Stellingen

behorende bij het proefschrift

Subsampling Methods for Image Sequence Coding

door R.A.F. Belfor

9 juni, 1994

1. Bij het ontwerpen van een digitaal compressiesysteem op basis van onderbemonstering dient het gebruik van een bandbegrenzend voor-filter zoveel mogelijk voorkomen te worden.

   (Dit proefschrift, Hoofdstuk 3, 4 en 5)

2. Digitale compressiesystemen op basis van onderbemonstering zijn gevoelig voor hoogfrequente ruis.

   (Dit proefschrift, Hoofdstuk 3, 4 en 6)

3. Binnen de klasse van compressiesystemen voor digitale beeldsequenties gebaseerd op onderbemonstering verdient een spatieel adaptief systeem met bewegingscompensatie de voorkeur.

   (Dit proefschrift, Hoofdstuk 7)

4. De op medische gronden vastgestelde limiet van maximaal twee scène-wisselingen per seconde is ook te verdedigen vanuit het oogpunt van compressie.

5. Het blijven streven naar volledige benedenwaartse compatibiliteit belemmert op den duur de vooruitgang van de techniek.

6. Het 'student-proof' maken van een systeem vereist een grotere inspanning dan het 'fool-proof' maken.

7. Door de passieve houding van de politie ten opzichte van fietsendiefstal is een fiets openbaar bezit geworden.

8. Technologisch onderzoek in ontwikkelingslanden dient zich niet in eerste instantie te richten op de verbetering van de huidige stand van de techniek, maar op het verwerven van een zelfstandige positie ten opzichte van de ontwikkelde landen.

9. Het getuigt van arrogantie van de westerse beschaving om Columbus te beschouwen als de *ontdekker* van het Amerikaanse continent.

10. Ontwikkelingshulp is geen gunst aan ontwikkelingslanden maar moet gezien worden als herstelbetaling voor de ontwrichting van de sociaal-economische structuur gedurende 300 jaar kolonisatie.

11. Het hebben van principes is een luxe-probleem.

Subsampling Methods for Image Sequence Coding

R.A.F. Belfor

Subsampling Methods for Image Sequence Coding

PROEFSCHRIFT

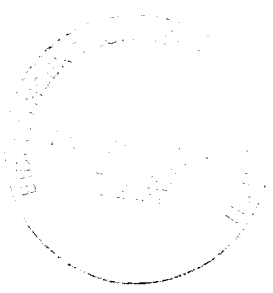
ter verkrijging van de graad van doctor
aan de Technische Universiteit Delft,
op gezag van de Rector Magnificus Prof. ir. K.F. Wakker
in het openbaar te verdedigen ten overstaan van een commissie,
door het College van Dekanen aangewezen,
op donderdag 9 juni 1994 te 10:30 uur

door

Ricardo Alexander Frederik BELFOR,

elektrotechnisch ingenieur,

geboren te Zeist.



Dit proefschrift is goedgekeurd door de promotor:
Prof. dr. ir. J. Biemond,

en de toegevoegd promotor:
Dr. ir. R.L. Lagendijk

CIP-DATA KONINKLIJKE BIBLIOTHEEK, DEN HAAG

Belfor, R.A.F.

Subsampling Methods for Image Sequence Coding / R.A.F. Belfor, – Delft : Technische Universiteit Delft, Faculteit der Elektrotechniek, – Ill.
Thesis Technische Universiteit Delft, – With ref. – With summary in Dutch.
ISBN 90-5326-017-X
Subject headings: image coding / video coding / image processing.

Table of Contents

SUMMARY

Recent developments in the area of digital signal processing hardware and the advent of broadband digital communication channels have made it possible to offer new audio-visual services to the consumer at home. These services cover almost every aspect of our daily life, such as business, education and entertainment. High definition television (HDTV) gives a better image and sound quality than conventional television. Other new audio-visual services are, for instance, the videophone, interactive television and the digital video recorder. All of these new services require the transmission of images, sounds and binary information. The important role of data compression is to reduce the amount of information to be stored or transmitted. If the information is coded as efficiently as possible, the transmission time or storage requirements decrease and the costs of the services are reduced.

Subsampling can be used as a method to compress digital image sequences. A digital image sequence consists of a spatial and temporal set of picture elements ("pixels"). In a subsampling system the information to be transmitted is reduced by transmitting only part of the pixels. At the receiver, the discarded pixels have to be recovered by using information which was transmitted across the channel (interpolation). To facilitate a good interpolation, it is important that the transmitted pixels contain sufficient information. This involves a trade-off between the required quality, the compression factor and the complexity of the coding system. In this thesis different subsampling methods are discussed.

An important choice in a subsampling system is the sampling lattice. This is a set of discrete points in the three-dimensional space. Sampling lattices are used to define the pixel positions in both the original and the subsampled image sequence. Different frequency components can be emphasized or suppressed by adjusting the shape of the sampling lattice. The density of the subsampling lattice determines the compression factor. The sampling lattice also determines the positions of the spectral replica of the baseband introduced by the subsampling process. To prevent aliasing errors, spectral replica should not overlap each other. Therefore the frequency components which give rise to aliasing have to be suppressed prior to subsampling with a lowpass filter, the *prefilter*. If there are no frequency components which cause aliasing, the prefilter becomes obsolete. At the receiver the missing pixels are recovered with a lowpass filter that recovers the original baseband spectrum: the *interpolation filter*.

Systems that use a fixed subsampling lattice make up the first group of spatio-temporal subsampling systems. In a simple fixed lattice subsampling system, fixed linear prefilters and interpolation filters are used. The reason for using these kind of systems is the reduced sensitivity of the human visual system for diagonal frequency components. These frequency components are not transmitted if a *quincunx* sampling lattice is used. Another application of simple fixed lattice subsampling systems is the subsampling of the color components of the

image sequence. Usually the energy contribution of color components is relatively low, allowing for the use of a lattice with a low sampling density. The simple fixed lattice subsampling system is not suitable for the image sequence luminance information.

The image quality of fixed lattice subsampling systems can be improved by using an adaptive interpolation filter instead of a fixed interpolation filter. The aim is to recover a part of the resolution lost during the subsampling process and therefore no prefilter should be used. The absence of a prefilter can introduce spatio-temporal aliasing which may not be canceled by the adaptive interpolation filters. Two classes of adaptive interpolation methods are discussed, namely motion adaptive interpolation and nonlinear interpolation. The interpolation can be adapted to motion in the image sequence by using motion compensated interpolation. With a motion compensated interpolation filter the spatio-temporal passband of the interpolation filter is adapted to the local motion vector. During the interpolation process no spatial resolution is lost, so in this case the spatial resolution of the interpolated image sequence is higher than for a fixed lattice subsampling system. For a good interpolation result, it is, however, necessary that the motion vectors are estimated with a high accuracy and correspond with the true motion. Because accurate motion estimation is not always possible, there must always be a mechanism to detect this situation and to perform an alternative interpolation. Detection can be done at the encoder by considering the magnitude of the interpolation error. A simple fixed lattice subsampling system with non-adaptive interpolation can serve as a fall-back system in the case of a locally inaccurate motion estimate.

The second class of adaptive interpolation filters is that of nonlinear interpolation filters which are based on the median filter. The advantage of these filters is that the interpolation result is not determined by the weighted sum of pixel values in a spatio-temporal window around the missing pixel. Instead, the pixel values majority determines the result. If the missing pixel belongs to this majority then the interpolation result is better than the result after fixed linear interpolation. The interpolation result can be improved further by using a hybrid filter, consisting of a combination of linear and nonlinear filters. A motion compensated interpolation filter can also be used in this context. Experiments show that nonlinear interpolation filters offer only a slight improvement over linear interpolation filters, mainly because of uncanceled aliasing.

The second group of spatio-temporal subsampling systems uses adaptive subsampling lattices instead of a fixed subsampling lattice. In a system with a spatially adaptive subsampling lattice the shape and the density of the local sampling lattice is adapted to the local image contents. The image is divided into blocks and for each block the optimal sampling lattice is determined. With the help of rate-distortion theory it can be shown that this approach always leads to a lower distortion than the distortion of a fixed lattice subsampling system. This theory can also be used to assign different sampling lattices to the blocks, when a target compression factor is specified.

A consequence of spatially adaptive subsampling is that at the receiver, interpolation should be done on a non-uniform sampling lattice. The sampling lattice within a block is, however, uniform. Using these pixels, polynomials of different orders can be estimated with least-squares estimation. At the decoder the polynomials are used to compute the missing pixels. An alternative interpolation method builds a hierarchical pyramid, where the lattices with a low sampling density are expanded to a lattice with a higher density. In this situation it is required that each sampling lattice is a subset of the lattices with a higher sampling density.

A temporally adaptive lattice can be accomplished by using motion information. If a block can be predicted from the previous image with the aid of motion information, there is no need to retransmit that block. As a consequence, the temporal sampling frequency can be adapted to the temporal activity. Experiments show an improvement with respect to fixed lattice subsampling systems, and a quality improvement if the size of the set of possible local sampling lattices is increased.

Another approach to obtain an adaptive subsampling lattice is to adapt the spatial lattice to the presence of motion. A stationary area in an image does not change in time, therefore the spatial sampling of this area can be spread over several images. The subsampling lattice is shifted for every image in such a way that over a period of several images, every pixel of the stationary area is transmitted. This technique is call "sub-Nyquist" sampling, because the spatial sampling frequency of each image is lower than the prescribed Nyquist frequency.

Sub-Nyquist sampling is used in several standardized compression systems for HDTV, namely the Japanese MUSE system and the European HD-MAC system. In practice, in a system there should always be a fall-back mode in case the image sequence contains non-stationary areas. In both systems, this is a fixed lattice subsampling system in combination with spatial interpolation. In the HD-MAC system the subdivision into stationary and non-stationary areas is done on a block basis, whereas in the MUSE system the subdivision is done on a pixel basis. In the HD-MAC system the decision as to which mode is used is transmitted as side information, but in the MUSE system the decision is repeated at the decoder. Therefore, in the HD-MAC system the complexity is concentrated at the encoder, whereas in the MUSE system both the encoder and the decoder have the same complexity.

Sub-Nyquist sampling can be applied to moving areas of the image sequence if the motion information is taken into account and the problem of critical velocities is solved. If a fixed sampling lattice is used, the combination of the subsampled images to obtain the original stationary image is no longer possible for certain velocities. The use of an adaptive lattice which is appropriately chosen solves this problem. An increase in the accuracy of the motion vector improves the overall result. Experiments show that motion compensated sub-Nyquist sampling outperforms non-motion compensated sub-Nyquist sampling because of the suitability of the method for non-stationary parts of the image sequence.

Subsampling can be combined with transform coding to reduce the complexity of the transform coding system. Combining the two systems can also be used to adapt the transform coding system to the human visual system, namely by excluding certain spectral regions. The

remaining part of the spectrum after subsampling can now be coded more accurately. Further, a motion adaptive system can be constructed by combining sub-Nyquist sampling and transform coding. In practice, only subsampling systems with a uniform sampling lattice can be combined with a transform coding system. The mean square error of a fixed lattice subsampling system combined with transform coding is always larger than the error of a system with only transform coding. However, in practice, this objective quality criterion is not the reason for combining these systems. When a transform coding system is combined with a sub-Nyquist sampling system, a theoretical analysis shows that the mean square error is always smaller than the error of a standard frequency transform coding system. Experiments show that the conditions under which this theoretical analysis is made are not valid in practice

When comparing different subsampling systems, we can conclude that the systems with a fixed sampling lattice are always outperformed by systems which use an adaptive subsampling lattice. An exception to this observation is the system with motion compensated interpolation, which also gives good results. The best results are obtained when a spatially and temporally adaptive subsampling lattice is used, because in this case the knowledge about the local image contents is used most effectively.

CHAPTER 1

INTRODUCTION

Recent technological innovations have made it possible to provide a broad range of new visual communication services to the consumer at home. Fast digital signal processing hardware and high speed digital communication channels make it possible that these new services will be implemented in the near future, and will cover almost every aspect of our daily lives (e.g. business, communication, education, entertainment). For example, high definition television (HDTV) provides video with a better image and sound quality and has applications in entertainment and education, while the videophone will change the manner of long-distance communication in business environments and private homes. Information can be gathered more flexibly and effectively using multimedia or interactive television. The future digital video recorder will enable the storage of high quality video. Common to these new services is the need to transmit sounds, images and binary data. The important role of data compression is in reducing the amount of information to be transmitted. If the information is transmitted as efficiently as possible, the transmission time decreases. The effective capacity of the storage devices also increases and hence more efficient use is made of the available resources. Further, the costs of the services are reduced and more services can be offered to each consumer.

Visual communication services require the transmission of image sequences. In an image sequence compression system, there are two reasons why the information which has to be transmitted can be reduced. The statistical redundancy in the image sequence is removed by exploiting the spatial and the temporal correlation. The information can be reduced even further by transmitting only the information which is relevant to the application. In the past, many different image coding system were proposed. Most originate from speech coding [Jaya84]. Simple systems are, for example, pulse code modulation (PCM) and differential pulse code modulation (DPCM). Examples of more advanced systems are vector quantization [Gers92], transform coding (e.g. [Jaya84] [Jain89] [Wood91]) and hybrid coding systems (e.g. [Nino82] [MPEG92]). In this thesis subsampling is discussed as an image coding technique.

The outline of this chapter is as follows. First the principles of subsampling are briefly discussed for a one-dimensional signal. Next the discussion is extended to the subsampling of image sequences. There are many different ways to carry out the subsampling of image sequences, and therefore an overview and classification is given of the different methods. The chapter is concluded with a discussion and explanation of the structure of this thesis.

1.1 Subsampling principle

As real-life scenes are continuous in both space and time, prior to subsampling, sampling is a necessary operation. Since sampling has such a close relation with subsampling, we first briefly review the sampling of a continuous signal, and follow with a discussion on the principle of subsampling.

1.1.1 Sampling

In digital signal processing, a signal is processed as a discrete set of samples. In Figure 1.1, sampling is illustrated for a one-dimensional time-continuous signal $u_c(t)$. On the left side $u_c(t)$ is shown and on the right side the time-discrete signal $u(t)$. We see that the samples are obtained by evaluating the continuous signal at regularly spaced instances in time, with a distance of T seconds between each sample. Thus $u(t)$ is equal to

$$u(t) = \begin{cases} u_c(t) & t = k \cdot T \\ 0 & \text{otherwise} \end{cases}, \quad k \in \mathbb{Z} \tag{1.1}$$

For perceptual purposes, the discrete signal should be converted back to a continuous signal when displayed. This is done by filling up the gaps between the samples with a time-continuous low-pass filter. This process is called *interpolation*. The dashed line in Figure 1.1 indicates the continuous signal that is obtained after interpolation. It can be observed that the fine details, corresponding with the high frequency components in the continuous signal, are lost in the sampling process. This yields a distortion of the original signal.

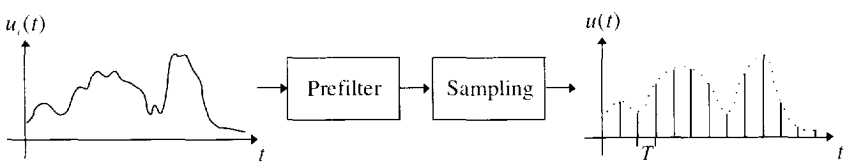


Figure 1.1: Sampling of a time-continuous signal.

The distortion is reduced if the sampling distance is decreased, because the continuous signal can now be followed more accurately. However, decreasing the sampling distance means that the number of samples increases. For an efficient representation of the signal, the number of samples must be as small as possible. The maximum distance between the samples is prescribed by the Nyquist theorem [Nyqu28]. This theorem state that, in order to avoid distortion, a signal has to be sampled with a sampling distance less than or equal to the reciprocal of twice the bandwidth W of that signal:

$$T \leq \frac{1}{2W} \tag{1.2}$$

This relation effectively limits the number of samples necessary to represent a given time-continuous signal.

The meaning of Equation (1.2) can also be reversed. Given a sampling distance T it determines the maximally allowed bandwidth W of the input signal. This explains the necessity of the low-pass filter preceding the actual sampling process. This *prefilter* should guarantee that there are no frequency components violating the Nyquist theorem. Note that therefore a prefilter is only necessary if there actually *are* frequency components above the maximal bandwidth. In practice, perfect low-pass filters cannot be realized, and therefore there are always frequency components that violate the Nyquist theorem. The prefilter has to be designed in such a way that the influence of these components is negligible.

1.1.2 Subsampling

Subsampling means that some of the samples of a discrete signal are discarded. Therefore fewer samples are used to represent the same signal, thus yielding a data reduction. Effectively this means that the sampling distance is increased. We saw in the previous section that, according to the Nyquist criterion, if the sampling distance is increased, the bandwidth of the signal should be decreased. A decrease of the bandwidth affects the fine details of the signal, corresponding with the high frequency contents of the signal. Hence a data reduction is achieved at the expense of some distortion.

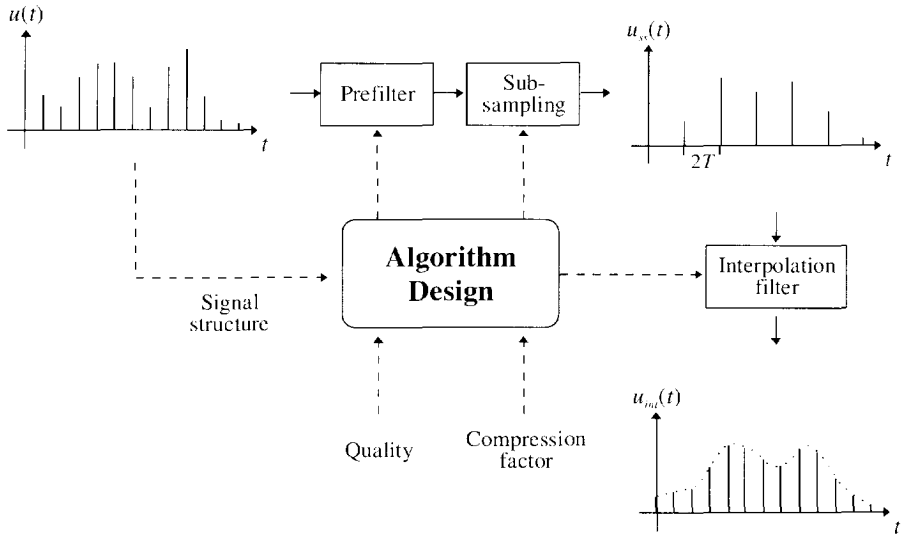


Figure 1.2: Subsampling of a time-discrete signal with a factor two.

An example of a subsampling system is given in Figure 1.2. The input of the system is the discrete signal $u(t)$ from Figure 1.1. The sampling distance is increased in the subsampling stage from T to $2T$, yielding a data reduction factor of two. The subsampled signal $u_{ss}(t)$ is now given by

$$u_{ss}(t) = \begin{cases} u(t) & t = k \cdot 2T \\ 0 & \text{otherwise} \end{cases}, \quad k \in \mathbb{Z} \tag{1.3}$$

hence the odd samples of the signal are discarded. Because the sampling distance is doubled, the bandwidth should be reduced to ½W. This is done by the prefilter prior to the subsampling. The time-discrete interpolation filter at the decoder is used to fill in the missing samples which were present in $u(t)$ but which are not present in $u_{ss}(t)$, hereby forming the interpolated time-discrete signal $u_{int}(t)$. Note that this is a discrete variant of the time-continuous interpolation from the previous section. In Figure 1.2 the dashed line connecting the samples of $u_{int}(t)$ represents the continuous signal which can be associated with the interpolated signal. Compared to the time-discrete signal shown in Figure 1.1, we see that more details are lost, which corresponds to a larger distortion.

The design of a subsampling algorithm is controlled by three major design parameters which are also indicated in Figure 1.2. The first parameter is the structure of the input signal. As an illustration of this consider the two discrete signals $u_1(t)$ and $u_2(t)$ shown on the left side of Figure 1.3. We see that $u_2(t)$ contains more details than $u_1(t)$. The two signals are fed into the same simple subsampling system which discards half the samples of the input signal. On the right side, the subsampled version $u_{ss1}(t)$ and $u_{ss2}(t)$ of the input signals are shown. $u_{ss2}(t)$ is distorted more than $u_{ss1}(t)$ because most of the detail information is lost. In the design process, the fact that the same subsampling scheme has a different impact on different input signals has to be taken into consideration. So the subsampling should be adapted to the structure of the input signal.

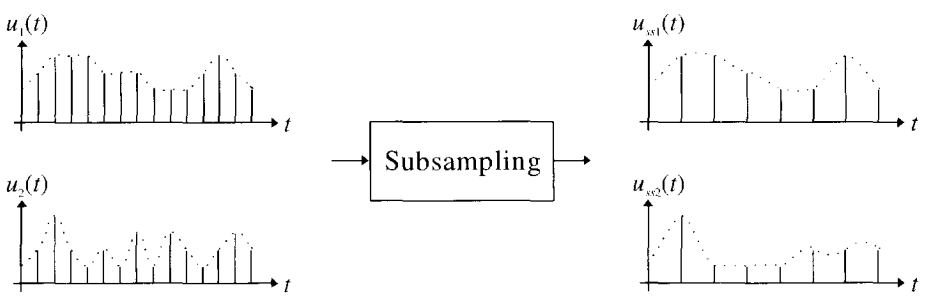


Figure 1.3: The subsampling of two different signals $u_1(t)$ and $u_2(t)$.

The two other design parameters are the desired quality and the compression factor. These parameters are usually prescribed by the application. In the simple subsampling system shown in Figure 1.2, they determine the necessary subsampling factor, and the bandwidth of the prefilter and interpolation filter. In every compression application the quality and the compression factor have to be traded off against each other. An increase in the compression factor leads to a decrease in the quality. Different compression schemes can be compared by comparing the increase in quality for a given decrease in compression factor. In Figure 1.4, a simple and a more advanced subsampling scheme are compared with each other. The distortion is used as a measurement of the quality. We see that the distortion of the simple scheme is generally larger than the distortion of the advanced scheme for the same

compression factor. Also the relative decrease in the distortion (increase in quality) caused by a decrease in the compression factor is larger for the simple scheme.

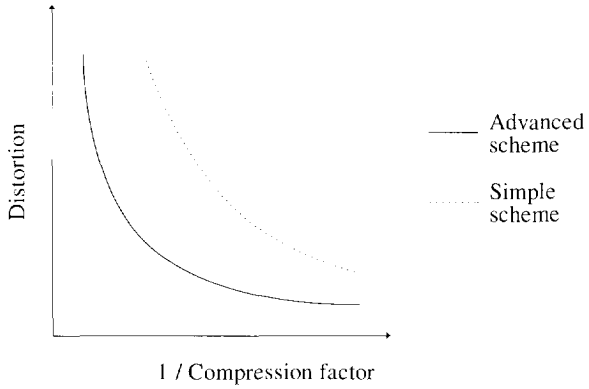


Figure 1.4: The relation between the distortion and the compression factor for two subsampling schemes.

1.2 Subsampling of image sequences

1.2.1 Relation to subsampling of one-dimensional signals

Subsampling, applied to image sequences, reduces the number of pixels to represent a given image sequence. It is obvious that a decrease in the number of pixels, reduces the required transmission bandwidth. In Figure 1.5 it is shown how subsampling can be integrated into a complete coding system. The role of the prefilter and the interpolation filter was discussed in the previous section. After applying subsampling, a set of samples, now called *pixels*, has to be transmitted. This set may be transmitted either analog or digital. Digital transmission can be done using PCM-encoding. If after subsampling there is still some correlation between the pixels, an additional coding scheme can be used. This is indicated by the dashed boxes before and after the channel in Figure 1.5.

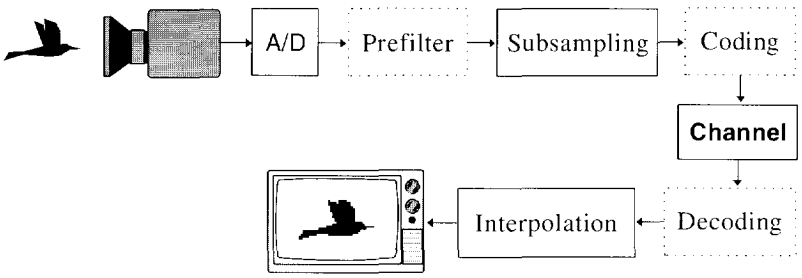


Figure 1.5: General coding scheme.

In the previous section we saw that the structure of the input signal is one of the algorithm design parameters. An important property of an image sequence is the spatial correlation between the pixels. If the spatial correlation is large, the bandwidth of the signal is narrow, and consequently the spatial sampling frequency can be reduced. In the extreme case that an image sequence consists of images with a constant intensity value, only one pixel per image is necessary to completely describe the image sequence. Another property is the amount of motion within the image sequence, as it determines the temporal correlation between the pixels. The temporal correlation is maximal if the contents of the scene are not moving. In that case, the temporal sampling frequency (image rate) can be reduced.

Besides the correlation, the variance of the image also plays an important role. It requires more information to represent a signal with a high variance than a signal with a low variance given the same amount of permissible distortion. In image sequences, generally, the variance of the luminance component is higher than the variance of the chrominance components. Therefore a distinction is made between the subsampling of the luminance component and the chrominance components. More sophisticated algorithms should be used for the luminance component than those necessary for the chrominance components.

The above discussion focused on the *number* of samples necessary to represent an image sequence, but another aspect of subsampling is the *position* of the samples. The main difference between image sequence compression applications and speech compression applications is that in image sequence compression we have a three-dimensional spectrum instead of a one-dimensional spectrum. In one dimension, the only degree of freedom in the sampling process is the distance between the samples. If the dimension is higher than one, the position of the samples also becomes important. The *sampling lattice* prescribes the specific location of the samples. The shape of the sampling lattice can be adapted to the spectrum of the image. Consider, for example, an image which contains more frequency components in the horizontal direction than in the vertical direction. The samples can then be distributed in such a way that there are more samples in the horizontal direction than in the vertical direction, hereby increasing the horizontal bandwidth.

The quality and the compression factor are controlled by the application. Some factors which are important in this case are the image dimensions, the bandwidth of the communication channel and the end-user of the image sequence. For instance, in a HDTV application, the quality is more important than in a videophone application. HDTV images also have larger dimensions than videophone images. These differences have to be taken into consideration in the algorithm design.

1.2.2 Overview of subsampling methods

There are various possibilities for the design of a subsampling algorithm for image sequence coding. To facilitate a structured discussion of the different approaches, the tree structure shown in Figure 1.6 is used. First a distinction is made between systems with a fixed subsampling lattice and those with an adaptive subsampling lattice. In a fixed lattice system, the most important aspect with respect to reducing the total distortion, is the interpolation.

Therefore a further subdivision is made based on the different interpolation methods. In a system with an adaptive subsampling lattice, the most important aspect is the basis on which the sampling lattice is adapted, namely spatially adaptive versus motion adaptive. Each of these methods can of course be combined with the more sophisticated interpolation methods from the other branch of the tree.

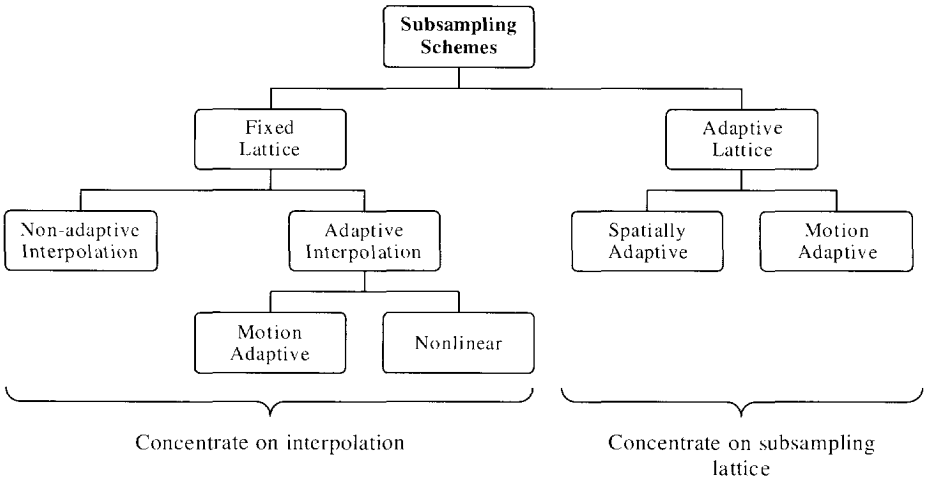


Figure 1.6: Overview of subsampling methods.

The simplest form of subsampling is fixed lattice subsampling with non-adaptive interpolation. The image is subsampled on a fixed lattice and reconstructed with a linear interpolation filter without taking the local structure into account. The theory on multi-dimensional sampling as described by Mersereau and Speake [Mers83] and by Dubois [Dubo85] applies to this situation. In [Mers83] the conversion from one sampling lattice to another lattice is discussed. In [Dubo85] the author elaborates more on this subject and several examples of subsampling lattices are given. In [Tong81] and [Reut85] more examples of fixed subsampling lattices are given. These are all applied to the luminance component of the image sequence. Girod and Geuen show in [Giro89] that fixed subsampling can also be applied to the chrominance components of the image sequence. Details about the necessary linear prefilters and interpolation filters can be found in [Pirs83], [Tong81] and [Sioh91].

A first improvement to the fixed lattice subsampling can be made by adapting the interpolation to the motion in the image sequence. Now motion compensated linear interpolation filters are used instead of fixed linear interpolation filters. In [Reut89] and [Erns91] this technique is used to interpolate missing frames in an image sequence and in [Giro85] motion compensation is used to convert an interlace scan image sequence into a progressive scan sequence. In [Hagh88] motion compensated interpolation is proposed as a branch in the HD-MAC encoding scheme to interpolate fields skipped at the encoder. In [Golz90] motion compensated interpolation was used to interpolate temporally subsampled high frequency components of an image sequence.

An alternative to linear interpolation is nonlinear interpolation. In [Hent89] a median filter is used to convert an interlace scan image sequence to a progressive scan image sequence. The advantage of nonlinear filters is the better preservation of high frequency details. Another advantage is the relatively simple implementation compared to linear filters with the same quality of the interpolation result. Median filtering does not work properly for all situations. Therefore FMH (FIR-median hybrid) filters introduced by Heinonen [Hein87] can be used instead. In [Leht90] and [Huuh92] these filters are used to interpolate an image subsampled with a fixed quincunx lattice. The choice between a median filter and an FIR filter is based on the local contents of the image. Therefore these filters fall into the class of adaptive interpolation techniques. In [Wang90] and [Haav92] motion compensated interpolation is combined with nonlinear filtering for de-interlacing.

The second major class of subsampling schemes are methods based on an adaptive subsampling lattice instead of a fixed lattice. The first possibility is to adapt the subsampling lattice to the local spatial frequency content. Tanimoto et al. [Tani88] describe a method where two different lattices are used: a dense sampling lattice is used for detailed regions and a sparse sampling lattice for regions containing little detail. In [Kish88], [Ashi88] and [Saku90] systems with three different sampling lattices are described. The number of different sampling lattices determines the efficiency of the algorithm. Increasing the number of sampling lattices improves the adaptation to the local statistics. In [Hesp93] and [Belf93b] a scheme is presented with an arbitrary number of sampling lattices. This system is extended in [Belf94] by including temporal adaptivity.

An adaptive sampling lattice is also used in motion adaptive subsampling schemes. An subsampling technique that is based on this approach is sub-Nyquist sampling [Tong87] [Scha87]. For stationary and non-stationary parts of the image sequence, a different subsampling lattice and interpolation filter is used. Sub-Nyquist sampling is used as an image compression method for the standardized HDTV transmission systems MUSE [Nino87], and HD-MAC [Vree89] [Hagh90], which are completely based on this technique. Recent proposals use this technique in combination with transform coding [Scha90] [Vos92]. In [Belf92a] motion compensated sub-Nyquist sampling is used instead of sub-Nyquist sampling. By using this technique, the application range of sub-Nyquist sampling is no longer limited to stationary parts of the image sequence. Sub-Nyquist sampling is now applicable to all parts of the image for which the motion can be estimated accurately.

1.3 Thesis overview

The organization of this thesis globally follows the tree structure shown in Figure 1.6. Chapter 2 concentrates on fixed lattice subsampling systems with linear non-adaptive interpolation filters. This system is used to discuss the basics of subsampling. We look at the different parts of the subsampling system, such as the sampling lattice, the prefilter and the interpolation filter. Furthermore we analyze the different types of errors in a subsampling system from a rate-distortion theory point of view. Also some properties of the human visual

system that are used in a subsampling system are pointed out. The chapter is concluded by discussing some applications of fixed lattice subsampling.

The interpolation process is, as has already been pointed out, the only aspect in a fixed lattice subsampling scheme open for improvement. Therefore, in Chapter 3, some alternatives for the linear non-adaptive interpolation are discussed. The first alternative is motion compensated interpolation where the interpolation is driven by an estimate of the motion vector. First, motion compensated interpolation is discussed from a theoretical viewpoint, where we look at the relation between the accuracy of the estimated motion vector and the quality of the interpolation result. Then the practical aspects are covered, which include the structure of the interpolation filer, the estimation of the motion and the handling of inaccurate motion vectors. A second alternative for linear interpolation filters are nonlinear interpolation filters based on median filters. Here we focus on the choice of the filter structure and the image structures that can properly be interpolated by each filter structure.

