explained in Section 4.5.2.

5.2 Real Data

To experiment on real data with out-of-focus blur, we captured several frames with different settings of the depth of focus of the camera while the camera was firmly set motionless. This results in an image with sharp objects in the foreground and out-of-focus blurred objects in the background, or vice versa depending on the depth of focus settings of the camera. We also used frames from a video sequence with the resolution of 1920x1080 which contains natural motion and out-of-focus blur and frames we captured with motion blur resulting from camera movement.

5.3 Performance of Block Matching in Blurred Frames

We evaluated the performance of the full-search block matching technique in the presence of blur. The block matching system was set to compare each frame of the video (target frame) against the previous frame (reference frame) to find the matching blocks between them. We would have:

$$NB = NNM + NM \tag{5.1}$$

The block matching process was performed using 4x4 blocks. Total number of 4x4 blocks in the target frame is presented as NB. Number of blocks which were not matched is presented as NNM and number of blocks which were matched with corresponding blocks in the reference frame are presented as NM. Then various extents of blur were added globally to every target frame and target frames were compared with the reference frames again. This way we know that every reference frame is being compared with a frame which has added blur. We would have:

$$NB = NNM_b + NM_b \tag{5.2}$$

Number of not-matched blocks and matched blocks after adding the blur are presented as NNM_b and NM_b respectively. Now we have:

$$NNB = (NB - NM_b) - (NB - NM) \tag{5.3}$$

The number of blocks which were not matched due to added blur is presented as NNB. In every case NM_b was smaller than NM and NNB was positive. We calculated the percentage of

reduction in the number of matched blocks after the blur was added which is shown as R according to the following:

$$R = 100 * \left(1 - \frac{NM_b}{NM}\right) \tag{5.4}$$

The average reduction in the blocked matches for tested videos was 20.29%.

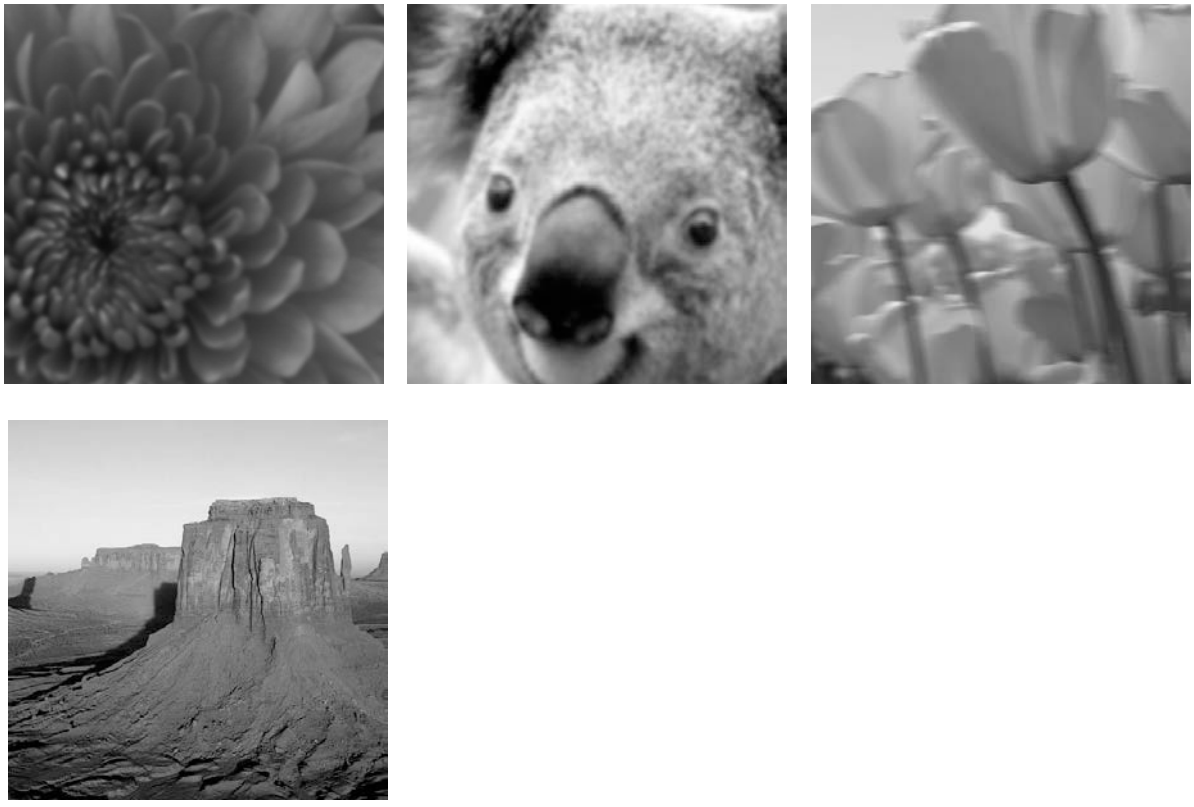
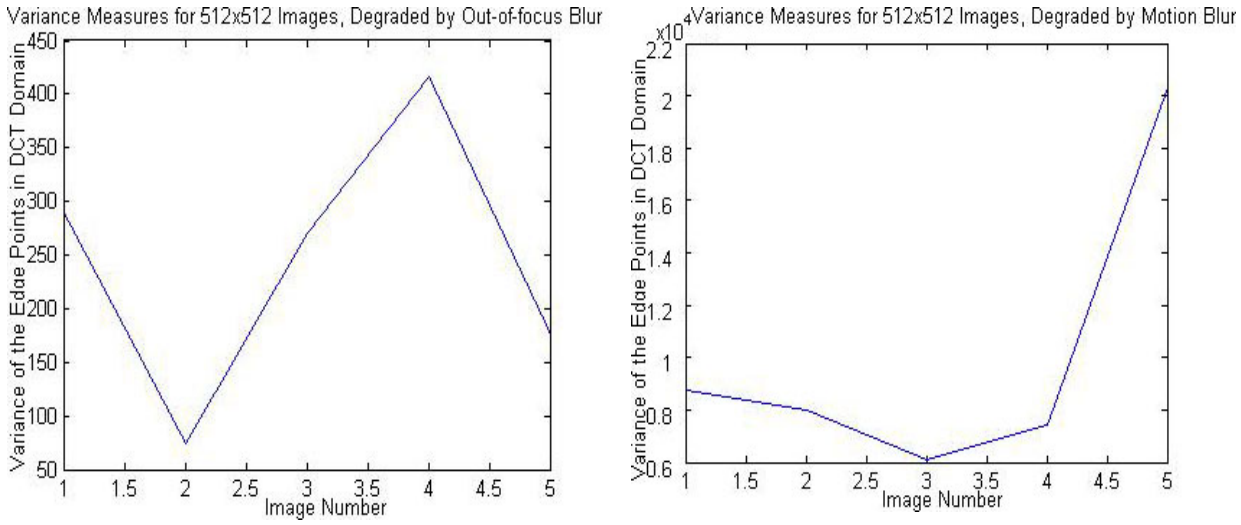


