MULTIFRAME SUPER RESOLUTION WITH JPEG2000 COMPRESSED
IMAGES

Thesis

Submitted to

The School of Engineering of the

UNIVERSITY OF DAYTON

In Partial Fulfillment of the Requirements for

The Degree of

Master of Science in Electrical Engineering

By

Barath Narayanan Narayanan

Dayton, Ohio

May, 2013

MULTIFRAME SUPER RESOLUTION WITH JPEG2000 COMPRESSED
IMAGES

Name: Narayanan, Barath Narayanan

APPROVED BY:

Russell C. Hardie, Ph.D.
Advisory Committee Chairman
Professor
Electrical and Computer Engineering

Eric Balster, Ph.D.
Committee Member
Assistant Professor
Electrical and Computer Engineering

John Loomis, Ph.D.
Committee Member
Professor
Electrical and Computer Engineering

John G. Weber, Ph.D.
Associate Dean
School of Engineering

Tony E. Saliba, Ph.D.
Dean, School of Engineering
& Wilke Distinguished Professor

© Copyright by
Barath Narayanan Narayanan
All rights reserved
2013

ABSTRACT

MULTIFRAME SUPER RESOLUTION WITH JPEG2000 COMPRESSED IMAGES

Name: Narayanan, Barath Narayanan
University of Dayton

Advisor: Dr. Russell C. Hardie

This research is primarily focused on combining the techniques of Image processing (Super Resolution) with Image compression. Historically, Joint Photographic Experts Group 2000 (JPEG2000) image compression and multiframe Super Resolution (SR) image processing techniques have evolved separately. In this research, we propose and compare novel processing architectures for applying multiframe SR with JPEG2000 compression. We focus on the adaptive wiener filter (AWF) method of SR and study the SR performance as JPEG2000 is incorporated in different ways. In particular, we perform compression prior to SR and compare this to compression after SR. We also compare independent frame compression with difference frame compression. We find that the effects of compression can be reduced by increasing the signal-to-noise ratio in the

correlation model for the AWF SR method, providing a novel approach to treating the compression artifacts. The experimental results include the use of simulated imagery for quantitative analysis. We also include real video results for subjective analysis. A chirp pattern is included in the scene to aid in the subjective analysis with regard to resolution and aliasing.

ACKNOWLEDGEMENTS

I enjoyed my study and research very much since I joined the research group led by Prof. Russell C. Hardie. So my greatest gratitude would be extended to my advisor, Prof. Russell C. Hardie, for his constant support and valuable instruction. Without his direct guidance, I could have not completed my graduate study at the University of Dayton. Besides his expertise in the research area, his dedication to Electrical Engineering contributes to my lifelong development, which is far behind my master's thesis. I will always appreciate it.

I would like to express my gratitude to other committee members, Prof. Eric Balster and Prof. John Loomis for their helpful discussions and kind motivations. I would like to extend my thanks to Prof. Guru Subramanyam for his constant support and encouragement. Additionally, I would like to thank Graduate School Committee for providing me Graduate Summer Fellowship Award in 2012 which really motivated to work hard in my research.

And last, but not least, I'm grateful for the support of my mom, Sukanya Narayanan who is currently residing in India and other members of my family & friends in various parts of world, thousands of miles away from Dayton. They always give me spiritual support and encouragement to accomplish the task.

TABLE OF CONTENTS

ABSTRACT.....	iv
ACKNOWLEDGEMENTS.....	vi
TABLE OF CONTENTS.....	vii
LIST OF FIGURES.....	ix
LIST OF TABLES.....	xi
CHAPTER	
1. INTRODUCTION	1
1.1 Super Resolution.....	1
1.2 Super Resolution with Image Compression.....	1
1.3 Objective of Study.....	2
1.4 Significance of the Study.....	3
1.5 Organization of the Thesis.....	4
2. ADAPTIVE WIENER FILTER ALGORITHM OF SUPER RESOLUTION	5
2.1 Observation Model.....	5
2.2 Adaptive Wiener Filter of SR Processing.....	8
3. ARCHITECTURES FOR SR WITH JPEG2000 COMPRESSION.....	11
3.1 SR after JPEG2000 Compression.....	11
3.1.1 Individual Frame Method.....	12

3.1.2 Difference Frame Method.....	13
3.2 JPEG2000 Compression after the Application of SR	15
4. EXPERIMENTAL RESULTS	17
4.1 Simulated Data.....	17
4.1.1 Bicubic Interpolation after JPEG2000 Compression.....	18
4.1.2 Individual Frame Method of JPEG2000 Compression before SR....	19
4.1.3 Difference Frame Method of JPEG2000 Compression before SR...21	
4.1.4 JPEG2000 Compression after the Application of SR.....	22
4.1.5 Overall Comparison of Various SR/Compression Methods.....	23
4.1.6 SNR Adjustment for Different Levels of Compression.....	26
4.1.7 Other Simulated Results.....	27
4.2 Real Video Data.....	33
5. SUMMARY	36
BIBLIOGRAPHY.....	38
APPENDIX.....	41

LIST OF FIGURES

Figures

2.1 Observation model with a set of corresponding LR frames.....	7
2.2 Alternative observation model replacing the motion and combining of LR frames.....	7
2.3 Overview of AWF SR algorithm.....	10
3.1 Overview of SR after JPEG2000 compression.....	12
3.2 Difference frame method of compression.....	14
3.3 Difference frame method of decompression.....	15
3.4 Overview of SR before JPEG2000 compression.....	16
4.1 Kodak lighthouse image.....	18
4.2 Simulated LR reference frame with CR =8 and $L=4$ bicubic interpolation.....	19
4.3 SR after individual frame method of compression (CR=8) using optimum SNR.....	20
4.4 SR after individual frame method of compression (CR=8) using SNR value of 4....	20
4.5 SR after difference frame method of compression (CR=8) using optimum SNR.....	21
4.6 Individual frame method of compression (CR=8) after SR using optimum SNR.....	22
4.7 Mean absolute error versus CR for the various SR/compression methods.....	24
4.8 Region of interest of the Kodak lighthouse image.....	25
4.9 Comparative plot between optimum SNR for first CR and for each CR.....	26
4.10 Ideal HR images used for simulation results from Kodak database.....	27

4.11 Kodak parrot HR image.....	28
4.12 Simulated LR reference frame with CR =8 and $L=4$ bicubic interpolation.....	29
4.13 SR after individual frame method of compression (CR=8) using optimum SNR....	29
4.14 SR after difference frame method of compression (CR=8) using optimum SNR....	30
4.15 Individual frame method of compression (CR=8) after SR using optimum SNR....	30
4.16 Region of interest of the Kodak parrot image.....	31
4.17 An individual frame in the original real video sequence.....	34
4.18 Region of interest (ROI – I) of real data for various SR/compression methods.....	35
4.19 Region of interest (ROI – II) of real data for various SR/compression methods.....	36

LIST OF TABLES

TABLES

4.1 Mean absolute error obtained using various SR/compression with CR=8 using optimum value SNR and $\rho = 0.7$	32
4.2 Mean square error obtained using various SR/compression with CR=8 using optimum value SNR and $\rho = 0.7$	32

CHAPTER 1

INTRODUCTION

1.1 Super Resolution

Multiframe super-resolution (SR) is a post processing technique designed to reduce aliasing and enhance resolution for some detector limited imaging systems [1]. As mentioned in [2] and [3], SR methods generally fuse a set of low-resolution (LR) observed frames with a common field of view. The result is a high-resolution (HR) image with reduced aliasing. These methods assume the presence of motion between frames that can be estimated or known with sub-pixel accuracy. The motion allows each frame to capture unique samples of the scene, effectively increasing the sampling frequency of the imaging sensor. SR can be applied to produce a single output frame or produce video output by employing a moving temporal window of frames [4]. Under the appropriate conditions, SR techniques have proven to be highly successful at providing meaningful resolution enhancement for images and videos.