The restriction of a fixed subsampling lattice prevents the effective use of all the image characteristics. Therefore, in Chapters 4 and 5, adaptive sampling lattices are used instead. In Chapter 4, the sampling lattice is adapted to the local image structure. This process is driven by both the amount of local activity and the shape of the local spatio-temporal spectrum. Rate-distortion theory is used to show the relative performance of this system compared to fixed lattice subsampling. Practical aspects which we address are how the sampling lattice can be adapted and how to perform the interpolation on a non-uniform sampling lattice.

In Chapter 5, sub-Nyquist sampling and motion compensated sub-Nyquist sampling is discussed. In these schemes the locally selected sampling lattice and interpolation method are directly coupled to the motion in the image sequence. In a sub-Nyquist sampling system, the sampling lattice is influenced by motion detection. The MUSE system and the HD-MAC system serve as an illustration of sub-Nyquist sampling. In a motion compensated sub-Nyquist sampling system, the sampling lattice is based on local motion information. The required accuracy of the motion estimate is an important topic in this chapter.

Chapter 6 considers the problem of combining transform coding with either fixed lattice subsampling or sub-Nyquist sampling. For the transform coding stage, subband coding or discrete cosine transform coding is used. The fundamental difference between subsampling and transform coding is discussed and an investigation is made whether transform coding can benefit from being combined with subsampling.

Finally, in Chapter 7, the different subsampling systems are compared with each other. This is done by combining the experiment results obtained in the preceding chapters. Both objective and subjective quality measurements are taken into consideration.

Chapter 2

FIXED LATTICE SUBSAMPLING

Fixed lattice subsampling is the least complicated method of subsampling. Certain pixels are simply discarded at the encoder and interpolated at the decoder without any regard for the actual image contents. Fixed lattice subsampling does not take local image statistics into account but relies on the three-dimensional frequency response of the human visual system. The advantage of fixed lattice subsampling is the straightforward implementation. The topics addressed in this chapter are not exclusively of importance for fixed lattice subsampling; they are also the foundation for the description of more sophisticated subsampling schemes in the following chapters.

The chapter starts with a concise description of the basics of multi-dimensional sampling [Dubo85]. Some examples of sampling lattices are also given [Tong81] [Reut85]. Next the different stages of a fixed lattice subsampling system are discussed in more detail. Special attention is given to the structure of the different filters used. To classify the different types of distortion introduced in the subsampling process, subsampling is described from a rate-distortion point of view [Berg71] [Belf94]. Next, several characteristics of the human visual system which are used in a subsampling system are reviewed [Kell79] [Nino87] [Giro88].

Toward the end of this chapter some applications and experiment results are discussed. A distinction is made between the application of fixed lattice subsampling on either the luminance or the chrominance components of the image.

2.1 Multi-dimensional sampling

Sampling is a basic operation in image processing and image communication systems. Figure 2.1. shows the sampling of an image sequence. First the continuous three-dimensional time-varying scene is projected onto a plane, thus forming a continuous two-dimensional time-varying version of the scene. For the purpose of digital processing and transmission, this three-dimensional intensity function is sampled and becomes discrete in both space and time. In this section first multi-dimensional sampling [Gaar72] [Dubo85] is discussed. Subsequent discussions of subsampling and interpolation can be based on this general theory.

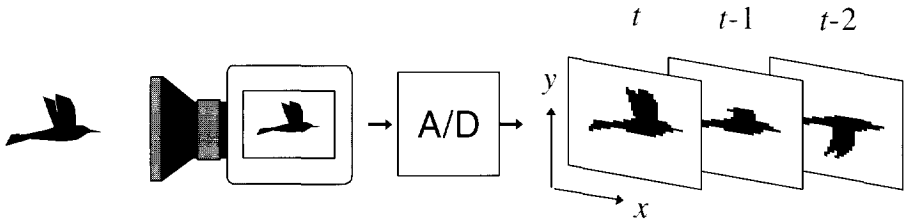


Figure 2.1: The sampling of an image sequence.

2.1.1 Sampling lattices

Sampling an image sequence means that the value of the analog intensity function is evaluated on a discrete set of points in $\mathbb{R}^D$; the *lattice*. First a method is given to formally describe the lattice.

Consider the non-singular matrix $\mathbf{S}$ with dimensions $D \times D$. A lattice L can be defined as the set of all linear combinations with integer coefficients n_i of the column vectors $\mathbf{s}_i$ of the matrix $\mathbf{S}$:

$$L = \left\{ n_1\mathbf{s}_1 + n_2\mathbf{s}_2 + \ldots + n_D\mathbf{s}_D \,\middle|\, n_i \in \mathbb{Z} \;, i = 1, \ldots, D \right\} \tag{2.1}$$

Thus a lattice consists of a set of discrete points $\mathbf{z}$ in $\mathbb{R}^D$ given by

$$\mathbf{z} = \mathbf{S} \cdot \mathbf{n} \,, \quad \mathbf{n} \in \mathbb{Z} \tag{2.2}$$

Note that the matrix $\mathbf{S}$ is not unique in describing the lattice L. If the matrix $\mathbf{S}$ is multiplied by an arbitrary matrix $\mathbf{E}$ with $|\det(\mathbf{E})| = 1$, then the matrix $\mathbf{E} \cdot \mathbf{S}$ describes the same lattice. In the one-dimensional case $\mathbf{S}$ degenerates to a scalar which represents the sampling distance. An example of a two-dimensional lattice is given in Figure 2.2(a). A matrix corresponding with this lattice is

$$\mathbf{S} = \begin{pmatrix} 2 & 1 \\ 0 & 1 \end{pmatrix} \tag{2.3}$$

The matrix $\mathbf{S}$ is often chosen such that is has an upper diagonal structure. This choice is possible for any given lattice. In digital image sequence processing, lattices are used to define the sampling points of the analog three-dimensional image intensity function $i_c(\mathbf{x},t)$ ($\mathbf{x} = (x,y)$):

$$i(\mathbf{x},t) = \begin{cases} i_c(\mathbf{x},t) & (\mathbf{x},t) \in L \\ 0 & \text{otherwise} \end{cases} \tag{2.4}$$

where $i(\mathbf{x},t)$ is the discrete intensity function describing the sampled image sequence. The *sampling density* $d(L)$ of a lattice is defined as

$$d(L) = \frac{1}{|\det(\mathbf{S})|} \tag{2.5}$$

The sampling density is unique for a particular lattice and thus independent of the specific basis vectors. It gives a measure for the spacing of the samples in the D-dimensional space. Increasing the value of the coefficients s_{ij} of the matrix $\mathbf{S}$ causes the samples to be more widely spaced. This is also reflected in the sampling density because the determinant of the matrix also increases. Hence decreasing the sampling density implies a decrease in the number of samples used to represent an image sequence.

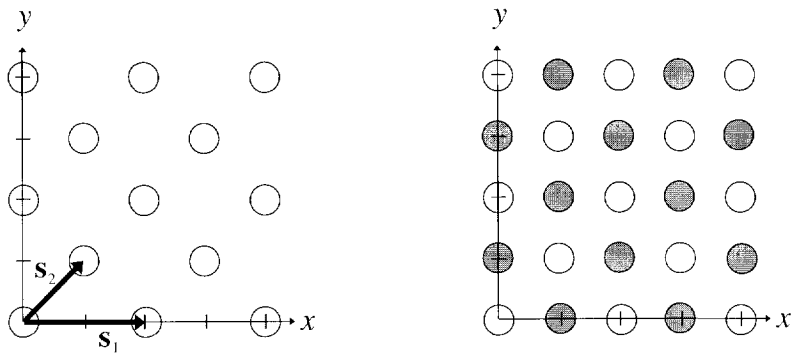


Figure 2.2: (a) Example of a two-dimensional lattice with basis vectors $\mathbf{s}_1$ and $\mathbf{s}_2$. (b) The cosets of the lattice. Samples with the same color belong to the same coset

Let L_O and L_S be lattices in $\mathbb{R}^D$. L_S is a *sublattice* of L_O if every element of L_S is an element of L_O. The set

$$\{\mathbf{c}+\mathbf{z} | \mathbf{z} \in L_S\}, \quad \mathbf{c} \in L_O \tag{2.6}$$

is called a *coset* of L_S in L_O. In other words, a coset is a shifted version of a sublattice L_S. Two cosets are either identical or disjoint. The number of distinct cosets is always an integer and is given by the ratio of the sampling densities of L_S and L_O. Figure 2.2(b) illustrates the concept of cosets. In this case L_O is the lattice defined by the identity matrix, and L_S is the lattice shown in Figure 2.2(a). In Figure 2.2(b) L_S is indicated by the light samples, which is according to the definition also a coset of itself. The other coset is, for instance, obtained by shifting L_S over the vector $(1,0)^T$, corresponding with the dark samples in Figure 2.2(b). Note that the union of all cosets completely covers L_O.

The D-dimensional Fourier transform $I(\mathbf{f})$ of a signal $i(\mathbf{z})$ defined on a lattice L is given by [Dubo85] [Mers83]:

$$I(\mathbf{f}) = \sum_{\mathbf{z} \in L} i(\mathbf{z}) \exp(-2\pi j \mathbf{f}^T \mathbf{z}) \qquad \mathbf{f} \in \mathbb{R}^D \tag{2.7}$$

where $\mathbf{f}$ is the D-dimensional frequency vector. Substituting $\mathbf{z} = \mathbf{S} \cdot \mathbf{n}$ ($\mathbf{n} \in \mathbf{Z}^D$) for the lattice point gives:

$$I(\mathbf{f}) = \sum_{\mathbf{n} \in \mathbb{Z}^D} i(\mathbf{Sn}) \exp(-2\pi j \mathbf{f}^T \mathbf{Sn}) \tag{2.8}$$

From the theory of sampling it is known that the Fourier transform of a sampled function has an infinite number of replicas in the spectral domain. The center points of the spectral replicas are located in the Fourier space on the points where the argument of the exponential function in Equation (2.8) is an integer multiple of $2\pi j$. From Equation (2.8) we conclude that these points $\mathbf{f}$ are located on a lattice defined by the matrix $\mathbf{R}$, which satisfies

$$\mathbf{R}^T \mathbf{S} = \mathbf{I} \Leftrightarrow \mathbf{R} = (\mathbf{S}^T)^{-1} \tag{2.9}$$

This lattice is called the *reciprocal lattice* L^R of L. The reciprocal lattice of the example in Figure 2.2(a) is defined by the matrix

$$\mathbf{R} = \begin{pmatrix} ½ & 0 \\ -½ & 1 \end{pmatrix} \tag{2.10}$$

and is shown in Figure 2.3. A result of Equation (2.9) is that the sampling density of L^R is the reciprocal of the sampling density of L. Decreasing the spatial sampling density results in a more densely packed spectrum.

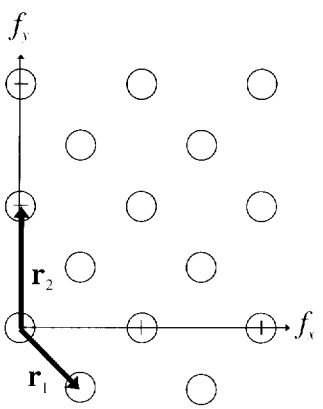


Figure 2.3: Reciprocal lattice with basis vectors $\mathbf{r}_1$ and $\mathbf{r}_2$.

The above theory is not sufficient to mathematically describe all sampling lattices; a lattice consisting of a combination of two lattice cosets cannot always be described. For more details the reader is referred to the tutorial paper by Dubois [Dubo85].

2.1.2 Examples of sampling lattices

In this section some illustrative examples are given of three-dimensional sampling lattices [Tong81][Dubo85][Reut85]. In Figure 2.4(a) an orthogonal lattice is shown. The basis vectors form an orthogonal basis, so this lattice can be represented by the identity matrix. The sampling density of this lattice is equal to 1. If this sampling structure is used to represent an image sequence, the sequence is called *progressively* scanned. An image at each instance of the time is called a *frame*. The spectral replicas are located on an orthogonal lattice in the frequency domain.

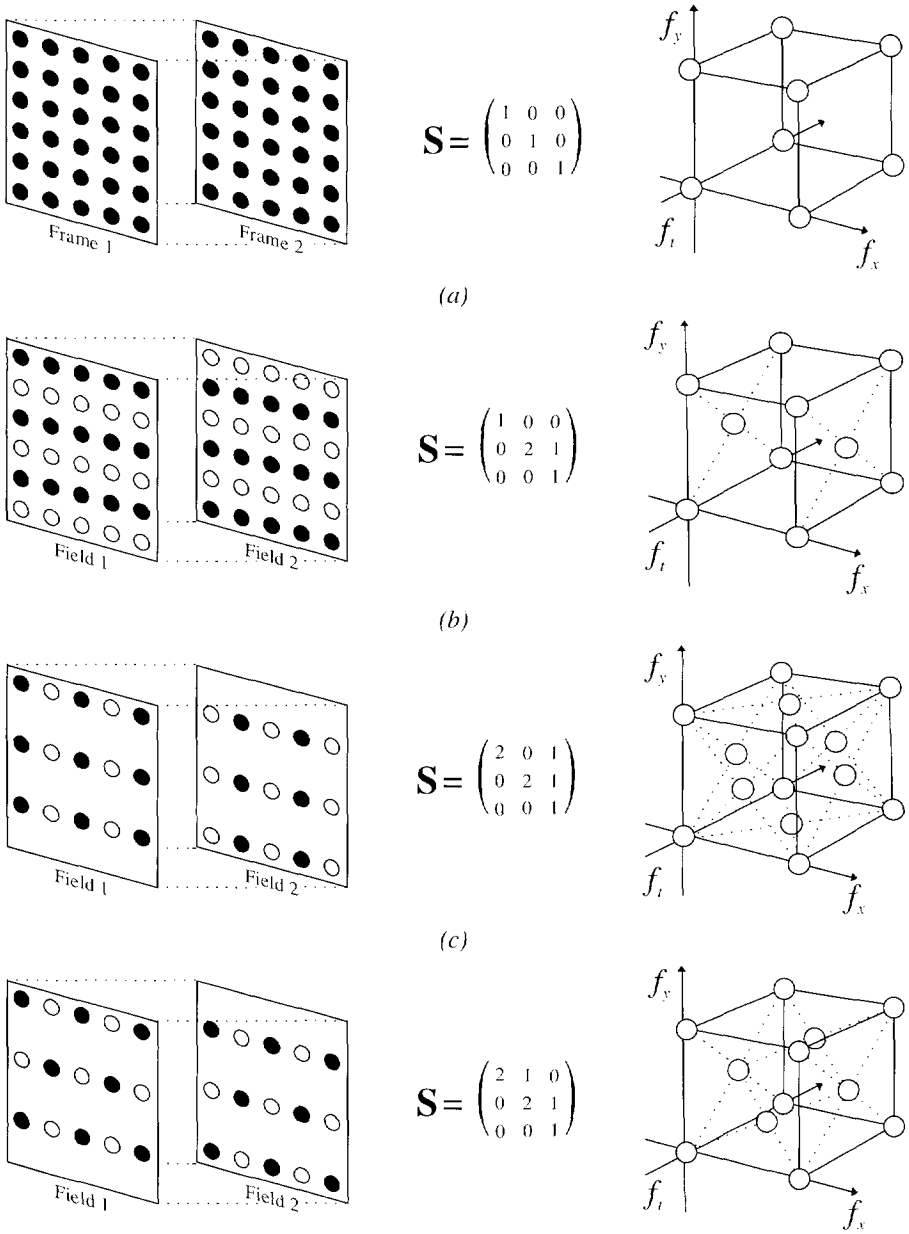


Figure 2.4: Examples of subsampling lattices: from left to right the sampling structure, the lattice matrix and the positions of the replicas in the frequency domain are shown for a: (a) orthogonal, (b) interlace, (c) field quincunx and (d) line quincunx lattice.

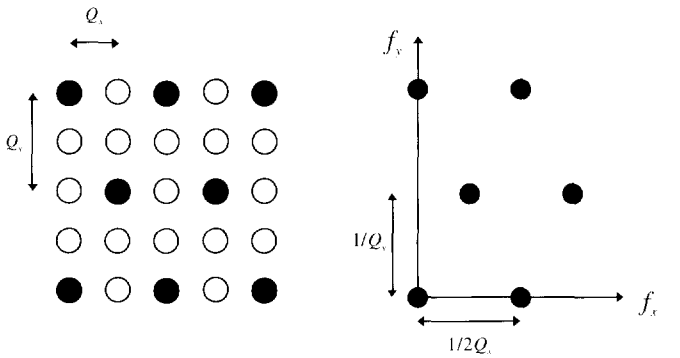


Figure 2.5: Quincunx sampling pattern and reciprocal lattice.

The scanning procedure which is commonly used in television is the *interlace* scan. The corresponding lattice is depicted in Figure 2.4(b). Half the number of rows of the frame are discarded, hereby reducing the vertical resolution. Effectively, this means that extra spectral replicas are introduced between the existing spectral replicas in the vertical direction. The pattern in which the rows are discarded is shifted from frame to frame. Therefore the extra replicas have an offset in the f_t-direction. After subsampling, the individual images are called *fields* instead of frames. The sampling density of this lattice is equal to ½. In practice, interlace sampling is used to increase the temporal resolution. While maintaining the same total number of samples as orthogonal sampling, the field rate is doubled with respect to the frame rate in the case of orthogonal sampling, hereby doubling the temporal resolution. As a result the last column vector of the lattice matrix **S** is changed to $(0,1,½)^T$, yielding a sampling density of 1 again.

Starting from an interlace input image sequence, two other subsampling structures are possible to further reduce the sampling density to ¼. These lattices are both of the class of *quincunx* sampling lattices, which can be described in two dimensions by the matrix

$$\mathbf{S} = \begin{pmatrix} 2Q_x & Q_x \\ 0 & Q_y \end{pmatrix} \tag{2.11}$$

The generic structure of a quincunx lattice is illustrated in Figure 2.5; also the corresponding reciprocal lattice is shown. The transmitted pixels are located like the number five on a dice. Another quincunx lattice has already been given in Figure 2.2(a).

The *field quincunx* sampling structure is shown in Figure 2.4(c). Half the pixels of a line are alternately discarded. The pattern in which the pixels are discarded is shifted from field to field. The consequence is that the resolution in the vertical and horizontal direction is the same but the resolution in the diagonal direction is reduced. The quincunx structure can be recognized by combining the two fields in the temporal direction. The *line quincunx* sampling structure is illustrated in Figure 2.4(d). Again half the pixels of a line are discarded but now

the pattern is shifted from line to line within each field. The vertical and horizontal resolution is again the same at the expense of the diagonal resolution. The difference between the field quincunx lattice and the line quincunx lattice is that their temporal resolution is different.

2.2 Fixed lattice subsampling

By fixed lattice subsampling we mean the conversion from an image sequence represented on an initial lattice L to a sequence represented on a new lattice L_{ss}, where $d(L_{ss}) < d(L)$. The data reduction factor is equal to $d(L)/d(L_{ss})$. A simple subsampling scheme is shown in Figure 2.6. The original image sequence is first prefiltered and next subsampled prior to transmission. The prefilter is enclosed by a dashed box because this filter is not always necessary. The remaining pixels after subsampling are either transmitted or fed into another coding scheme for a further data compression. This depends on whether the subsampling scheme can be seen as a complete coding scheme or as a part of a larger scheme. At the decoder the bandlimited image sequence is represented on the original lattice L by using a linear spatial interpolation filter. The various operations are now discussed in more detail.

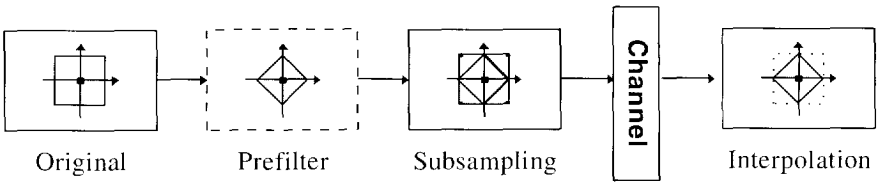


Figure 2.6: Basic subsampling scheme. The figures in the boxes illustrate the operations in the spectral domain

2.2.1 Prefiltering

The purpose of the prefilter is to confine the image spectrum to a region in the three-dimensional space. This region is the so-called *unity cell*. A unity cell U_L of a lattice L is a region in $\mathbb{R}^D$ which satisfies the following two criteria:

- The copies of U_L centered at each lattice point completely cover $\mathbb{R}^D$:

$$\bigcup_{\mathbf{x}\in L}(U_L+\mathbf{x})=\mathbb{R}^D \tag{2.12a}$$

- The copies of U_L centered at each lattice point do not overlap each other:

$$(U_L+\mathbf{x})\bigcap(U_L+\mathbf{y})=\varnothing,\quad \mathbf{x},\mathbf{y}\in L,\mathbf{x}\neq\mathbf{y} \tag{2.12b}$$

In other words, $\mathbb{R}^D$ is partitioned into unity cells. In the literature a unity cell is also called a Voronoi region or a Brillouin zone [Dubo85]. A unity cell is not unique. In Figure 2.7 two different examples of unity cells are given for the reciprocal lattice shown in Figure 2.3.

The concept of the unity cell is relevant to both the sampling and subsampling of image sequences. To avoid conflicts between the different spectral replicas after the sampling of an

analog image sequence, the spectrum of an image sequence should be confined to a unity cell of the reciprocal lattice L^R. This prevents the different spectral replicas from overlapping each other. The error caused by overlapping spectral replicas is called *aliasing*.

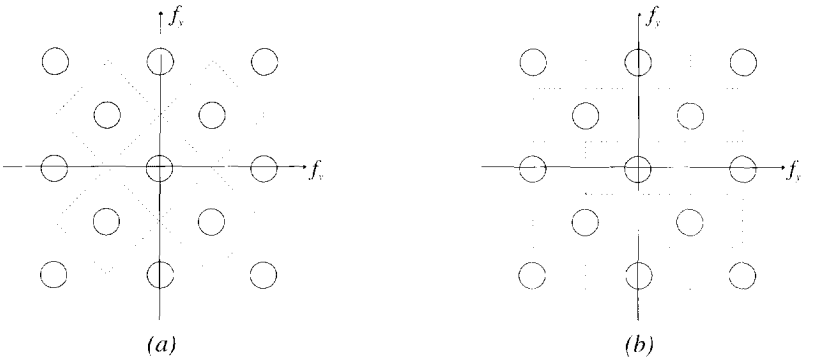


Figure 2.7: Two examples of unity cells for the reciprocal lattice shown in Figure 2.3. The unity cells are marked by the dashed lines.

To prevent aliasing in the case of subsampling, the spectrum should be confined to a unity cell of the new reciprocal lattice $L_{ss}{}^R$. Essentially, the concept of the unity cell extends the Nyquist sampling theorem ([Nyqu28] [Jerr77]) to multiple dimensions. If the spectrum of the original image sequence is *not* already confined to a unity cell, a prefilter should be used. If the original spectrum already has the shape of the unity cell, prefiltering is not necessary. The shape of the unity cell determines the resolution of the subsampled image sequence in the various directions. In Figure 2.7(a) the diagonal resolution is reduced in favor of the horizontal and vertical resolution. In Figure 2.7(b) the horizontal frequency components are over-emphasized compared to the vertical frequency components.

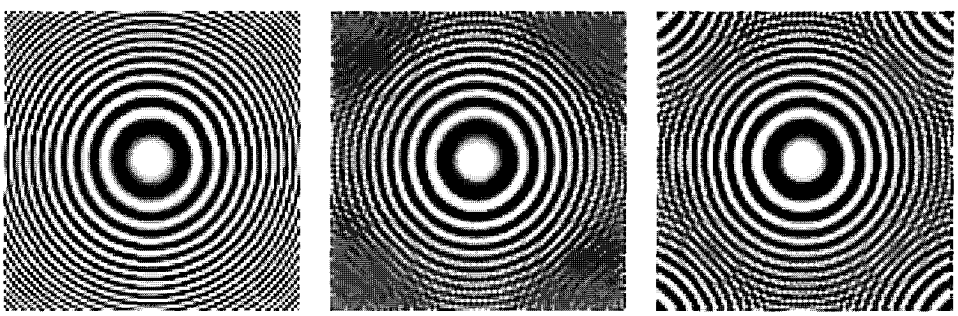


Figure 2.8: (a) Zone plate. (b,c) Quincunx subsampled and interpolated zone plates: (b) with prefiltering and (b) without prefiltering.

In Figure 2.8 a zone plate is used to show the consequences of not using a prefilter. The zone plate is used because as we can see in Figure 2.8(a) it contains a broad range of spatial frequencies. The spatial domain now gives a good indication of the consequences of

subsampling in the frequency domain. The quincunx subsampling lattice depicted in Figure 2.2(a) is used. In Figure 2.8(b) a diamond-shaped prefilter as defined by the unity cell shown in Figure 2.7(a) is applied prior to subsampling. We see that the diagonal spatial frequency components are attenuated by the prefilter. In Figure 2.8(c) no prefiltering is used, causing clearly visible aliasing in the diagonal spatial frequency components. We see that the errors caused by aliasing are far more severe than the attenuation of the frequency components caused by the prefilter.

2.2.2 Subsampling

If the new sampling lattice is a sublattice of the original lattice, then the actual subsampling can be implemented by simply discarding pixels not present in the new lattice. If this is not the case, an intermediate sampling structure L_i must be used, which has the following relation with both the original and the new lattice:

$$L \subset L_i \wedge L_{ss} \subset L_i \tag{2.13}$$

with

$$d(L_i) \geq d(L) \geq d(L_{ss}) \tag{2.14}$$

The lattice L is first converted to the lattice L_i. From this intermediate lattice the subsampling lattice L_{ss} can be deduced [Reut86].

2.2.3 Interpolation

At the receiver the original image sequence has to be reconstructed at the original sampling lattice, which corresponds to removing the extra replicas introduced by the subsampling process. This is done with an interpolation filter. The most important design criterion of the interpolation filter is that it should not cancel the frequency components within the unity cell. Otherwise the interpolation causes an additional loss of resolution. Therefore, for the unity cell in Figure 2.7(a) a diamond-shaped filter should also be used for the interpolation.

2.2.4 Filter requirements

In fixed lattice subsampling normally two filtering operations are necessary. The prefilter is designed to suppress parts of the spectrum that would otherwise cause aliasing. The interpolation filter has to be designed in such a way that the extra replicas introduced by the subsampling process are canceled. These considerations result in a specific shape for each of the filters. In this design process some frequency components need extra attention. These are the *critical frequencies* [Pirs83] [Sioh91], which are the center positions of the replicas included in ${L_{ss}}^R$ and not in L^R (${L_{ss}}^R \cap L^R$). These critical frequencies are replicas of the DC-component of the baseband spectrum. In Figure 2.9(a) the critical frequencies are shown for a quincunx lattice with $Q_x = 1$ and $Q_y = 2$ which is interpolated to an orthogonal lattice.

In video applications, it is important that the DC gain of the overall system is equal to one, otherwise annoying oscillation artifacts are introduced in areas with a constant luminance [Pirs83]. The human visual system is especially sensitive to errors in this part of the

spectrum. To obtain a unity gain, the filter should have a zero frequency response for the critical frequencies. This should be the case for both the prefilter and the interpolation filter. This design requirement can be translated to the spatial domain by demanding that the sum of all the polyphase components [Croc75] should be equal. A filter design procedure that takes this constraint into consideration is given in [Sioh91]. In Table 2.1 a filter is shown for the interpolation of the quincunx lattice from Figure 2.9(a) to an orthogonal lattice. The modulation transfer function of the filter is shown in Figure 2.9(b). The filter gain is zero for (f_x,f_y) equal to (0,½) and (½,¼). These points correspond with the critical frequencies.

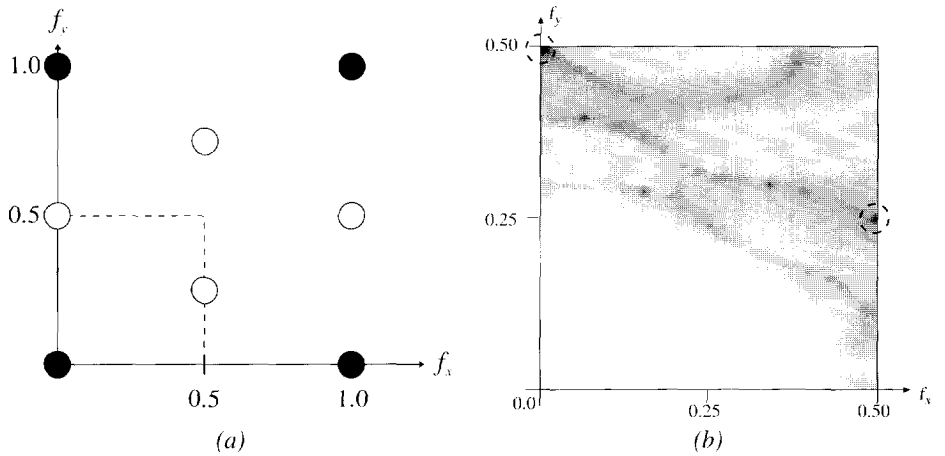


Figure 2.9: (a) Position of replica. The open dots are the critical frequencies and the dashed box indicates the baseband spectrum, (b) Modulation transfer function of the interpolation filter (the dashed circles indicate the zeroes of the filter).

Table 2.1: Filter coefficients (multiplied by 64) of a two-dimensional interpolation filter (taken from [Sioh91]).

x \ y	0	±1	±2
0	52	25	0
±1	41	19	-2
±2	19	5	-4
±3	3	-3	-4
±4	-2	-2	-1

In an image sequence the unity cell is a three-dimensional region. This would imply that the prefiltering and interpolation is done in three dimensions. However filtering in the temporal direction is avoided in practical situations. Temporal filtering is also not always necessary because of the high degree of tolerance of the human visual system to aliasing in the temporal direction [Giro88]. This reduces the filtering process to a spatial operation which only involves single images. A further simplification can be made by using two one-dimensional filters operating in different directions instead of a non-separable two-dimensional filter. In [Giro89] it is shown that if separable one-dimensional filters are used for the conversions

between an orthogonal lattice and a quincunx lattice then an intermediate sampling lattice is necessary.

One particular class of one-dimensional filters which is interesting in a subsampling application is the class of maximally-flat filters [Tong81] [Guma78]. One of the properties of these filters is that for a filter $h(k)$ with $2N+1$ filter taps:

$$h(0) = \frac{1}{2} \wedge h(k) = 0, \quad k = \pm 2, \pm 4, \ldots, \pm N \tag{2.15}$$

In the frequency domain this requirement can be interpreted as skew symmetry around the cut-off frequency. If this filter is used on an alternating input source of zeroes and original pixel values, the original pixel values are not changed by the filtering process. This property is especially useful if the filter is used for interpolation.

2.3 Subsampling and rate-distortion theory

As subsampling is a data reduction method, its properties can be described by rate-distortion theory. Although rate-distortion theory has a limited use in practice, it can give some insight into the potentials of subsampling. The rate-distortion function $R(D)$ [Berg71] provides a lower bound for the bit rate R necessary to transmit a source with an average distortion D. We derive the rate-distortion function for subsampling as a data reduction method to show the conditions under which subsampling is appropriate and to study the source of the different errors. The discussion in this section is done for the one-dimensional case.

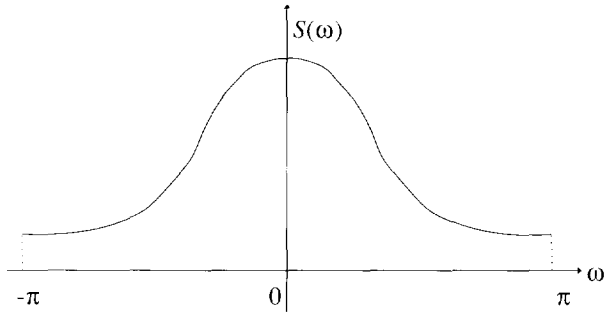


Figure 2.10: Power density function.

We assume a PCM encoded spatially-discrete signal with a power spectral density function $S(\omega)$. An example of such a function is shown in Figure 2.10. According to Parseval's theorem the variance σ^2 of this source is

$$\sigma^2 = \frac{1}{2\pi} \int_{-\pi}^{+\pi} S(\omega) d\omega \tag{2.16}$$

In a fixed lattice subsampling scheme, the spectrum of the input source is low-pass filtered and the signal is then subsampled according to the Nyquist theorem. If the bit rate required to transmit the original source using PCM coding is B_o, then the new rate B after subsampling is

$$B = \frac{W_{ss}}{\pi} B_0 \quad \text{(bits/pixel)} \tag{2.17}$$

where W_{ss} is the radial bandwidth after prefiltering. Rewriting this relation gives

$$W_{ss} = \frac{B}{B_0} \cdot \pi \tag{2.18}$$

We now define B/B_0 as the *relative* bit rate R, so we can rewrite the previous equation to

$$W_{ss} = R \cdot \pi \tag{2.19}$$

Hence the relative bit rate is reduced proportionally to the bandwidth reduction.