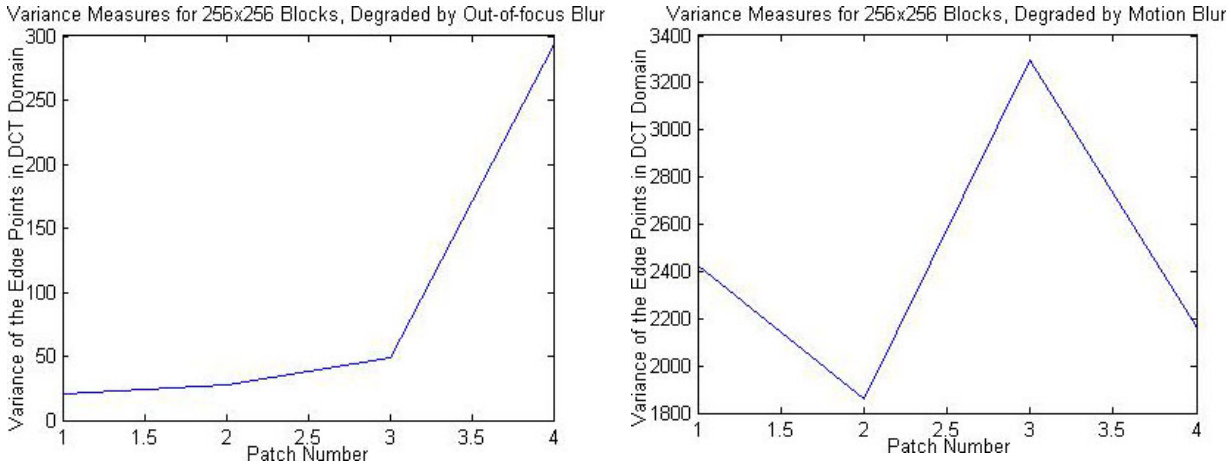
Figure 5.1: Examples of frames used in our experiments.



(a) (b)

Figure 5.2: Variance values of 512x512 images degraded by motion and out-of-focus blur

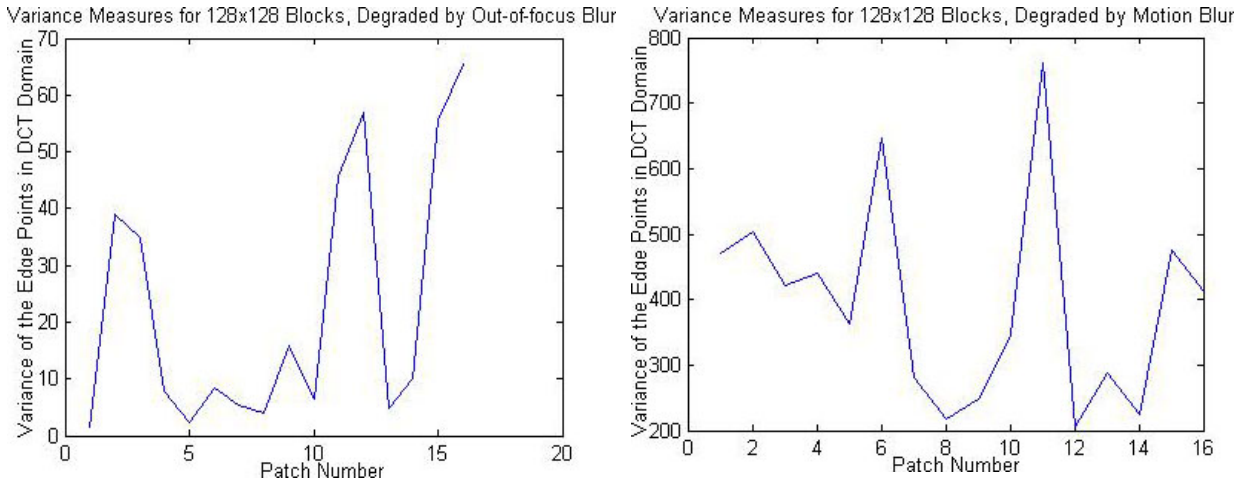
(a) Variance for Out-of-focus blur, (b) Variance for motion blur



(a) (b)

Figure 5.3: Variance values of 256x256 blocks degraded by motion and out-of-focus blur

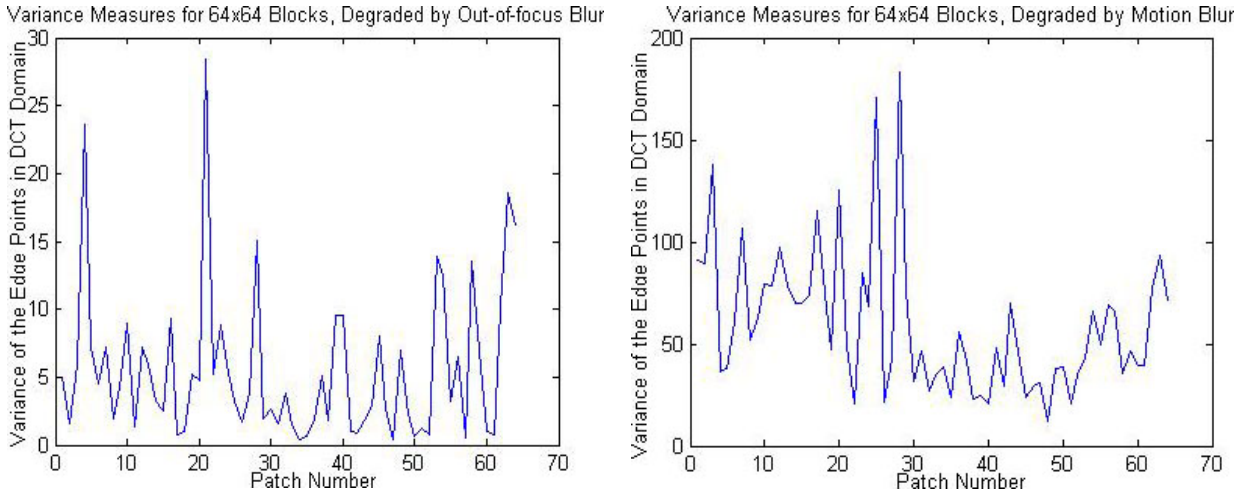
(a) Variance for Out-of-focus blur, (b) Variance for motion blur