1.2 Super Resolution with Image Compression

In previous work, some SR techniques have been applied to compressed imagery and video signals. For video signals, most of this work has focused on processing LR imagery

that has been compressed by MPEG and H.26x methods. In [6] and [7], a compression method of sub-sampling video prior to MPEG4 compression and SR applied post decompression shows SNR improvement over merely MPEG4 compressing the original full resolution signal. SR techniques have been applied to video signals in several ways. In [8], P- and B-frames in an H.264 compression technique are sub-sampled prior to compression. At the decompression end, SR is used to resolve the P- and B-frames using the I-frames as the training samples. In addition to compressed video, SR techniques have been applied to independently compressed images. In [9], SR is applied to imagery after JPEG compression and shows degradation to the SR performance when compression is high. In [10], a SR technique is designed which is robust to JPEG compression artifacts. Also, in [11] a SR technique is developed which applies SR in the transform domain, enabling decompression engines with embedded SR enhancements. Also, [12] has developed an SR technique to restore details in imagery that have been degraded due to JPEG2000 compression. With the exception of [12], the combination of SR and JPEG2000 has not been widely studied. We believe this is an important area to investigate.

1.3 Objective of the Study

SR research has grown significantly in recent years [1]. However, the majority of SR research has focused on the use of raw uncompressed image data obtained directly from an imaging sensor. In many practical imaging applications, the acquired video frames must be compressed in order to be transmitted through a band-limited channel or stored using a limited file size. A powerful compression method is the Joint Photographic Experts Group 2000 (JPEG2000) compression [5]. Recently, studies have shown that

JPEG2000 is a good choice for compression for high-quality, high-resolution video [13]. JPEG2000 compression uses wavelet domain techniques and supports both the lossy and lossless modes. In 2004, the motion picture industry, specifically Digital Cinema Initiatives, announced JPEG2000 as the standard for digital delivery of all motion pictures [14]. In light of this, as well as the emergence of some important new classes of SR algorithms, important questions are raised regarding how to best incorporate the benefits of both super-resolution and JPEG2000 image compression. For example, how does compression before SR compare to compression applied after SR? Also, how does SR performance degrade with compression ratio using JPEG2000? What can be done to established SR methods to allow them to perform their best with compressed LR input frames? We shall attempt to address these and other questions in the current research.

1.4 Significance of the Study

In this research, we provide a novel study of the performance of SR with JPEG2000 compression. We employ a relatively new SR technique based on the Adaptive Wiener filter (AWF). The AWF is a computationally efficient SR method, suitable for real-time implementation, with generally good performance [4]. Here we propose several architectures for combining AWF SR with JPEG2000 compression. These include systems that apply compression prior to SR and those that do compression after SR processing. We study how SR performance is impacted by a wide range of compression ratios (CRs). We also investigate methods for modifying the correlation models used by the AWF SR method to minimize the impact of compression of the SR output. Based on our findings, we make some practical and important recommendations and observations regarding the joint use of JPEG2000 and AWF SR.

1.5 Organization of the Thesis

Chapter 1 introduces the multiframe Super Resolution and JPEG2000 Compression. Chapter 2 develops the Adaptive wiener filter method algorithm of Super Resolution. Chapter 3 presents the various architectures of combining of Super resolution with JPEG2000 compression. Chapter 4 gives the experimental results obtained using various architectures of Super resolution with JPEG 2000 compression. Chapter 5 summarizes the thesis outline.

CHAPTER 2

ADAPTIVE WIENER FILTER ALGORITHM OF SUPER RESOLUTION

The AWF SR method is introduced in [4]. We provide some of the key algorithm details here for the reader's convenience. We begin with the observation model and then describe the AWF SR algorithm.

2.1. Observation Model

The AWF SR method is based on the observation model depicted in Fig. 2.1. Here we have P low resolution (LR) frames that are related to the desired continuous scene, $d(x, y)$, through a shift, point spread function (PSF) blur, and sampling with additive noise as shown. In the case of translational motion, the shift and PSF models commute, allowing us to equivalently use the observation model in Fig. 2.2. Here we have switched the motion and PSF blocks and combined the motion and sampling steps into a single non-uniform sampling operation. Details on the commutation of the motion and PSF blur are addressed in [15]. The PSF blur in Fig. 2.2 yields the intermediate image

$$f(x, y) = d(x, y) * h(x, y), \quad (1)$$

where $h(x, y)$ is the PSF and $*$ is 2D convolution. This blurred image is assumed to be sampled nonuniformly based on the motion parameters and the detector pitch of the focal

plane array (FPA) as described in [4]. These samples are represented in lexicographical notation as $\mathbf{f}=[f_1, f_2, \dots, f_N]^T$. With additive noise these samples are given by $\mathbf{g}=\mathbf{f}+\mathbf{n}$, where $\mathbf{n}$ is an $N \times 1$ array of noise samples. The noise will be assumed to be zero-mean independent and identically distributed Gaussian noise with a variance of σ_n^2 .

To model the PSF, we follow the approach in [16] that models diffraction and detector integration. For diffraction limited optics, the spatial cut-off frequency is $\omega_c = 1/(\lambda \times f\text{-number})$. Here, λ is the wavelength of light and f-number is the ratio of the focal length to the effective diameter of the optics. Another critical parameter for an imaging system is the detector pitch, p , for the focal plane array. The pitch is the spacing between the detector elements and one divided by the pitch is the sampling frequency. The Nyquist criterion dictates that to avoid aliasing we must have $1/p > 2\omega_c$. However, due to a complex trade-space in imaging system design, many imaging systems do not meet the Nyquist criterion. This issue is addressed well in [17]. Resolution in such under sampled imaging systems may be thought of as limited by the detector array, rather than optically limited [2]. These systems may benefit from multiframe SR processing, such as the AWF SR method.

Figure 2.1: Observation model with a set of corresponding LR frames

Figure 2.2: Alternative observation model replacing the motion and combining of LR frames

2.2. Adaptive Wiener Filter of SR Processing

The AWF SR processing of the observed data to produce an HR image estimate is illustrated in Fig. 2.3. Let the resulting HR image be upsampled by a factor of L in each spatial dimension relative to the LR input frames. The AWF SR algorithm uses a moving window that passes over the HR grid of nonuniformly sampled data in $\mathbf{g}$. Let the samples in the small moving observation window about the i 'th HR output pixel be denoted $\mathbf{g}_i$. The output of the AWF is an estimate of the desired high resolution (HR) image and is given by a weighted sum for each HR pixel as follows

$$\hat{d}_i = \mathbf{w}_i^T \mathbf{g}_i, \quad (2)$$

where $\mathbf{w}_i$ is a vector of weights. The minimum mean squared error (MSE) weights are employed and these are given by

$$\mathbf{w}_i = \mathbf{R}_i^{-1} \mathbf{p}_i, \quad (3)$$

where $\mathbf{R}_i = E\{\mathbf{g}_i \mathbf{g}_i^T\} = E\{\mathbf{f}_i \mathbf{f}_i^T\} + \sigma_n^2 \mathbf{I}$ is the autocorrelation matrix for the observation vector, and $\mathbf{p}_i = E\{d_i \mathbf{g}_i\} = E\{d_i \mathbf{f}_i\}$ is the cross-correlation vector between the desired sample and observation vector.

The correlations employed are model-based and they vary with the spatial distribution of the samples in $\mathbf{g}_i$. In particular, the continuous desired image autocorrelation is assumed to be

$$r_{dd}(x, y) = \sigma_d^2 \rho^{\sqrt{x^2 + y^2}}, \quad (4)$$

where σ_d^2 is the desired image variance and ρ controls the drop in correlation as a function of distance. From this, it can be shown that the other key correlation functions can be computed as follows

$$r_{df}(x, y) = r_{dd}(x, y) * h(x, y) \quad (5)$$

and

$$r_{ff}(x, y) = r_{dd}(x, y) * h(x, y) * h(-x, -y). \quad (6)$$

Based on the distances between the observed samples in $\mathbf{g}_i$ as they appear on the HR grid, we evaluate Eqn. (6) to populate $\mathbf{R}_i$. Similarly, given the distances between the desired sample position and the observed samples, we evaluate Eqn. (5) and populate the vector $\mathbf{p}_i$. This allows us to compute $\mathbf{g}_i$ using Eqn. (3). Note that by estimating the HR image in this fashion, the AWF is simultaneously performing nonuniform interpolation, deconvolution, and noise reduction, all with a single weighted sum operation. This sets it apart from other fast SR methods that perform nonuniform interpolation and restoration as independent processing steps. The combined approach of the AWF has computational and performance robustness advantages [4].