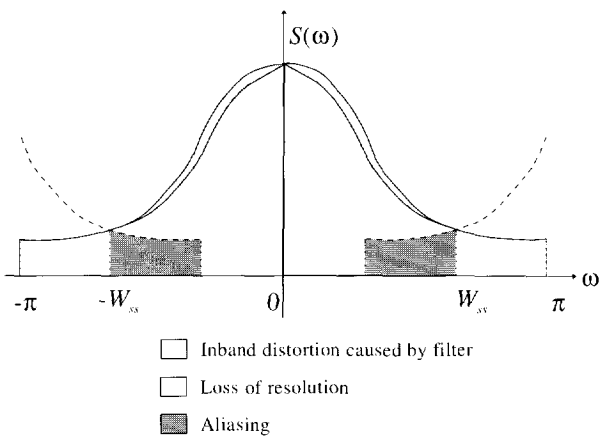


Figure 2.11: Power density function after subsampling. The shaded areas are the components which contribute to the distortion.

We model the overall response of the system with the low-pass filter $H(\omega)$ with a cut-off frequency at W_{ss}. This response includes both the prefilter and the interpolation filter. The resulting mean square error distortion after interpolation is given by

$$D = \underbrace{\frac{1}{\pi}\int_0^{W_{ss}} (1-|H(\omega)|)^2 S(\omega)d\omega}_{1} + \underbrace{\frac{1}{\pi}\int_{W_{ss}}^{\pi} (1-|H(\omega)|)^2 S(\omega)d\omega}_{2} + \underbrace{\frac{1}{\pi}\int_{W_{ss}}^{\pi} |H(\omega)|^2 S(\omega)d\omega}_{3} \tag{2.20}$$

where we have taken into account the symmetry of $S(\omega)$ and $H(\omega)$ around the origin. The three areas which contribute to the distortion are the shaded regions in Figure 2.11. The first two parts of the equation represent the distortion introduced by the low-pass filter and the third part is the aliasing error caused by an imperfect low-pass filter. The distortion can be classified in the following way (the numbers correspond with the relevant part of Equation (2.20)):

1. The interval from 0 to W_{ss} contributes to the *inband* distortion ([Berg71]). This is the distortion of the part of the spectrum which is coded.
2. The interval from W_{ss} to π contributes to the *outband* distortion which is the distortion introduced by the part of the spectrum which is not coded.

3. The aliasing error folds into the part of the spectrum which is coded and therefore contributes to the inband distortion.

If an ideal low-pass filter is used with transfer function

$$H(\omega) = \begin{cases} 1, & |\omega| < W_{ss} \\ 0, & W_{ss} < |\omega| < \pi \end{cases} \tag{2.21}$$

then the inband distortion is equal to zero and Equation (2.20) reduces to

$$D = \frac{1}{\pi} \int_{W_{ss}}^{\pi} S(\omega) d\omega \tag{2.22}$$

leaving only the outband distortion. Substituting Equation (2.19) in (2.22) and using Equation (2.16) gives the rate-distortion function $D(R)$:

$$D(R) = \sigma^2 - \frac{1}{\pi} \int_{0}^{\pi R} S(\omega) d\omega \tag{2.23}$$

An illustration of rate-distortion functions is given in Figure 2.12. The curves shown are derived from an AR(1) process with a correlation coefficient ρ, driven by white noise with unity variance. Hence the power spectral density function is equal to

$$S(\omega) = \frac{1}{\left|1 - \rho e^{-j\omega}\right|^2} \tag{2.24}$$

We see that with increasing correlation between the samples the source can be coded more easily.

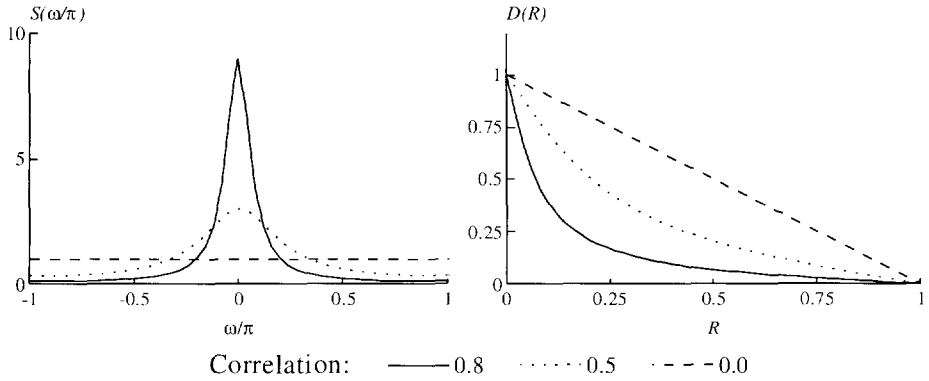


Figure 2.12: Examples of rate-distortion functions for AR(1) processes with different correlation coefficients ρ. *Left the power density functions are shown and right the corresponding rate-distortion functions.*

Let us consider the properties of the rate-distortion function. The first derivative of $D(R)$ is given by:

$$\frac{\partial D}{\partial R} = -S(\pi R) \tag{2.25}$$

The first derivative of the rate-distortion function is *always* monotonically decreasing because $S(\omega)$ is always greater or equal to zero. Thus (2.25) is the trivial result that for each coding scheme the distortion increases if the rate decreases. Another property of rate-distortion curves is convexity. This is a required property because it implies that the parts of the spectrum with the least relevance to the entire signal are discarded first. A function is convex if the second derivative is monotonically decreasing. From Equation (2.25) we obtain:

$$\frac{\partial^2 D}{\partial R^2} = -\pi S'(\pi R) \tag{2.26}$$

where $S'(.)$ is the first derivative of $S(.)$. We see that if the power spectral density function $S(\omega)$ is monotonically decreasing with increasing ω (i.e., $S'(\omega) < 0$) then the rate-distortion function is convex. Thus in subsampling schemes, convexity of the rate-distortion curve is directly coupled to the decreasing character of $S(\omega)$. If the power spectrum density is non-decreasing the rate-distortion may become non-convex and subsampling is no longer optimal.

The above discussion can be extended to two dimensions. After filtering and subsampling $S(\omega_x,\omega_y)$, does not extend over a specific bandwidth but covers the unity cell prescribed by the subsampling lattice. Equations (2.17) and (2.23) now become:

$$B = \frac{\text{Area}(U_{L^R})}{(2\pi)^2} B_o \tag{2.27a}$$

$$D = \sigma^2 - \frac{1}{(2\pi)^2} \iint_{(\omega_x,\omega_y)\in U_{L^R}} S(\omega_x,\omega_y)\, d\omega_x d\omega_y \tag{2.27b}$$

Without any specific knowledge about the two-dimensional subsampling lattice it is not possible to obtain a close form relation for the rate-distortion function.

2.4 Resolution of the human visual system

Fixed lattice subsampling will always result in a reconstructed image sequence with a lower resolution than the original image sequence, unless the spectrum of the original image sequence lies within a unity of the subsampling lattice. Therefore a fixed lattice subsampling scheme has to take advantage of the spatio-temporal resolution of the human visual system (HVS) in order to avoid a *visible* loss of resolution. Hence any perceptual irrelevancy should be removed.

In Figure 2.13(a) the two-dimensional frequency threshold sensitivity of the HVS is shown. We see that the sensitivity is lower for diagonal frequency components than for horizontal and vertical frequency components. Even if the HVS was isotropic, on an orthogonal lattice the resolution in the diagonal direction is $\sqrt{2}$ times larger than the resolution in the horizontal and vertical direction. Therefore a quincunx pattern is better suited than an orthogonal lattice. The spatial frequency support of an image with dimensions 1440×1125 sampled with a

quincunx lattice ($Q_x = 2$, $Q_y = 1$) viewed at a distance of three times the picture height is also shown as the dashed line in Figure 2.13(a). We see that the resolution is still well within the passband of the HVS. Also if a visible loss of degradation is permitted, quincunx subsampling is the best possible choice because it represents the best compromise between the resolution and the passband of the HVS.

In Figure 2.13(b) the threshold sensitivity of the HVS is shown for different velocities. The relative sensitivity decreases with increasing velocity [Kell79] [Nino87]. This would imply that for image sequences containing moving objects, the temporal resolution can be reduced. However the characteristics in Figure 2.13(b) were measured for eyes which were fixed on one specific position. If a scene contains moving objects, the human eye can *track* these objects at high speed and will perceive them to be stationary [Giro88]. Thus the human eye has the capability of compensating for the motion, and will still notice a loss of resolution.

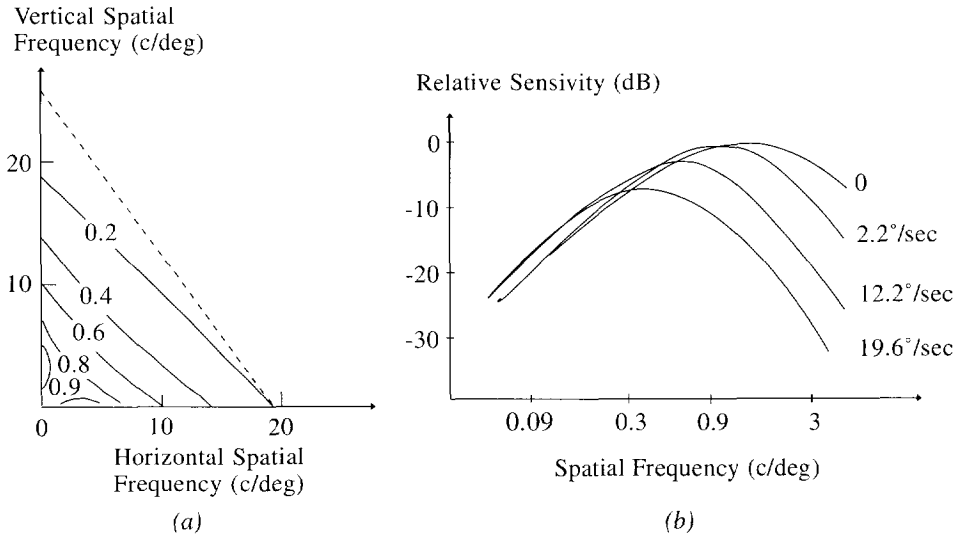


Figure 2.13: (a) Spatial frequency threshold sensitivity of the human visual system. (b) Threshold sensitivity of the human visual system for moving objects (both taken from [Nino87]).

2.5 Applications

In this section some applications are discussed that use fixed lattice subsampling. Interlacing has already been discussed as a way to increase the temporal resolution at the expense of the spatial resolution. Interpolation is not always necessary because the display devices can be designed to be used with interlaced lattices. However, displaying an image sequence on an interlaced display device gives annoying artifacts such as line crawl and line flicker (e.g. [Hent89]). To avoid these artifacts, an interlaced sequence can be interpolated back to an orthogonal lattice. This operation is referred to as de-interlacing [Giro85][Wang90].

In Section 2.4 it was observed that a part of the diagonal resolution can be sacrificed without introducing any visible degradation. This property can be used as a pre-processing operation prior to the actual coding. For instance, in the MUSE system [Nino87] the highest transmitted resolution of the luminance information is a quincunx lattice with $Q_x = 2$ and $Q_y = 1$. In the HD-MAC system [Vree89] the highest resolution of the luminance information is a quincunx lattice with $Q_x = 1$ and $Q_y = 1$.

Another application of fixed lattice subsampling is the subsampling of the color information of an image sequence. The representation of the image sequence in the YUV space concentrates most of the energy in the Y-component of the image sequence. Therefore the U and V components have a low variance and can be subsampled with a fixed lattice without this leading to severe degradation. One factor which also works to the advantage of this scheme is the low sensitivity of the HVS to degradation in the color information.

There are different sampling lattices which are used to subsample the color information. The CCIR recommendation 601 [CCIR82] prescribes a horizontal decimation of both the color components. This standard is referred to as the 4:2:2 standard. The MPEG standard [MPEG92] [LeGa92] goes even further and proposes both a horizontal and vertical decimation of the color information prior to coding. This is the so-called 4:2:0 standard. In the MUSE system the color information is subsampled with a quincunx sampling lattice with $Q_x = 3$ and $Q_y = 2$ for each field, yielding an overall reduction of 12. In the HD-MAC system two different sampling lattices are used for the color information. One of those lattices is a quincunx lattice with $Q_x = 2$ and $Q_y = 4$ used for each field with an overall reduction of 16.

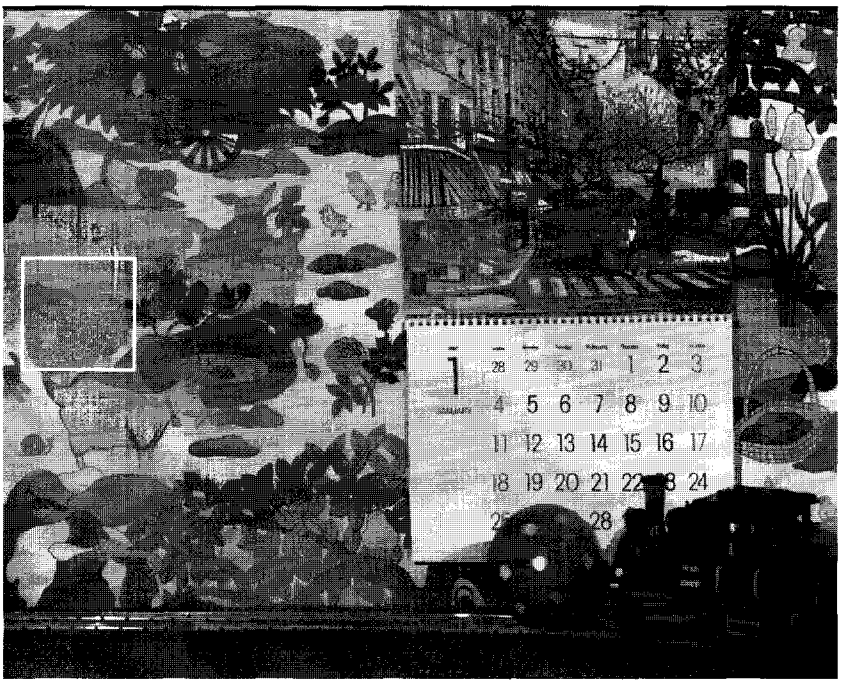
2.6 Experiment results

The purpose of the experiments is to illustrate different aspects of coding schemes to gain some insight into their peculiarities and performance. Therefore throughout the thesis a consistent set of experiments is reported. The same test sequences are used in many different subsampling schemes. Thus the results presented in each chapter can be compared with each other.

In this section the experiment results using fixed lattice subsampling are described. First, some aspects of the experiments in general are discussed, namely the test sequences and the way in which the subsampling methods are here evaluated. Next, experiments are done on both the luminance and the chrominance components. These components are treated separately because we show that the application of fixed lattice subsampling is useful for the chrominance components, but has limited possibilities for the luminance component.

2.6.1 Description of the test images

For experiments that do not involve motion, the commonly used *LENA* image is used. This image is chosen because of its status as a non-official standard image, which facilitates comparison with other coding methods.



(a)

(b)

Figure 2.14: (a) First frame of MOBILE sequence. (b) Detail of the first frame of the MOBILE sequence. The position of the detail is indicated in (a).

In Figure 2.14(a) the first image of the *MOBILE* sequence is shown. The sequence is sampled according to the CCIR 601 standard (720 pixels × 576 lines, 4:2:2 color sampling). This sequence is used in experiments which require either single images or image sequences. The sequence is chosen because of the following features:

- The sequence is progressively scanned. The advantage of this lattice is that different subsampling lattices can easily be derived from this lattice.
- The sequence contains a considerable amount of high spatial frequencies. In most coding schemes the high frequency content is affected. To better illustrate the different coding artifacts a detail of the image is used as indicated by the rectangle in Figure 2.14(a). The dimensions of this detail is 96×96 pixels and is shown in Figure 2.14(b). The detail encompasses the entire spatial frequency range with emphasis on the high spatial frequency components.
- A substantial part of the image is moving translationaly because of a camera pan. This is useful if we consider schemes which use motion information. A completely stationary scene or a scene containing only complex motion would not show the advantages of motion adaptive schemes.
- To balance this advantage there are also objects moving in a non-translational manner (the ball), areas which are occluded because of the motion (areas around the train) and objects moving in a translational manner with a non-integer speed (the calendar).
- There is some noise in the sequence (approx. 28 dB).
- The image contains a broad range of different colors.

2.6.2 Evaluation of subsampling schemes

The reduction factor is chosen in such a way that coding artifacts are just visible. If an image is coded with a high reduction factor, the relation with the original image is small and a comparison with the original is harder.

In order to evaluate subsampling schemes, different approaches are chosen. An objective measurement can be obtained by using the mean square error (MSE) defined as

$$\mathrm{MSE} = \frac{1}{\#\,pixels} \sum_{\forall \mathbf{x}} (i_{int}(\mathbf{x},t) - i(\mathbf{x},t))^2 \tag{2.28}$$

where $i(\mathbf{x},t)$ is the original image intensity at the location $(\mathbf{x},t)$ and $i_{int}(\mathbf{x},t)$ the reconstructed image intensity. To normalize the MSE, the signal to noise ratio (SNR) given by

$$\mathrm{SNR} = 10 \cdot {}^{10}\log\left(\frac{\sigma^2}{\mathrm{MSE}}\right) \tag{2.29}$$

can be used, where σ^2 is the variance of the original image.

It is, however, generally known that the SNR and MSE do not properly reflect the actual image quality as it is perceived. Some of the aspects which are neglected by these objective measures are:

- That the loss of fine detail and high spatial frequency components does not result in a high error value because the energy contribution of these signal components is relatively low.

- If the noise present in the original image is not properly reconstructed, the noise variance will also contribute to the MSE. This is an undesirable effect because the noise should not be considered as an essential part of the scene.

Thus besides the MSE or SNR we need some kind of other mechanism to evaluate the different coding schemes. The approach taken in this thesis is to visualize the stretched difference between the coded image and the original image. The advantages of this are:

- The stretching amplifies the artifacts and makes subtle errors more noticeable.
- The noise can be identified because for noise there is no correlation between the error value and the image contents.

The stretching factor used throughout the thesis is equal to 4. For display purposes, an offset of 128 is added to the difference.

2.6.3 Luminance subsampling

In this section only spatial sampling lattices are considered. As we will see in the remainder of the thesis, three-dimensional sampling lattices require a more complex interpolation. Three different subsampling schemes were used, all with a data reduction of two. These are horizontal subsampling, vertical subsampling and quincunx subsampling. For the prefiltering a 21-taps filter was used [Giro89] (Table 2.2) and for the interpolation a 7-taps maximally flat filter [Guma78] (Table 2.3). The filter characteristics are shown in Figure 2.15. These filters were chosen because the long prefilter gives a good suppression of the replica, whereas the relative short interpolation filter prevents the occurrence of annoying ringing artifacts. For horizontal and vertical subsampling the response at the critical frequencies is zero because of the zeroes of the filters at $\omega = \pi$. For quincunx subsampling the filters are used twice in both the diagonal directions, canceling the critical frequencies at $(\omega_x,\omega_y) = (\pm\pi,\pm\pi)$.

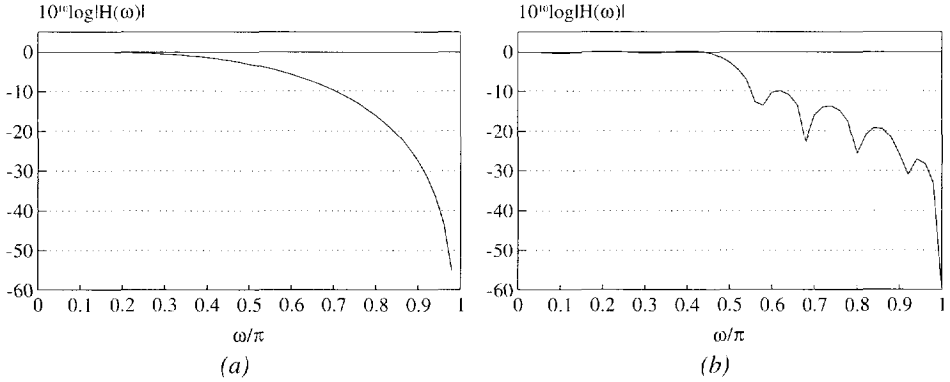


Figure 2.15: Filter transfer function of: (a) the prefilter and (b) the interpolation filter.

Table 2.2: Prefilter coefficients (× 512).

0	*±1*	*±2*	*±3*	*±4*	*±5*	*±6*	*±7*	*±8*	*±9*	*±10*
254	160	-3	-55	1	32	-5	-22	7	13	1

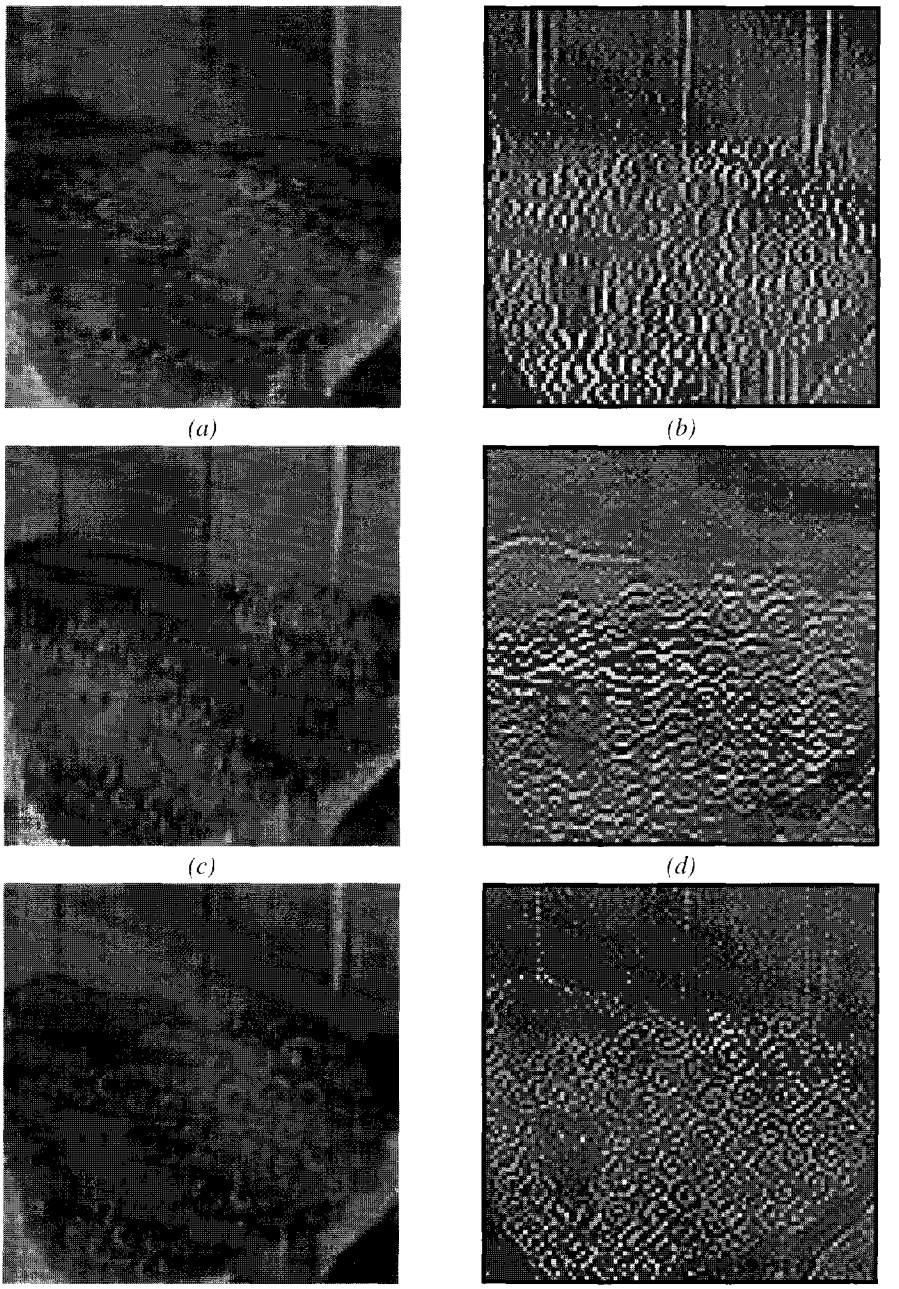


Figure 2.16: Image detail and stretched difference after subsampling and interpolation: (a,b) horizontal subsampling, (c,d) vertical subsampling and (e,f) quincunx subsampling.

Table 2.3: Interpolation filter coefficients (× 32).

0	±1	±2	±3
16	9	0	-1

In Figure 2.16 the results of the experiments are shown for the detail of Figure 2.14(b). The stretched difference images are also shown. We see that each subsampling scheme result in a loss of sharpness of specific edges of the image depending on the direction of the edge. Visually the quincunx lattice gives the best performance because of the relative insensitivity of the HVS to loss in diagonal detail. However, even at this low reduction factor the loss of resolution is visible. The MSE and SNR are shown in Table 2.4. These values can be used as a reference for future experiments.

Table 2.4: Average SNR and MSE for fixed lattice subsampling using the MOBILE sequence.

Subsampling method	*SNR (dB)*	*MSE*
Horizontal	18.1	26.5
Vertical	17.9	27.4
Quincunx	19.5	18.8

2.6.4 Chrominance subsampling

Color subsampling is one of the most widespread applications of fixed lattice subsampling. To illustrate the relevance of color subsampling, Table 2.5 lists the variance of the three color components for the *MOBILE* sequence. We see that the relative energy contribution of the U and V components is small compared to the contribution of the Y component.

Table 2.5: Average variance of the image components for MOBILE sequence.

Component	*Variance*	*Contribution*
Y	1709.79	83.7 %
U	129.40	6.3 %
V	204.32	10 %

In Figure 2.17, the different subsampling schemes used in the experiments for color subsampling are shown in the spatial and frequency domain. The original image has a 4:2:2 color sampling. First the image is subsampled in the vertical direction, resulting in a 4:2:0 sampling lattice (scheme 1). Next the image is repeatedly subsampled with a quincunx lattice (schemes 2 to 4). Note that subsampling a quincunx lattice again with a quincunx lattice yields an orthogonal lattice.

These different color subsampling schemes are used on the *MOBILE* sequence. The resulting average MSE is shown in Figure 2.18. Because of the relative insensitivity of the HVS to distortions in the color information the first artifacts appear using subsampling scheme 3. The artifacts are clearly visible if subsampling scheme 4 is used (i.e., a data reduction of 16). The artifacts are a *smearing* of the colors over the edges which surrounds an object with a unique

color and a vanishing of fine color details. The conclusion which can be drawn from these experiments is that simple fixed subsampling schemes can be used for the chrominance component. These simple schemes give a high compression at the expense of a small loss in image quality.

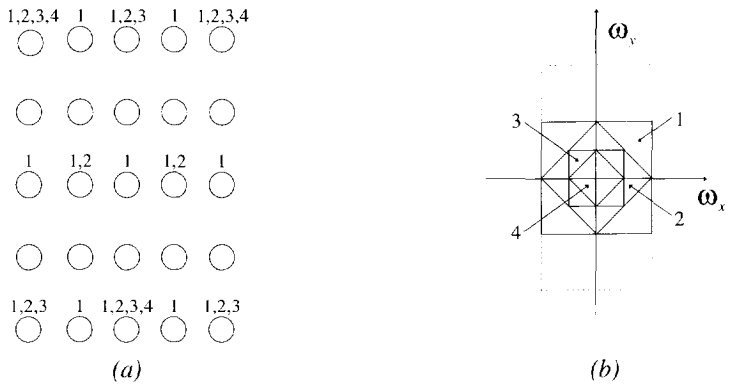


Figure 2.17: Color subsampling schemes: (a) Spatial domain (The numbers indicate which pixels are kept in each scheme), (b) Spectral domain (The numbers indicate the subsampling scheme, and the dashed box is the support of the original spectrum).

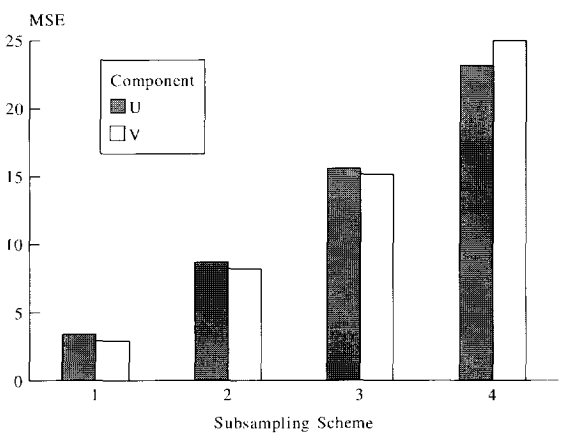


Figure 2.18: Average MSE for color subsampling using the MOBILE sequence.

Chapter 3

ADAPTIVE INTERPOLATION TECHNIQUES

A simple subsampling scheme was introduced in the previous chapter. It used a fixed sampling lattice in combination with a fixed linear interpolation filter. Before we introduce in the next chapter more sophisticated subsampling schemes using spatially adaptive subsampling, we first look at what improvements can be achieved when using fixed lattice subsampling. Attention is focused on *adaptive interpolation* of the subsampled image sequence. One important aspect of adaptive interpolation techniques is whether they are able to recover some of the resolution lost because of the subsampling. If a prefilter is used prior to subsampling, the high frequency information is permanently lost, and adaptive interpolation acts as an image enhancement technique [Wang89]. Therefore the use of a prefilter should be avoided as much as possible. The resolution is also not recovered if there is any aliasing. Hence aliasing should also be avoided as much as possible. These two aspects, namely the use of a prefilter and aliasing, are often in conflict with each other and a choice should be made. In this chapter the choice is made to use *no* prefilter.

This chapter discusses two alternatives for the previously described linear interpolation. The first alternative is motion compensated linear interpolation [Giro85] [Wang89] [Erns91] [Reut89]. Motion compensated interpolation relies on the temporal correlation within an image sequence. Even in regions where the spatial correlation is not large, the temporal correlation can be large. If an image sequence is subsampled, the missing samples can be interpolated by using the corresponding pixels along the motion trajectory in the previous or the next image. This requires motion estimation. The use of motion information also enables the use of three-dimensional sampling lattices as it provides a means to adapt the interpolation filter to the three-dimensional spectrum. Special precautions have to be taken when motion compensated interpolation is not possible (e.g. in occluded regions). The detection of these situations and possible solutions are also addressed in this chapter.

The second adaptive interpolation method discussed in this chapter is nonlinear spatial interpolation based on median filters instead of linear spatial interpolation [Renf90] [Leht90]. In a linear interpolation scheme *all* pixels in a specific window contribute to the output value: each pixel value is multiplied by a filter coefficient and the sum is taken. This causes blurring if the pixel values differ strongly from each other (e.g. at edges). If median filtering is used, the pixel values *majority* determines the output, under the assumption that it is most likely that the missing pixel belongs to this majority. If the pixel does indeed belong to the majority, the output value is correct and there are no blurring artifacts. However, if the assumption is not true the interpolation result is erroneous. These errors are again particularly noticeable at edges.

Next, the two above interpolation techniques are combined into a single spatial interpolation filter. This motion adaptive nonlinear spatial interpolation filter possesses the advantages of both the interpolation techniques. The chapter is concluded with the discussion of the results of several experiments using different interpolation methods.

3.1 Motion Compensated Interpolation

The main difference in a motion adaptive subsampling scheme as compared to traditional subsampling is that in the interpolation stage a motion compensated interpolation filter is used instead of a fixed interpolation filter. In this section the main features and advantages of motion compensated interpolation are discussed. Fields where motion compensated interpolation is also applicable are other three-dimensional interpolation problems, such as de-interlacing [Giro85] [Wang89], temporal interpolative coding [MPEG92] and frame rate conversion [Erns91] [Reut89].

3.1.1 Principle

The principle of motion compensated interpolation is first illustrated by an example. In Figure 3.1(a) an image sequence is depicted containing an object moving with a constant velocity in front of a stationary background. At the encoder, half the images of the sequences are skipped, corresponding to a temporal subsampling with a factor two. At the decoder the missing images have to be interpolated. The following simple temporal interpolation filter just averages the previous and the next image:

$$i_{int}(\mathbf{x},t) = \tfrac{1}{2} i(\mathbf{x},t-1) + \tfrac{1}{2} i(\mathbf{x},t+1) \tag{3.1}$$

In Figure 3.1(b) the result $i_{int}(\mathbf{x},t)$ of this filter is shown when it is applied to the image sequence in Figure 3.1(a). The filter works well for the stationary background but produces a blurred version of the moving object. Fixed temporal interpolation may be acceptable for image regions moving at high velocities, but in regions with a low velocity, where the eye is capable of tracking the motion, the blurring is visible. To obtain a good image quality both the temporal and the spatial resolution must be preserved while maintaining the spatial resolution.