(a) (b)

Figure 5.4: Variance values of 128x128 blocks degraded by motion and out-of-focus blur

(a) Variance for Out-of-focus blur, (b) Variance for motion blur



(a) (b)

Figure 5.5: Variance values of 64x64 blocks degraded by motion and out-of-focus blur

(a) Variance for Out-of-focus blur, (b) Variance for motion blur

5.4 A Novel DCT Based Blur Type Detection Algorithm

To evaluate the algorithm which we proposed in Section 3.4, we examined frames containing motion and out-of-focus blur degradation. Since the algorithm should be applied to non-smooth macroblocks, we picked frames with small smooth areas and high amounts of texture. Macroblocks of sizes 512x512, 256x256, 128x128 and 64x64 were tested in this experiment. Figure 5.1 shows examples of the test frames. The frames shown contain various textures and small smooth areas.

As described in Section 3.4, we analyze the type of the blur degradation by analyzing the variance value resulting from our algorithm. Figure 5.2, Figure 5.3, Figure 5.4 and Figure 5.5 show the variance values resulting from dividing a 512x512 frame into various block sizes. This shows that between a motion blurred block and an out-of-focus blurred block, there is a significant difference in the variance levels. We used this signifact difference to determine the type of the blur in each level.

We globally added motion and out-of-focus blur with a low to high range of blur extents to several frames, Table 5.1 shows the average results of the blur type detection. Here, we defined the percentage of error in blur type detection as follows:

$$Error = 100 * \frac{NF}{NT} \tag{5.5}$$

Total number of blocks with a type of blur that was not correctly detected is presented as NF, and the total number of examined blocks is presented by NT. The results show that in

frames with small smooth areas and high amounts of texture, our proposed algorithm has a 92.36% accuracy for blocks with the size of 128x128.

Size of the macroblocks	Average error in blur type detection
512x512	8.89%
256x256	10.28%
128x128	7.64%
64x64	10.00%

Table 5.1: Average error in blur type detection for various patch sizes

5.5 Improvement in Motion Blur Angle Estimation from a Single Frame

Here we evaluate the algorithm we proposed in Section 3.5.2. Table 5.2 shows the average improvement made to the angle estimation of motion blur. The average estimation error was calculated as follows:

$$Error = \frac{|\theta - \theta_e|}{180} * 100 \tag{5.6}$$

The blur angle and the estimated blur angle are presented as θ and θ_e respectively. The $|\theta - \theta_e|$ is divided by 180 since there are 180 possible angles of motion blur. It should be noted that a blur angle of θ° is the same as a blur angle of $(\theta + 180)^\circ$, therefore the range of motion blur angle is [0, 180) and not [0, 360).

We used the steerable filters to estimate the initial angle and searched for a better match in the range of -20 to +20 degrees of the neighbors of the initially estimated angle. Frames with small smooth areas were used in this experiment to ensure the reliability of the results. We greatly improved the accuracy of the angle estimation. As seen in the results, the estimation is more accurate in the larger block sizes; this is due to the fact that there are more samples and information to be analyzed in the spectrum domain.

5.6 Motion Estimation in Blurred frames Using Phase Correlation

Various translations and extents of motion blur were applied to frames to evaluate the displacement estimation. Examples of frames naturally degraded by blur are shown in Figure 5.7. Table 5.3 shows a few examples of the type of tests we performed on 512x512 patches of the Desert frame shown in Figure 5.1 and the performance of our algorithm, followed by Table 5.4 which shows the average results acquired from our experiments on several artificially blurred frames.

Patch size	Average error of the blur angle estimated by steerable filters	Average error of the blur angle estimated by our algorithm
512x512	3.77%	0.67%
256x256	3.78%	1.92%
128x128	4.79%	2.59%

Table 5.2: Comparison between the estimated angle and the corrected estimated angle in various patch sizes.

To compare our phase correlation algorithm with the phase-squared method presented in [21] we examined both algorithms in the presence of added noise. Gaussian and uniform noises were added to blurred and non-blurred frames and the motion vectors were calculated by both methods under the same conditions. The variety of frame patches examined is demonstrated in Figure 5.3. As seen in this figure, different patches of the frame contain various textures and smooth areas; therefore, different patches of the same frame would result in different amounts of estimation error.

Blur type of frame patch	Blur length	Blur angle	Error in motion vector estimation (x, y)	Error in blur angle estimation	Error in blur length estimation
No blur	NA	NA	(0, 0)	NA	NA
Motion blur	15	90	(0,0.5)	0	0
Motion blur	15	135	(0.5, 0.5)	0	0.79
Motion blur	10	0	(0.5, 0)	0	0
Out-of-focus blur	NA	NA	(0.5, 0)	NA	NA

Table 5.3: Example of tests performed on frame patches to test the performance of our phase correlation algorithm

Size	Average error in motion vector estimation (x, y)	Average error in blur angle estimation	Average error in blur length estimation
512x512	(0, 0)	0	1.76
256x256	(0. 0)	0	1.95
128x128	(0.75, 0.75)	0.10	2.85
64x64	(0.78, 0.78)	0.27	1.72

Table 5.4: Average performance of our phase correlation algorithm

The results of the comparison are shown in Table 5.5. As seen in this table, our proposed algorithm shows much more accuracy than the phase-squared method; also, our experiments on patches similar to frames in Figure 5.6 show that unlike the phase-squared method, our algorithm shows high accuracy in patches with larger smooth areas. As discussed before, in general, as patches get smaller, the chance of phase correlation producing erroneous results gets larger. This is due to the fact that there would be too little information in the spectrum domain to process. The result would highly depend on the content of the frame patch. Image patches with more edges and detail produce more reliable results while patches with large smooth areas are prone to error.