The tuning parameters impacting performance are ρ and the signal-to-noise (SNR) ratio σ_d^2 / σ_n^2 . We shall show that adjusting the SNR for the level of compression is one means of reducing the degradation effects of compression on this type of AWF SR

processing. In a sense, we can treat the compression artifacts as an additional noise source.

Figure 2.3: Overview of AWF SR algorithm

CHAPTER 3

ARCHITECTURES FOR SR WITH JPEG2000 COMPRESSION

There are a number of ways to combine SR algorithms with JPEG2000 compression. Here we compare applying AWF SR after performing compression on the LR input frames. We also consider performing AWF SR and then compressing the resulting HR images. When using compression on the multiple input frames prior to SR, we consider both independent and difference frame compression methods as described below.

3.1 SR after JPEG2000 Compression

The most common scenario is that the imagery from the sensor is compressed for storage and/or transmission immediately after it is acquired. Access to the raw uncompressed imagery for SR may not be possible. In this case, SR can only be applied after the compression as illustrated in Fig. 3.1. Thus, understanding the robustness of SR operating on such compressed imagery is an important problem.

Figure 3.1: Overview of SR after JPEG2000 compression

3.1.1 Individual Frame Method

Perhaps the most straightforward method for treating the multi-frame input with compression is to compress each input frame individually and independently. This allows each frame to be decompressed independently, providing an advantage over MPEG, for example. We shall refer to this as the Individual Frame method. All processing here is done in MATLAB and the JPEG2000 is achieved with the “imwrite” command using the default JPEG2000 parameters.

It should be noted that SR is most beneficial for significantly undersampled imaging systems where aliasing is present. For such an imaging system, the LR observed frames do not tend to compress well because of the high spatial frequency content. However, since a set of frames suitable for SR must overlap in field of view, these frames are also likely to exhibit significant inter-frame correlation. Thus, we also consider compression of registered difference frames as described below.

3.1.2 Difference Frame Method

The difference frame compression method is illustrated in the block diagram in Fig. 3.2. In this method, the last (i.e., most recent) observed LR frame is considered to be the reference image, denoted $\mathbf{r}$. This reference frame is $\mathbf{g}(P)$ as shown in Fig. 2.1. After JPEG2000 compression and then decompression, the reference frame is denoted $\bar{\mathbf{r}}$ and $\hat{\mathbf{r}}$, respectively. We set the CR for the reference image to be $1/Q$ that of the CR for the difference frames generated where Q is a tuning parameter. The parameter Q is determined depending on the application of the compressed frames. In this case, 8 is found to be a good choice.

Next, the decompressed reference frame is shifted to match each of the remaining $P-1$ LR frames and $P-1$ difference frames are computed and compressed. The shifts are estimated from the LR frames the same way they are for SR [4]. The difference frames are denoted $\mathbf{e}(k)$ for $k=1,2,\dots,P-1$. After compression these difference frames are denoted $\bar{\mathbf{e}}(k)$. To be stored and/or transmitted from this compression stage, we have the compressed reference, $\bar{\mathbf{r}}$, along with the compressed difference frames $\bar{\mathbf{e}}(k)$ and shifts $\mathbf{s}(k)$, for $k=1,2,\dots,P-1$.

Figure 3.2: Difference frame method of compression

The process of decompression for the difference frame method is illustrated in Fig.3.3. The process begins by decompressing the reference and difference frames as shown. Next, the decompressed reference is added to the decompressed difference frames to recover the individual frames. This is given by $\hat{g}(k) = \hat{e}(k) + \hat{r}$, for $k=1,2,\dots,P-1$. Finally, note that the last image is simply the reference and is given by $\hat{g}(P) = \hat{r}$.

Figure 3.3: Difference frame method of decompression

3.2. JPEG2000 Compression after the Application of SR

The final architecture considered here applies JPEG2000 compression after SR. This is shown in Fig. 3.4. In this mode, AWF SR is applied directly to the raw uncompressed imagery from the sensor. The resulting SR image is then compressed with JPEG2000 and transmitted/stored. This mode would be most practicable when SR processing can be done in real-time at the sensor. It is also possible to use this mode if SR is applied forensically to stored uncompressed data, the results of which are to be later disseminated in compressed format.

While the pixel dimensions of the SR image are increased, the aliasing is reduced, making the image SR generally easier to compress. Furthermore, the SR process gets the benefit of working on data with no compression artifacts. In our experiments, we combine 16 LR frames to produce a single SR image with upsampling in each dimension of $L=4$. Thus, the total number of HR and LR pixels is the same. In a video-to-video application, the SR video frame rate may be the same as the LR video rate. In this case, we have L^2 the number of SR pixels as we do input pixels. However, the possibility of difference frame compression on the SR video exists for enhanced compression of the SR video.

Figure 3.4: Overview of SR before JPEG2000 compression

CHAPTER 4

EXPERIMENTAL RESULTS

In this section we present the experimental results that include simulated LR data and a real visible video sequence. The simulated data allow for quantitative performance analysis and the real data allow for subjective in a real application. In addition to comparing the methods described in Section 3, we also examine modifying the AWF correlation model in an attempt to more robustly treat the compression artifacts, especially at high CRs.

4.1 Simulated Data

The first set of simulated data is based on the uncompressed 8 bit image Lighthouse image from the Kodak database [18] shown in Fig. 4.1. This image is 512×768 pixels at 8 bits per pixel (bpp) grayscale. We artificially degrade this image to simulate the observation model in Fig. 2.1. In particular, we simulate $P=16$ LR frames with random translational shift, blur and noise. The simulation PSF model is based on parameters matching the real video data used in Section 4.2. The f-number of the optics is 4, and the detector pitch in the both horizontal and vertical directions is 5.6 mm. We assume a 100% fill factor [15] and a wavelength is $0.55 \mu\text{m}$. The down sampling factor, relating the LR and SR image sizes, is $L=4$.

Finally, white Gaussian noise with a variance of 1 digital unit (DU) to simulate low level electronic noises. Image results using the Lighthouse simulation are shown in Figs. 4.2-4.6 and quantitative results can be found in Figs. 4.7 and 4.8. Further quantitative results for this image and others are listed in Tables 4.1 and 4.2.

Figure 4.1: Kodak lighthouse image

4.1.1 Bicubic Interpolation after JPEG2000 Compression

In this research, we determine the compression level of an image from its CR value. CR is calculated using the formula $CR = 8 / \text{bpp}$ in grayscale. For instance, an image is said to have CR of 4 if its bits per pixel value is 2. Figure 4.2 shows the LR reference frame compressed by $CR = 8$ and then $L = 4$ bicubic interpolation.

Figure 4.2: Simulated LR reference frame with $CR=8$ and $L=4$ bicubic interpolation

4.1.2 Individual Frame Method of JPEG2000 Compression before SR

One of the $P=16$ frames generated using the individual frame compression method as described in 3.1.1. The result of applying AWF SR with $L=4$ on the individually compressed LR images is shown in Fig. 4.3. Here the SNR in the AWF correlation model that minimizes the MAE is applied and $\rho=0.7$ [4]. The SR images look notably sharper than the image in Fig. 4.2. But, we can observe that the LR images are rather hard to compress due to aliasing. As a result of the shifts between LR frames, the compression artifacts tend to vary somewhat from frame to frame. This can lead to a SR image with artifacts seen in Fig. 4.3. We can reduce the artifacts by using lower subjective SNR and sharper image can be obtained as shown in Fig 4.4

Figure 4.3: SR after individual frame method of compression (CR=8) using optimum SNR.

Figure 4.4: SR after individual frame method of compression (CR=8) using SNR value of 4.

4.1.3 Difference Frame Method of JPEG2000 Compression before SR

The next result, shown in Fig. 4.5, is for AWF SR applied to LR frames compressed with difference frame compression described in Section 3.1.2. Note that in the difference frame compression method, the group of $P=16$ LR frames is set to have an overall compression ratio of $CR=8$ to match the individual frame method. The optimum SNR of 20.86 is used with $\rho=0.7$. Note that this result appears far superior to that obtained with individual frame compression (for the same overall CR). Here, the redundancy between LR frames is exploited by the difference frame compression. At the same time, the differences among the LR frames are exploited by SR. We believe this is perhaps the best way to combine compression and SR. Image data suitable for multiframe SR is probably also suitable for difference frame compression, due to the overlapping field of view.