The solution for this problem is to use *motion compensated* interpolation instead of fixed temporal interpolation [Giro85] [Erns88]. A motion compensated interpolation filter temporally interpolates the image sequence in the direction of the velocity, in this way adapting the interpolation filter's spatio-temporal passband to the spatio-temporal shape of the spectrum of the subsampled sequence. The interpolation filter of Equation (3.1) now takes on the following form:

$$i_{int}(\mathbf{x},t) = \tfrac{1}{2} i(\mathbf{x} - \mathbf{d}_b(\mathbf{x},t), t-1) + \tfrac{1}{2} i(\mathbf{x} - \mathbf{d}_f(\mathbf{x},t), t+1) \tag{3.2}$$

where $\mathbf{d}_b(\mathbf{x},t)$ is the displacement vector of the object from the image at time t to the image at t-1 and $\mathbf{d}_f(\mathbf{x},t)$ the displacement from the image at time t to the image at time t+1. The interpolation result of this filter is illustrated in Figure 3.1(c). Both the stationary background

and the moving object are reconstructed correctly. This discussion shows that motion compensated interpolation has the potential to improve both the spatial and the temporal resolution after subsampling as opposed to using straightforward temporal interpolation.

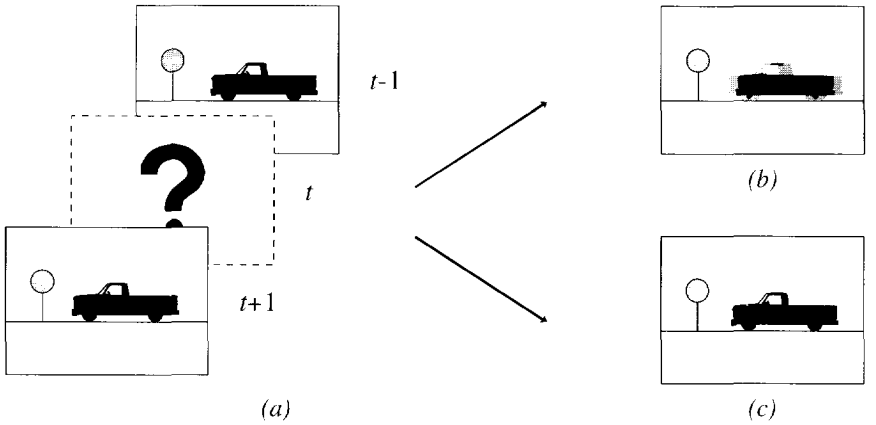


Figure 3.1: (a) The interpolation problem. Interpolation with: (b) a fixed temporal interpolation filter, (c) a motion compensated temporal interpolation filter.

Note that the motion compensated interpolation filter from Equation (3.2) is equivalent to applying a 3-taps one-dimensional FIR filter with coefficients ½, 1 and ½ along the motion trajectory. The most simple temporal interpolation filter is the 2-taps filter with coefficients 1 and 1. This filter is implemented as:

$$i_{int}(\mathbf{x},t) = i(\mathbf{x} - \mathbf{d}_b(\mathbf{x},t), t-1) \tag{3.3}$$

This interpolation filter involves only the previous image, and is effectively a motion compensated extrapolation.

3.1.2 Spectral analysis

In this section, we discuss motion compensated interpolation in the spectral domain and compare it with fixed temporal interpolation. The information presented in this section is used in the next section where we discuss the necessary accuracy of the motion estimation for a given motion compensated interpolation filter. To simplify the discussion in this section and the next section we assume an image sequence with only *global* motion and that all the objects in the scene are moving with the velocity $\mathbf{v} = (v_x, v_y)^T$. The displacement vectors $\mathbf{d}_b(\mathbf{x},t)$ and $\mathbf{d}_f(\mathbf{x},t)$ introduced in the previous section are linked to the velocity as being the distance traveled between two consecutive images. We also assume that the illumination does not change in time. The entire sequence is then described by the following equation:

$$i(\mathbf{x},t) = i(\mathbf{x} - \mathbf{v} \cdot t, 0) \tag{3.4}$$

To do a spectral analysis of motion compensated interpolation, it is first necessary to determine the three-dimensional spectrum of an image sequence. According to the definition of the three-dimensional Fourier transform, the three-dimensional spectrum $I(\omega_x,\omega_y,\omega_t)$ of $i(\mathbf{x},t)$ is given by

$$I(\omega_x,\omega_y,\omega_t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} i(\mathbf{x},t)\exp(-j(\omega_x x+\omega_y y+\omega_t t))\,dxdydt \tag{3.5}$$

Substituting Equation (3.4) into Equation (3.5) gives after some basic manipulations

$$\begin{aligned} I(\omega_x,\omega_y,\omega_t) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} i(\mathbf{x},0)\exp(-j(\omega_x x+\omega_y y))dxdy \cdot \\ &\quad \int_{-\infty}^{\infty}\exp(-j(\omega_x v_x+\omega_y v_y+\omega_t))dt \end{aligned} \tag{3.6}$$

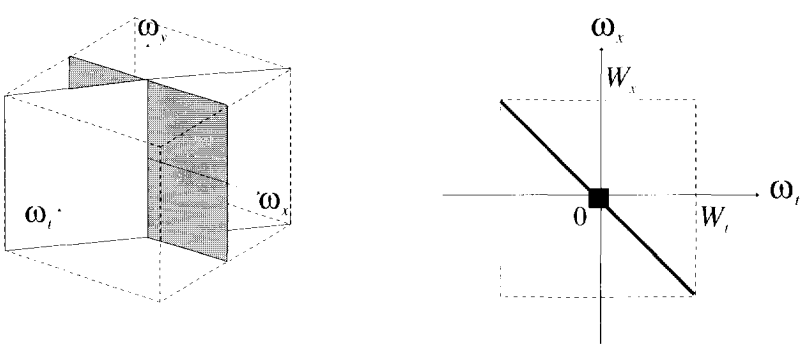
which can be simplified to

$$I(\omega_x,\omega_y,\omega_t) = I_o(\omega_x,\omega_y)\cdot\delta(\omega_x v_x+\omega_y v_y+\omega_t) \tag{3.7}$$

The first part of this equation, $I_o(\omega_x,\omega_y)$ describes the two-dimensional spatial Fourier transform of the image $i(\mathbf{x},0)$. This part of Equation (3.7) is not a function of the temporal frequency, hence it is valid for every ω_t. The second part of the equation represents a delta function whose value is only non-zero when the argument is zero. Non-zero entries occur only when

$$\omega_x v_x+\omega_y v_y+\omega_t = 0\,,\quad |\omega_x|\le W_x, |\omega_y|\le W_y \tag{3.8}$$

which denotes a plane in the three-dimensional Fourier space perpendicular to the vector $(\mathbf{v},1)^T$. Hence the resulting Fourier transform of an image sequence containing global motion is formed by the spatial frequency components intersecting the plane perpendicular to the motion vector. This plane is bounded in the ω_x direction by the horizontal bandwidth W_x and in the ω_y direction by the vertical bandwidth W_y. Figure 3.2(a) serves as an illustration of this. The dark plane in Figure 3.2(a) indicates the support of the spectrum for a stationary sequence, and the light plane indicates the support of the spectrum for an image sequence which can be described by a velocity of $\mathbf{v} = (1,0)^T$. Because of the velocity the original image plane rotates, introducing temporal frequency components.



(a) (b)

Figure 3.2: The effect of motion on the image spectrum: (a) The dark plane is support of the spectrum when there is no motion, and the light plane is the support of the spectrum for a velocity $\mathbf{v} = (1,0)^T$. (b) Intersection of the light plane with the plane $\omega_y = 0$.

To simplify the graphical representation, in this section only motion in the horizontal direction is considered ($v_y = 0$), so only the ω_x-ω_t plane is relevant. The intersection of the support of the three-dimensional spectrum with this plane is the line

$$\omega_t = -\omega_x v_x \ , \quad |\omega_x| \leq W_x \tag{3.9}$$

In Figure 3.2(b) an example is given for a velocity of 1 pixel/image in the horizontal direction, so $\omega_t = -\omega_x$.

With this knowledge about the three-dimensional spectrum, we can discuss the different aspects of motion compensated interpolation in the spectral domain. The same subsampling lattice as in Figure 3.1(a) is used, so the images are alternately discarded. This subsampling lattice introduces extra replicas between the original replicas in the direction of the ω_t-axis. We assume a velocity of 1 pixel/image in the horizontal direction, as was already used in Figure 3.2. The top half of Figure 3.3 illustrates a subsampling scheme in the spectral domain in the case of a fixed temporal interpolation filter at the decoder, whereas the bottom half depicts the situation in which motion compensated interpolation is used. Only the baseband ranging from $-\pi$ to $+\pi$ is shown.

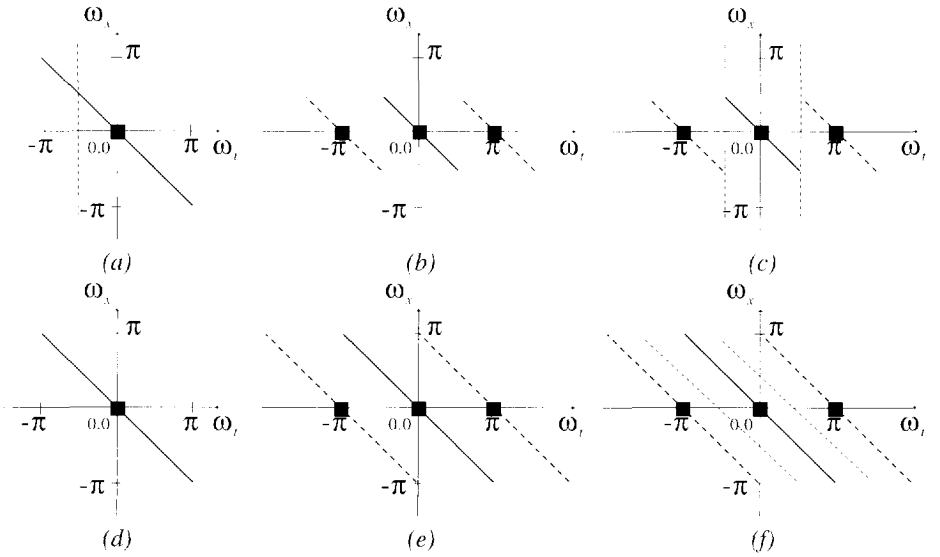


Figure 3.3: Spectral description of subsampling scheme. The lines show the support of the spectral components. (a-c) Fixed temporal interpolation. (d-f) Motion compensated interpolation. The shaded areas are passbands of the necessary low-pass filters.

If a fixed temporal interpolation filter is used, high temporal frequency components may cause aliasing, therefore these have to be eliminated prior to subsampling. To achieve this the input images are temporally low-pass filtered, with a filter whose passband corresponds with the shaded area in Figure 3.3(a). Next the image sequence can be properly subsampled in the temporal direction (Figure 3.3(b)). The dashed lines in the figure are the replicas introduced

by the temporal subsampling of the image sequence. At the receiver, a temporal low-pass filter is used to separate the baseband spectrum and the replicas without introducing aliasing (shaded area in Figure 3.3(c)). In this case the unity cell is a rectangular region and both the spatial and temporal resolution are reduced.

In the case of motion compensated interpolation (Figure 3.3(d-f)) *no* prefiltering is used prior to subsampling (Figure 3.3(d)). In the subsampling stage (Figure 3.3 (e)) the replicas enter the baseband, but do not overlap each other. The interpolation filter from Figure 3.3(c) would give rise to aliasing at the decoder, causing degradation of the interpolated image sequence. Therefore, at the decoder, a motion compensated interpolation filter is used to interpolate the image instead (Figure 3.3(f)). The shape and the orientation of the unity cell is adapted to support of the baseband spectrum of the subsampled sequence. Aliasing is now avoided in the interpolation process without affecting the spatial resolution. It is however necessary that the decoder knows the velocity, since the interpolation filter is applied in the direction of the velocity.

3.1.3 Motion compensated temporal interpolation filters

Some properties of motion compensated temporal interpolation filters are discussed in this section. First we consider the velocity range covered by an interpolation filter. If we know the velocity range of one filter then we know what the distance between the interpolation filters must be to cover all possible velocities. From this result the necessary accuracy of the motion estimation can be derived. In this section we still assume a sequence with one global velocity, but in contrast with the previous section we now consider velocities in both the horizontal and vertical direction.

To cover the entire range of velocities it is not necessary to have a filter for each possible velocity. The two-dimensional velocity plane depicting all the possible horizontal and vertical velocity components can be subdivided into regular regions. Each region can be associated with one specific motion compensated interpolation filter. The set of velocities required to cover the entire velocity plane are called the *nominal velocities* $\mathbf{v}_n = (v_{nx}, v_{ny})^T$.

First the velocity range covered by a given nominal velocity is derived for the two-dimensional case using Figure 3.4, where we only take the ω_x-ω_t plane into consideration. The solid line is the support of the spectrum if the global velocity is equal to the nominal velocity. In Figure 3.4 we use a velocity of 1 pixel/image. The shaded area is the passband of the motion compensated interpolation filter used to cover the velocity range around the nominal velocity. In the two-dimensional case the boundaries of the passband are the lines given by

$$\omega_t = -v_{nx}\omega_x \pm \beta \quad (\beta > 0) \tag{3.10}$$

where β is the cut-off frequency of the one-dimensional interpolation filter which is applied along the direction of the velocity. In Figure 3.4 a perfect low-pass filter is assumed, hence $\beta = \frac{1}{2}\pi$. The modulation transfer function of this filter is shown as the solid line in the bottom part of Figure 3.4.

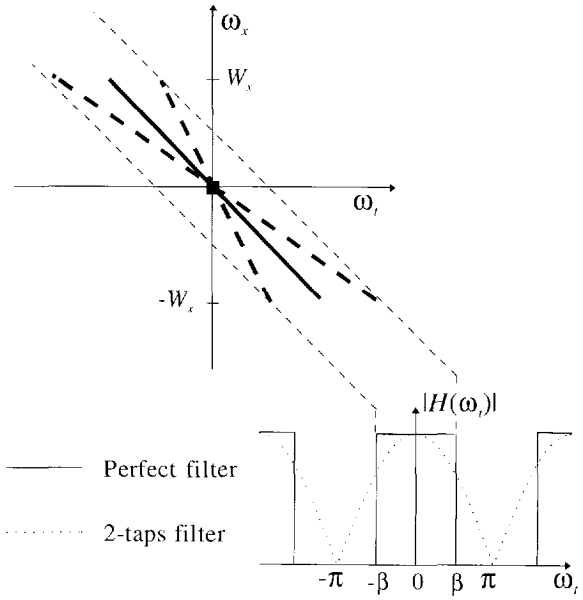


Figure 3.4: Accuracy of the motion compensated filter. The top part shows the motion compensated interpolation filter in the ω_x-ω_t plane. The bottom part shows the modulation transfer function for both a perfect filter and a simple 2-taps filter.

To determine which velocities are covered by the filter's passband we vary the velocity v_x of the input signal, depicted by the dashed lines in Figure 3.4. The input signal is bounded by the horizontal bandwidth W_x, so using Equation (3.9) the boundaries can be described by

$$\omega_t = -W_x v_x \tag{3.11}$$

Next we compute the intersection of the boundaries of the input signal with the filter's passband boundaries. Combining the Equations (3.10) and (3.11) gives for the intersections

$$(v_{nx} - v_x)W_x = \pm\beta \tag{3.12}$$

which, if we take the exact condition under which each intersection occurs into account, can be rewritten to

$$|v_x - v_{nx}|W_x = \beta \tag{3.13}$$

The range covered by the nominal velocities is the distance between the two boundaries and is equal to $2\beta/W_x$. All horizontal velocities are covered by the motion compensated interpolation filters if the nominal velocities are chosen at

$$v_{nx} = \frac{2\beta}{W_x}k + c_x, \quad k \in \mathbb{Z} \quad \text{(pixels/image)} \tag{3.14}$$

where c_x is an arbitrarily chosen constant. If the original image sequence was sampled according to the Nyquist criterion ($W_x = \pi$), then in Figure 3.4 the intersections are at $v_x = ½$

pixel/image and v_x = 1½ pixels/image. The distance between the nominal velocities is equal to 1, therefore the nominal velocities can be chosen at integer locations ($c_x = 0$):

$$v_{nx} = k\,, \quad k \in \mathbb{Z} \qquad \text{(pixels/image)} \tag{3.15}$$

The dotted line in the lower part of Figure 3.4 is the transfer function of a 2-taps temporal interpolation filter. This filter can only be used for the region in which the passband characteristic is reasonable flat. We see in Figure 3.4 that this is only the case for a small region, so β should be chosen smaller than in the case of a perfect low-pass filter. The range covered by a nominal velocity is therefore also smaller for this filter. Note that the short filter has a zero for $\omega_t = \pi$. Hence in the case that the actual velocity exactly equals the nominal velocity the aliasing is completely canceled, providing perfect reconstruction.

In the three-dimensional case the passband of a motion compensated interpolation filter is bounded by planes instead of lines, so Equation (3.10) must be rewritten to

$$\omega_t = \pm\beta - v_{nx}\omega_x - v_{ny}\omega_y \quad (\beta > 0) \tag{3.16}$$

The boundaries of the input signal in three-dimensions are equal to

$$\omega_t = -v_x W_x - v_y W_y \tag{3.17}$$

where W_x and W_y are the horizontal and vertical bandwidths. Substituting Equation (3.17) in Equation (3.16) gives for the intersection points

$$(v_{nx} - v_x)\omega_x + (v_{ny} - v_y)\omega_y = \pm\beta \tag{3.18}$$

which can be rewritten to

$$|v_x - v_{nx}|W_x + |v_y - v_{ny}|W_y = \beta \tag{3.19}$$

This equation describes a diamond-shaped region covered by the nominal velocity in the velocity plane. For an efficient subdivision of the velocity plane the regions covered by each nominal velocity should not overlap with other regions and the entire velocity plane should be covered. Therefore the nominal velocities should be chosen as follows:

$$v_{nx} = \frac{2\beta}{W_x}k + c_x\,, \quad v_{ny} = \frac{2\beta}{W_y}l + c_y \quad k,l \in \mathbb{Z} \qquad \text{(pixels/image)} \tag{3.20}$$

where c_x and c_y are arbitrarily chosen constants.

Since the velocity of a moving object is derived from the motion in a fixed time interval, the accuracy of the velocity is coupled to the accuracy of the motion estimation. From Equation (3.20) it follows that the necessary accuracy of the motion estimation is given by

$$\Delta d_x = \frac{2\beta}{W_x}\,, \quad \Delta d_y = \frac{2\beta}{W_y} \quad \text{(pixels)} \tag{3.21}$$

where Δd_x is the accuracy in the horizontal direction and Δd_y the accuracy in the vertical direction. If the numerical values from the previous example are substituted in Equation (3.21), then the required accuracy is

$$\Delta d_x = 1, \quad \Delta d_y = 1 \quad \text{(pixels)} \tag{3.22}$$

The values prescribed by Equation (3.21) are the minimally required accuracy. If it is possible to realize a motion estimation with a higher accuracy, Equation (3.21) can be solved instead for a different value of β. A lower value for Δd_x and Δd_y results in a smaller value for β, making it possible to use a shorter temporal interpolation filter. This principle is explained by giving an example.

Figure 3.5 illustrates the effect of increasing the accuracy of the motion estimation for the 2-taps temporal interpolation filter. The horizontal frequency response of the filter is shown as a function of the horizontal velocity. When the velocity is no longer within the range of one nominal velocity, another nominal velocity takes over. In the case of a perfect interpolation filter the frequency response is equal to 1 for all values of ω_x and v_x. In Figure 3.5(a) pixel accuracy is used for the motion estimation ($\Delta d_x = \Delta d_y = 1$ pixel). The horizontal velocity range covered by each nominal velocity is the interval $[v_{nx} - ½, v_{nx} + ½]$. We see that especially for high-frequency components the filter response deviates much from the perfect value of 1. This gives rise to blurring artifacts in the interpolated image.

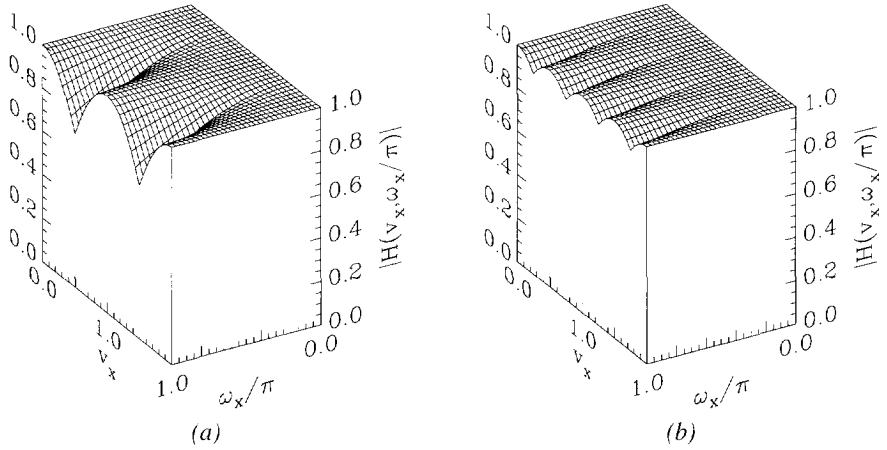


Figure 3.5: Characteristics of the motion compensated interpolation filter as a function of the horizontal velocity v_x and the horizontal frequency ω_x for different accuracy of the motion estimation: (a) Δd_x = 1 pixel, (b) Δd_x = ½ pixel

In Figure 3.5(b) half pixel accuracy is used instead ($\Delta d_x = \Delta d_y = ½$ pixel). Solving Equation (3.21) for half pixel accuracy and with $W_x = W_y = \pi$, gives a value of $¼\pi$ for β. The nominal velocities are now given by

$$v_{nx} = ½k, \quad v_{ny} = ½l, \quad k,l \in \mathbb{Z} \quad \text{(pixels/image)} \tag{3.23}$$

The horizontal velocity range covered by each nominal velocity in Figure 3.5(b) is now the interval $[v_{nx} - ¼, v_{nx} + ¼]$. Over this smaller range the simple 2-taps filter is a good

approximation of a perfect filter, effectively improving the overall filter response. Hence in practice a short filter should be used in conjunction with a high motion estimation accuracy.

3.2 Motion compensated interpolation in practice

In the previous section, the properties of motion compensated interpolation filters were derived under specific conditions. In this section we look at how these conditions must be translated to a practical situation.

Motion compensated filtering is done by temporally filtering an image sequence in the direction of the motion. In the previous discussions we assumed a constant velocity over a number of images. In [Dubo92] it is shown that this is not a necessary assumption and that it is sufficient to follow an object along a motion *trajectory*. The assumption made in this case is that the characteristics of the object do not change dramatically from image to image caused by variations in the illumination, object deformations or occlusions.

In Section 3.1.1 two temporal interpolation filters were introduced, namely the 2-taps filter with coefficients 1 and 1, and the 3-taps filter with coefficients 1, ½ and 1. These filters have a poor frequency response, but are particularly interesting because they impose small memory requirements. If longer filters are used, the overall characteristics improve; however, also the number of images involved in the filtering process increases, and thus more image stores are required. Another disadvantage of long temporal filters is that the assumption that the objects do not change in time is likely to be less valid over a longer period of time.

In the previous section we saw that the result of motion compensated filtering also depends on the accuracy of the motion estimation. To apply motion compensated interpolation on a subsampled image sequence, the displacement should be defined on the sampling lattice. In order to meet this accuracy demand, it is therefore necessary to expand the spatial resolution of the subsampled image sequence back to the spatial resolution on which the motion was estimated. In the case of fractional accuracy of the motion vectors, this means that the original image sequence has to be spatially upsampled to an image sequence with bigger dimensions. If the upsampling filters involved in this process are not sufficiently long this may cause a loss of resolution.

3.2.1 Motion estimation requirements

An important aspect of motion compensated interpolation is the estimation of the motion vectors. If the motion estimation does not provide an accurate motion vector, an inaccurate temporal interpolation filter is used and the result becomes useless.

A requirement for the motion estimation is that the estimation is done at the encoder side. At the decoder side not all the samples of the image sequence are available so reliable motion estimation may become difficult. A consequence of this requirement is that the motion vectors are transmitted to the decoder as side information. To reduce the amount of side information, the motion estimation is done on a block basis and only one vector is transmitted

for each block. An alternative not chosen here is to estimate the motion on a pixel basis and to encode the vector field prior to transmission. The fact that the motion information is transmitted also limits the maximum accuracy of the motion estimation. Increasing the accuracy increases the amount of side information.

If the 3-taps interpolation filter from Equation 3.2 is used, two motion vectors $\mathbf{d}_b(\mathbf{x},t)$ and $\mathbf{d}_f(\mathbf{x},t)$ are necessary. To reduce the amount of side information we can assume that $\mathbf{d}_f(\mathbf{x},t)$ is equal to $-\mathbf{d}_b(\mathbf{x},t)$ so only $\mathbf{d}_b(\mathbf{x},t)$ is transmitted. This assumption is only valid if the object is moving with a constant velocity and the estimated motion vector represents the true motion vector. The last requirement imposes large constraints on the motion estimation algorithm.

3.2.2 Hierarchical block matching

There are several block based algorithms which are capable of estimating true motion, such as hierarchical full search block matching [Bier88], recursive block matching [DeHa93] or phase plane correlation [Thom87]. In this thesis hierarchical full search block matching is used. To introduce hierarchical full search block matching we first discuss full search block matching [Musm85]. The actual image is divided into blocks. Next a search is made for the best corresponding block in the previous image based on some matching criterion. The difference between the position of the block in the actual image and the position of the corresponding block in the previous image is the estimated motion vector (Figure 3.6(a)). For computational reasons it is not possible to search over the entire image. Therefore the search area is limited to a region around the current block. The size of the region determines the maximal displacement.

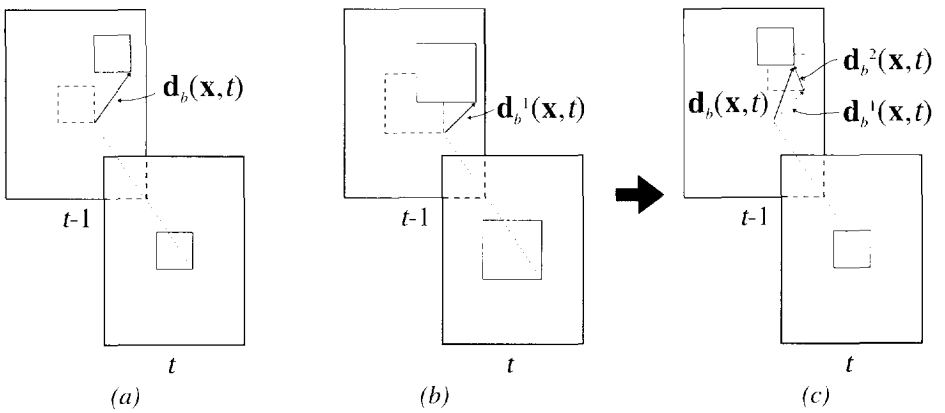


Figure 3.6: (a) The principle of full search block matching. (b,c) The principle of two level hierarchical block matching: (b) level 1, (c) level 2.

This algorithm is easily distracted by the local image contents and does not necessarily produces a motion field which corresponds with the true motion. In Figure 3.7(a) an example of an estimated motion vector field is given. Figure 3.7(b) shows the result after motion compensated interpolation, where $-\mathbf{d}_b(\mathbf{x},t)$ is used instead of $\mathbf{d}_f(\mathbf{x},t)$. The image contains a

rotating object. We see that the estimated vectors do not correspond with the true motion vector and the assumption that $-\mathbf{d}_b(\mathbf{x},t)$ is equal to $\mathbf{d}_f(\mathbf{x},t)$ leads in some places to large errors.

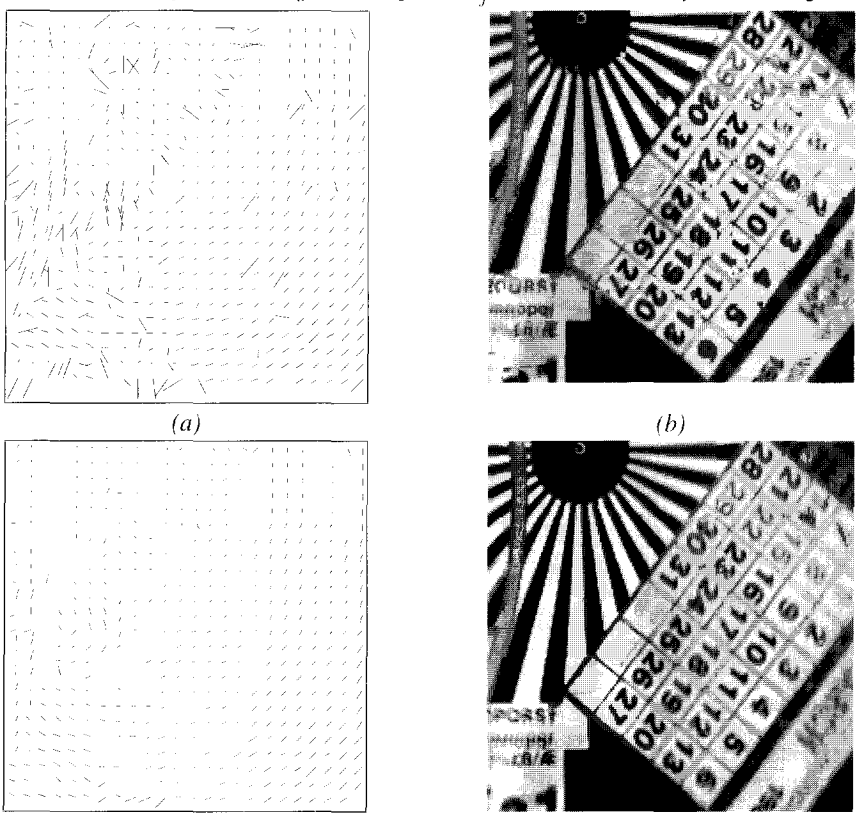


Figure 3.7: (a,b) Full search block matching: (a) vector field, (b) result of motion compensated interpolation. (c,d) Hierarchical block matching: (c) vector field, (d) result of motion compensated interpolation.

To obtain a motion estimation which corresponds better with the actual motion, the hierarchical block matching algorithm [Bier88] can be used (Figure 3.6(b,c)). The estimation process starts with large blocks to make a rough initial estimate $\mathbf{d}_b^1(\mathbf{x},t)$ of the motion vector. This initial estimate is used in a following stage as a start point for a finer estimation $\mathbf{d}_b^2(\mathbf{x},t)$ of the motion using smaller blocks than the initial estimate. The new estimate can be used again to obtain a finer estimate. This algorithm is a compromise between the need for small blocks for a good local motion estimate, computational complexity and homogeneity of the motion vectors. On a coarse scale the estimation process is not easily distracted by the local image contents, so the motion estimation corresponds better with the true motion vector. This approach fails at edges of the motion field, because in this case the coarse scale motion vector may not correspond with the local motion vector.

The evaluation of the matching criterion for each block does not only use the pixels of the block, but a window larger than the block is placed around it. Now the pixels in the window are used for the computation of the matching criterion. This action increases the homogeneity of the motion vector because information from the neighboring blocks is also considered, but it increases the number of computations. The number of computations is reduced again by not evaluating the error function for *all* the pixels in the window. Only the pixels at a regularly spaced distance from each other on an orthogonal lattice are used. Aliasing is avoided by using a mean filter prior to the motion estimation.

Figure 3.7(c) and Figure 3.7(d) respectively show the estimated motion vector field and the result after motion compensated interpolation. We see that the estimated motion field is more homogeneous than the field obtained with full search block matching and corresponds better with the true motion.

3.2.3 Handling inaccurate motion vectors

In motion compensated interpolation schemes, particular precautions have to be taken in regions where the motion estimation is inaccurate. The motion estimation is inaccurate if the interpolation result has no relation to the original image. This happens in occluded regions or regions moving at high velocity. The question is how these situations can be detected and what corrective actions should be taken. Therefore, first the errors resulting from motion compensated subsampling are modeled. Two distinctive situations are considered: the case when an accurate motion estimation is made, and the situation when an inaccurate motion estimation is made. These two situations each introduce different types of interpolation errors.