As seen in Table 5.5, for this frame, both methods show very high accuracy for various patches of size 512x512. But for patches of size 256x256 and 128x128, the phase-squared method shows very high error for some patches of the frame while our method is more accurate. In the phase-squared method, the phase is squared with the assumption that there is no added noise, but in fact the resilience of phase correlation to noise is reduced.

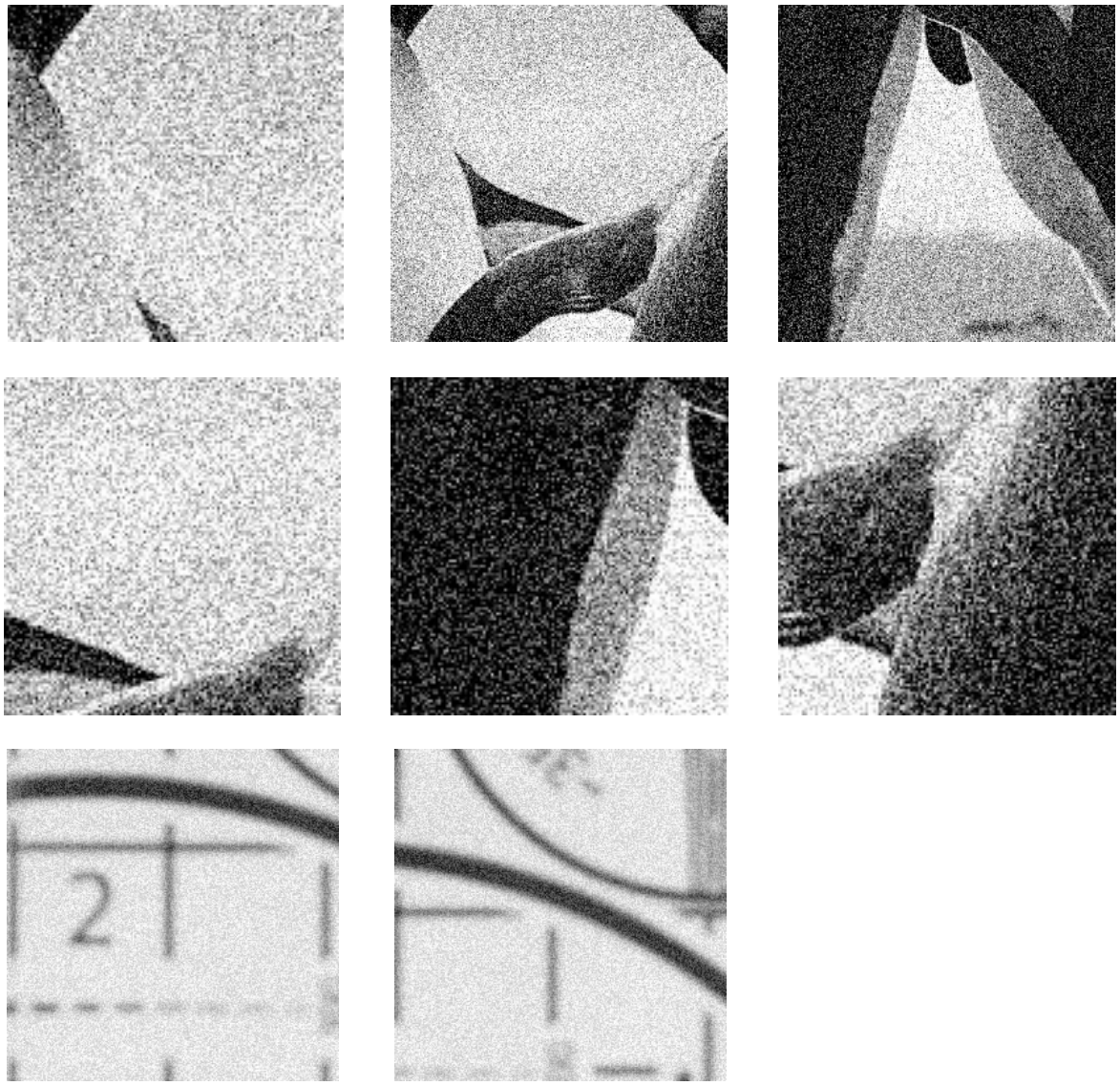


Figure 5.6: Examples of some 256x256 blurred frames with added noise used in our experiments. Various patches have different amount of smooth areas and various textures.

We also added high amounts of noise to blurred frames in order to find the lowest PSNR in which our algorithm performs well. The results are shown in Table 5.7. The results show that our proposed algorithm performs well with very low PSNR values until it begins to fail.