Figure 4.5: SR after difference frame method of compression ($CR=8$) using optimum SNR.

4.1.4 JPEG2000 Compression after the Application of SR

The image obtained using compression after SR, as described in Section 3.2, is shown in Figure 4.6. Again a compression ratio of $CR = 8$ is used to facilitate comparison. The optimum SNR of 40 and $\rho = 0.7$ is used to facilitate comparison. This result also appears to be much better than SR after individual frame compression. Since we chose $P = 16$ and $L = 4$, the SR image has the same pixel count as the set of LR frames. As mentioned above, in a video-to-video application with input and output frame rates that are the same, this approach would face the challenge of an uncompressed throughput increase of $16\times$. Note that difference frame compression could be applied to the resulting SR video to help deal with the increase. But, to make a fair comparison among the SR/compression techniques, frame rate needs to be reduced for this method according to its compression level.

Figure 4.6: Individual frame method of compression ($CR=8$) after SR using optimum SNR.

4.1.5 Overall Comparison of Various SR/Compression Methods

Figure 4.7 shows the variation of mean absolute error (MAE) with CR for all the different architectures of Multiframe SR with JPEG2000 compressed images as mentioned in Section 3 for the Lighthouse image. The plots clearly shows that SR is clearly beneficial, even out to high CRs. Individual frame compression prior to SR is very good at low CRs, but degrades quickly. The difference frame compression performance is considerably better than individual frame for higher CRs. SR after compression provides the lowest MAE here. For the applications with same frame rates; the SR before compression technique would be a good choice. For other cases, SR before compression needs to be compressed $16 \times$ the CR of the SR after compression techniques.

Figure 4.8 shows a region of interest (ROI) for the original and various processed images.

Figure 4.7: Mean absolute error versus CR for the various SR/compression methods

(a)

(b)

(c)

(d)

(e)

Figure 4.8 : Region of interest of the Kodak lighthouse image. All the SR/compression methods are performed using $L=4$ and $\rho=0.7$. All the compressed images have a CR=8.

(a) Kodak lighthouse HR image (Uncompressed). (b) Bicubic interpolation on the lighthouse HR image compressed using individual JPEG2000 compression. (c) AWF SR on the individually compressed LR images with an optimum SNR of 20.86. (d) AWF SR on the difference frame compressed LR images with optimum SNR of 244.35. (e) Individual frame compression is applied after the AWF SR is applied on the LR images with optimum SNR of 20.86.

4.1.6 SNR Adjustment for Different Levels of Compression

Figure 4.9 shows the how adjusting the SNR in the AWF SR correlation model can improve results as CRs increase. Two curves are shown in Fig. 4.9 showing MAE versus CR for SR after individual LR frame compression. The top curve is MAE for SR using a fixed SNR value of 199.49 for all CRs. While this is the optimum SNR for CR=4, it may not produce the best results at higher CRs. The lower error curve uses the optimum SNR for each CR tested. The curves clearly show that as the CR increases, a lower SNR produces significantly better results. In this way, compression artifacts are managed as an additional noise source.

Figure 4.9: Comparative plot between optimum SNR for first CR and for each CR

4.1.7 Other Simulated Results

To provide a more comprehensive performance analysis, four additional images are processed. These additional images are also from the Kodak dataset [19] and are shown in Figure 4.10 (a)-(d).

Figure 4.10: Ideal HR images used for simulation results (a) Kodak parrot image (b) Kodak mountain stream image (c) Kodak propeller plane image (d) Kodak window with bushy flowers.

Figures 4.11 – 4.16 shows the images obtained for various SR/Compression methods for a Parrot image from Kodak Dataset to further strengthen our results.

Table 4.1 lists the mean absolute errors (MAEs) calculated between the true HR image and various processed images for CR=8. The corresponding mean squared errors (MSE) are shown in Table 4.2. The MAE optimum SNRs are used for all SR and $\rho=0.7$ These results show that the difference frame compression method prior to SR consistently produces the lowest error. Note also that even using the inferior individual frame compression prior to SR we still obtain a lower error than using bicubic interpolation in all cases tested.

Figure 4.11: Kodak parrot HR image

Figure 4.12: Simulated LR reference frame with $CR = 8$ and $L=4$ bicubic interpolation

Figure 4.13: SR after individual frame method of compression ($CR=8$) using optimum SNR

Figure 4.14: SR after difference frame method of compression (CR=8) using optimum SNR

Figure 4.15: Individual frame method of compression (CR=8) after SR using optimum SNR

(a)

(b)

(c)

(d)

(e)

Figure 4.16: Region of interest of the Kodak parrot image. All the SR/compression methods are performed using $L=4$ and $\rho=0.7$. All the compressed images have a CR =8. (a) Kodak lighthouse HR Image (Uncompressed). (b) Bicubic Interpolation on the lighthouse HR image compressed using individual JPEG2000 compression. (c) AWF SR on the individually compressed LR Images with an optimum SNR of 20.86. (d) AWF SR on the difference frame compressed LR images with optimum SNR of 244.35 (e) Individual frame compression is applied after AWF SR is applied on the LR images with optimum SNR of 20.86.

Input Image	Bicubic after compression (Individual Frame)	SR After Compression		SR Before Compression
		Individual Frame	Difference Frame	
Light House	9.84	8.15	6.85	6.81
Parrot	4.60	3.64	3.09	2.83
Propeller Plane	5.71	4.56	4.01	3.70
Mountain Stream	18.49	16.72	13.68	13.62
Window with bushy flowers	8.41	5.74	3.94	3.82

Table 4.1: Mean absolute error obtained using various SR/compression with CR=8 using optimum value SNR and $\rho=0.7$

Input Image	Bicubic after compression (Individual Frame)	SR After Compression		SR Before Compression
		Individual Frame	Difference Frame	
Light House	286.26	188.45	149.10	132.10
Parrot	93.44	47.18	38.87	35.14
Propeller Plane	150.94	82.73	71.21	65.12
Mountain Stream	588.67	565.03	416.90	390.91
Window with bushy flowers	177.23	87.92	54.96	47.42

Table 4.2: Mean square error obtained using various SR/compression with CR=8 using optimum value SNR and $\rho=0.7$

4.2 Real Video Data

In this section, the algorithms are demonstrated for real video sequences. Figure 4.17 shows a single frame in the real video sequence captured using an Imaging Source DMK 21BU04 visible camera. The camera acquires 640×480 8-bit grayscale images with a Sony ICX098BL CCD sensor with 5.6mm pitch detectors. The camera is fitted with an F/4 lens with a focal length of 5mm. The central wavelength is assumed to be $\lambda = 0.55 \mu\text{m}$. Note that this imaging system is theoretically $5.09\times$ undersampled. In practice, we find that there is very little residual aliasing using $L=4$, so we believe this is a good choice for SR processing with this sensor. In all of the SR results, we use $L=4$ and assume an SNR of 40 and use $\rho=0.7$ in the AWF correlation model. For results incorporating compression, a CR of 8 is used in all of the results in this section.

Figure 4.18 shows images for an ROI centered on a 2D chirp pattern. The true chirp pattern is made up of concentric circles with increasing spatial frequency moving away from the center. The SR result with no compression is shown in Fig. 4.18 (a). This image provides a good representation of the true chirp pattern. The image formed using $L=4$ bicubic interpolation after individual frame compression is shown in Fig. 4.18 (b). Here Moire patterns are clearly visible on the chip due to aliasing. The image is also noticeably blurrier than in (a). AWF SR after individual frame compression is shown in Fig. 4.18 (c). While this result is quite good, some compression artifacts can be seen in the high spatial frequencies. AWF SR after difference frame compression is shown in (d) and SR before compression is shown in (e). Both of these results look comparable to the uncompressed result in (a). Similar results for a different ROI are provided in 4.19. This ROI include lettering on book spines. Jagged diagonal edges and degraded lettering are

clearly an issue with bicubic interpolation here. All of the SR methods significantly reduce these artifacts and improve readability of the text.