In a practical situation the intensity function describing an image sequence does not only contain information about the actual scene, but also noise introduced during the image acquisition process. The noise is modeled by an additive term:

$$i(\mathbf{x},t) = i_o(\mathbf{x},t) + n(\mathbf{x},t) \tag{3.24}$$

where $i_o(\mathbf{x},t)$ is the actual intensity, $i(\mathbf{x},t)$ the observed intensity and $n(\mathbf{x},t)$ the noise. We assume that the variance σ_n^2 of the noise is small compared to the variance σ_i^2 of the actual intensity and that the noise is uncorrelated with the actual intensity. If an object is translating in the image plane according to the vector $\mathbf{d}_b(\mathbf{x},t)$, then $i_o(\mathbf{x},t)$ is equal to

$$i_o(\mathbf{x},t) = i_o(\mathbf{x} - \mathbf{d}_b(\mathbf{x},t), t-1) \tag{3.25}$$

Now consider $\hat{\mathbf{d}}_b(\mathbf{x},t)$ to be the *estimated* motion vector. If an accurate motion estimation is made, $\hat{\mathbf{d}}_b(\mathbf{x},t)$ is equal to $\mathbf{d}_b(\mathbf{x},t)$. In practice $i_o(\mathbf{x} - \hat{\mathbf{d}}_b(\mathbf{x},t), t-1)$ is approximated by the observed pixel $i(\mathbf{x} - \hat{\mathbf{d}}_b(\mathbf{x},t), t-1)$ in the previous image given by

$$i(\mathbf{x} - \hat{\mathbf{d}}_b(\mathbf{x},t), t-1) = i_o(\mathbf{x} - \hat{\mathbf{d}}_b(\mathbf{x},t), t-1) + n(\mathbf{x} - \hat{\mathbf{d}}_b(\mathbf{x},t), t-1) \tag{3.26}$$

The interpolation error value $e(\mathbf{x},t)$ is the difference between the observed pixel value and the observed interpolated pixel value:

$$e(\mathbf{x},t) = i(\mathbf{x},t) - i(\mathbf{x} - \hat{\mathbf{d}}_b(\mathbf{x},t), t-1) \tag{3.27}$$

Substituting Equations (3.24) and (3.26) into Equation (3.27) and using Equation (3.25) gives

$$e(\mathbf{x},t) = n(\mathbf{x},t) - n(\mathbf{x}-\hat{\mathbf{d}}_b(\mathbf{x},t),t-1) = \Delta n(\mathbf{x},t) \tag{3.28}$$

where $\Delta n(\mathbf{x},t)$ denotes the part of the error introduced by noise. Thus in the case of a correct motion estimation, the introduced error is equal to noise colored by the motion vector field. If we assume white noise then the variance of the reconstruction error is approximately equal to

$$\begin{aligned} \mathrm{E}\left[e(\mathbf{x},t)^2\right] &= \mathrm{E}\left[\Delta n(\mathbf{x},t)^2\right] \\ &\approx 2\,\mathrm{E}[n(\mathbf{x},t)]^2 = 2\sigma_n{}^2 \end{aligned} \tag{3.29}$$

where E[.] is the expectation operator. There are no clearly visible errors because the interpolation error variance is of the same order as the original noise variance and has no relation to the image contents.

If an inaccurate motion estimation is made, Equation (3.25) no longer holds. The reconstruction error is now given by

$$\begin{aligned} e(\mathbf{x},t) &= i_o(\mathbf{x},t) - i_o(\mathbf{x}-\hat{\mathbf{d}}_b(\mathbf{x},t),t-1) + n(\mathbf{x},t) - n(\mathbf{x}-\hat{\mathbf{d}}_b(\mathbf{x},t),t-1) \\ &= \Delta i_o(\mathbf{x},t) + \Delta n(\mathbf{x},t) \end{aligned} \tag{3.30}$$

An extra error term $\Delta i_o(\mathbf{x},t)$ is introduced as compared to Equation (3.28). The variance of the interpolation error can now, again under the assumption of white noise, be approximated by:

$$\begin{aligned} \mathrm{E}\left[e(\mathbf{x},t)^2\right] &= \mathrm{E}\left[\left(\Delta i(\mathbf{x},t) + \Delta n(\mathbf{x},t)\right)^2\right] \\ &\approx \sigma_{\Delta i}{}^2 + 2\sigma_n{}^2 \end{aligned} \tag{3.31}$$

where $\sigma_{\Delta i}{}^2$ is the variance of $\Delta i_o(\mathbf{x},t)$. If the intensity value changes dramatically from one image to another in the direction of the estimated motion vector then $i_o(\mathbf{x}-\hat{\mathbf{d}}_b(\mathbf{x},t),t-1)$ differs significantly from $i_o(\mathbf{x},t)$ and $\sigma_{\Delta i}{}^2$ is much larger than $\sigma_n{}^2$. The $\Delta i_o(\mathbf{x},t)$ term now dominates, introducing conspicuous errors, because the variance of the error now depends on the image sequence contents. From this discussion it is clear that the errors made in the case of an accurate motion estimation differ significantly from the errors made in the case of an inaccurate motion estimation. This result is illustrated next with an experiment.

The histogram of the error value $e(\mathbf{x},t)$ was determined for the second image of the *MOBILE* sequence. The sign of the error value is not relevant so the absolute error value is used. In Figure 3.8 the histogram of the interpolation error for a subsampling scheme with motion compensated interpolation (MC-INT) is shown. As a reference, the histogram of the interpolation error of a fixed subsampling scheme with a quincunx subsampling lattice is shown (QNX-INT). The figure consists of three parts. On the upper part the complete histogram is shown. The two bottom graphs both show a specific part of the horizontal axis. The left part shows the behavior of the histogram for small error values whereas the right part shows the behavior for large error values.

As can be seen from the figure the two bottom parts differ significantly. The motion compensated scheme (solid line) has a high probability for small error values. This corresponds with Equation (3.28) which described the characteristics of the reconstruction

error when an accurate motion estimation is available. The interpolation error is equal to $\Delta n(\mathbf{x},t)$, which has a variance in the same order as the noise in the original image sequence. However there is also a non-zero occurrence for large error values. These correspond to areas with an inaccurate motion estimation as modeled in Equation (3.30). In this case the error consists of $\Delta i_o(\mathbf{x},t)$ and $\Delta n(\mathbf{x},t)$. These kind of errors cause a limited number of artifacts in the interpolated image, which are nevertheless clearly visible.

The fixed lattice subsampling scheme (dashed line) has a smaller occurrence of small error values but a zero occurrence of high error values. This results in a completely blurred reconstructed image but without clearly visible isolated artifacts. Experiments on different image sequences show similar results.

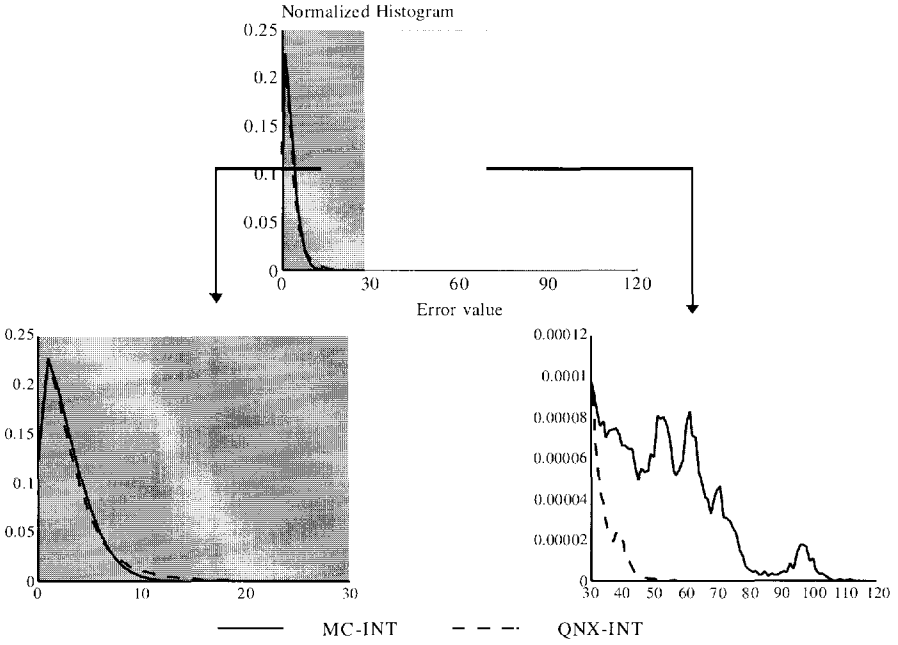


Figure 3.8: Histogram of the interpolation error for the second image of the MOBILE sequence.

In a practical coding scheme, a combination of these two histograms can be made in a system with two branches (Figure 3.9). Subsampling is combined with motion compensated interpolation if the interpolation error is small. In the case of large error values a fall-back mode is used. A requirement for the fall-back mode is that it should not introduce clearly visible artifacts. We saw in Figure 3.8 that fixed lattice subsampling satisfies this requirement. The decision as to which branch to use is made at the encoder and transmitted as side information to the decoder. Thus the interpolation is also done at the encoder. This system gives a high resolution image in the case of an accurate motion estimation but without clearly visible artifacts in the case of an inaccurate motion estimation.

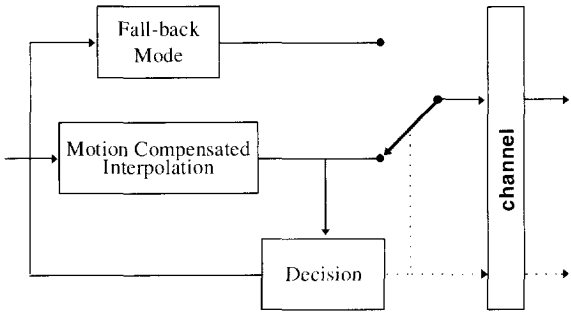


Figure 3.9: Motion compensated interpolation scheme with fall-back mode.

An additional step to improve the interpolation result is to reduce the variance $\sigma_{\Delta i}^2$, by effectively reducing the area under the tail of the histogram. This can be accomplished by improving the interpolator. An interpolator which is intended to give better results in areas with occlusions is

$$i_{int}(\mathbf{x},t) = \begin{cases} \frac{1}{2}\, i(\mathbf{x}-\mathbf{d}_b(\mathbf{x},t),t-1) + \frac{1}{2}\, i(\mathbf{x}-\mathbf{d}_f(\mathbf{x},t),t+1) \\ i(\mathbf{x}-\mathbf{d}_b(\mathbf{x},t),t-1) \\ i(\mathbf{x}-\mathbf{d}_f(\mathbf{x},t),t+1) \end{cases} \tag{3.32}$$

where the interpolator which gives the smallest interpolation error is chosen. If a region is covered in the next image, some information about that region may be available in the previous image. The same will hold for uncovered regions. The information as to which interpolator to use should also be transmitted to the decoder.

3.3 Nonlinear spatial interpolation

3.3.1 Principle

Median filtering is a nonlinear filtering technique which is known for preserving sharp intensity transitions in signals. The median is defined as the center sample of an ordered set of samples arranged according to the magnitude. If the number of samples is even, there are two center samples and the average of these two center samples is taken. In this section median filters are used for the spatial interpolation of quincunx subsampled images [Renf90] [Leht90]. The application of median filters is however not limited to spatial quincunx lattices and the filters can be adapted to other lattices.

First, median filter interpolation is illustrated using the *3-point vertical median* filter. The missing pixels of the subsampling lattice are interpolated as follows

$$i_{int}(\mathbf{x}) = \text{med}\left[i(\mathbf{x}+\begin{pmatrix}0\\-1\end{pmatrix}), i(\mathbf{x}+\begin{pmatrix}0\\1\end{pmatrix}), i(\mathbf{x}+\begin{pmatrix}-1\\0\end{pmatrix}) \right] \tag{3.33}$$

where med[.] stands for the median operator and $i_{int}(\mathbf{x})$ for the spatially interpolated result. The pixels which are already present in the subsampling lattice are not changed. The 3-point vertical median filter is demonstrated in Figure 3.10. A vertical edge is quincunx subsampled. After applying the 3-point vertical median filter to the subsampled image the original edge is recovered, shown by the interpolated image on the right side. The edge is preserved because the proper edge pixels are always in the majority and thus determine the filter output. If a two-dimensional FIR filter is used, the weighted sum is taken from the pixels surrounding the discarded pixels. This causes a smearing of the edge. Hence, median filtering has the potential for a better preservation of the high-frequency structures after interpolation.

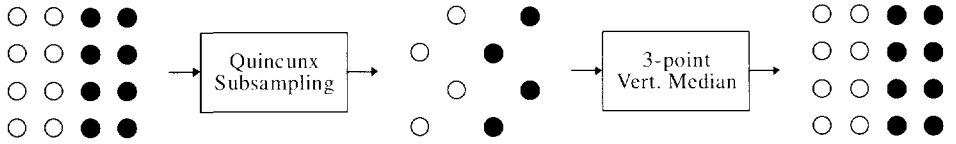
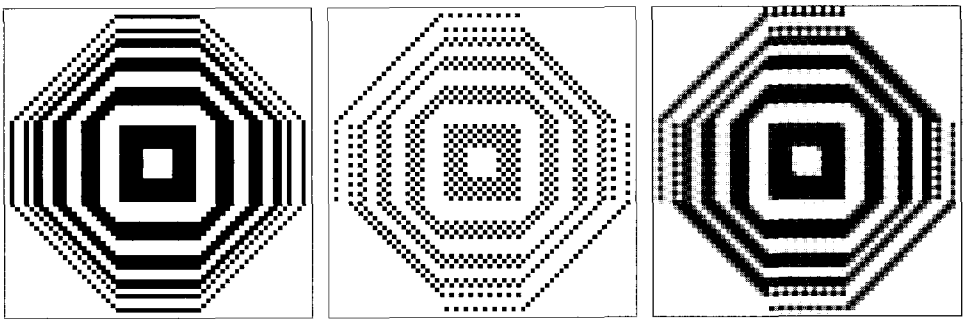


Figure 3.10: Illustration of nonlinear interpolation using a 3-point vertical median filter.

3.3.2 Median interpolation filters

Median filters are nonlinear which complicates their mathematical analysis. In the past, the study of median filters was based on their *root signals*. These are defined as the signals which are unchanged after filtering [Gall81]. This approach is especially useful to study the noise filtering capabilities of median filters. However, for interpolation we are more interested in the signal structures which are properly interpolated by the median filter. In Figure 3.10 we saw that a vertical edge is a signal structure which is properly interpolated by a 3-point vertical median filter.



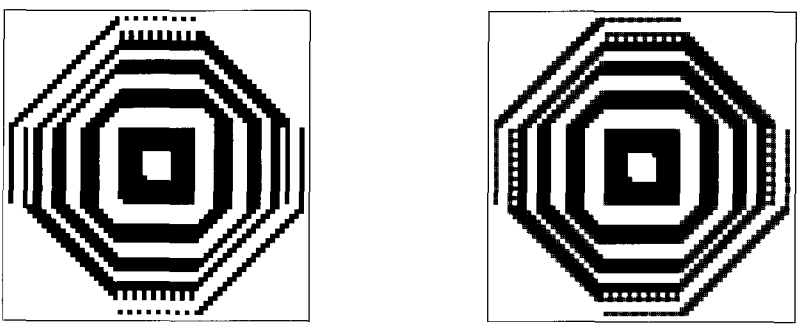
(a) *(b)* *(c)*

Figure 3.11: Test image for nonlinear interpolation: (a) Original image, (b) Quincunx subsampled image, (c) Interpolation with a 3×3 taps FIR-filter.

In Figure 3.11(a) a test image is shown which is used to show the interpolation qualities of the different nonlinear interpolation filters. The image is constructed in such a way that it is possible to see which image structures are interpolated by the different median filters. The test image contains horizontal, vertical and diagonal edges and thin lines. The thin lines are important because in a gray-value image they correspond to structures of one pixel wide with

the same intensity. These structures occur frequently in images containing fine detail. The quincunx subsampled test image is shown in Figure 3.11(b). *No* prefiltering is used prior to subsampling. Prefiltering blurs the edges and the edge-preserving property of the median filter is no longer apparent. The consequence is that there are some aliasing artifacts (e.g. some diagonal lines disappeared). In Figure 3.11(c) the image is interpolated with an FIR filter and we see that for this image FIR-filters are clearly not suitable.

In Figure 3.12(a), the interpolation result is shown after a 3-point vertical median filter is applied on the subsampled test image. We see that both the horizontal and the vertical edges are preserved. Also the vertical thin lines are preserved. However, the horizontal thin lines are converted into vertical lines because of the vertical orientation of the filter. The diagonal edges are also distorted. Figure 3.13 shows the cause of the distortion of thin lines. We see that viewed on a small scale both vertical and horizontal thin lines produces the same output on a quincunx sampling lattice. Thus with only the subsampled image available, it is not possible to detect the proper orientation of a thin line and an arbitrary choice has to be made.



(a) *(b)*

Figure 3.12: Interpolation results: (a) 3-point vertical median, (b) 4-point median filter.

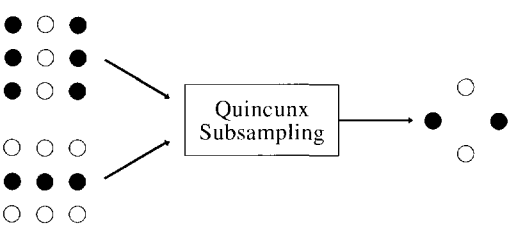


Figure 3.13: Subsampling of horizontal and vertical thin line structures.

An alternative to the 3-point vertical median filter is the 4-point median filter which uses the four pixels around the missing pixel:

$$i_{int}(\mathbf{x}) = \text{med}\left[i(\mathbf{x}+\begin{pmatrix}-1\\0\end{pmatrix}), i(\mathbf{x}+\begin{pmatrix}1\\0\end{pmatrix}), i(\mathbf{x}+\begin{pmatrix}0\\-1\end{pmatrix}), i(\mathbf{x}+\begin{pmatrix}0\\1\end{pmatrix}) \right] \tag{3.34}$$

Note that the number of samples involved is even, so the output value is not necessarily equal to one of the input samples. In Figure 3.12(b) we see that again the horizontal and vertical

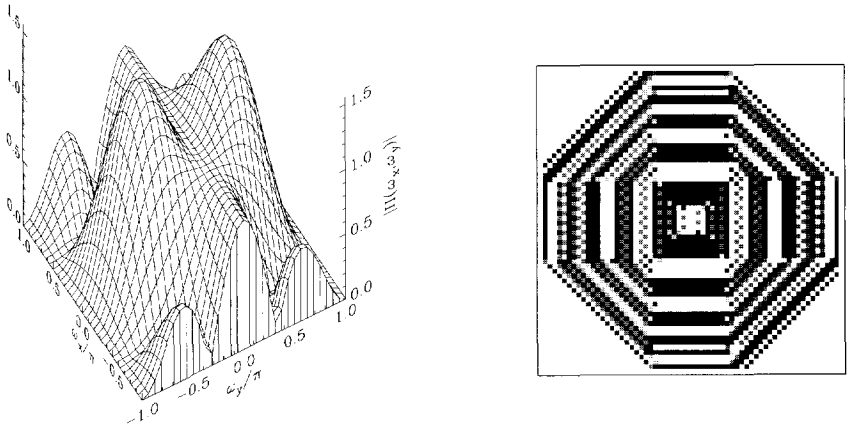
edges are preserved. In the areas containing thin horizontal and vertical lines, a compromise is made between a white and a black pixel and a gray pixel is chosen.

3.3.3 FIR-median hybrid interpolation filters

The main drawback of median filters is the poor interpolation in regions with thin details. To solve this problem FIR-median hybrid (FMH) filters [Hein87] can be used [Leht90]. An FMH filter consists of a set of FIR filters. The output values of these FIR filters are fed into a median filter. The FIR filters are designed in such a way that they give a good reconstruction for thin line structures with a specific orientation. An example of a filter capable of reconstructing horizontal edges and thin line structures on a quincunx lattice is shown in Table 3.1. The filter modulation transfer function, shown in Figure 3.14(a), confirms this. In Figure 3.14(b) the filter is applied to the test image. We see that the horizontal edges and thin lines are indeed reconstructed correctly. When the x and y coordinates of the filter coefficients are exchanged the filter properly reconstructs vertical edges.

Table 3.1: Horizontally oriented filter ([Leht90]).

x \ y	-2	-1	0	1	2
-1	-¼	0	½	0	-¼
0	0	½	1	½	0
1	-¼	0	½	0	-¼



(a) *(b)*

Figure 3.14: (a) Amplitude transfer function of the filter from Table 3.1, (b) Filter output from test image.

An example of an FMH filter is the following interpolation filter:

$$i_{int}(\mathbf{x}) = \text{med}\left[i(\mathbf{x}+\begin{pmatrix}-1\\0\end{pmatrix}), i(\mathbf{x}+\begin{pmatrix}0\\-1\end{pmatrix}), i(\mathbf{x}+\begin{pmatrix}1\\0\end{pmatrix}), i(\mathbf{x}+\begin{pmatrix}0\\1\end{pmatrix}), i_{hor}(\mathbf{x})\right] \tag{3.35}$$

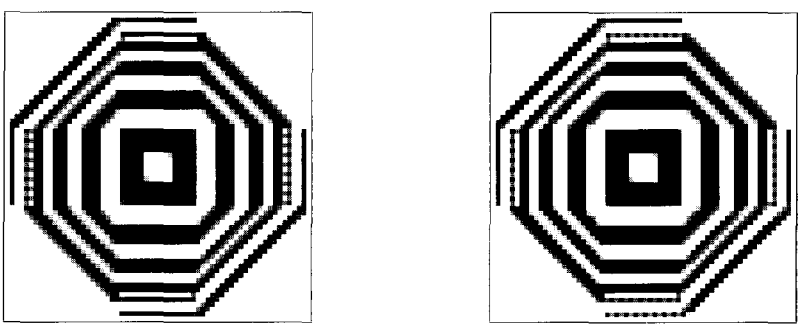
This a combination of the 4-point median filter with the output $i_{hor}(\mathbf{x})$ of the horizontally oriented filter. In Figure 3.13, we saw that the 4-point median filter was not able to make a

decision for thin lines. For horizontal lines, the oriented filter now gives the decisive vote. We see in Figure 3.15(a) that the horizontal edges and thin lines are well preserved. To improve the result for vertical edges, the oriented filter can be applied in the vertical direction and the result $i_{ver}(\mathbf{x})$ can be included in Equation (3.35):

$$i_{int}(\mathbf{x}) = \text{med}\left[i(\mathbf{x}+\begin{pmatrix}-1\\0\end{pmatrix}), i(\mathbf{x}+\begin{pmatrix}0\\-1\end{pmatrix}), i(\mathbf{x}+\begin{pmatrix}1\\0\end{pmatrix}), i(\mathbf{x}+\begin{pmatrix}0\\1\end{pmatrix}), i_{hor}(\mathbf{x}), i_{ver}(\mathbf{x}) \right] \quad (3.36)$$

Figure 3.15(b) shows the result of this interpolation filter. The vertical edges are improved at the expense of the horizontal edges.

In the literature, many other variations on FMH filters have been proposed (e.g. [Huuh93]). They differ in the number of pixels involved in the interpolation and in the FIR filters which are used. However the interpolation qualities of the filters do not differ much. To further improve the performance, motion information needs to be brought into the nonlinear filters.



(a) (b)

Figure 3.15: Interpolation results: (a) horizontal FMH filter, (b) combined horizontal and vertical FMH filter.

3.3.4 Motion compensated nonlinear interpolation

The main problem in nonlinear spatial interpolation is to detect whether a pixel belongs to a particular edge in the image. In the section dealing with motion compensated interpolation, we saw that the motion compensated pixel value from the previous image can be a good estimate for the actual pixel. This temporally predicted pixel is now used to give the decisive vote in the spatial median filter. Hence, a straightforward extension of the spatial nonlinear interpolation filters from the previous section is

$$i_{int}(\mathbf{x},t) = \text{med}\left[i(.,t), \ldots, i_{mc}(\mathbf{x},t) \right] \quad (3.37)$$

where $i(.,t)$ stands for the necessary spatial pixels and with

$$i_{mc}(\mathbf{x},t) = i_{int}(\mathbf{x} - \mathbf{d}_b(\mathbf{x}), t-1) \quad (3.38)$$

Because the actual previous image is not available, the motion compensated interpolation is based on the interpolated version of the previous image. Therefore the quality of the interpolation result relies on both the quality of the previously interpolated image and on the accuracy of the motion estimation. This interpolation filter has an implicit provision for

handling inaccurate motion vectors. If the predicted pixel value $i_{mc}(\mathbf{x},t)$ is inaccurate, its value differs much from the other pixels in Equation (3.37), and in the median filter the motion compensated pixel is automatically rejected. For instance, at the start of the sequence and at scene changes motion compensation is not possible or not meaningful; then simple spatial nonlinear interpolation filtering is *automatically* used as a fall-back option.

Another variation on the median filters is the class of *weighted* median filters [Yli91]. In a weighted median filter the input values are repeated, and a weight factor determines how many times a particular sample is repeated. By adjusting the different weights an emphasis can be placed on certain pixels. In [Haav92] an interpolator was proposed which uses a spatio-temporal weighted median filter. This filter uses no motion compensation but a detection of the motion activity. The corresponding pixel in the previous image $i(\mathbf{x},t\text{-}1)$ is combined with pixels surrounding the missing pixel in the current image. If little motion activity is detected, the weight of the pixel from the previous image is increased and when there is much motion activity the weight is decreased. Hence the interpolator switches between spatial and temporal interpolation depending on the temporal predictability of the images.

This spatio-temporal weighted median filter can also be combined with motion compensated interpolation. The following motion compensated interpolation filter can be constructed:

$$i_{int}(\mathbf{x},t) = \mathrm{med}\left[i(.,t),\ldots,w\Diamond i_{mc}(\mathbf{x},t)\right] \tag{3.39}$$

where $\Diamond$ denotes the replication operator (e.g. $3 \Diamond x = x,x,x$). The weight factor w determines how much emphasis is placed on the motion compensated pixel value. In a motion compensated interpolation scheme the absolute displaced frame difference ($|DFD|$) defined as

$$|DFD| = |i(\mathbf{x},t) - i_{mc}(\mathbf{x},t)| \tag{3.40}$$

is an indicator of the accuracy of the motion vector. A low $|DFD|$ may indicate a good motion estimation and a high $|DFD|$ a bad motion estimation. Because the actual pixel value is not available at the decoder an *estimate* of the $|DFD|$ is made by using the average of the $|DFD|$ of surrounding pixels. These pixels are available at the decoder. If the average $|DFD|$ exceeds a certain threshold, a low value is chosen for w and otherwise a high value is chosen. This scheme switches adaptively between spatial and temporal interpolation based on the quality of the motion compensated prediction.

3.4 Experiment results

In this section, both the results obtained with motion compensated temporal interpolation and nonlinear spatial interpolation are discussed. The motion compensated nonlinear spatial interpolation filters are discussed in a separate section.

3.4.1 Motion compensated linear interpolation

Here, the results obtained with motion compensated interpolation are presented. Half the number of images is skipped at the encoder and reconstructed at the receiver. The motion is

estimated with the hierarchical block matching algorithm with two levels and a full search on each level. The settings at each level are summarized in Table 3.2. The block size at the second level is equal to 8×8 pixels. Experiments are done with both pixel and half pixel accuracy of the motion estimator. The necessary upsampling for fractional accuracy is done with a 7-taps maximally flat filter, so that the original pixels are not distorted. The resulting vector fields for pixel and half pixel accuracy are shown in Figure 3.16 for the second image of the *MOBILE* sequence. We can see that especially the calendar is moving with a fractional velocity.

Table 3.2: Hierarchical block matching parameters.

	Level 1	*Level 2*
Block size	16×16	8×8
Window size	64×64	16×16
Maximum displacement	± 7	± 3
Subsampling factor	4	1
Mean filter size	5×5	3×3

The following five interpolation schemes are used:

①: 2-taps interpolation filter from Equation 3.3, with pixel motion accuracy.

②: 2-taps interpolation filter from Equation 3.3, with half-pixel motion accuracy.

③: 3-taps interpolation filter from Equation (3.2), with half-pixel motion accuracy.

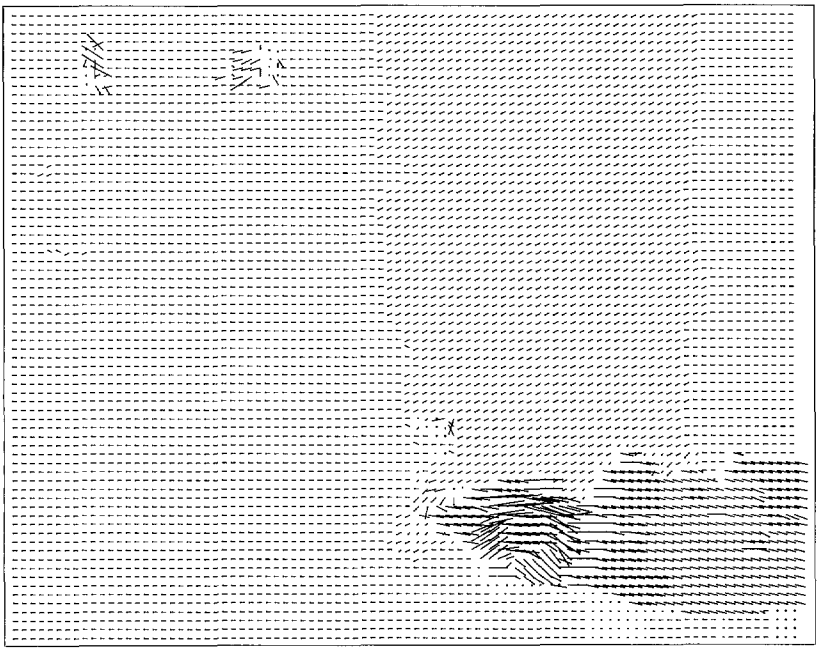
④: 3-taps interpolation filter from Equation (3.2), with half-pixel motion accuracy and $-\mathbf{d}_b(\mathbf{x},t)$ is used instead of $\mathbf{d}_f(\mathbf{x},t)$

⑤: The combined interpolator from Equation (3.32), with half-pixel motion accuracy.

If the side information is not coded, interpolator ② requires twice the amount of side information as interpolator ① because of the increased accuracy. Interpolator ③ requires twice the amount of side information as interpolator ② because two vectors are needed instead of one vector. Interpolator ④ requires the same amount of side information as interpolator ② because again only one vector is needed. Interpolator ⑤ requires the largest amount of side information because beside the two motion vectors, also information about which interpolator should be used is needed at the decoder.

Figure 3.17 shows the results of the different interpolators. The interpolated images are compared with the original images. We see that for this sequence that there is a large benefit if half-pixel motion accuracy is used instead of pixel accuracy, because the calendar is moving with fractional velocity. This confirms the observation from Figure 3.5 that a closer spacing of the nominal velocities improves the overall interpolation filter characteristic. The benefit of using fractional motion accuracy can also be noticed if we compare the stretched difference images in Figure 3.18. In Figure 3.18(a) the difference is shown for the second image of the *MOBILE* sequence for interpolator ① and in Figure 3.18(b) for interpolator ②.

(a)



(b)

Figure 3.16: Backward estimated motion vectors $\mathbf{d}_b(\mathbf{x},t)$ for the second image of the MOBILE sequence: (a) pixel accuracy, (b) half-pixel accuracy.

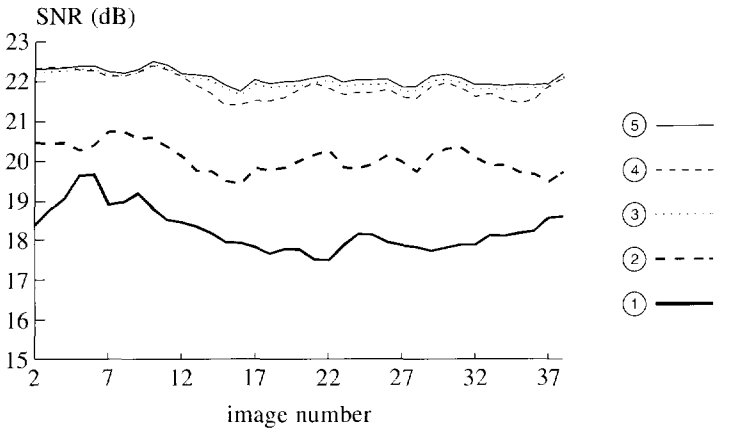
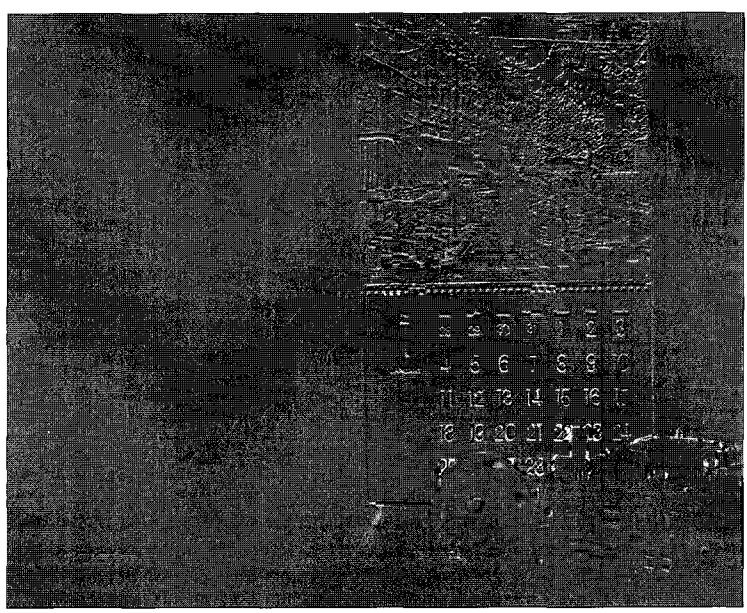


Figure 3.17: Interpolation results for the MOBILE sequence.