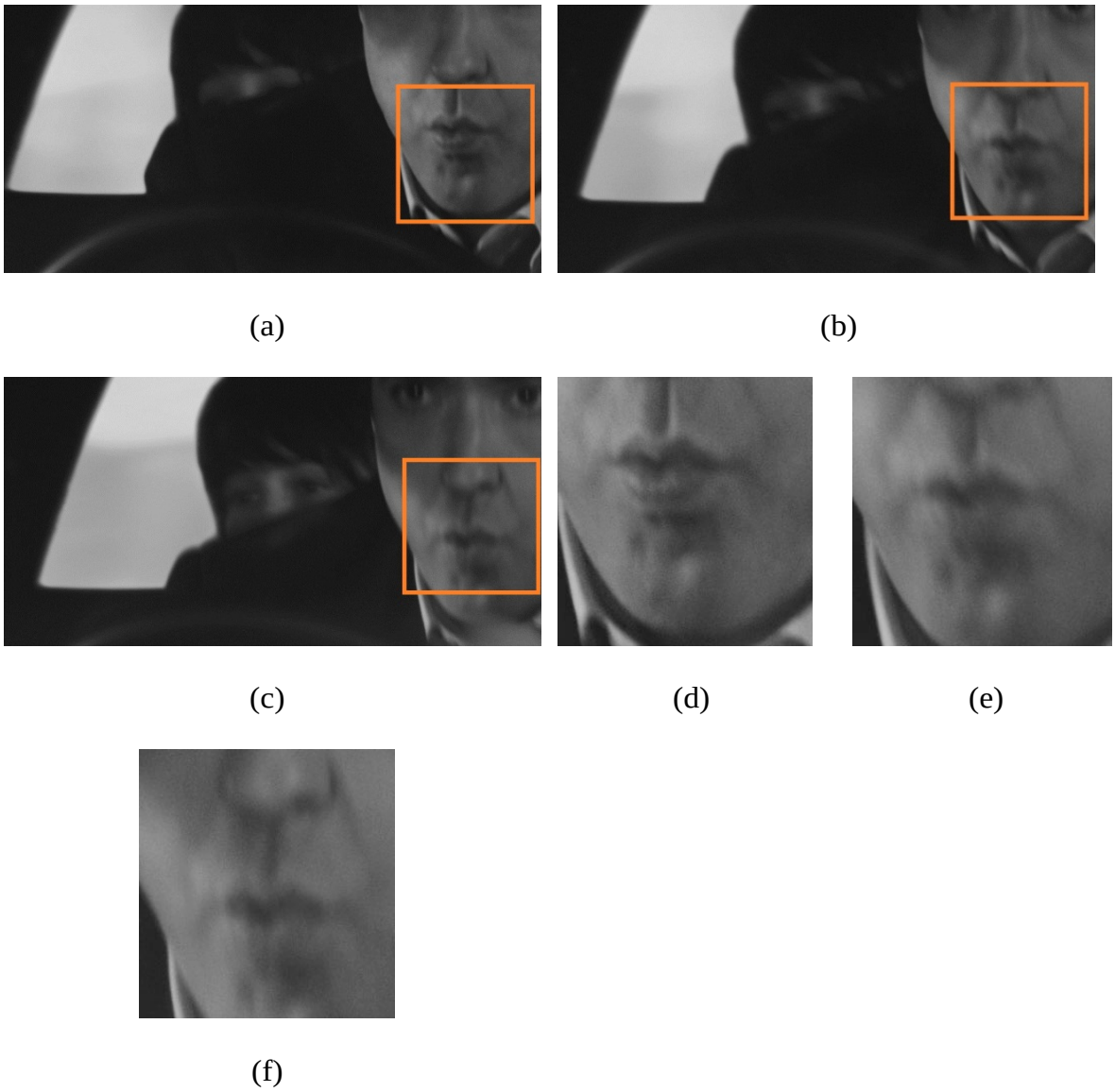


Figure 5.7: Examples of naturally blurred frames from video sequences used in our experiments.[5]

(a) Frame t which is a reference frame, (b) Frame $t + 1$, (c) Frame $t + 2$, (d) Patch from frame $t + 1$, (e) Patch from frame $t + 2$.

[5] Images taken from "2012" the movie, 2009, Centropolis Entertainment (as Centropolis), Columbia Pictures (presents), The Mark Gordon Company, Farewell Productions, Sony Pictures Home Entertainment.

(a) (b)

Figure 5. 8: Examples of naturally blurred frames from video sequences used in our experiments

(a) Frame t' which is a reference frame, (b) Frame $t' + 1$.

As it was shown in the results, the phase correlation algorithm was modified by our proposed algorithm for usage in frames degraded by blur. The algorithm is highly robust against noise as is expected from phase correlation and the overall results show that our results are more accurate than the results from the phase-squared method. The algorithm performs well in smaller patches of frames and is therefore suitable for local motion estimation in the presence of blur.

To examine the algorithm in the presence of naturally blurred frames and real case scenarios, we gathered some video frames with motion blur and translation motion. Some examples of the examined frames are show in Figure 5.7 and Figure 5.8. The average results are shown in Table 5.6.

Errors were calculated using the following equation:

$$Error = |x - x_e| \tag{5.7}$$

When x is the correct value and x_e is the estimated value.

Patch size	Average error in motion estimation by the phase-squared method (x, y)	Average error in motion estimation by the proposed method (x, y)
512x512	(0.13, 0.13)	(0, 0.12)
256x256	(31.34, 29.59)	(2.90, 3.91)
128x128	(13.65, 14.64)	(4.65, 5.12)
64x64	(9.57, 9.31)	(7.32, 7.53)

Table 5.5: Comparison between the phase-squared algorithm and our algorithm

Size	Average error in motion estimation by the proposed method (x, y)
(512, 512)	(1, 2.5)
(256, 256)	(1, 0.5)

Table 5.6: Average performance of our proposed phase correlation algorithm.

PSNR	Average error in motion estimation by the proposed method (x, y)
17.04	(0.5, 0.5)
14.59	(0.5, 0.5)
10.11	(1.5, 1.5)
7.31	(1, 1)
6.39	(0.5, 0.5)
6.25	(0.5, 0.5)
6.16	(20, 19.5)
6.12	(19, 97.5)

Table 5.7: Performance of our phase correlation algorithm in frames with very low PSNRs.

It should be noted that to improve the results when selecting small patches from the frame, Gaussian window was applied to the patches. Applying a window would reduce the ringing effect in frequency domain which is increased when taking a patch of the frame [24]. The artifacts would be more severe when the patches get smaller; therefore, applying a window is necessary.

5.7 Bit-rate Calculation

In this section we evaluate the bit-rate reduction obtained from our blur compensation system. As described in Section 5.3, to evaluate the performance of the block-matching

technique in the presence of blur artifacts, we added various extents of blur degradation to several video sequences which previously contained no or low amounts of blur degradation. We set the experiment in the way that every blurred block was compared against the non-blurred blocks in the previous frame for a match and evaluated the performance of block-matching before and after the blurring was added to frames. The results are show in Table 5.8. The results show that the percent of non-matched blocks is increased by a high amount and therefore, the number of intra-coded is increased; therefore, the compression ratio is decreased in frames degraded by blur artifacts and blur compensation is necessary to inter-code the blurred blocks.

Average percent of non-matched blocks/total number of blocks	Average percent of non-matched blocks/total number of blocks after blur was added
14.69%	34.98%

Table 5.8: Performance of the block-matching techniques in the presence of added blur degradation

After identifying the blur properties of blurred macroblocks, we simulate the blur in the matching non-blurred macroblocks. We then encode the residual error between the blurred blocks and artificially-blurred blocks using entropy coding after being quantized and transferred to DCT domain. The information about the filters is encoded using our Huffman tables and the overall reduction in bit-rate is calculated. We used 8x8 blocks in our coding.