Figures 4.18 and 4.19 clearly illustrate that the results obtained with the real video data follow the same pattern seen with the simulated data. In particular, AWF SR before compression is the best method among the architectures tested. This is followed closely in performance by AWF SR after difference frame compression. AWF SR after individual frame compression, while perhaps inferior to the other methods, still outperforms bicubic interpolation by a significant margin.

Figure 4.17: An individual frame in the original real video sequence

Figure 4.18: Region of interest (ROI – I) of real data for various SR/compression methods. The optimum SNR of 40, $\rho = 0.7$ and $L = 4$ is used. All the compressed images have a CR = 8.

(a) Uncompressed AWF SR. (b) Bicubic Interpolation on the real video sequence compressed by individual JPEG2000 compression. (c) AWF SR on the individually compressed real video frames. (d) AWF SR on the difference frame compressed LR images. (e) Individual frame compression is applied after AWF SR is applied on the LR images.

(a)

(b)

(c)

(d)

(e)

Figure 4.19: Region of interest (ROI – II) of real data for various SR/compression methods. The optimum SNR of 40, $\rho = 0.7$ and $L = 4$ is used. All the compressed images have a CR = 8. (a) Uncompressed AWF SR. (b) Bicubic Interpolation on the real video sequence compressed by individual JPEG2000 compression. (c) AWF SR on the individually compressed real video frames. (d) AWF SR on the difference frame compressed LR images. (e) Individual frame compression is applied after AWF SR is applied on the LR images.

CHAPTER 5

SUMMARY

We have proposed and tested novel methods for combining multiframe SR and JPEG2000 compression. The results obtained suggest that SR prior compression provides the best results when compared to other architectures. However, this may require real-time SR processing at the sensor or storage of full resolution video for later processing. This may not always be feasible. Furthermore, in video-to-video SR applications with SR prior to compression, the data throughput is significantly increased if the frame rate is the same for the LR and SR frames (since the SR frames are upsampled by L). Often, a more practical scenario is to apply SR after JPEG2000 compression of the LR frames. With this approach, we have demonstrated that SR processing still provides improvement, even for relatively high CRs. We also observed that with high CRs, the AWF SR correlation model can be adjusted by decreasing the assumed SNR to help compensate for the compression artifacts. Essentially, using a lower SNR treats the compression artifacts as an additional noise source.

When compression is done prior to SR, we have shown that difference frame compression is superior to individual frame compression. Note that a set of frames suitable for multiframe SR must overlap in field of view such that accurate registration possible. With such overlapping frames, there will tend to be significant inter-frame

correlation. With difference frame compression, this redundancy between LR frames is exploited. On the other hand, it is the differences among the LR frames that are exploited by SR to reduce aliasing. This provides a potentially practical and efficient approach to combining SR and JPEG2000 compression.

In summary, AWF SR processing can be effectively combined with JPEG2000 compression. Even at relatively high CRs using simple individual frame compression prior to SR we see improvement over bicubic interpolation. Difference frame compression prior to SR provides improved performance without increasing the video throughput. The modeled SNR in the AWF correlation model can be used as a tuning parameter to help treat compression artifacts. In terms of application, applying AWF SR effectively for JPEG2000 compressed images would benefit us in the field of Aerial mapping and Airborne imaging [20].

BIBLIOGRAPHY

- [1] Peyman Milanfar, “*Super Resolution Imaging (Digital Imaging and Computer Vision)*”, CRC press, Taylor & Francis group, 2011
- [2] R. C. Hardie, R. R. Schultz, and K. E. Barner, “Super-resolution Enhancement of Digital Video” *EURASIP JASP*, Vol. 2007, Article ID 20984, (3 pages).
- [3] Moon Gi Kang, Chaudhuri.S , “Super-Resolution image reconstruction”, *IEEE signal processing magazine*, vol. 20, no. 3, pp. 19-20, 2003
- [4] R. C. Hardie, “A Fast Image Super-Resolution Algorithm using an Adaptive Wiener Filter”, *IEEE Transactions on Image Processing*, Vol. 16, No. 12, Dec. 2007 pp. 2953-2964.
- [5] David S.Taubman and Micheal W.Marcellin, “*JPEG2000 Image Compression Fundamentals, Standards and Practice* “,Kluwer Academic Publishers, Norwell, MA (2001)
- [6] R. Molina, A. K. Katsaggelos, L. D. Alvarez, and J. Mateos. “Towards a new video compression scheme using super-resolution”. In *Proc. of SPIE, visual communication and image processing*, page 607706.1=607706.13, Jan 2006.

- [7] D. Barreto, L. D. Alvarez, R. Molina, A. K. Katsaggelos, and G. M. Callico. "Region-based super-resolution for compression". *Multidimensional Systems and Signal Processing*, 18(2-3):59-81, 2007.
- [8] F. Brandi, R. de Querioz, and D. Mukherjee. "Super resolution of video using key frames". In *Proc. IEEE Int. Symp. On Circuits and Systems*, pages 1608-1611, May. 2008.
- [9] W.T. Freeman, T.R. Jones and E.C. Paszlor. "Example-based Super Resolution". In *IEEE Computer graphics and Applications*, 22(2):56-65, 2002.
- [10] Z. Xiong, X. Sun and F. Wu. "Super-resolution for low quality thumbnail images". In *Proc. IEEE Int. Conf. on Multimedia and Expo*, pages 181-184, June. 2008.
- [11] M. Pickering, G. Ye, M. Frater, and J. Arnold. "A transform-domain approach to super-resolution mosaicking of compressed images". In *Proc. 4th AIP Int. Conf. and the 1st Congress of the IPIA*, 2008.
- [12] Z. Xiang, Y. Jie, and S.D. Du. "Super-resolution reconstruction of image sequences compressed with DWT-based techniques". In *Proc. IEEE Int. Conf. on Wavelet Analysis and Pattern Recognition*, pages 555-560, Nov. 2007.
- [13] M. Smith and J. Villasenor, "Intra-frame JPEG-2000 vs. inter-frame compression comparison: The benefits and trade-offs for very high quality, high resolution sequences", In *Proc. SMPTE Technical Conf. and Exhibition*, Oct.2004.
- [14] "Analog Devices Video Compression Technology Enables Motion Picture Industry to Adopt JPEG2000 as Digital Cinema Standard", Business wire, 2004.

- [15] R. C. Hardie, Kenneth J. Barnard, and Raul Ordonez, “Fast super-resolution with affine motion using an adaptive Wiener filter and its application to airborne imaging”, *Optics Express*, Vol. 19, Issue 27, pp. 26208-26231 (2011).
- [16] R. C. Hardie, K. J. Barnard, J. G. Bogner, E. E. Armstrong, and E. A. Watson, “High-resolution image reconstruction from a sequence of rotated and translated frames and its application to an infrared imaging system,” *Opt. Eng.*, Vol. 37,no. 1, pp. 247–260, Jan 1998.
- [17] R. D. Fiete, “Image quality and λ FN/ p for remote sensing systems,” *Optical Engineering* 38, 1229–1240 (1999).
- [18] <http://r0k.us/graphics/kodak/kodim21.html>
- [19] <http://r0k.us/graphics/kodak>
- [20] R. C. Hardie, K. Barnard, and R. Ordonez, “Fast super-resolution with affine motion using an adaptive Wiener filter and its application to airborne imaging,” *OSA Optics Express*, Vol. 19, Issue 27, 26208-26231 (2011).