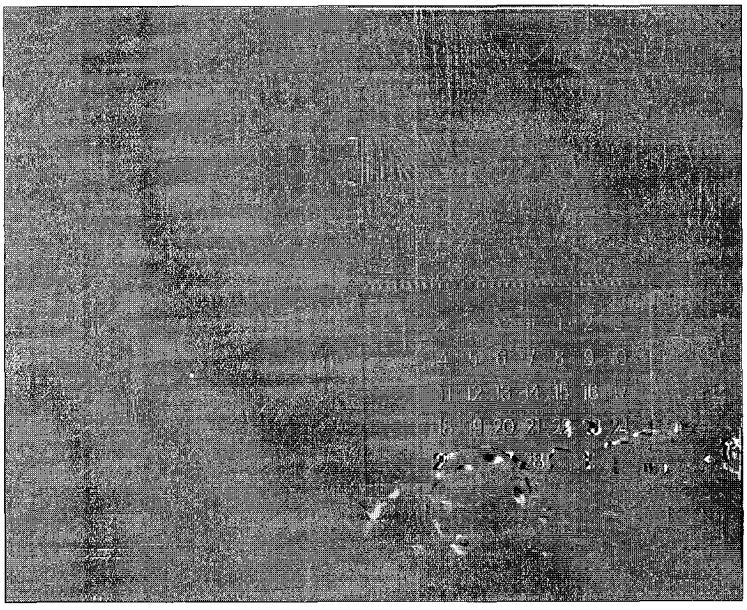
Experiments reported in [Iu92] and [Giro93] show that increasing the accuracy to ¼ pixel accuracy does not improve the results as much as the improvement obtained by increasing the accuracy from 1 pixel to a ½ pixel.

In Figure 3.17 we can also see that a 3-taps temporal interpolation filter improves the result if we compare interpolator ② with ③. The filter cut-off frequency β of the longer interpolation filter better approximates the required value of $\frac{1}{4}\pi$, again improving the overall filter characteristics. The stretched difference image (Figure 3.18(c)) also shows an improvement compared to interpolator ②.

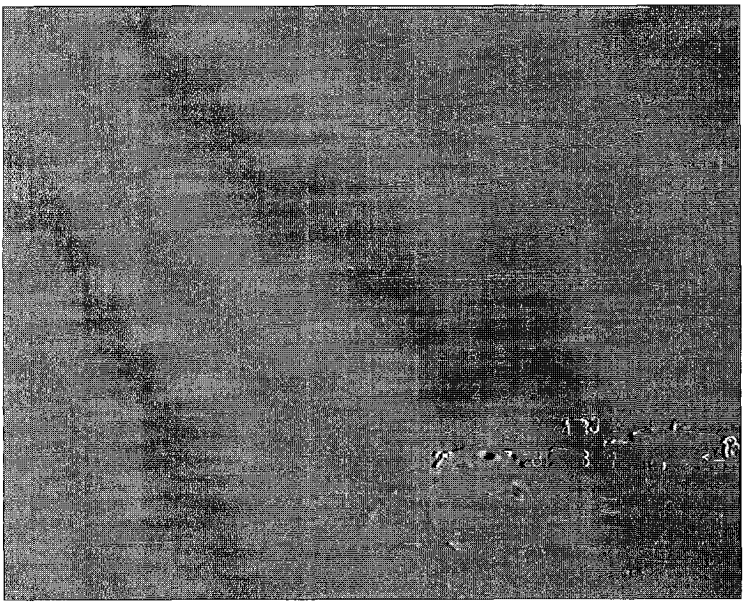


(a)

Figure 3.18: (continued on next page).



(b)



(c)

Figure 3.18: Stretched difference images for the second image of the MOBILE sequence for interpolator: (a) ①, (b) ② and (c) ③.

We see that the loss in SNR because of using $-\mathbf{d}_b(\mathbf{x},t)$ instead of $\mathbf{d}_f(\mathbf{x},t)$ is not large by comparing the results of interpolator ③ with interpolator ④. As the *MOBILE* sequences does not contain accelerating motion, the motion field is more or less stationary in the temporal direction. As can be expected, interpolator ⑤ gives the best result at the expense of extra side information. The gain is not large because the test sequence does not contain many uncovered and covered regions.

Figure 3.18 also serves as a confirmation of the kind of errors encountered after motion compensated interpolation. Namely, in regions where the motion is estimated accurately, the interpolation errors are more or less uncorrelated with the image content. However, in regions which are not accurately interpolated the errors are dependent on the image content and also more visible. Most of these regions correspond with areas which are occluded.

3.4.2 Nonlinear spatial interpolation

To clearly show the various artifacts at edges of a specific orientation, another detail of the *MOBILE* sequence is used. This image is shown in Figure 3.19. We see that it contains edges of different orientations. A quincunx subsampling lattice is used with $L = M = 1$. Four different interpolation methods are used. These are listed in Table 3.3 together with the numerical interpolation results. Figure 3.20 shows the interpolated images and the stretched difference images. Only, prior to interpolation with the FIR filter, a prefilter is used. In this case the same prefilter and interpolation filter are used as in Chapter 2.

Figure 3.19: Detail of MOBILE sequence.

Table 3.3: Average SNR and MSE of the interpolation results for the MOBILE sequence.

Interpolation Method	*SNR (dB)*	*MSE*
FIR filter	19.5	18.8
3-point horizontal median (Equation (3.33))	17.0	33.5
4-point median (Equation (3.34))	19.2	20.2
Horizontal and Vertical FMH filter (Equation (3.36))	19.9	17.4

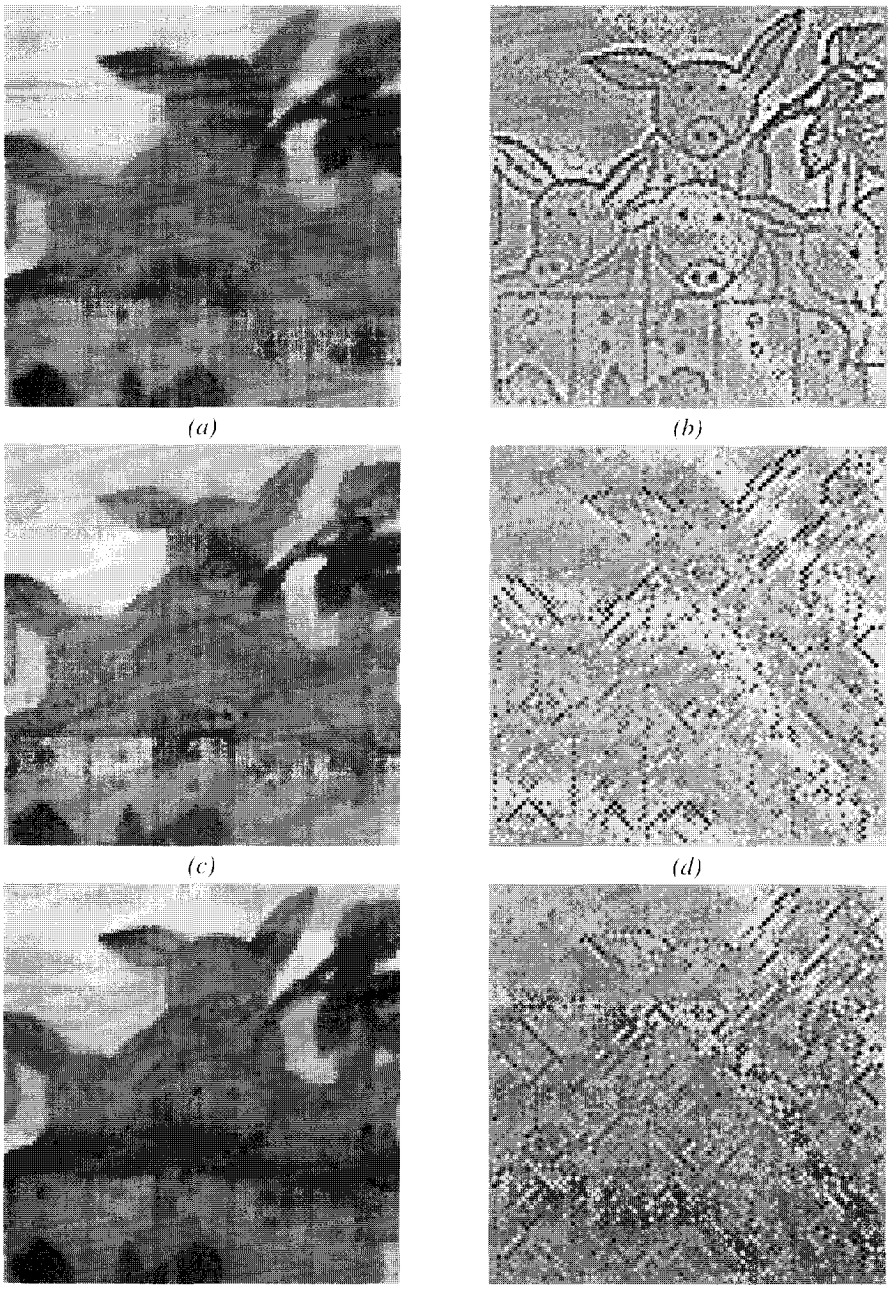
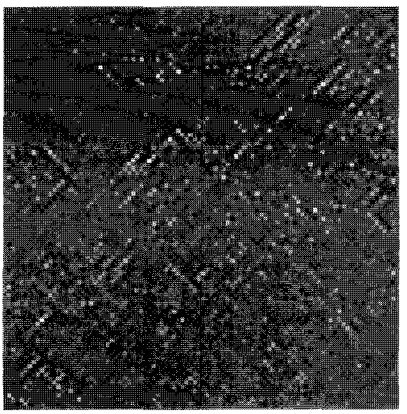


Figure 3.20: Interpolation results and stretched difference: (a,b) FIR filter, (c,d) 3-point horizontal median filter, (e,f) 4-point median filter.

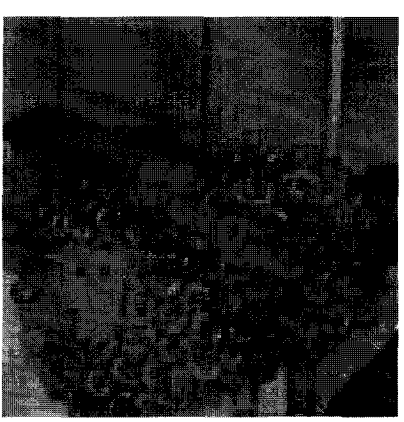
(g)



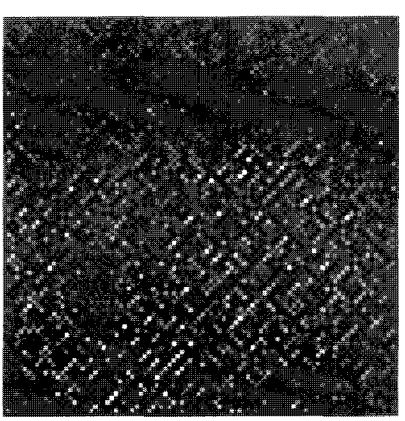
(h)

Figure 3.20 (continued): Interpolation results and stretched difference: (g,h) combined horizontal and vertical FMH filter.

The image in Figure 3.20(a) is obtained with an FIR filter. The difference image, shown in Figure 3.20(b) contains the edge information which is lost because of the prefilter and interpolation filter. A sharper image is achieved with a 3-point median filter (Figure 3.20(c)). The difference image in Figure 3.20(d) shows an improvement for horizontal edges in the image. The results for the 4-point median filter (Figure 3.20(e,f)) and the combined FMH filter (Figure 3.20(g,h) do not differ much from each other. Both visually and numerically there is a slight preference for the combined FMH filter. Most of the horizontal and vertical details are recovered. The information about the diagonal edges is lost because of the aliasing. The aliasing also causes the high frequency noise to fold back to the low frequencies, making it more visible. Some of the edges are locally interrupted because the median filter uses information from the wrong side of the edge. This artifact causes a temporal inconsistency of the edges. If an edge is interrupted in one image and properly reconstructed in the next image the interrupted edge is a more annoying visual artifact.



(a)



(b)

Figure 3.21: Interpolation results (a) and stretched difference (b) using a combined horizontal and vertical FMH-filter.

The result of the FMH filter for the standard detail image is shown in Figure 3.20. The artifacts are similar to the artifacts described for the previous image. Because the image contains more diagonal details, the overall quality is worse.

3.4.3 Motion compensated nonlinear spatial interpolation

In this section, the results obtained with a motion compensated nonlinear interpolation filters are presented. Again a quincunx subsampling lattice is used. First the following two motion compensated filters are used:

MC-MED4: This is the filter from Equation (3.37) where the motion compensated pixel value is combined with the 4-point median filter from Equation (3.34).

MC-FMH: This filter is formed by combining the pixels involved in the combined horizontal and vertical FMH filter (Equation (3.36)) with the motion compensated pixel into the median filter.

The results are compared with the results of spatial nonlinear interpolation filters, namely the 4-point median filter (*MED4*) and the combined horizontal and vertical FMH filter (*FMH*). The first image in the sequence is interpolated with the spatial interpolation filter. A vector field estimated with pixel accuracy is used in all the experiments.

Figure 3.22 shows the results of the experiments. The result of the *MC-FMH* filter is always better for all the images than the corresponding spatial FMH interpolation filter. Also for the second half of the sequence where the motion becomes more violent the *MC-FMH* filter is able to maintain a better quality. The *MC-MED4* filter is not able to cope with the increased motion activity and the result becomes worse than the spatial interpolation filter. This is because with the *MC-MED4* filter, four spatial pixels and one temporal pixel are involved, whereas with the *MC-FMH* filter six spatial and one temporal pixel are involved. Hence the ratio between temporal and spatial pixels is smaller for the *MC-FMH* filter and the filter becomes less sensitive to the amount of motion.

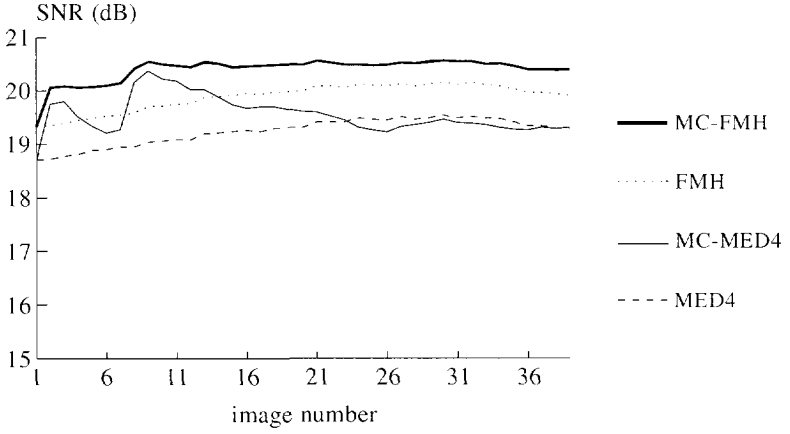


Figure 3.22: Motion compensated nonlinear interpolation filter compared with spatial nonlinear interpolation filters for the MOBILE sequence.

Next, two other motion compensated nonlinear interpolation filters are used:

MC-WFMH: This filter is similar to the *MC-FMH* filter but the motion compensated pixel is weighted with a factor three.

MC-AWFMH: This filter is a combination of the *MC-FMH* filter and the *MC-WFMH* filter. If the average absolute *DFD* is larger than four the *MC-FMH* filter is used and the *MC-WFMH* is used if the average absolute *DFD* is less than or equal to four.

The results for these filters are shown in Figure 3.23. The *MC-WFMH* filter outperforms the *MC-FMH* filter for images which are predicted well but the performance deteriorates when the motion becomes more complex. We see that *MC-AWFMH* provides a good compromise for all the images. Also shown in the figure is the switching behavior of the *MC-AWFMH* filter between the two alternatives. The *MC-WFMH* filter is chosen relatively more often in regions where the performance is better than the *MC-FMH* filter.

The dominating artifact in the interpolated image is again the aliasing because of the lack of a prefilter. No artifacts are visible which are caused by incorrect motion vectors or covered and uncovered areas. The median operator is aware that the predicted pixel from the previous image is not reliable and properly switches to spatial interpolation.

The conclusion that can be drawn is that the quality gain of using nonlinear interpolation techniques instead of linear interpolation is small. Because of the aliasing, it is not possible to design a nonlinear interpolation filter which performs equally well for all edge directions. Thus the simple fixed lattice subsampling system cannot be improved significantly. Therefore in the next two chapters we look at more advanced systems which use an adaptive instead of a fixed lattice.

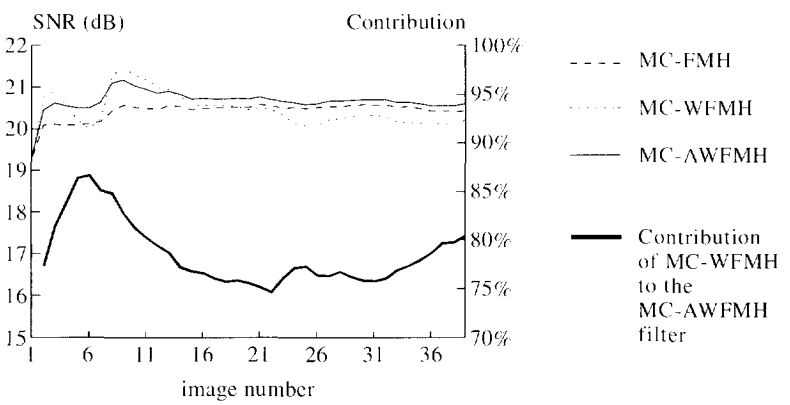


Figure 3.23: Results of the MC-WFMH filter and MC-AWFMH filter compared with the MC-FMH for the MOBILE sequence. Also shown is the switching behavior of the MC-AWFMH filter.

Chapter 4

SPATIALLY ADAPTIVE SUBSAMPLING

In a spatially adaptive subsampling scheme the local spatial sampling frequency of the image is adapted to the local image contents [Tani88] [Kish88]. A larger number of samples is used to make up the detailed regions of the image, and fewer samples are used to make up the image regions with low spatial activity. The specific shape of the sampling lattice is also adapted to the local spatial frequencies. By using this technique, at the same data reduction factor, an improvement of the quality over fixed lattice subsampling can be obtained at the expense of a greater complexity. A problem which arises is the division of the samples over the different regions of the image at the encoder side. This is the *mode allocation* problem. Another problem is the interpolation of the irregularly distributed samples at the decoder side.

In this chapter we start with a theoretical analysis of spatially adaptive subsampling. The quality improvement of spatially adaptive subsampling over fixed lattice subsampling is investigated. After that the practical implementation of a spatially adaptive subsampling scheme is discussed. Two different approaches to the interpolation are presented in separate sections: the least-squares interpolation and the hierarchical interpolation. Then the mode allocation, which is the most essential part of a spatially adaptive subsampling scheme, is discussed. We show that the mode allocation can only be solved analytically if only two different sampling lattices are used. To increase the effectiveness of spatially adaptive subsampling, the number of different lattices should be greater than two. Therefore two algorithms are discussed which enable an increase of the number of different lattices.

Next an extension of the algorithm is presented which also uses temporal adaptivity [Belf93b] [Belf94]. The same principles on which the spatial adaptivity is based can be used in the temporal direction. If the temporal activity is low, a lower temporal sampling rate is required then in the case of violent motion. The temporal adaptivity can be exploited further by using motion compensation.

This chapter is concluded by reporting several experiment results. The various mode allocation schemes and interpolation methods are illustrated. Also the effect of the different system parameters is investigated. The motion compensated system is compared with a system with only spatially adaptive subsampling.

4.1 Theoretical analysis of spatially adaptive subsampling

The simple spatially adaptive coding scheme shown in Figure 4.1 is used to examine the advantage of spatially adaptive subsampling over fixed lattice subsampling. The aim is to

show how a spatially adaptive subsampling scheme can take advantage of a non-stationary input signal. For the sake of simplicity the analysis is done in one dimension using the results derived for the rate-distortion function discussed in Chapter 2.

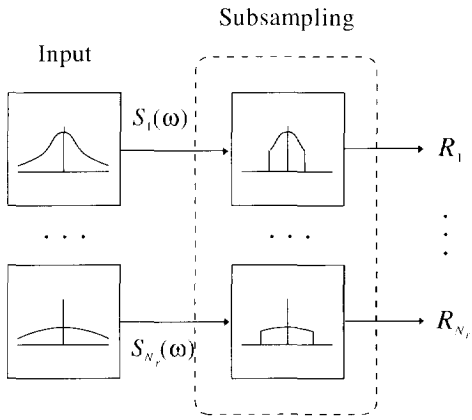


Figure 4.1: A simple spatially adaptive subsampling coding scheme.

To account for the non-stationary nature of the input signal, we consider a one-dimensional signal that is subdivided in N_r distinct equally sized regions (e.g. lines in an image) with different statistics, so that the power spectral density functions $S_k(\omega)$ differ from each other. We assume that the power spectral density functions exist and are monotonically decreasing, so the corresponding rate-distortion function $D_k(R_k)$ of each region is convex. All regions have the same number of samples and all samples are fed into a single coding scheme. The coding scheme consists of parallel subsampling operations, where each individual operation is applied to one region. The subsampling factor of each region varies. In Section 2.3 we have seen that the bit rate varies proportionally with the subsampling factor. The regions are also interpolated independently from each other.

The total relative bit rate R_t is given by the average of the relative bit rate R_k of each region:

$$R_t = \frac{1}{N_r} \sum_{k=1}^{N_r} R_k \ , \quad (R_k \geq 0) \tag{4.1}$$

Because the regions are interpolated independently of each other, the total mean square error distortion $D_t^A(R_t)$ after coding is given by:

$$D_t{}^A(R_t) = \frac{1}{N_r} \sum_{k=1}^{N_r} D_k(R_k) \tag{4.2}$$

The aim of the bit allocation is now to minimize the total distortion as defined in (4.2) under the constraint of Equation (4.1). We define the *optimal* bit allocation as the point $(R_1,\ldots,R_{N_r})$

in the N_r-dimensional space for which the total distortion $D_t^A(R_t)$ is minimal. We now show how we can express R_k as a function of $S_k(\omega)$ for an optimal bit allocation.

The first part of the proof is similar to what was done for subband coding in [Pear91]. We introduce the N_r-dimensional vector $\mathbf{p} = (p_1,\ldots,p_{N_r})$ with the elements p_k equal to:

$$p_k = \frac{R_k}{N_r R_t} \tag{4.3}$$

Each p_i represents the fraction of the contribution of each region k to the total average bit rate. Substituting this relation into (4.1) gives

$$\sum_{k=1}^{N_r} p_k = 1, \quad (p_k \geq 0) \tag{4.4}$$

If we substitute Equation (4.3) into (4.2) then (4.2) can be rewritten as

$$D_t{}^A(\mathbf{p}) = \frac{1}{N_r} \sum_{k=1}^{N_r} D_k(p_k N_r R_t) \tag{4.5}$$

where D_t^A is no longer a function of R_t but of $\mathbf{p}$.

The *Kuhn-Tucker* theorem states that if f($\mathbf{p}$) is a convex function over the convex region defined by $\mathbf{p} = (p_1,\ldots,p_{N_r})$ ($p_k \geq 0$, $\sum_{k=1}^{N_r} p_k = 1$), then for some constant θ the conditions

$$\frac{\partial \mathrm{f}(\mathbf{p})}{\partial p_k} = \begin{cases} \geq \theta, & p_k = 0 \\ \\ = \theta, & p_k > 0 \end{cases}, \quad k \in \{1,\ldots,N_r\} \tag{4.6}$$

are necessary and sufficient for a minimum point of f($\mathbf{p}$). A proof of this theorem can be found in [Gall68]. The constant θ is the maximum value of the partial derivatives of f($\mathbf{p}$) at the minimum point.

This theorem can be applied to Equation (4.5) to obtain the minimum of the distortion $D_t^A(\mathbf{p})$ because with Equation (4.4) the constraints for the p_k's are satisfied. We now get the following sets of relations:

$$\frac{\partial}{\partial p_k}\left(\frac{1}{N_r}\sum_{k=1}^{N_r} D_k(p_k N_r R_t)\right) = \begin{cases} \geq \theta, & p_k = 0 \\ \\ = \theta, & p_k > 0 \end{cases}, \quad k \in \{1,\ldots,N_r\} \tag{4.7}$$

which can be simplified to

$$\frac{1}{N_r}\frac{\partial D_k(p_k N_r R_t)}{\partial p_k} = \begin{cases} \geq \theta, & p_k = 0 \\ \\ = \theta, & p_k > 0 \end{cases}, \quad k \in \{1,\ldots,N_r\} \tag{4.8}$$

Solving this equation for the R_k's gives

$$\frac{\partial D_k(R_k)}{\partial R_k} = \begin{cases} \geq \dfrac{\theta}{R_t}, & R_k = 0 \\ \\ = \dfrac{\theta}{R_t}, & R_k > 0 \end{cases}, \quad k \in \{1,\ldots,N_r\} \tag{4.9}$$

Note that the threshold θ is negative because the rate-distortion function is monotonically decreasing. From Equation (2.25) we know that the first derivative of the rate-distortion function is equal to $-S_k(R_k\pi)$, so the previous equation can be rewritten to:

$$S_k(\pi R_k) = \begin{cases} \leq -\dfrac{\theta}{R_t}, & R_k = 0 \\ \\ = -\dfrac{\theta}{R_t}, & R_k > 0 \end{cases}, \quad k \in \{1,\ldots,N_r\} \tag{4.10}$$

This relation determines the number of bits assigned to each region at the optimal bit allocation. The interpretation of this relation is as follows. If the power density function is below the threshold $-\theta/R_t$ then the necessary part of the Kuhn-Tucker states that no bits are assigned to this region and this region is *not* coded. Otherwise, if the power density function intersects with the threshold, that region is coded. The bit rate assigned to this source is proportional with the intersection value on the ω-axis.

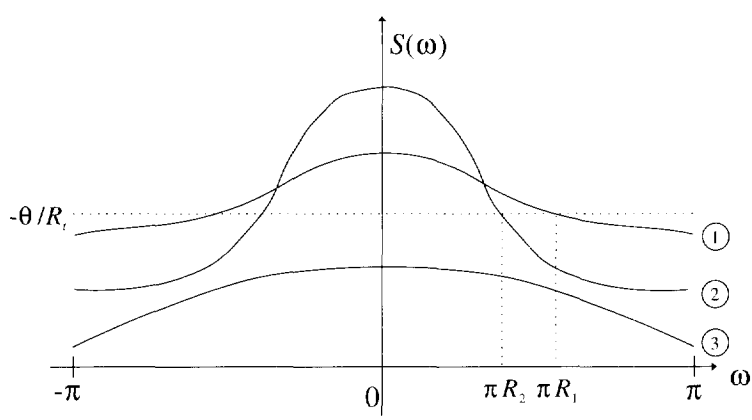


Figure 4.2: Physical interpretation of the optimal allocation. Shown are the power spectral density functions of three different regions fed into one coding scheme.

An illustration of this is given in Figure 4.2. In Figure 4.2 region 3 is not coded because its power spectral density function is below the threshold. At this specific bit rate the energy contribution of this region is low, hence omitting it has little effect on the total distortion. The intersection of region 1 is at a greater distance from the origin than region 2, so more bits are assigned to region 1.

The conclusion is that if the available bits are optimally divided over the different regions instead of assigning a fixed amount to each region, the total distortion reaches a global minimum. As the bit rate is directly coupled with the subsampling factor, this is the main principle on which spatially adaptive subsampling is based. Different subsampling factors are used for different regions in the image depending on the shape of the local power density function.

If fixed lattice subsampling is used, the bit rate R_k of each source is a priori fixed to

$$R_k = \frac{1}{N_r} R_t \tag{4.11}$$

so the total distortion D_t^F is equal to

$$D_t^F(R_t) = \frac{1}{N_r} \sum_{k=1}^{N_r} D_k(\frac{R_t}{N_r}) \tag{4.12}$$

By using the same line of reasoning and using the necessary part of the *Kuhn-Tucker* theorem, the conclusion is that fixed subsampling is only optimal if

$$S_k(\pi \frac{R_t}{N_r}) = S_l(\pi \frac{R_t}{N_r}), \quad \forall k,l \in \{1,\ldots,N_r\} \tag{4.13}$$

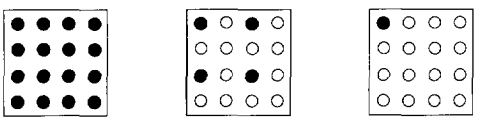
Only in this case the intersection of each source with the threshold is identical. However in most of the practical situations the power spectral densities differ and the performance of spatially adaptive subsampling is therefore better than the performance of non-adaptive subsampling. Despite the fact that the above analysis is one-dimensional, the same conclusion holds for the two-dimensional case.

4.2 Spatially adaptive subsampling in practice

Now that we have shown that theoretically spatially adaptive subsampling performs better than fixed lattice subsampling, we consider its practical implementation. The ideal case would be to segment the image into homogeneous regions which require the same spatial sampling frequency and sample each region according to this frequency. Such a solution would require a detailed analysis of the image and a large amount of side information would be needed to transmit the shape of the regions. Therefore we subdivide the image into square blocks and within each block one specific sampling lattice is used. The size of the blocks is an important system parameter. If large blocks are chosen the amount of side information is low, but the ability to adapt to the local spatial frequency contents would be insufficient. Small blocks cause a large overhead but warrant a better adaptation.

Another consideration in a practical system is the sampling lattice which is used for each block. Ideally each block should be sampled with a sampling lattice optimally suited to that particular block. The algorithm described in [Cort93], which gives all the possible sampling lattices given a specific compression factor, can be used for this purpose. Again this implies a large amount of side information. Therefore only a limited set of possible sampling lattices is used here. This set is designed in such a way that it gives a good coverage of the range of all

necessary spatial frequencies. Following [Tani88] we call each specific sampling lattice a *mode*. In Figure 4.3 some examples of modes are given with different data reduction factors. For instance in mode 3 only one pixel is kept out of 16 pixels, giving a data reduction factor of 16. Mode 1 can be used for highly detailed regions whereas mode 3 can be used for areas with a slowly varying luminance. The number of possible modes is affected by the block size, because for decreasing block size the number of possible sampling lattices within the block decreases as well. The minimum bit rate is achieved if the mode with the lowest sampling density is assigned to all blocks. In this example the minimum relative bit rate is equal to 1/16.



Mode 1 Mode 2 Mode 3

Figure 4.3: Examples of different modes. The solid dots are the pixels which are transmitted.

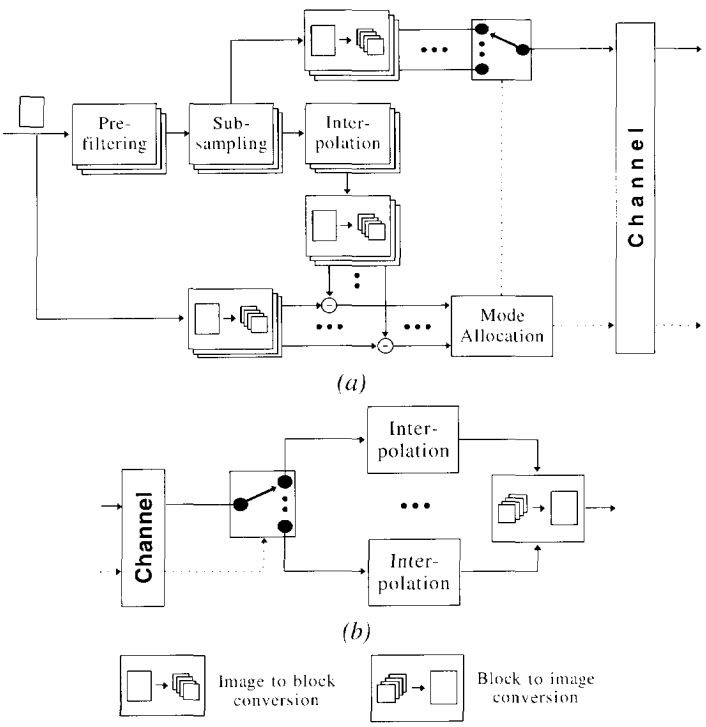


Figure 4.4: Spatially adaptive coding scheme: (a) encoder, (b) decoder.

The overall encoding scheme is shown in Figure 4.4(a). The entire input image is first prefiltered and subsampled for each mode. The subsampled images are fed into an

interpolation module. After that the mean square difference with the original image is evaluated for each image block and each mode. These error values are used in the mode allocation which assigns a particular mode to each block. Based on the mode allocation for each block, one mode is selected and the corresponding samples are transmitted together with the mode information. At the receiver the blocks are reconstructed again, where each mode requires a different interpolation scheme (Figure 4.4(b)). Next the different blocks are combined to form the decoded image.

The prefiltering operation in Figure 4.4(a) needs some special attention. The aim in a spatially adaptive subsampling scheme is to adapt the sampling lattice to the local image contents. If a block containing high spatial frequency components is sampled with only a few samples without any prefiltering, then the aliasing causes large errors. These large errors make it easy to detect this situation, and as a corrective action the mode allocation can assign more samples to this block. If a prefilter is used, the severity of assigning a wrong number of samples to a block decreases because of the absence of aliasing errors. Now only the loss of resolution (the outband distortion) is used to detect the presence of high spatial frequency components. This argument leads to the conclusion that a prefilter is not desirable.