Frames were captured by changing the depth of focus of the camera level and making sure that the camera is without any movement. Our goal was to find a suitable mathematical model to simulate out-of-focus blur. We evaluated the results obtained from the general Gaussian distribution model and the pillbox model. We found the best possible amounts of PSNR achievable by our blur model using and iteration technique explained before. Our experiments showed that in all cases, our pillbox-based model achieves a higher PSNR in the simulated blur; therefore, we based our algorithm on the pillbox model.

Table 5.9 shows our results obtained from out-of-focus blur compensation using our modified Pillbox model on frames degraded by natural blur. Percent of reduction in bit-rate was calculated using the following equation:

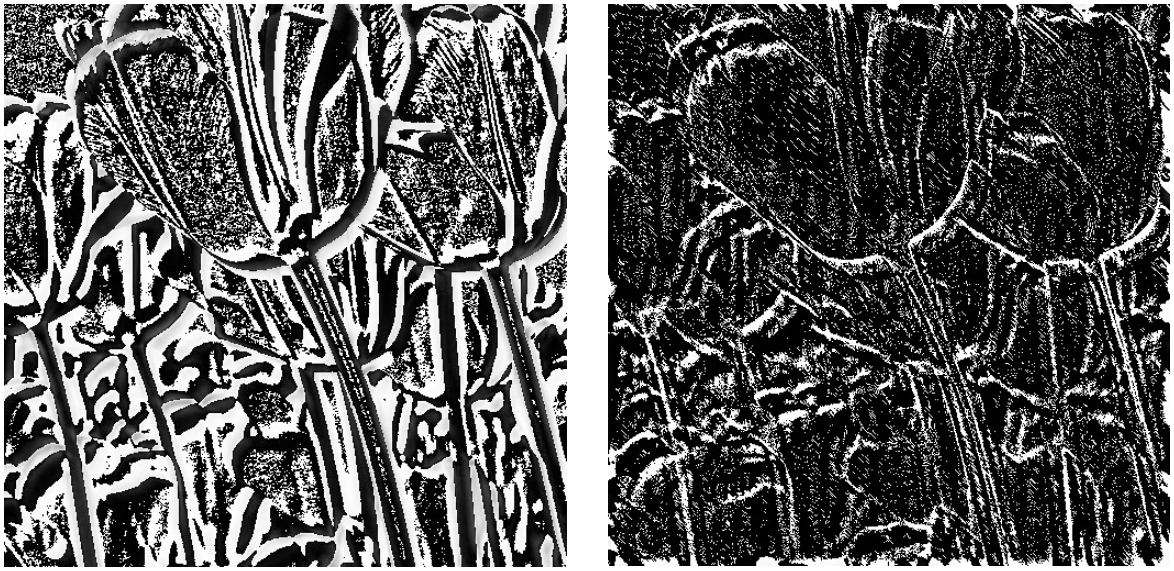
$$R = 100 * \left(1 - \frac{B}{B_n}\right) \tag{5.8}$$

When R is the percent of reduction in the bit-rate, B is the bitrate before blur compensation and B_n is the new bitrate after blur compensation.

Average PSNR before blur compensation	Average PSNR after blur compensation	Reduction in bit rate
19.26	29.58	20.78%

Table 5.9: Bit-rate reduction by compensation of out-of-focus blur

To compensate for motion blur, we followed the same procedure we used for out-of-focus blur. We identified the motion by using our phase correlation technique and obtained the angle from the motion assuming that the blurring pattern is translational. The blur length is obtained from the distance of the peaks resulting from phase correlation. Furthermore since blur length can be erroneous we iterate over a few neighboring lengths close to the obtained length to identify the most suitable length. The motion blur is then simulated based on the model in Equation (3.3). The residual error and filter properties are encoded similar to the case of out-of-focus blur explained before and the reduction in bit-rate is evaluated.



(a) (b)

Figure 5.9: The residual error before and after blur compensation

(a) The residual error before motion blur compensation, (b) The residual error after motion blur

The results obtained from the artificial and natural data are shown in Table 5.10 and Table 5.11 respectively. The highest reduction in bit-rate for natural motion blur in our experiments was 11.11%. The natural data is obtained from video sequences. To analyze our

blur compensation algorithm on artificial data, we applied global translational blur and global displacement to test images. Figure 5.9 demonstrates the residual errors between and after motion blur compensation in a test image; brighter pixels represent larger errors. It is seen that most of the error is caused around the edges in the image and the error is reduced by blur compensation. The bit reduction is higher in larger macroblocks due to the better performance of our phase correlation algorithm in larger macroblocks. In the real case scenarios, the motions are commonly more complex than simple translational motions. Both camera and object motions include complex motions such as rotation and changes in scale which result in rotational and scaling patterns of blur degradation. Our current system is unable to compensate for these types of motions and blur degradations; therefore, the overall performance of our motion blur compensation is lower than the performance of our out-of-focus blur compensation since the tested videos contain a great amount of complex motions.

Size	Average PSNR before blur compensation	Average PSNR after blur compensation	Average reduction in bit rate
512x512	24.42	48.05	54.53%
256x256	24.46	43.57	40.75%
128x128	24.04	36.83	27.92%
64x64	19.40	21.19	10.43%

Table 5.10: Bit-rate reduction by compensation of motion blur in images degraded by artificial motion blur

Average PSNR before blur compensation	Average PSNR after blur compensation	Reduction in bit rate
28.99	31.42	4.99%

Table 5.11: Bit-rate reduction by compensation of motion blur

Chapter 6

Conclusion and Future Work

6.1 Concluding Remarks

The main objective of the research presented in this thesis was to improve the video compression ratio in the video frames degraded by blurring artifacts. Current compression techniques are unable to compensate for the reduction in compression ratio caused by blur degradation. This is due to the fact that in the presence of blur, the predicted block cannot be accurately reconstructed from the reference block using motion vectors. We also showed that the block-matching motion estimation technique and the phase correlation technique are not resilient to blurring artifacts; therefore, the number of blocks predicted from the reference blocks is reduced. In this research we presented a novel approach to blur compensation. In this approach, areas of the video frames degraded by blur are identified using the blur detection techniques such as DCT-based techniques. We presented a novel

algorithm to detect the two main types of blur: motion blur and out-of-focus blur. The PSF of the blur degradation is then estimated using the blur identification techniques. We presented an algorithm to improve the accuracy of the identified angle estimated by current blur identification techniques. We have proposed a novel algorithm to modify the phase correlation technique for usage in video frames degraded by blur and showed the effectiveness and high accuracy of the proposed algorithm. We showed that using phase correlation we can estimate the length and angle of a linear motion blur and recreate the blur using the current blur model. We also showed that we can recreate out-of-focus blur degradations by the Pillbox-based model and using an iterative technique. We evaluated our algorithms on frames degraded by natural and artificial blurring artifacts. Our results show the effectiveness and great potential of the proposed blur compensation technique. We also highlighted the current limitations of the algorithm in different scenarios of motion and blur degradation.