APPENDIX

MATLAB script for various SR/compression methods for simulated data

```
clear all;
clc;
close all;

%% Load input image

org_image = imread('lighthouse.png');% Original Image
im(org_image);
% title('The Original Color Image');

hr_true = double( org_image(:,:,2) );
im(hr_true);
% title('True HR Image');
print -depsc HR_Image.eps;

%% Parameters of PSF

% F-number of optics
fnum = 4;

% Physical Detector Array Parameters
sensor_dx_um = 5.6; % detector pitch in x direction
sensor_dy_um = 5.6; % detector pitch in y direction

% a, b are the physical dimensions of the active area of a detector
% a_um <= sensor_dx_um; b_um <= sensor_dy_um;
a_um = 5.6; % Changed for 100% fill factor
b_um = 5.6; % Changed for 100% fill factor

M = 25; % size of psf in x-direction
```

```

N = 25;           % size of psf in y-direction

lam_um = 0.55;    % center wavelength of HR image
lam_mm = lam_um/1000;
rho_c = 1./(lam_mm.*fnum); % Optics cutoff (limiting factor in system)
nyquist_dx_mm = 1./(2.*rho_c); % pixel pitch (mm) for Nyquist sampling
nyquist_dx_um = nyquist_dx_mm.*1000 % convert to (um)

% For no aliasing, need to have pixel pitch of dx=dy=0.5*nyquist_dx_um
% Undersampling
undersampling = sensor_dx_um./nyquist_dx_um

L = 4 % Downsample factor

% Impulse invariant PSF model (MxN PSF Filter for SR grid)
% output is psf model and theoretical (true) OTF
% Sampling (x,y detector pitch is for HR grid, i.e., divided by
% the amount of upsampling we plan between LR & HR grids
psf = model_psf( fnum, lam_um, a_um, b_um, ...
    sensor_dx_um/L, sensor_dx_um/L, M, N );
% Note the /L terms above because this is a PSF on the SR grid, not on the
% sensor native resolution grid.

figure;
mesh( [-(M-1)/2 : (M-1)/2 ], [-(N-1)/2 : (N-1)/2 ], psf );
xlabel('n_1');
ylabel('n_2');
zlabel('h(n_1,n_2)');
% title_text = sprintf(...
    %'Impulse Invariant PSF Model on HR Grid');
% title(% title_text,'FontWeight','bold');
axis tight;

% Generate high resolution MTF from psf found in "model_psf.m" function
H = fftshift(fft2( psf, 256, 256 ));
f = linspace(-1/2, 1/2, 256 );

figure;
surf(f,f,abs(H));
shading interp;

```

```

colormap(gray(256));
axis tight;
% title_text=sprintf('Impulse Invariant MTF Model');
% title(% title_text,'FontWeight','bold');
xlabel('f_1 (cyc/sample)'); ylabel('f_2 (cyc/sample)');
zlabel('H(f_1,f_2)');

%% LR Image Generation

num_images = 16; % No. of Frames
rand('state',1)

Shifts = rand(num_images,2); % Create x,y shifts (fraction of LR pixel)
Shifts(1,:) = [0,0]; % Set first shift as (0,0) as reference

% Plot Shifts
figure;
plot(Shifts(:,1),Shifts(:,2),'rs');
xlabel('x (LR Pixels)'); ylabel('y (LR Pixels)');
% title_text = sprintf(...
%'Shifts for Creating LR Image Set (on LR Grid)');
% title(% title_text,'FontWeight','bold');
grid on;

% Create simulated LR frames with blur and shift
% Shifts are on the HR grid because we operate with HR image to
% create LR images
lr_image = fwd_model_shift( hr_true, psf, L, L*Shifts, 'bic' );

% Additive noise
std_n = 1;
var_n = std_n^2;

randn('state',1)

noise = std_n*randn( size(lr_image) );
lr_observed = lr_image + noise;

```

```

im(lr_observed(:,:,end));
% title_text = sprintf('Sample Output LR Image');
% title(% title_text,'FontWeight','bold');
print -depsc LR_Image.eps;
% figure;
% dispvideo3di( lr_observed );

%% AWF SR Parameters

% SR Parameters
params.L = L;

% Must be odd sizes
params.obsx = 5;    % in LR pixels
params.obsy = 5;    % in LR pixels

% Number of frames used for each output
params.num_frames = num_images; % num_images;

% Filter parameters impacting the correlation model
filt.SNR = 20; % Adjust for best performance
filt.var_d = 1; % Normalize and fix
filt.var_n = filt.var_d / filt.SNR; % Set SNR via noise variance

filt.rho = 0.75; % 0.75; % one step HR pixel correlation value

% AWF filter size in HR pixels
filt.wx = params.obsx*params.L;
filt.wy = params.obsy*params.L;

% Registration related parameters
filt.ref = 1; % Which frame forms the reference frame
filt.pre_std = 1.5; % Gaussian prefilter for registration std in LR pixels
% filt.method = 'bil'; % Interpolatoin used for registration
% filt.method = 'bic'; % Interpolatoin used for registration
filt.method = 'bic'; % Interpolatoin used for registration ('spline' is slow.)

filt.awf_crop = (params.obsx*params.L-params.L)/2;

```



```

stats = awf_model_correlations( filt.var_d, filt.rho, ...
    filt.wx, filt.wy, psf, 0 );

%% Individual Frame Compression Before SR

% Individual Frame method
cr_desired = 8.1 % Individual Compression

% Cleaned up function and removed subfunction to streamline
[ decompressed, cr_actual ] = individual_frame_jp2k( lr_observed, cr_desired, 8 );

cr_actual

% figure;
% dispvideo3di( decompressed );
% The temporal artifacts from compression are obvious

%% Find optimum SNR

var_n_opt = fminbnd( @(var_n) AWF_SR_dis_error( var_n, decompressed, hr_true,
params, stats, filt, 'mae' ),.001,.1);
snr_opt = 1/var_n_opt

% Perform the AWF on decompressed
awf_on_decompressed = AWF_SR_dis( decompressed, params, stats, var_n_opt, ...
    filt.ref, filt.pre_std, filt.method );

im( awf_on_decompressed( filt.awf_crop+1:end-filt.awf_crop, ...
    filt.awf_crop+1:end-filt.awf_crop ) );
% title(sprintf('AWF SR After JP2K (Individual), SNR=%2.2f (Opt.)', snr_opt) )
print -depsc individual_frame_true.eps

crop = filt.awf_crop*2;

err_awf_opt = dif2( hr_true( crop+1:end-crop, crop+1:end-crop ), ...

```

```

    awf_on_decompressed( crop+1:end-crop, crop+1:end-crop ));

disp(' Error Individual Frame method' )
err_awf_opt

%% Bicubic

out_bic = do_shift_interp( decompressed(:, :, 1), [0,0], 'bic', L );

im( out_bic( filt.awf_crop+1:end-filt.awf_crop, ...
    filt.awf_crop+1:end-filt.awf_crop ) );
% title('Bicubic Interpolation after JP2K (Individual)')
print -depsc Bicubic_True.eps;
err_bic = dif2( hr_true( crop+1:end-crop, crop+1:end-crop ), ...
    out_bic( crop+1:end-crop, crop+1:end-crop ));

disp(' bicubic Interpolation')
err_bic

%% Now do difference frame compression

reg.buff = 5;
reg.thresh = .05;
reg.pre_std = 1.5;
reg.method = 'bic';

cr_ref = 1.062;
cr_err = 8 * cr_ref;
bits = 8;
num_std = 16;

[ decompressed2, cr_actual2 ] = difference_frame_jp2k_v2( lr_observed, reg, cr_ref,
    cr_err, bits, num_std );
cr_actual2

% figure;
% dispvideo3di( cat(2,decompressed,decompressed2) );
% dif2( lr_observed, decompressed )

```

```

dif2( lr_observed, decompressed2 )

%% AWF SR on difference frame

var_n_opt2 = fminbnd( @(var_n) AWF_SR_dis_error( var_n, decompressed2, hr_true,
params, stats, filt, 'mae' ),.001,.1);
snr_opt2 = 1/var_n_opt2

% Perform the AWF on decompressed
awf_on_decompressed2 = AWF_SR_dis( decompressed2, params, stats, var_n_opt2, ...
    filt.ref, filt.pre_std, filt.method );

im( awf_on_decompressed2( filt.awf_crop+1:end-filt.awf_crop, ...
    filt.awf_crop+1:end-filt.awf_crop ) );
% title(sprintf('AWF SR After JP2K (Difference), SNR=%2.2f (Opt.)', snr_opt2) )
print -depsc differenceframe.eps

crop = filt.awf_crop*2;

err_awf_opt2 = dif2( hr_true( crop+1:end-crop, crop+1:end-crop ), ...
    awf_on_decompressed2( crop+1:end-crop, crop+1:end-crop ));
disp(' Difference Frame compression :');
err_awf_opt2

%% SR Before Compression

awf_on_uncompressed = AWF_SR_dis(lr_observed, params, stats, var_n_opt, ...
    filt.ref, filt.pre_std, filt.method );

[ decompressed_sr, cr_actual ] = individual_frame_jp2k(awf_on_uncompressed,
cr_desired, 8 );

err_awf_sr_before = dif2( hr_true( crop+1:end-crop, crop+1:end-crop ), ...
    decompressed_sr( crop+1:end-crop, crop+1:end-crop ));

im(decompressed_sr( filt.awf_crop+1:end-filt.awf_crop, ...
    filt.awf_crop+1:end-filt.awf_crop ) );

% title(sprintf('AWF SR Before JP2K'));