There are, however, two other arguments in favor of a prefilter. First, the action of assigning more samples to a block if there are any aliasing errors, implies that there are still samples available. If however the desired bit rate is low, then it may not be possible to assign more samples to the block. In this case the decoded image contains aliasing errors. This can be prevented by the use of a prefilter. A second argument in favor of a prefilter is the fact that if the aliasing errors are small, then these errors may not be detected by the mode allocation. Especially the high frequency noise components cause this problem. Now the decoded image also contains aliasing errors, which in the case of noise is visible as noise with a low spatial frequency. In this situation a prefilter is also necessary. In practice a compromise can be made by using a spatial prefilter with only a few filter taps. A short filter does not totally eliminate the aliasing errors but only reduces them.

4.3 Interpolation

Figure 4.4(a) and 4.4(b) both contain a module that involves interpolation. The first interpolation is done before the error computation block. To do an assignment of the different modes to each block, the distortion of each mode for all blocks must be evaluated. This can be done using either an *a priori* or an *a posteriori* estimator. An a priori estimation can be done by defining some kind of *detail* parameter (see [Elme88]) and using this value as an indicator of the expected distortion. This method will not be discussed further because of its heuristic nature. An a posteriori estimation is based on the difference between the original image and the reconstructed image, which requires interpolation at the encoder side. At the receiver, interpolation is of course also necessary to reconstruct the subsampled image.

The problem of the interpolation at the decoder is that the subsampled image is only available on a highly irregular sampling lattice because of the mix of modes used to represent the

image. Therefore straightforward filtering methods cannot be used. A solution to this problem is the introduction of a common intermediate sampling lattice into which each mode can be converted. From this lattice the boundary pixels for the different modes can be deduced and interpolation is possible. The problem of this method is that no optimal use is made of the specific structure of the modes. Therefore next two alternatives are presented, namely the least-squares interpolation and the hierarchical interpolation [Hesp93].

4.3.1 Least-squares interpolation

The least-squares interpolation method is an interpolation scheme which involves only the pixels within each block. The pixels of each block can be represented by a two-dimensional polynomial. If the order of the polynomial increases the estimation becomes more accurate, but the number of coefficients of the polynomial also increases. In [Will91] this approach was described for the coding of an image sequence at very low bit rate. The coefficients c_{kl} of the interpolating polynomial were transmitted instead of the image intensities. Here this method is adapted for a subsampling application. The different modes are now formed by constructing polynomials of different orders. For instance mode 2 from Figure 4.3 can be represented by the polynomial

$$f(x, y) = c_{00} + c_{01}x + c_{10}y + c_{11}xy \tag{4.14}$$

For each mode the coefficients c_{kl} can be estimated at the encoder using a least-mean-squares estimation. With this method the interpolation error is minimized. In a subsampling application it is not the polynomial coefficients that are transmitted but image intensities. Thus the polynomial coefficients have to be converted to image intensities. The coefficients can be transformed to intensities using the fact that a set of $P \times Q$ samples uniquely prescribes the coefficients c_{kl} of the two-dimensional polynomial $f(x,y)$ given by

$$f(x, y) = \sum_{k=0}^{P-1} \sum_{l=0}^{Q-1} c_{kl} x^l y^k \tag{4.15}$$

Therefore the image intensities can be obtained by evaluating the estimated polynomial $f(x,y)$ at regularly spaced positions. At the decoder, the missing pixels are obtained by transforming the intensities back to a polynomial, followed by an evaluation of the polynomial on a full sampling lattice. Because the polynomial coefficients are real numbers and the pixel values are integer numbers, there may be some truncation errors, causing a slight difference between the polynomial estimated at the encoder and the decoder.

The method is first illustrated by a one-dimensional example. The 10 input samples $u(t)$ are given in Figure 4.5(a). In Figure 4.5(b) a fourth-order polynomial is estimated based on these input samples. This polynomial can be described by five coefficients. Because we want to transmit image intensities instead of polynomial coefficients, the estimated polynomial is evaluated at five positions (Figure 4.5(c)). These sample values are transmitted. Thus effectively the number of samples describing the original input samples is halved. At the decoder, the fourth-order polynomial is estimated again, based on the five samples which are transmitted. The missing samples are obtained by evaluating the polynomial at the missing positions (Figure 4.5(d)).

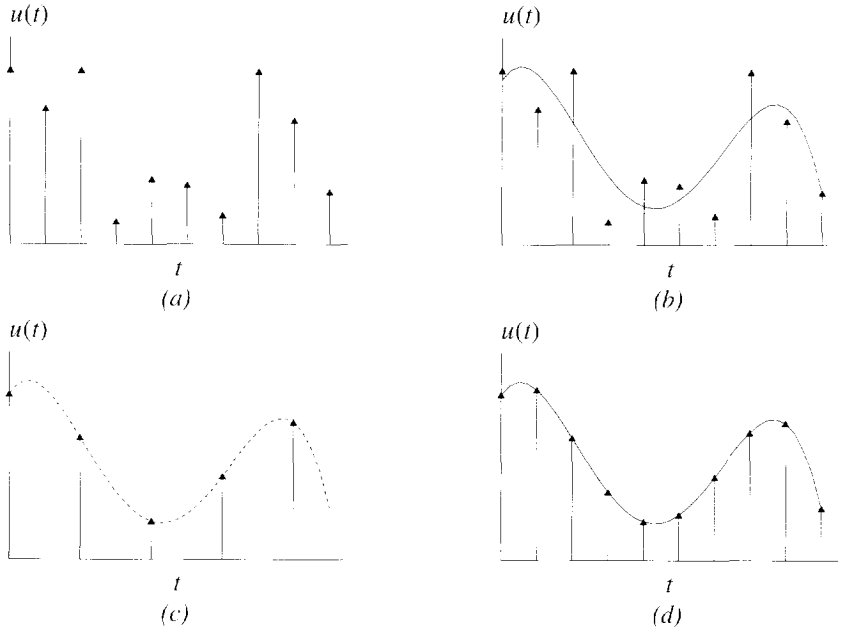


Figure 4.5: Example of least-squares interpolation: (a) Input samples, (b) Estimation of the polynomial based on the input samples, (c) Conversion of polynomial to samples, (d) interpolation at the decoder.

Next we look at how the coefficients of the polynomial are estimated. We assume that a block with dimensions $P_O \times Q_O$ is subsampled to a block with dimensions $P_{SS} \times Q_{SS}$. To represent this block, a polynomial with order Q_{SS}-1 in the x direction and order P_{SS}-1 in the y direction is necessary. This polynomial is estimated by constructing the polynomial matrix $\mathbf{A}$ with dimensions $P_O Q_O \times P_{SS} Q_{SS}$ where each row of the matrix consists of the evaluation of the monomials $x^l y^k$ ($1 \le l \le P_{SS}$, $1 \le k \le Q_{SS}$) for each position (x_O, y_O) ($1 \le x_O \le Q_O$, $1 \le y_O \le P_O$) in the original block. The image intensities $i(x_O, y_O)$ of the original block are arranged in the column vector $\mathbf{i}$. The coefficient vector $\mathbf{c}$ representing the coefficients c_{kl} of $f(x,y)$ are now obtained by solving the matrix equation (e.g. [Stra76]):

$$\mathbf{c} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{i} \tag{4.16}$$

The polynomial prescribed by these coefficients can be used to interpolate the missing pixels after subsampling at the encoder and the decoder side.

The advantage of this method is that the interpolation result at the decoder is the same as the interpolation result at the encoder side because no information from neighboring blocks is necessary. However this also implies that the interpolation is not continuous at block boundaries, thus introducing annoying blocking artifacts in regions with slowly varying intensities.

4.3.2 Hierarchical interpolation

The blocking artifacts introduced by least-squares interpolation can be reduced by using larger interpolation kernels which filter across block boundaries. At the encoder this is not a problem because all the modes for all the blocks are available. However, at the decoder the pixels necessary for the interpolation may not be available if a neighboring block is sampled with a different mode. To avoid such problems, a hierarchical set of modes has to be used. In a hierarchical set, mode n+1 is always a subset of mode n (we assume that the modes are ordered by decreasing sampling density):

$$\{\mathbf{x}|\mathbf{x} \in \text{mode}_{k+1}\} \subset \{\mathbf{x}|\mathbf{x} \in \text{mode}_k\}, \; k \in \{1,\ldots,M-1\} \tag{4.17}$$

where M represents the number of modes. For instance, the modes given in Figure 4.3 form a hierarchical set. The different modes can now be interpolated by low-pass filtering the previous mode in the hierarchy.

The interpolation process is illustrated in Figure 4.6. At the lowest level, the pixels corresponding with the pixels necessary for the mode with the smallest sampling density (mode 3) are combined into one image with a regular sampling lattice. This includes also the pixels from blocks which are subsampled with a different mode. This image can be interpolated because the required boundary pixels on this low resolution sampling lattice are always present in the neighboring blocks. Next, the pixels necessary for mode 2 are added to the interpolated image after which another interpolation follows. Finally the same procedure is repeated for mode 1. Hence the image is interpolated by building it up from the lower resolution levels and by adding the necessary pixels at each level of the hierarchy.

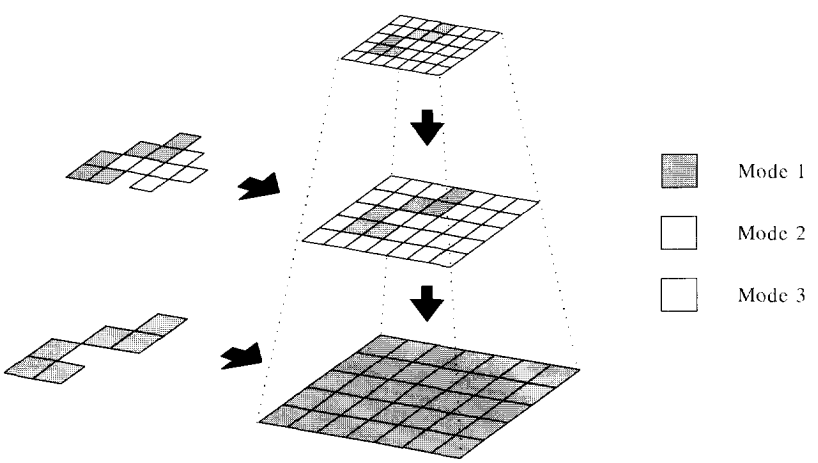


Figure 4.6: Hierarchical interpolation.

This interpolation scheme gives a poorer interpolation result at the decoder than the interpolation made at the encoder when all the required pixels were available. This is because an *estimation* of the boundary pixels is used, instead of the original boundary pixels. However, the blocking artifacts are no longer dominating. Note that the blocking is a visual

artifact and that an elimination of this artifact is not reflected in the MSE. The severity of this loss of performance is examined in Section 4.6.

4.4 Mode allocation

The mode allocation is of great significance as it influences the output quality considerably. The optimal mode allocation problem in a constant bit rate application can be formulated as follows:

> *In a system with M different modes and N_b blocks, assign a mode k ($k \in \{1,\ldots,M\}$) to each block l ($l \in \{1\ldots N_b\}$) in such a way that the total distortion is minimal under the constraint that the bit rate is less or equal to the desired bit rate.*

An alternative formulation in a constant *quality* application is to seek for the minimum bit rate which achieves a given distortion. As the allocation process is driven by the total distortion, the distortion measure used in the mode allocation process is an important factor. In the ideal case the system should be driven by the HVS. Because of the lack of an objective measure based on the HVS, the MSE is used as a distortion measurement.

The objective of minimizing the overall distortion implies that the total distortion can be computed at the time the mode allocation is carried out. As we saw in the previous section, this is not the case if hierarchical interpolation is used at the receiver. Using this interpolation scheme, boundary pixels of neighboring blocks are necessary and these are only available *after* the mode allocation. Hence the mode allocation is performed on an *estimate* of the distortion and not on the actual distortion. The effect of this approximation is investigated in Section 4.6.

A brute force search for the mode allocation tries all the modes on each block and takes the combination of modes with the smallest distortion. In a practical situation this is not feasible because the number of possible allocations is equal to M^{N_b}. As the number of blocks grows linearly, the number of allocations increases exponentially. In a system with 3 different modes and a block size of 8×8, an image with dimensions 720×576 pixels requires $\pm 10^{3091}$ iterations. Therefore in the following sections more sophisticated mode assignment schemes are discussed.

4.4.1 Two-mode allocation

In this section a mode allocation scheme based on an analytical solution is described. The equations which have to be satisfied in order to achieve a given relative bit rate R_t are:

$$\sum_{k=1}^{M} \alpha_k = 1 \wedge \sum_{k=1}^{M} \alpha_k R_k = R_t \tag{4.18}$$

where α_k is the fraction of the blocks which are assigned to mode k, and R_k is the relative bit rate associated with mode k ($R_k > R_{k+1}$). The first relation states that a mode is assigned to

every block and the second relation states that the weighted sum of the bit rates of all modes should equal the total relative bit rate R_k. This pair of equations with M unknowns can only be solved analytically if the number of modes is equal to two [Tani88] or if the rate-distortion functions $D_k(R_k)$ are known analytically [Huan63]. In the latter case the equations can be solved using the Lagrange-multiplier method, which leads to the result in Equation (4.9).

If only two modes are used, then Equation (4.18) becomes

$$\begin{aligned} \alpha_1 + \alpha_2 &= 1 \\ \alpha_1 R_1 + \alpha_2 R_2 &= R_t \end{aligned} \quad . \tag{4.19}$$

From these equations α_1 and α_2 can be solved:

$$\alpha_1 = \frac{R_t - R_2}{R_1 - R_2}, \quad \alpha_2 = \frac{R_1 - R_t}{R_1 - R_2} \tag{4.20}$$

To minimize the distortion, first mode 2 is assigned to all the blocks. Next mode 1 is assigned to a fraction α_1 of the blocks with the highest distortion. This leads to an optimal mode assignment in the sense of minimizing the overall distortion. In Section 4.4.3 we see that the two-mode allocation algorithm is a special case of the convex hull allocation algorithm. A system with only two modes is also not very useful.

4.4.2 Histogram mode allocation

In [Kish88] and [Saku90] a heuristic algorithm is presented to solve the mode allocation problem. A variation of this algorithm for different mode structures was also described in [Ashi88]. This algorithm is only suitable for three modes and is based on the two-dimensional histogram of the error differences between the different modes. The algorithm has some similarities with the *greedy* bit allocation algorithm used for vector quantization (e.g. [Gers92]). The principle of this algorithm is illustrated using Figure 4.7.

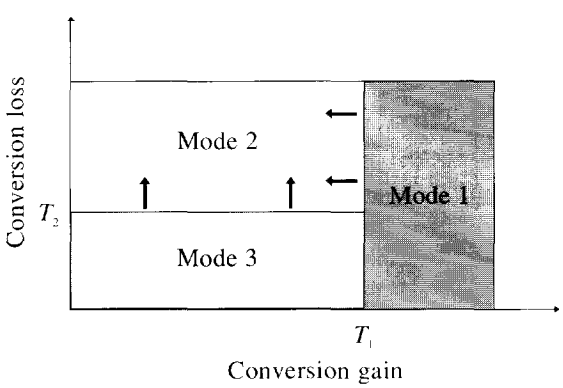


Figure 4.7: The principle of the histogram mode allocation.

First for all blocks the distortions D_1,D_2 and D_3 are computed which are caused by assigning respectively mode 1, 2 and 3 to each block. Next mode 2 is assigned to all blocks and for each blocks two values are computed:

- The *conversion gain*: the decrease of the distortion caused by the assignment of mode 1 to a block instead of mode 2 (D_2 - D_1).
- The *conversion loss*: the increase of the distortion caused by the assignment mode 3 to a block instead of mode 2 (D_3 - D_2).

These two values form a coordinate of a point in the two-dimensional histogram shown in Figure 4.7.

Now two thresholds T_1 and T_2 are introduced:

- If the conversion gain of a block is higher than T_1, mode 1 is assigned to that block. A decrease of this value lowers the distortion but increases the bit rate. The initial value of T_1 is equal to the maximum conversion gain.
- If the block is not assigned to mode 1 and the conversion loss is lower than T_2, mode 3 is assigned to that block. An increase of this value raises the distortion and lowers the bit rate. The initial value of T_2 is equal to zero.

Note that the initial values of these thresholds are consistent with the initial mode assignment.

If the bit rate using the initial mode assignment is higher than the required bit rate, the bit rate is decreased by increasing T_2 until the desired bit rate is reached. If the bit rate is lower than the required bit rate then T_1 is lowered instead. Now the total distortion is computed. Next mode 1 is assigned to the block with the highest conversion gain, increasing the bit rate but lowering the total distortion. The bit rate is again lowered by increasing T_2, hereby increasing the total distortion. If changing the two thresholds decreases the overall distortion, the process is repeated. The process stops when the overall distortion no longer decreases. An example of a histogram allocation result is given in Figure 4.8. The dots characterize each individual block of the image, and the shaded regions are prescribed by the final values of T_1 and T_2.

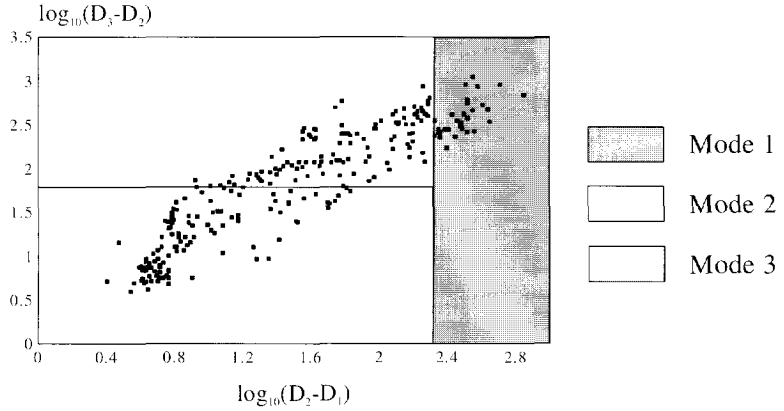


Figure 4.8: An example of histogram mode allocation. The algorithm is used on the LENA image with a block size of 16×16 pixels and a data reduction factor of 4.

The two disadvantages of this algorithm are:

- It is only suitable for a system with three different modes. This causes a poor adaptation to the local image content because of the limited size of the set of modes.
- Optimality is not guaranteed because starting from an initial allocation no systematic attempt is made to find a global minimum of the distortion and the algorithm stops in what may be a local minimum.

4.4.3 Convex hull mode allocation

Here we describe a mode allocation algorithm that allows for an arbitrary number of modes [Belf93b][Hesp93]. The algorithm is optimal if the blocks are interpolated independent of each other. It is based on the convex hull bit allocation scheme used for assigning quantizers to the subbands in a subband coding scheme [West88a]. Two translations have to be made to this algorithm:

- The subbands which have to be coded correspond to the blocks into which the image is subdivided.
- The quantizers correspond to the different modes.

The algorithm is based on the combined rate-distortion function $D_t^A(R_t)$ as defined in Section 4.1, which is formed by the lower convex hull in the plot of all possible mode allocations. All the rate-distortion functions $D_k(R_k)$ of the blocks contribute to this combined rate-distortion function. According to the definition of the rate-distortion function, the optimal mode allocation must lie on the combined rate-distortion function. By evaluating only the points which lie on the combined rate-distortion function, eventually the optimal mode allocation, given a required bit rate, must be reached.

First it is necessary to determine the convex hull of the rate-distortion function $D_k(R_k)$ of each block by removing the modes which cause the rate-distortion function to be non-convex. In Section 2.3 we have seen that the rate-distortion function is not always convex and that the convexity depends on the shape of the power spectral density function of the block. For an entire image this may be a valid assumption, but for a small block in an image this no longer holds in general. Convexity is also not possible if different sampling lattices with the same sampling density are used. These lattices all give rise to the same bit rate but not necessarily to the same distortion.

For the optimal mode allocation, next the combined convex hull of all the blocks together must be determined. If each block is coded independently, then the total relative bit rate R_t is equal to the average of the relative bit rate of each block:

$$R_t = \frac{1}{N_b} \sum_{k=1}^{N_b} R_k \tag{4.21}$$

If the blocks are interpolated independent of each other, then the total distortion D_t is equal to

$$D_t{}^A(R_t) = \frac{1}{N_b} \sum_{k=1}^{N_b} D_k(R_k) \tag{4.22}$$

Now suppose that a mode with a relative bit rate R_{k1} is assigned to the block k giving a distortion D_{k1}. We assume that this point lies on the convex rate-distortion function of the block. Next we choose a new mode for this block with a higher relative bit rate R_{k2} and a new distortion D_{k2}. The slope of the rate-distortion function $D_k(R_k)$ in this point is equal to

$$slope_k = \frac{(D_{k2} - D_{k1})}{(R_{k2} - R_{k1})} \tag{4.23}$$

A change of the allocation of one block also affects the slope of the combined rate-distortion function. The slope of the combined rate-distortion can be computed with the Equations (4.21) and (4.22):

$$\begin{aligned} slope_t &= \frac{(D_1+\ldots+D_{k2}+\ldots+D_{N_b}) - (D_1+\ldots+D_{k1}+\ldots+D_{N_b})}{(R_1+\ldots+R_{k2}+\ldots+R_{N_b}) - (R_1+\ldots+R_{k1}+\ldots+R_{N_b})} \\ &= \frac{(D_{k2} - D_{k1})}{(R_{k2} - R_{k1})} = slope_k \end{aligned} \tag{4.24}$$

The conclusion is that the block with the smallest slope on the block rate-distortion function also gives the smallest slope on the combined rate-distortion function because the two slopes are equal. Now we can use the convexity of the rate-distortion function. If, starting from a point on the combined rate-distortion function, we choose the transition with the smallest slope, this new point must lie on the combined rate-distortion function because the slope of this function is monotonically increasing. The same line of reasoning can be applied to any subsequent transition. This gives us a method to combine the block rate-distortion functions of each block into a combined rate-distortion function.

The mode allocation algorithm starts by assigning to each block the mode with the highest distortion. This point gives the lowest bit rate and highest distortion so it is guaranteed that this point lies on the combined rate-distortion function. Starting from this point for each block the slope after assigning the next mode to this block is computed. The block with the lowest slope is assigned a new mode. Because of the convexity of the combined rate-distortion function this new mode allocation again lies on both the block rate-distortion function as the combined rate-distortion function. Starting from the new allocation the described procedure is repeated. The algorithm terminates when the desired bit rate is reached. Hence instead of evaluating all the possible mode allocations, only those which lie on the combined rate-distortion function are examined. This causes a drastic decrease of the computational load while still guaranteeing optimality.

The complexity of each iteration step is of order $\log(N_b)$, because it requires a search for a minimum in an ordered list. The maximum number of iterations is equal to $N_b \times M$ in the case were every block traverses through each mode. An example of the points which are evaluated is given in Figure 4.9. Also the points examined in a brute force allocation are shown. We see that the convex hull allocation follows the lower convex hull of all possible mode allocations. The brute force allocation requires 81 evaluations of the rate with the corresponding

distortion, whereas the convex hull allocation requires maximally 9 iterations depending on the required bit rate.

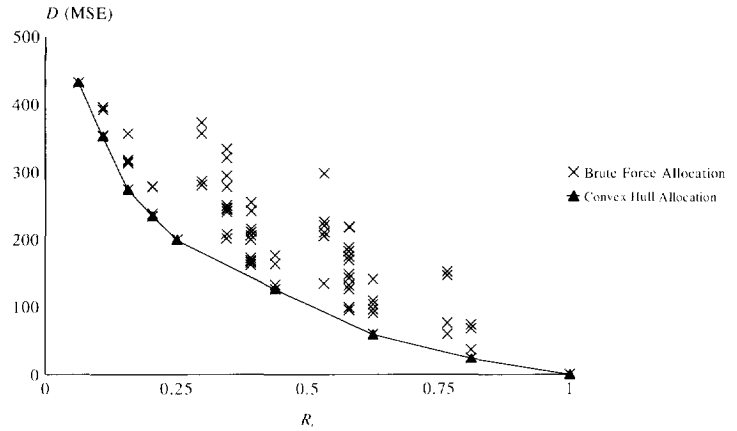


Figure 4.9: Evaluated points in a mode allocation with 4 blocks and 3 different modes using the LENA image.

4.5 Motion compensated spatially adaptive subsampling

Motion compensation has been used in many coding schemes to exploit the temporal correlation in image sequences (e.g. [Nino82], [Wang92]). If a correct motion compensated prediction is made then no additional information has to been transmitted besides the motion vectors. A spatially adaptive subsampling scheme can benefit from this property if for a particular region the spatial correlation is low but the temporal correlation is high. In this case spatially adaptive subsampling would require a lot of samples, whereas motion compensation requires none.

The overall system is shown in Figure 4.10. The shaded area contains the components which were also present in the spatially adaptive subsampling scheme shown in Figure 4.4(a). A motion compensated prediction of the actual image is made using the previously reconstructed image stored in the image memory and the motion vectors. The prediction error is determined by subtracting the original image from the motion compensated reconstructed image. The prediction error is fed together with the interpolation errors from the other modes into the mode allocation. The mode allocation now starts by assigning to each block a mode with zero pixels. The interpolation for this mode is based on the motion compensated reconstructed image. If the motion compensated prediction error is large, then only spatially subsampling is applied without using any motion information. Hence both the spatial and temporal correlation are exploited.

An advantage of this scheme is that it implicitly adjusts the threshold for the decision between temporal and spatial subsampling. If the desired bit rate is high then the algorithm is biased toward spatial subsampling, and if the bit rate is low temporal subsampling is

preferred. Another advantage of this scheme is that there is a mode which requires no additional pixels, so the maximal compression factor is no longer bounded by the block size, as is the case when only spatially adaptive subsampling is used.

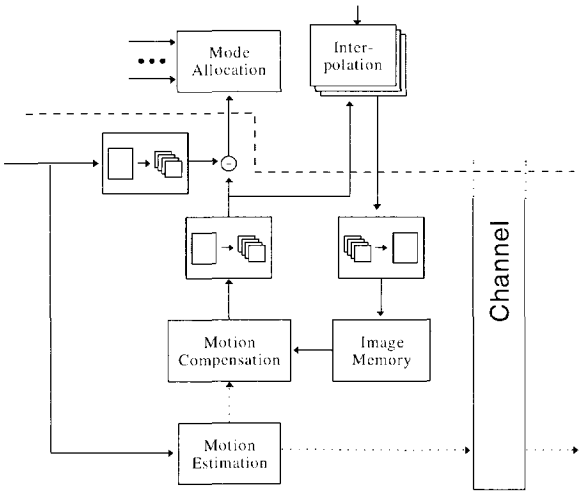


Figure 4.10: Motion compensated spatially adaptive subsampling scheme.

4.6 Experiment results

In this section, the effects of the different system parameters are investigated for spatially adaptive subsampling and motion compensated spatially adaptive subsampling.

4.6.1 Implementation details

Three different mode schemes are used in the experiments:

scheme1: This scheme is the mode structure as shown in (Figure 4.11(a)).

scheme2: The second scheme consists of five different modes (Figure 4.11(b)). Starting from a block containing all samples, each time quincunx subsampling is used on the previous block. This results in alternating orthogonal and quincunx sampling lattices.

scheme3: The third scheme is a modification of *scheme1* (Figure 4.11(c)). In between the existing modes, two extra modes are introduced. One mode is a horizontal subsampling of the previous mode and the second extra mode is a vertical subsampling of the previous mode. The number of modes is increased to seven. Note that this is *not* a hierarchical set, so for some modes an intermediate sampling lattice is necessary. For example mode 7 is first interpolated to mode 4, and from this image the necessary boundary pixels for mode 5 and 6 are obtained.

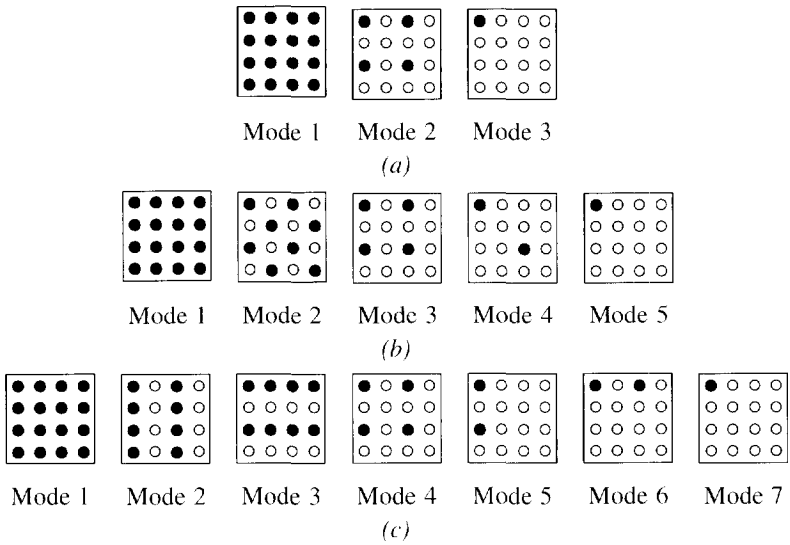


Figure 4.11: The different mode structures which are used (open dots are discarded pixels): (a) scheme 1, (b) scheme2, (b) scheme3.

As a reference in all experiments, unless otherwise stated, a system with the mode structure *scheme1*, a block size of 4 by 4 pixels and hierarchical interpolation is used. This system is referred to as the *standard* system. The convex hull mode allocation scheme is used driven by the mean square error. Side information is accounted for in all experiments by reserving $\log_2(M)$ bits per block if M different modes are used. To eliminate noise aliasing effects a prefilter with three taps is used. For the interpolation a maximally flat interpolation filter with seven taps is used.

4.6.2 Spatially adaptive subsampling

In Figures 4.12 and 4.13 a coded detail image of the first image of the *MOBILE* sequence is shown together with the difference image and the mode assignment. The number of transmitted pixels is halved with respect to the original number of pixels. The mode with the highest sampling density is shown as a black block and the mode with the lowest sampling density as a white block. It can be observed that the regions with a constant luminance are assigned a mode with a low sampling density whereas detailed regions are assigned a mode with a high sampling density.

The coding artifacts are the loss of *isolated* fine spatial detail (e.g. the fence posts in the background) and elimination of noise in flat regions. Large areas of high spatial activity are easily identified as such and are assigned a large number of pixels as far as the desired bit rate permits. However, if the bit rate is low, some of the spatial detail has to be sacrificed. This may result in spatial inconsistencies of the mode allocation. The corresponding artifacts are

isolated blocks with a low sampling density surrounded by blocks coded with a high sampling density. The coding results are also shown numerically in Table 4.1.

Table 4.1: Coding results of MOBILE detail.

Configuration	*Expected MSE*	*Actual MSE*
Standard	26.2	32.9
Standard but with histogram allocation	26.5	31.9
Standard but with scheme3	19.7	23.5
Standard but with scheme3 and least-squares interpolation	13.6	13.9

Histogram allocation and convex hull allocation are compared with each other in Figure 4.12. The difference between the two allocation schemes is small, which is also reflected by the small differences in the expected MSE. Note that there is a difference between the expected MSE and the actual MSE. The actual MSE for histogram allocation is even smaller than convex hull allocation. This is caused by the distortion introduced by hierarchical interpolation which is not accounted for in the mode assignment. The visual quality is not good because there are still some image structures visible in the difference image, indicating that significant parts of the image are distorted.

Hierarchical interpolation is compared with least-squares interpolation in Figure 4.13. The number of modes is increased to seven. As can be theoretically expected, least-squares interpolation gives a better performance. The image details are captured more efficiently than they are in hierarchical interpolation. There is still a slight difference between the expected MSE and the actual MSE because of truncation errors introduced by the conversion of the polynomial coefficients to the pixel intensities. The blocking artifacts introduced by least-squares interpolation are not visible in this example. The visual quality is significantly higher as most of the image details are no longer present in the difference image.

If we compare Figures 4.12 and 4.13 the real benefit of convex hull allocation compared to histogram allocation becomes clear. The performance of the histogram allocation does not differ much from the optimal allocation. However, using the convex allocation scheme it is also possible to benefit from the fact that increasing the number of modes has a positive influence on the performance.

4.6.3 Effect of the system parameters

The parameters of a spatially adaptive subsampling system are now discussed in more detail. This is done using the *LENA* image. The results of the different experiments are shown in Figure 4.14. To compare the experiments with each other, in all the experiments the solid line represents the standard configuration as defined in Section 4.6.1.