6.2 Discussion on Potential Future Research Directions

We currently use an iterative process to obtain some of the blurring values (radius of the out-of-focus blur and the length of the motion blur) which result in the highest PSNR; developing faster methods for this purpose would be beneficial. As discussed before, one of the shortcomings of our current system is to compensate for some of the blurring and motion types. Compensation of rotational and scaling blur types would greatly improve the performance of the blur compensation system. As discussed before, phase correlation can be modified to estimate the rotational and scaling motions in non-blurred frames; therefore,

analyzing the effects of blur degradation on these techniques could be the first step to analyze the rotational and scaling blur types; also, the current motion blur model is unable to model the translational and scaling blur degradations; therefore, developing a model to accurately compensate these types of blur is beneficial. We discussed that current blur detection techniques are unable to accurately distinguish between the blurred and smooth areas of the frame. Doing so would be very beneficial in blur analysis as we currently have a DCT based method under development which shows great promise for future work.

References

[1] Q. Wang and R. J. Clarke, “Motion Estimation and Compensation for Image Sequence Coding,” *Signal Process.: Image Commun*., vol. 4, pp. 161–174, Apr. 1992.

[2] M. Budagavi, “Video compression using blur compensation,” *IEEE International Conference on Image Processing (ICIP)*, vol. 2, pp. II.882–II.885, Sep. 2005.

[3] C. Pradhan, “Survey on Block Matching Algorithms for Motion Estimation,” *International Journal of Computer Applications*,” vol. 46, no.16, pp. 6–10, Sep. 2012.

[4] S.I.A. Pandian, G.J. Bala, and B.A. George, “A Study on Block Matching Algorithms for Motion Estimation,” *International Journal on Computer Science and Engineering*, vol. 3, no. 1, pp. 34–44, Jan. 2011.

[5] R. Li, B. Zeng, and M. L. Liou, “A New Three-step Search Algorithm for Block Motion Estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, pp. 438–442, Aug. 1994.

[6] W.F. Al Maki and S. Sugimoto, “Blind Deconvolution Algorithm for Spatially-Invariant Motion Blurred Images Based on Inverse Filtering and DST,” *Int. J. Circuits Syst. Signal Process*., vol. 1, no. 1, pp. 92–100, Apr. 2007.

[7] N.L. Max and D.M. Lerner, “A Two-and-a-Half-D Motion-Blur Algorithm,” *Computer Graphics*, vol. 19, no. 3, pp. 85-93, Jul. 1985.

[8] A. Bovik, “The Essential Guide to Image Processing,” *Academic Press*, 2009

[9] S. Smith, “The Scientist and Engineer's Guide to Digital Signal Processing,” *California Technical Pub.*, 1997

[10] R. Liu, Z. Li, and J. Jia, “Image Partial Blur Detection and Classification,” *CVPR: Computer Vision and Pattern Recognition*, pp. 1-8, Jun. 2008.

[11] E. Kalalembang, K. Usman and I. P. Gunawan, “DCT-based Local Motion Blur Detection,” *International Conference on Instrumentation, Communication, Information Technology, and Biomedical Engineering*, pp. 1-6, Nov. 2009.

[12] D. Grimaldi, Y. Kurylyak, and F. Lamonaca,” Detection and Parameters Estimation of Locally Motion Blurred Objects,” *IDAACS: IEEE Conference on Intelligent Data Acquisition and Advanced Computing Systems*, vol. 1, pp. 483–487, Sep. 2011.

[13] http://www.mediacollege.com/video/camera/shutter/shutter-vs-fps.html

[14] I. M. Rekleitis, “Steerable Filters and Cepstral Analysis for Optical Flow Calculation from a Single Blurred Image,” *Vision Interface*, pp. 159–166, May. 1996.

[15] W. T. Freeman, and E. H. Adelson, “The Design and Use of Steerable Filters,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 9, pp. 891–906, Sep. 1991.

[16] M. E. Moghaddam and M. Jamzad, “Blur Identification in Noisy Images Using Radon Transform and Power Spectrum Modeling,*” in Proceedings of the 12th IEEE International Work- shop on Systems, Signals and Image Processing (IWSSIP ’05),* pp. 347–352, Sep. 2005.

[17] J. Gibson, “The Communications Handbook,” *CRC Press*, 1997

[18] R. Fabian and D. Malah, “Robust Identification of Motion and Out-of-Focus Blur Parameters from Blurred and Noisy Images,” *CVGIP: Graphical Models and Image Processing*, vol. 53, no. 5, pp. 403-412, Sep. 1991.

[19] M. E. Moghaddam and M. Jamzad, “Motion Blur Identification in Noisy Images Using Fuzzy Sets,” *In: Proceedings of the Fifth IEEE International Symposium on Signal Processing and Information Technology*, pp. 862–866, 2005

[20] P. A. Sarginson, “MPEG-2: Overview of the system layer,” *BBC Research and Development Report*, BBC RD 1996/2, 1996

[21] S. Tang, Y. Wang, and Y. Chen, “Blur Invariant Phase Correlation in X-Ray Digital Subtraction Angiography” *CME 2007: IEEE/ICME International Conference on Complex Medical Engineering*, pp. 1715-1719, May. 2007.

[22] B.S. Reddy and B.N. Chatterji, “An FFT-based Technique for Translation, Rotation and Scale-invariant Image Registration,” *IEEE Transactions on Image Processing 5*, pp. 1266–1271, 1996.

[23] G. J. Brostow and I. Essa, “Image-Based Motion Blur for Stop Motion Animation,” *In Fiume, E., ed.: SIGGRAPH 2001, Computer Graphics Proceedings*, pp. 561–566, 2001.

[24] J. Proakis and D. Manolakis, “Digital signal processing,” *Pearson Prentice Hall*, 2007

[25] G. K. Wallace, “The JPEG Still Picture Compression Standard,” *IEEE Trans. Consumer Electronics*, Vol. 38, No. 1, Feb. 1992.