```

```

print -depsc sr_before_compression.eps
disp(' SR Before compression ');
err_awf_sr_before

```

MATLAB script to study SNR adjustment for different levels of compression

```

% Comparison of individual frame and difference frame compression prior to
% AWF SR

```

```

%% Define paths and clear variables

```

```

clear all;

```

```

clc;

```

```

close all;

```

```

% Flag to control if we output .eps figure

```

```

make_eps = 1;

```

```

%% Load input image

```

```

org_image = imread('lighthouse.png');% Original Image

```

```

im(org_image);

```

```

title('The Original Color Image');

```

```

% Define true HR image as the green field

```

```

hr_true = double( org_image(:,:,2) );

```

```

%% Parameters of PSF

```

```

% F-number of optics

```

```

fnum = 4;

```

```

% Physical Detector Array Parameters

```

```

sensor_dx_um = 5.6; % detector pitch in x direction

```

```

sensor_dy_um = 5.6; % detector pitch in y direction

```

```

% a, b are the physical dimensions of the active area of a detector

```

```

% a_um <= sensor_dx_um; b_um <= sensor_dy_um;

```

```

a_um = 5.6; % Changed for 100% fill factor

```

```

b_um = 5.6; % Changed for 100% fill factor

```

```

M = 25; % size of psf in x-direction

```

```

N = 25; % size of psf in y-direction

```

```

lam_um = 0.55;      % center wavelength of HR image
lam_mm = lam_um/1000;

rho_c = 1./(lam_mm.*fnum); % Optics cutoff (limiting factor in system)
nyquist_dx_mm = 1./(2.*rho_c); % pixel pitch (mm) for Nyquist sampling
nyquist_dx_um = nyquist_dx_mm.*1000 % convert to (um)

% For no aliasing, need to have pixel pitch of dx=dy=0.5*nyquist_dx_um
% Undersampling
undersampling = sensor_dx_um./nyquist_dx_um

L = 4; % Downsample factor

% Impulse invariant PSF model (MxN PSF Filter for SR grid)
% output is psf model and theoretical (true) OTF
% Sampling (x,y detector pitch is for HR grid, i.e., divided by
% the amount of upsampling we plan between LR & HR grids
psf = model_psf( fnum, lam_um, a_um, b_um, ...
    sensor_dx_um/L, sensor_dx_um/L, M, N );
% Note the /L terms above because this is a PSF on the SR grid, not on the
% sensor native resolution grid.

figure;
mesh( [-(M-1)/2 : (M-1)/2 ], [-(N-1)/2 : (N-1)/2 ], psf );
xlabel('n_1');
ylabel('n_2');
zlabel('h(n_1,n_2)');
axis tight;

if make_eps
    print -deps psf_plot

end

title_text = sprintf(...
    'Impulse Invariant PSF Model on HR Grid');
title(title_text,'FontWeight','bold');

% Generate high resolution MTF from psf found in "model_psf.m" function
H = fftshift(fft2( psf, 256, 256 ));

```

```

f = linspace(-1/2, 1/2, 256 );

figure;
surf(f,f,abs(H));
shading interp;
colormap(gray(256));
axis tight;
xlabel('f_1 (cyc/sample)'); ylabel('f_2 (cyc/sample)');
zlabel('H(f_1,f_2)');

if make_eps
    print -deps mtf_plot
end

title_text=sprintf('Impulse Invariant MTF Model');
title(title_text,'FontWeight','bold');

%% LR Image Generation
num_images = 16; % No. of Frames

rand('state',1)
Shifts = rand(num_images,2); % Create x,y shifts (fraction of LR pixel)
Shifts(1,:) = [0,0]; % Set first shift as (0,0) as reference

figure;
plot(Shifts(:,1),Shifts(:,2),'rs');
xlabel('x (LR Pixels)'); ylabel('y (LR Pixels)');
grid on;

if make_eps
    print -deps shifts_plot
end

title_text = sprintf(...
    'Shifts for Creating LR Image Set (on LR Grid)');
title(title_text,'FontWeight','bold');

% Create simulated LR frames with blur and shift
% Shifts are on the HR grid because we operate with HR image to
% create LR images

```

```

lr_image = fwd_model_shift( hr_true, psf, L, L*Shifts, 'bic' );

% Additive noise
std_n = 1;
var_n = std_n^2;
randn('state',1)
noise = std_n*randn( size(lr_image) );
lr_observed = lr_image + noise;
im(lr_observed(:, :, 1));

title_text = sprintf('Sample Output LR Image');
title(title_text, 'FontWeight', 'bold');

% figure;
% dispvideo3di( lr_observed );

%% AWF SR Parameters

% SR Parameters
params.L = L;

% Must be odd sizes
params.obsx = 5;    % in LR pixels
params.obsy = 5;    % in LR pixels

% Number of frames used for each output
params.num_frames = num_images; % num_images;

% Filter parameters impacting the correlation model
filt.SNR = 20; % Adjust for best performance
filt.var_d = 1; % Normalize and fix
filt.var_n = filt.var_d / filt.SNR; % Set SNR via noise variance

filt.rho = 0.75; % 0.75; % one step HR pixel correlation value

% AWF filter size in HR pixels
filt.wx = params.obsx*params.L;
filt.wy = params.obsy*params.L;

% Registration related parameters

```

```

filt.ref= 1; % Which frame forms the reference frame
filt.pre_std = 1.5; % Gaussian prefilter for registration std in LR pixels
% filt.method = 'bil'; % Interpolatoin used for registration
% filt.method = 'bic'; % Interpolatoin used for registration
filt.method = 'bic'; % Interpolatoin used for registration ('spline' is slow.)
filt.awf_crop = (params.obsx*params.L-params.L)/2;

stats = awf_model_correlations( filt.var_d, filt.rho, ...
    filt.wx, filt.wy, psf, 0 );

%% Individual Frame Compression Before SR
cr_desired = linspace(1,50,5);

for i = 1: length(cr_desired)
% Cleaned up function and removed subfunction to streamline
[ decompressed, cr_actual(i) ] = individual_frame_jp2k( lr_observed, cr_desired(i), 8 );

% figure;
% dispvideo3di( decompressed );
% The temporal artifacts from compression are obvious

%% Find optimum SNR

var_n_opt (i) = fminbnd( @(var_n) AWF_SR_dis_error( var_n, decompressed, hr_true,
params, stats, filt, 'mae' ),.00001,1);
snr_opt (i)= 1/var_n_opt(i);

% Perform the AWF on decompressed
awf_on_decompressed = AWF_SR_dis( decompressed, params, stats, var_n_opt(i), ...
    filt.ref, filt.pre_std, filt.method );

%
% im( awf_on_decompressed( filt.awf_crop+1:end-filt.awf_crop, ...
%     filt.awf_crop+1:end-filt.awf_crop ) );
% title(sprintf('AWF SR After JP2K (Individual), SNR=%2.2f (Opt.)', snr_opt) )
% print -depsc individual_frame_true.eps

crop = filt.awf_crop*2;