[26] B. M. T. Pourazad, C. Doutre, M. Azimi and P. Nasiopoulos, “HEVC: The New Gold Standard for Video Compression: How Does HEVC Compare with H.264/AVC?” IEEE *Consumer Electronics Magazine*, Vol. 38, no. 3, pp. 36-46, Jul. 2012.

[27] B. Fufht, “A Survey of Multimedia Compression Techniques and Standards. Part I. JPEG Standard,” *Real Time Imaging*, vol. 1, pp. 49–67, Apr. 1995

[28] N. Kamaci and Y. Altunbasak, “Performance Comparison of the Emerging H.264 video Coding Standard with the Existing Standards,” in *Proc. 2003 International Conference on Multimedia and Expo.*, vol. 1, pp. I.345-I.348, Jul. 2003.

[29] T. Wiegand, G. J. Sullivan, G. Bjontegaard and A. Luthra, “Overview of the H.264/AVC Video Coding Standard,” *IEEE Transactions on Circuits and Systems*,” vol. 13, no. 7, pp. 560–576, Jul. 2003.

[30] J. Nightingale, Q. Wang and C. Grecos, “HEVStream : A Framework for Streaming and Evaluation of High Efficiency Video Coding (HEVC) Content in Loss-prone Networks,” *IEEE Transactions on Consumer Electronics*, vol. 58, no. 2, pp. 404–412, May. 2012.

[31] T. L. Harrington and M. K. Harrington, “Perception of Motion Using Blur Pattern Information in the Moderate and High-velocity Domains of Vision,” *Acta Psychologica*, vol. 48, no. 1-3, pp. 227-237, Jan. 1981.

[32] Y. W. Huang, C. Y. Chen, C. H. Tsai, C. F. Shen, and L. G. Chen, “Survey on Block Matching Motion Estimation Algorithms and Architectures with New Results,” *J. VLSI Signal Process.*, vol. 42, no. 3, pp. 297–320, Mar. 2006.

[33] M. Potmesil and C. Indranil, “Modeling Motion blur in Computer-Generated Images,” *In Proceedings of the 10th annual conference on Computer graphics and interactive techniques, ACM Press*, vol. 17, pp. 389–399, Jul. 1983.

[34] C. Shimizu, A. Shesh, and B. Chen, “Hardware Accelerated Motion Blur Generation,” *Technical report, Department of Computer Science, University of Minnesota at Twin Cities*, vol. 22, no. 3, 2003.

[35] M. Cannon, “Blind Deconvolution of Spatially Invariant Image Blurs with Phase,” *IEEE Trans. Acoust. Speech, Signal Processing*, vol. 24, no. 1, pp. 58–63, Feb. 1976.

[36] J. Oliveira, M. Figueiredo, and J. Bioucas-Dias, “Blind Estimation of Motion Blur Parameters for Image Deconvolution,” *presented at the Iberian Conf. Pattern Recognition and Image Analysis. Part II*, pp. 604-611, Jun. 2007.

[37] Q. Shan, J. Jia, and A. Agarwala, “High-Quality Motion Deblurring from a Single Image,” *ACM Transactions on Graphics*, vol. 27, no. 3, article no. 73, Aug. 2008.

[38] M. Ben-Ezra, and S. K. Nayar, “Motion-based Motion Deblurring,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 6, pp. 689–98, Jun. 2004

[39] Q. Han, M. Zhang, C. Song, Z. Wang, and X. Niu, “A Face Image Database for Evaluating Out-of-Focus Blur,” *in ISDA*, vol. 2, pp. 277–282, Nov. 2008

[40] X. Marichal, W-Y. Ma, and H. Zhang, “Blur Determination in the Compressed Domain Using DCT Information,” *in Proc. IEEE Int. Conf. Image Processing*, vol. 2, pp. 386–390, Oct. 1999.

[41] S. Wu, Z. Lu, E. P. Ong and Lin, W, “Blind Image Blur Identification in Cepstrum Domain,” *16th International Conference on Computer Communications and Networks*, pp. 1166–1171, Aug. 2007.

[42] D. Li, R. M. Mersereau, and S. Simske, “Blur Identification Based on kurtosis Minimization,” *IEEE Proc. Int. Conf. Image Process.*, vol. 1, pp. 905–908, Sep. 2005.

[43] L. Xu and J. Jia, “Two-Phase Kernel Estimation for Robust Motion Deblurring,” *In ECCV*, pp. 157-170, 2010.

[44] M. Sakano, N. Suetake and E. Uchino, “Robust Identification of Motion Blur Parameters by Using Angles of Gradient Vectors,” *ISPACS: International Symposium on Intelligent Signal Processing and Communications*, pp. 522-525. Dec. 2006.

[45] M. E. Moghadam and M. Jamzad, “Linear Motion Blur Parameter Estimation in Noisy Images Using Fuzzy Sets and Power Spectrum,” *EUROSIP J. Applied Signal Processing*, pp.1-8, 2007

[46] H. Tong, M. Li, H. J. Zhang, and C. Zhang, “Blur Detection for Digital Images Using Wavelet Transform,” *in Proc. IEEE Int. Conf. Multimedia Expo*, pp. 17–20, Jun. 2004.

[47] F. Crete, T. Dolmiere, P. Ladret, and M. Nicolas, “The Blur Effect: Perception and Estimation with a New Noreference Perceptual Blur Metric,” *SPIE-IS&T Electronic Imaging*, vol. 6492, pp. 1–11, Jan. 2007.

[48] I. M. Rekleitis, “Visual Motion Estimation based on Motion Blur Interpretation,” M.Sc. thesis, McGill Univ., Montreal, Canada, 1995

[49] C. Fogg, D. LeGall, J. Mitchell, and W. Pennebaker, “MPEG Video Compression Standard,“ *International Thomason Publishing*, 1997.

[50] K. Rijkse, “H.263: Video Coding for Low-Bit-Rate Communication,” *IEEE Commun. Mag*., vol. 34, no. 12, pp. 42–45 Dec. 1996.

[51] ITU-T Rec. H.263, “Video Coding for Low Bitrate Communication,” 1996

[52] P. Toft, "The Radon Transform: Theory and Implementation," PhD thesis, Technical Univ. of Denmark, 1996.