```



```

err_awf_opt = dif2( hr_true( crop+1:end-crop, crop+1:end-crop ), ...
    awf_on_decompressed( crop+1:end-crop, crop+1:end-crop ));

err_mae_ind_opt(i)=err_awf_opt.mae;
snr_first=1/var_n_opt(1);

% Perform the AWF on decompressed
awf_on_snr_first = AWF_SR_dis( decompressed, params, stats,var_n_opt(1), ...
    filt.ref, filt.pre_std, filt.method );

%
% im( awf_on_decompressed( filt.awf_crop+1:end-filt.awf_crop, ...
%     filt.awf_crop+1:end-filt.awf_crop ) );
% title(sprintf('AWF SR After JP2K (Individual), SNR=%2.2f (Opt.)', snr_opt) )
% print -depsc individual_frame_true.eps

crop = filt.awf_crop*2;

err_awf_opt = dif2( hr_true( crop+1:end-crop, crop+1:end-crop ), ...
    awf_on_snr_first( crop+1:end-crop, crop+1:end-crop ));

disp(' Error Individual Frame method' )
err_mae_ind_first(i)=err_awf_opt.mae;
end

figure;
plot(cr_desired,err_mae_ind_opt,'k-o',cr_desired,err_mae_ind_first,'k-*');
%title(' Optimum vs First Value of SNR');
legend(' Optimum SNR Value ',' First Optimum SNR','location','best');
xlabel(' Compresion Ratio');
ylabel(' Mean Absolute Error');
textString = sprintf('SNR Value = %.2f', snr_opt(1));
hold on;
text(2,6.5 , textString);
textString = sprintf('SNR Value = %.2f', snr_opt(2));
text(12,9 , textString);
textString = sprintf('SNR Value = %.2f', snr_opt(3));
text(26,11, textString);
textString = sprintf('SNR Value = %.2f', snr_opt(4));
text(36,12, textString);

```

```

textString = sprintf('SNR Value = %.2f', snr_opt(5));
text(41.5,14, textString);
axis([0 56 6 16]);
print -depsc snr_plot.eps;

```

MATLAB function for Individual frame method of JPEG2000 Compression

```

function [ decompressed, cr_actual ] = individual_frame_jp2k( ...
    lr_image, cr_desired, bits )
%
% [ decompressed, cr_actual ] = individual_frame_jp2k( ...
%   lr_image, cr_desired, bits )
%
% Does individual frame JP2K compression and decompression on an image or
% array of images. Returns the decompressed image and the actual
% compression ratio.
%
% REQUIRED INPUTS
%   lr_image      2D or 3D array (should be uint8 or uint16)
%   cr_desired    Desired compression ratio
%   bits          Input bits per pixel (8 or 16 only)
%
% OUTPUTS
%   decompressed   Compressed then decompressed image(s)
%   cr_actual      Actual compression ratio
%
% Author : Barath Narayanan
% Date   : 08/28/2012
% Modified by R. C. Hardie 12-5-12

% Get size
[sy,sx,sz] = size(lr_image);
% Recast as needed
if bits == 8
    lr_image = uint8(lr_image);
else
    lr_image = uint16(lr_image);
end

% Initialize variable storing number of bytes after compression

```

```

total_compressed_bytes = 0;

% Temporary JP2K file name
file_name = 'Temp_File.jp2';

% Loop over frames
for k = 1 : sz

    % Do JP2K Compression
    imwrite( lr_image(:, :, k), file_name, 'CompressionRatio', cr_desired );

    % Decompress
    decompressed(:, :, k) = double( imread(file_name) );

    % Accumulate file size for actual compression ratio calculation
    file_info = imfinfo(file_name);
    total_compressed_bytes = total_compressed_bytes + file_info.FileSize;

end

% Complete the actual compression ratio calculation
if bits == 8
    % One byte per pixel in uncompressed image
    cr_actual = (sx*sy*sz) / total_compressed_bytes;
else
    % Two bytes per pixel in uncompressed image
    cr_actual = (sx*sy*sz*2) / total_compressed_bytes;
end

```

MATLAB function for difference frame method of JPEG2000 Compression

```

function [ out, cr_actual, S ] = difference_frame_jp2k_v2( ...
    in, reg, cr_ref, cr_err, bits, num_std )
%
% This Function compresses the given image sequence using difference frame
% method based on global shift registration.
%
% Syntax:
%
% [ out, cr_actual, S ] = difference_frame_jp2k_v2( ...

```

```

% in, reg, cr_ref, cr_err, bits, num_std )
%
% REQUIRED INPUTS
% in      3D array with input images, in(:, :, 1) must be reference
%
% reg      Structure with registration parameters
% reg.buff
% reg.thresh
% reg.pre_std
% reg.method
%
% cr_ref    CR for reference frame (based on bits)
% cr_err    CR for difference frames (also based on bits, but actual
%           error compression is done on 16 bits)
%
% bits      Bits per pixel in input imagery (8 or 16 only)
%
% num_std    Num of +/- standard deviations of the difference frame
%           values will be scaled to fit in a 16 bit dynamic range for
%           compression.
%
% OUTPUTS
% out      compressed and then decompressed version of "in"
%
% cr_actual  Structure of achieved compression ratios
% cr_actual.ref
% cr_actual.diff
% cr_actual.overall
%
% Author : Barath Narayanan
% Date : 06/21/2012
%
% VERSION 2
% Modified by R. C. Hardie 12-8-12
% Use individual_frame_jp2k.m for reference (eliminate other subfunctions)
% Add controls over the error scaling (num_std)
% 8 or 16 bit input option
% use register_shift.m (has prefilter option to improve registration)
% Better comments

```

```

% Get number of frames
sz = size( in, 3 );

% Make sure in is double precision for registration
in = double(in);

% Extract reference image
ref_frame = in(:, :, 1); % 1st frame is considered as the reference frame

% Recast reference as needed
if bits == 8
    ref_frame = uint8( ref_frame );
    cr_err = cr_err*2; % map cr_err for 16 bit error, from 8 bit data
    % You need to double the 16 bit compression to get compression relative
    % to 8 bit data
else
    ref_frame = uint16( ref_frame );
end
% Compress and decompress reference frame
[ out(:, :, 1), cr_actual.ref ] = individual_frame_jp2k( ...
    ref_frame, cr_ref, bits );
% Compute and compress difference frames
for k = 2 : sz

    % Register reference to each other frame
    S(k, :) = register_shift( in(:, :, 1), in(:, :, k), reg.thresh, ...
        reg.buff, reg.pre_std, reg.method );

    % Apply shift
    shift_ref_frame_k = do_shift( out(:, :, 1), S(k, :), reg.method );

    % Compute the difference frame
    error = in(:, :, k) - shift_ref_frame_k;
    % Scale error into unsigned 16 bit range (error should be zero mean)
    dynamic_range = 2^16;
    Sd = std(error(:));
    G = dynamic_range/(num_std*2*Sd);
    error_scaled = G*error + dynamic_range/2;

    % Compress difference frame (treat as 16 bit, regardless of input)

```

```

[ out(:,k), cr_diff(k-1) ] = individual_frame_jp2k( ...
error_scaled, cr_err, 16 );

% Note: cr_diff is based on a 16 bit input. If in( ) is 8 bit we need
% to adjust the final compression ratio accordingly.
if bits == 8
    cr_diff(k-1) = cr_diff(k-1)/2;
end

% Scale error back to a signed double precision value
out(:,k) = ( out(:,k) - (dynamic_range/2) ) / G;

% Get frame estimate by adding back the decompressed reference
out(:,k) = out(:,k) + shift_ref_frame_k;

end

% Compute the compression ratios
cr_actual.diff = (1/(sz-1))*sum( cr_diff );
cr_actual.overall = (1/sz)*cr_actual.ref + (1/sz)*sum( cr_diff );

```

MATLAB function to determine error between the given HR image and SR image

```

function error = AWF_SR_dis_error( var_n, lr_observed, hr_true, params, stats, filt, type
)
%
% error = AWF_SR_dis_error( var_n, lr_observed, hr_true, params, stats, filt, type );
%
% stats must be such that var_d = 1;
% type = 'mse', 'mae', 'snr_dB', 'psnr_dB'
% filt.ref
% filt.pre_std
% filt.method
% Perform the AWF on decompressed
awf = AWF_SR_dis( lr_observed, params, stats, var_n, ...
    filt.ref, filt.pre_std, filt.method );
% Compute error
crop = (params.obsx*params.L-params.L);
err = dif2( hr_true( crop+1:end-crop, crop+1:end-crop ), ...
    awf( crop+1:end-crop, crop+1:end-crop ));
error = getfield( err, type ); % Export the selected field

```