First, spatially adaptive subsampling is compared with fixed lattice subsampling for different bit rates in Figure 4.14(a). For the fixed lattice subsampling, either horizontal, vertical or quincunx subsampling is used and each time the sampling lattice which gave the best result is

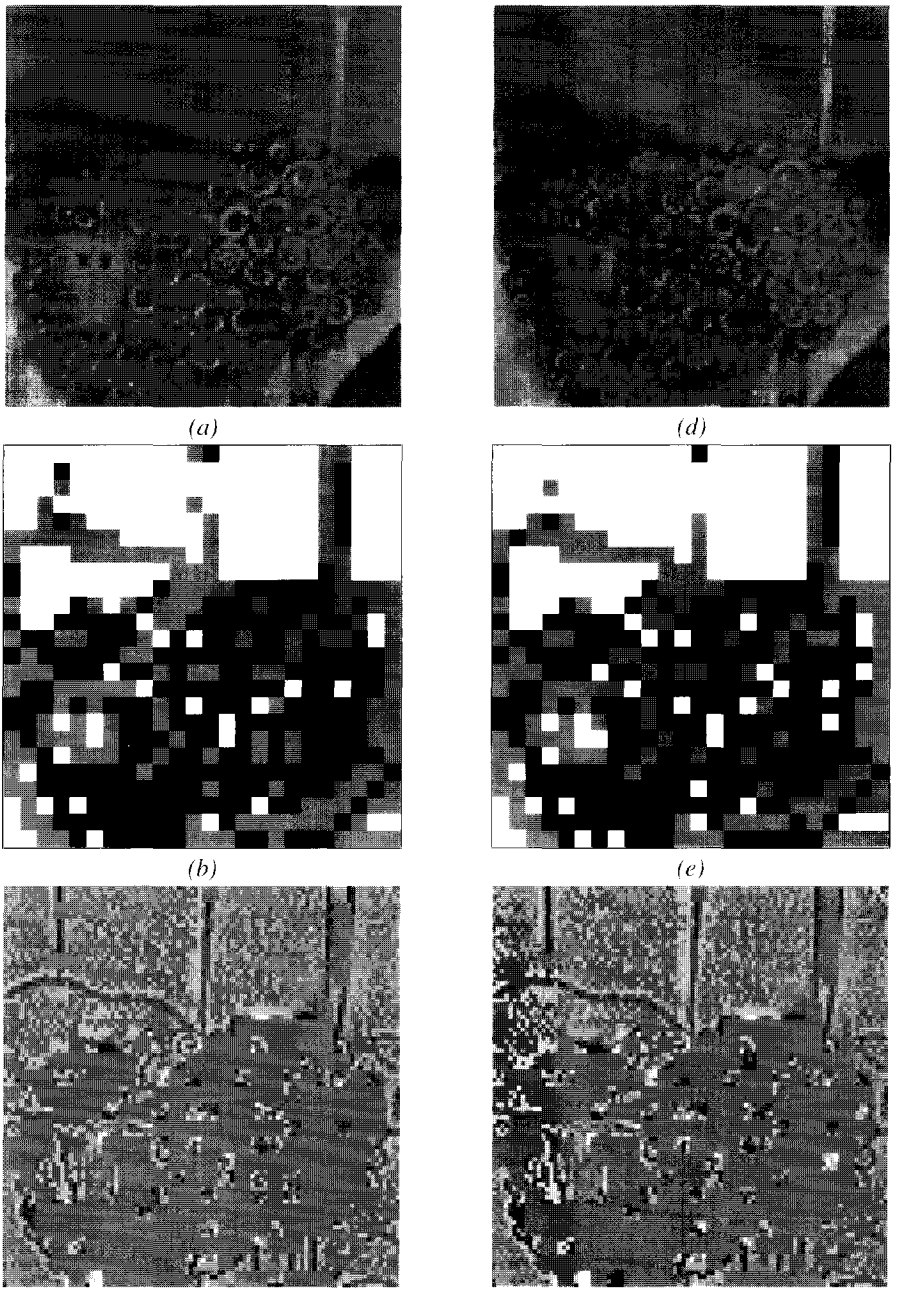


Figure 4.12: Comparison with mode scheme "scheme1" between histogram allocation: (a) coding result, (b) mode assignment, (c) stretched difference image, and convex hull allocation: (d) coding result, (e) mode assignment, (f) stretched difference image.

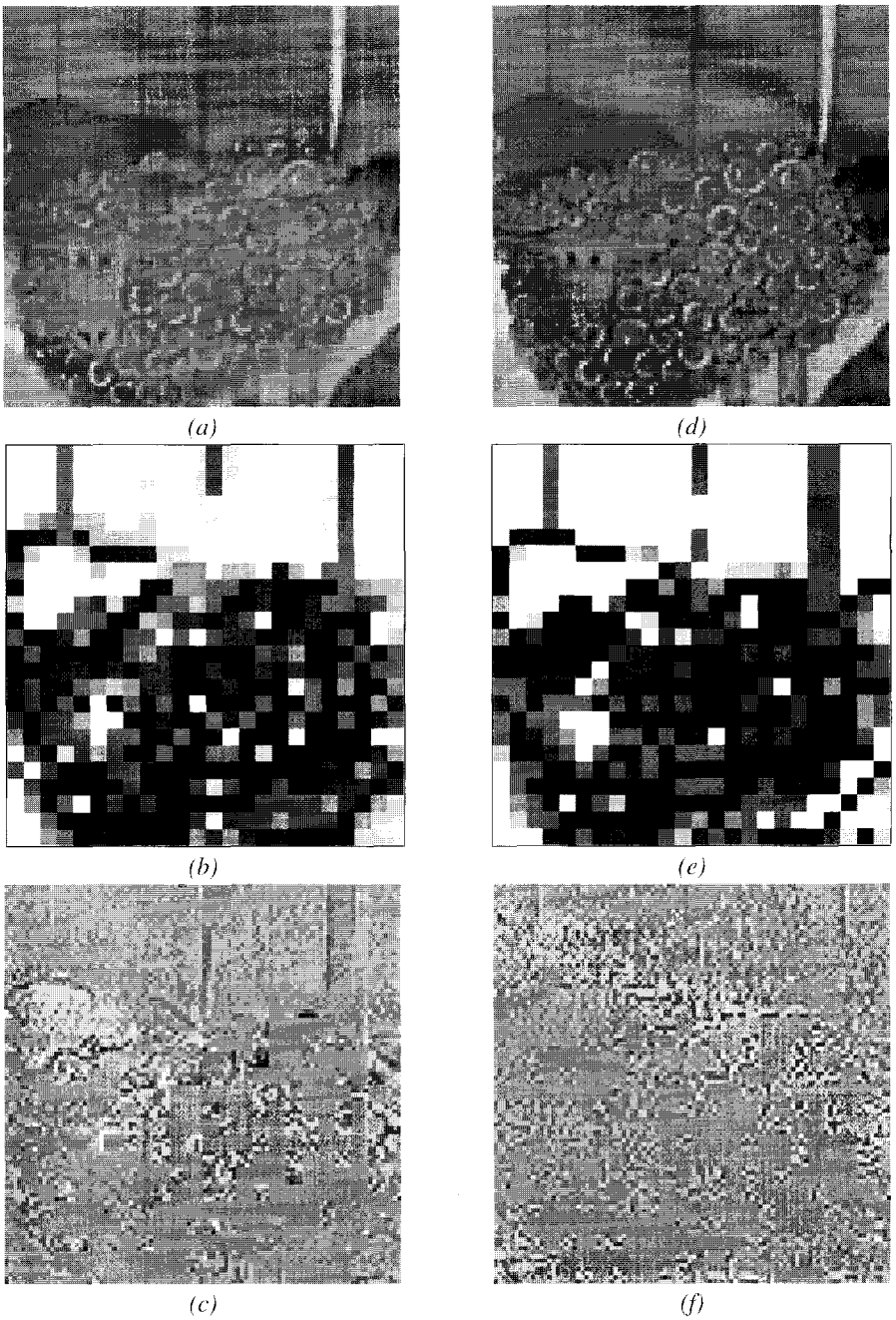


Figure 4.13: Comparison with mode scheme "scheme3" between hierarchical interpolation: (a) coding result, (b) mode assignment, (c) stretched difference image, and least-squares interpolation: (d) coding result, (e) mode assignment, (f) stretched difference image.

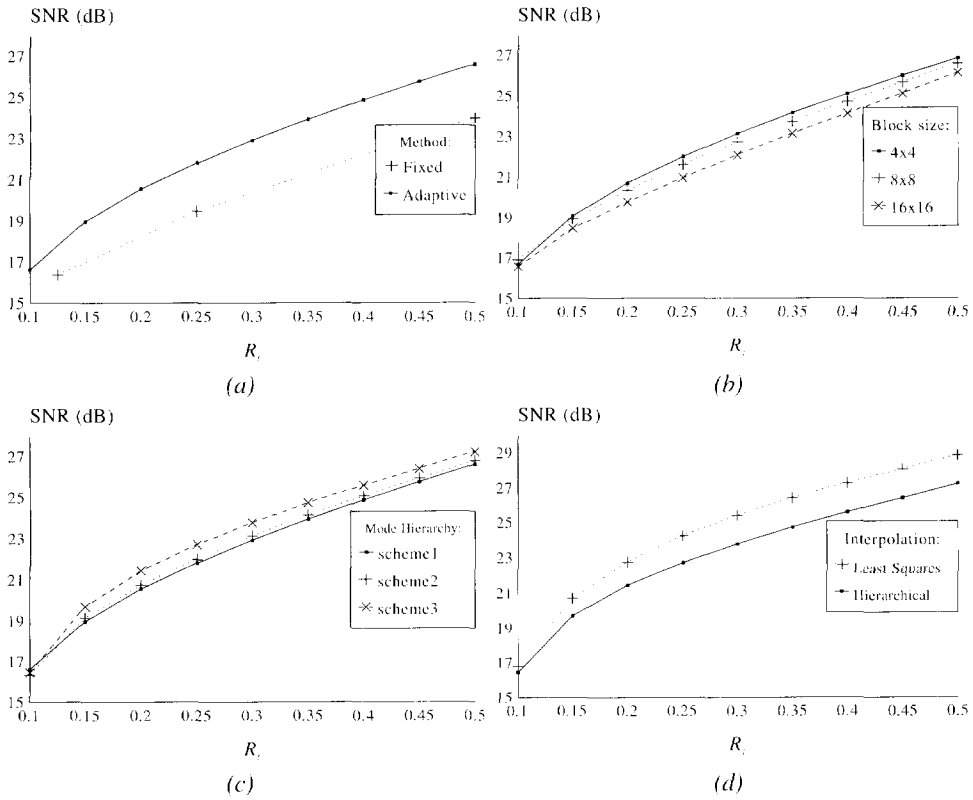


Figure 4.14: Simulation results of LENA with different, (a) subsampling schemes, (b) block sizes, (c) mode schemes, (d) interpolation methods.

selected. We see that spatially adaptive subsampling gives a significant improvement over the best fixed subsampling result. The difference decreases with decreasing rate because eventually, for zero rate, the mean square error is equal to the image variance in both coding schemes.

In Figure 4.14(b) the effect of different block sizes is shown. Small blocks give a better performance than large blocks because of a better adaptation to the local spatial frequency contents. For low bit rates, the advantage of small blocks decreases because of the relative increase of the amount of side information compared to the total bit rate. In Figure 4.14(c) the effect of different mode structures is shown. If the number of different modes increases, the SNR increases as well. This is because a better adaptation is possible and a closer match can be found between the necessary local spatial sampling frequency and the available sampling frequencies. In Figure 4.14(d) different interpolation methods are used. Least-squares interpolation gives a better result than hierarchical interpolation.

The optimal combination which follows from the investigated variations in this experiment is a system with a block size of 4 by 4 pixels, the mode structure *scheme3* and least-squares interpolation.

4.6.4 Motion compensated spatially adaptive subsampling

Motion compensated spatially adaptive subsampling (*MC/SA-SS*) is compared with the following other coding schemes:

SA-SS: Spatially adaptive subsampling. This scheme uses *no* motion information and was discussed in the previous section.

MC/NSS: Motion compensated coding with no subsampling. In this scheme there are only two modes: if a good prediction is made then no additional information is transmitted and if a bad prediction is made the original image is transmitted. Hence this scheme uses only temporal and no spatial adaptivity.

The simulations were done using the *MOBILE* sequence. The size of the blocks was 4 by 4 pixels and for the spatially adaptive subsampling the mode scheme *scheme3* was used with least-squares interpolation. This configuration of the parameters is chosen because the experiments in the previous section indicated that these parameters give the best result. The motion vectors were estimated using hierarchical block matching with two levels with pixel accuracy of the motion vectors (for details see Chapter 3). The results are averaged over the entire sequence and shown in Figure 4.15 for five different relative bit rates.

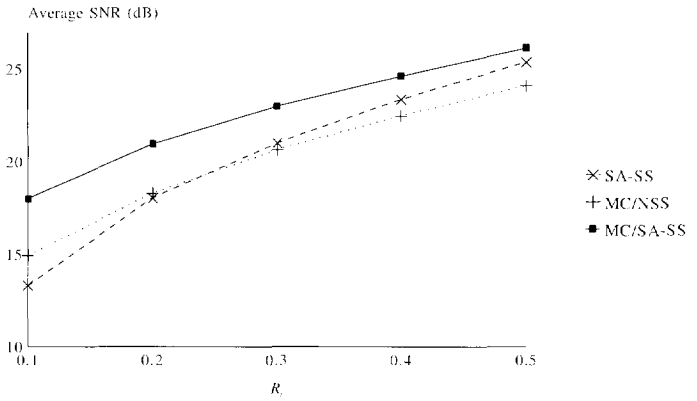


Figure 4.15: A comparison of the different algorithms. The average SNR for the MOBILE sequence is shown.

We see that spatially adaptive subsampling without motion compensation gives the poorest performance. This is what can be expected because no use is made of the temporal correlation in the image sequence. At zero bit rate the SNR will be equal to 0 dB. After motion compensation, the redundancy in the sequence is reduced considerably as is shown by the *MC/NSS* scheme. Now the SNR at zero bit rate is no longer dictated by the 0 dB boundary but controlled by the prediction error variance and the image variance. Finally, motion compensated spatially adaptive subsampling gives the best SNR results because both the

spatial and temporal correlation can be exploited. The difference with the other schemes increases for low bit rate when the efficiency of the algorithm becomes more important.

Typical artifacts which are introduced in the motion compensated spatially adaptive subsampling system, compared to the spatially adaptive subsampling system, is that besides spatial inconsistencies, also temporal inconsistencies are introduced. If a block cannot be predicted temporally then it is coded spatially, which gives different visual artifacts.

Chapter 5

MOTION ADAPTIVE SUBSAMPLING

In this chapter we focus on a subsampling technique known as sub-Nyquist sampling and some of its variations. Using this technique, the image sequence is spatially sampled with a frequency lower than the necessary Nyquist frequency, without taking precautions against aliasing. The aliasing can be undone at the decoder using temporal interpolation instead of spatial interpolation. In this way the full spatial resolution is preserved. Sub-Nyquist sampling can only be used for the stationary parts of the image sequence. If the image sequence contains regions with non-stationarities, for example because of motion, an alternative subsampling method has to be used. Hence, motion detection has to be used to segment the image into the two types of regions. Two practical coding schemes which use sub-Nyquist sampling are discussed: the Japanese MUSE [Nino87] system and the European HD-MAC system [Vree89] [Hagh90].

Motion compensated sub-Nyquist sampling [Belf92a] can be used to enlarge the velocity range for which sub-Nyquist sampling is possible. Now motion estimation has to be performed instead of motion detection. A problem which has to be solved is the problem of the *critical velocities* [Giro85] [Belf91]. These prevent a straightforward use of motion information. Aspects which are also addressed are how motion compensated sub-Nyquist sampling can be modified to incorporate motion information estimated with fractional accuracy [Belf93a] and what precautions have to be taken in case of an inaccurate motion estimate.

In the experiment results section, a comparison is made between fixed lattice subsampling, sub-Nyquist sampling and motion compensated sub-Nyquist sampling. The latter scheme is investigated in more detail.

5.1 Sub-Nyquist Sampling

First, the principle of sub-Nyquist sampling is discussed. Next, a specific explanation is given for image sequences. A description is given in both the spatial and frequency domain.

5.1.1 Principle

The principle of sub-Nyquist sampling is illustrated using a simple example [Anne86]. For the reason of simplicity we consider only the horizontal spatial dimension. We assume a two-dimensional spatially-continuous and time-discrete signal $i_c(x,t)$ with a spatial bandwidth W. We assume that the signal does not change in the temporal direction, therefore the time index

is dropped in the following discussion. This signal is spatially sampled according to the Nyquist theorem with a sampling period T equal to $1/2W$, which gives at each instance of the time the discrete signal $i(x)$ (see Figure 5.1(a)):

$$i(x) = \sum_{k=-\infty}^{+\infty} i_c(kT)\delta(x - kT) \tag{5.1}$$

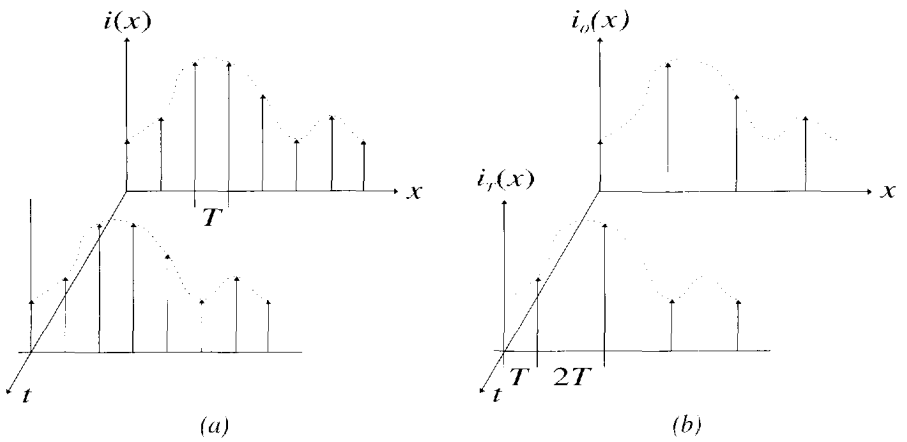


Figure 5.1: The principle of sub-Nyquist sampling in two dimensions: (a) normal sampling, (b) sub-Nyquist sampling.

In the spectral domain, sampling gives rise to replicas spaced by a distance $2W$, so the Fourier transform $I(\omega)$ of the sampled signal is (see Figure 5.2(a)):

$$I(\omega) = \frac{1}{T} \sum_{k=-\infty}^{+\infty} I_c\left(\frac{\omega}{2\pi} - \frac{k}{T}\right) \tag{5.2}$$

where $I_c(\omega)$ is the Fourier transform of $i_c(x)$. From this sampled signal the original spatially continuous signal can be recovered with a perfect low-pass filter with a cut-off frequency at $\omega = W$.

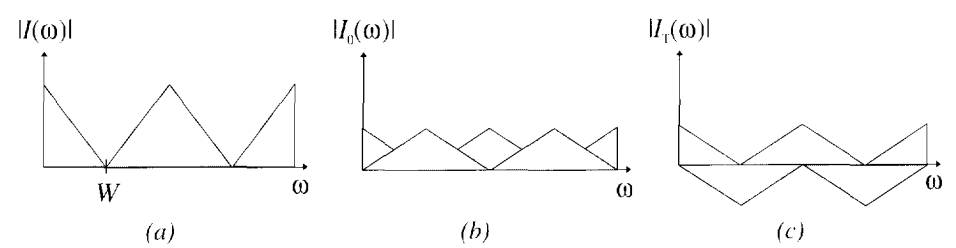


Figure 5.2: (a) Original spectrum, (b) Spectrum at t = 0, (c) Spectrum at t = 1.

In sub-Nyquist sampling, data reduction is achieved by using a sampling period which is larger than T. In this example the continuous signal is sampled with a sampling period of $2T$ and the first sample is taken at $x = 0$. We call this signal $i_0(x)$ (see Figure 5.1(b)):

$$i_0(x) = \sum_{k=-\infty}^{+\infty} i_c(2kT)\delta(x-2kT) \tag{5.3}$$

The Fourier transform $I_0(\omega)$ is given by

$$\begin{aligned} I_0(\omega) &= \frac{1}{2T}\sum_{k=-\infty}^{+\infty} I_c\left(\frac{\omega}{2\pi}-\frac{k}{2T}\right) \\ &= \frac{1}{2}\left(\frac{1}{T}\sum_{k=-\infty}^{+\infty} I_c\left(\frac{\omega}{2\pi}-\frac{k}{T}\right)+\frac{1}{T}\sum_{k=-\infty}^{+\infty} I_c\left(\frac{\omega}{2\pi}-\frac{k+½}{T}\right)\right) \end{aligned} \tag{5.4}$$

The first part of this expression is equal to the expression given for $I(\omega)$ in Equation (5.2). The second part represents the extra replicas introduced at $\omega = 2\pi\cdot(k+½)/T$ which are responsible for the aliasing. These components are shown with dark shading in Figure 5.2(b). Since the Nyquist criterion is not satisfied, the original signal can not be recovered.

At $t = 1$ the same signal is sampled again with a sampling period of $2T$ but with the first sample taken at $x = T$. This sampled signal $i_T(x)$ (see Figure 5.1(b)) is:

$$i_T(x) = \sum_{k=-\infty}^{+\infty} i_c(2kT-T)\delta(x-2kT) \tag{5.5}$$

It is obvious that the original discrete signal $i(x)$ can be reconstructed by taking the sum of $i_0(x)$ and $i_T(x)$:

$$i(x) = i_0(x) + i_T(x) \tag{5.6}$$

This argument of course holds in the frequency domain as well. The Fourier transform $I_T(\omega)$ of $i_T(x)$ is (see Figure 5.2(c)):

$$\begin{aligned} I_T(\omega) &= \frac{1}{2T}\sum_{k=-\infty}^{+\infty} I_c\left(\frac{\omega}{2\pi}-\frac{k}{2T}\right)\cdot\exp(-j\omega T) \\ &= \frac{1}{2}\left(\frac{1}{T}\sum_{k=-\infty}^{+\infty} I_c\left(\frac{\omega}{2\pi}-\frac{k}{T}\right)-\frac{1}{T}\sum_{k=-\infty}^{+\infty} I_c\left(\frac{\omega}{2\pi}-\frac{k+½}{T}\right)\right) \end{aligned} \tag{5.7}$$

This expression has the same form as Equation (5.4) but the extra replicas have a negative phase with respect to the extra replicas of $I_0(\omega)$. Adding $I_0(\omega)$ and $I_T(\omega)$ cancels out the extra replicas and gives the spectrum $I(\omega)$ of the originally sampled signal.

In conclusion, in sub-Nyquist sampling data reduction is achieved by using the fact that at each time instance only a part of the total number of required samples are taken. At the decoder, the original signal can be recovered by combining the samples taken at the different instances in time. This operation can be seen as a simple temporal interpolation filter.

5.1.2 Sub-Nyquist sampling of image sequences

Consider an image sequence that contains two identical consecutive images (Figure 5.3). Now sub-Nyquist sampling is applied to this sequence. The original image is defined on an orthogonal lattice L_o. The lattice used for the subsampled image at $t = k\text{-}1$ is L_{k-1}. The sampling density is halved compared to the original orthogonal sampling lattice. At $t = k$ the lattice

$$\begin{aligned} L_k &= \{\mathbf{x} \in L_o \mid \mathbf{x} \notin L_{k-1}\} \\ &= \{\mathbf{x} \mid \mathbf{x} = \mathbf{y} + (1.0)^T, \mathbf{y} \in L_{k-1}\} \end{aligned} \tag{5.8}$$

is used instead of L_{k-1}. The second line of this equation shows that L_k is a coset of the lattice L_k. As was shown in Section 2.1.1, the union of these two lattices completely covers the entire image. Thus if these two lattices are combined using a fixed 2-taps temporal interpolation filter, the original image can be reconstructed exactly at the receiver. Hence this technique results in an output image with the full original spatial resolution. This is only true if the content of the scene is stationary.

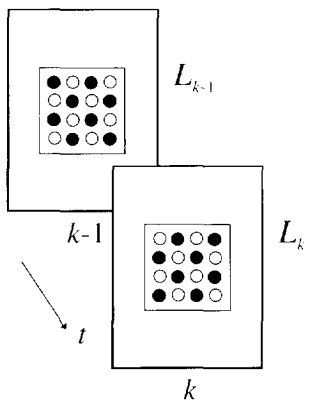


Figure 5.3: Sub-Nyquist sampling of an image sequence using a quincunx lattice.

If the sampling lattice L_{k-1} contains more than two cosets, i.e. the sampling density is lower, then the number of image involved in the interpolation increases. A higher compression ratio can be achieved at the expense of a longer temporal interpolation filter. As a consequence, the image sequence has to be stationary over a longer period of time.

In Figure 5.4, sub-Nyquist sampling is described in the spectral domain for a completely stationary image sequence using the same sampling structure as Figure 5.3. For the sake of simplicity, only the intersection of the spectrum with the ω_x-axis is shown. The replicas introduced because of the spatial subsampling are indicated as dashed lines. They are located in the empty space between the original temporal replica. Because of the quincunx subsampling lattice, the replicas are shifted along the ω_x-axis with respect to the original spectral components. The temporal space is empty because there is no motion causing the temporal bandwidth to be equal to zero. Therefore it can be said that in this case the temporal resolution is sacrificed in favor of the spatial resolution. If the sampling lattice has a lower

sampling density, more replicas are introduced and the area between the original replicas will become more densely packed. The shaded area in Figure 5.4 indicates the passband of the temporal filter. After temporal filtering only the replicas from the original image sequence are retained. Note that the original replicas cannot be recovered by using a spatial interpolation filter.

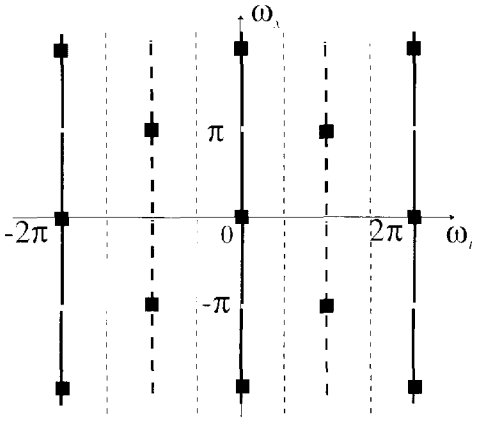


Figure 5.4: Sub-Nyquist sampling in the spectral domain in the case of no motion. The shaded area indicates the pass band of the temporal interpolation filter.

5.1.3 Velocity range

Sub-Nyquist sampling is not limited to completely stationary image sequences but is also possible for a small range of velocities. Although a fixed temporal interpolation filter is used and no further motion information is taken into account, it is also applicable to image sequences containing little motion. The velocity range for sub-Nyquist sampling directly depends on the interpolation filter used. If a low-pass filter is used with cut-off frequency β then, using Equation (3.20), the velocity range for perfect interpolation is given by

$$v_x = \pm\frac{\beta}{W_x},\ v_y = \pm\frac{\beta}{W_y} \quad \text{(pixels / image)} \tag{5.9}$$

In the case that $W_x = W_y = \pi$ and a perfect low-pass filter with a cut-off frequency at $\frac{1}{2}\pi$ is used, the range is $\pm\frac{1}{2}$ pixels/image in horizontal and vertical direction. If a lattice with a lower sampling density is used, the replicas come closer to the origin. In this case β must also decrease, reducing the velocity range.

If the scene contains a camera pan, a large portion of the screen is moving with the same velocity. To still use sub-Nyquist sampling in this situation, an estimate of the global motion vector for the entire image is required. The global motion vector is used to compensate the entire image, hereby increasing the stationary area. Essentially this means that a single fixed motion compensated interpolation filter is used for the entire image.

5.2 Implementation of sub-Nyquist sampling

The problem encountered in a practical implementation of sub-Nyquist sampling is that sub-Nyquist sampling is only useful for a limited range of velocities. Therefore some kind of fall-back mode has to be used for non-stationary regions. Also motion detection is necessary. In this section two practical implementations of sub-Nyquist sampling are discussed. The schemes are simplified versions of the Japanese MUSE system [Nino87] and European HD-MAC system [Vree89]. Other examples of sub-Nyquist sampling can be found in [Tong87] and [Scha87].

5.2.1 MUSE

The first system is the MUSE (Multiple sub-Nyquist Sampling Encoding) system. It was developed by the Japanese broadcasting corporation (NHK) for the analog transmission of HDTV signals through direct broadcasting satellites using FM modulation. The scanning format of the input signal is 1125 lines/1440 pixels/60 fields/2:1 interlace with an aspect ratio of 16:9. The bandwidth of the input signal is 22 MHz, but the signal is sampled at a sampling frequency of 48.6 MHz. The reason for this is explained later on. A variation on the MUSE system is narrow-MUSE [Tana90] which is intended for terrestrial broadcasting.

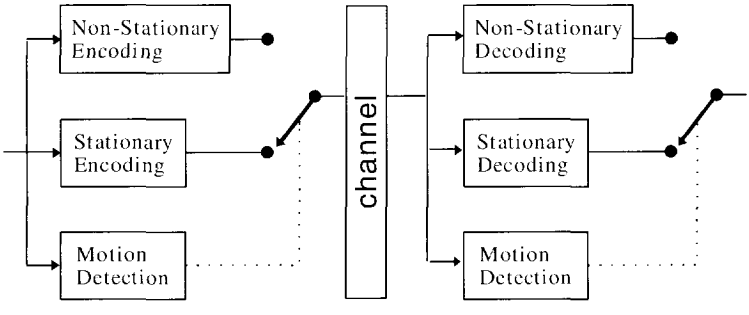


Figure 5.5: Simplified version of the MUSE encoder and decoder.

A simplified version of the encoding and decoding scheme is shown in Figure 5.5. The encoding strategies for stationary and non-stationary parts differ and are therefore discussed separately. The segmentation in a stationary and non-stationary part is in this case done on a *pixel* basis. This detailed information can obviously not be transmitted to the decoder, as it would cause a large overhead. Therefore a scheme has been devised which enables the detection of motion at the decoder based only on the transmitted pixels. The consequence of this is that the subsampled stationary parts should have the same sampling structure as the subsampled non-stationary parts.

The steps taken to encode the stationary parts are shown in Figure 5.6. The sampling lattice in the spatial domain, the spectrum along the horizontal frequency axis and the support of the two-dimensional spectrum are shown. The original image is shown in Figure 5.6(a). First the

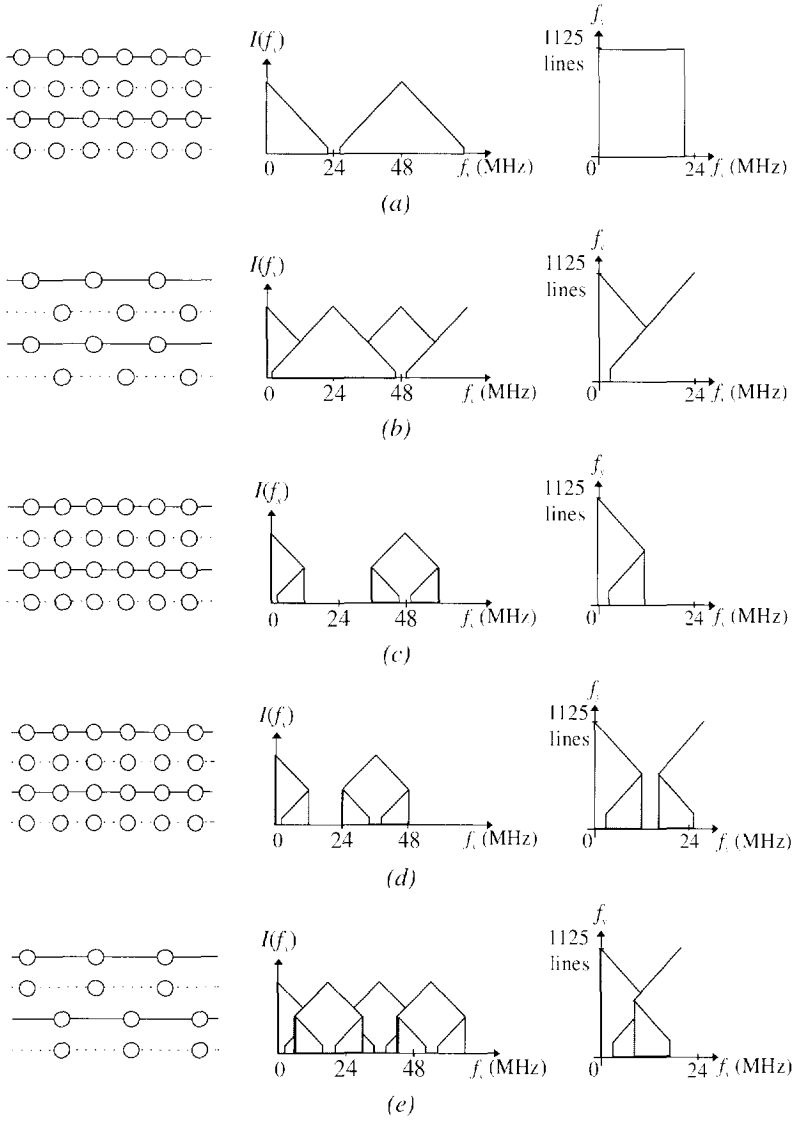


Figure 5.6: Encoding of the stationary parts in the MUSE system (the number between the parenthesis indicates the horizontal sampling frequency): (a) original (48.6 MHz), (b) field quincunx subsampling (24.3 MHz), (c) 12 MHz low-pass filter (48.6 MHz), (d) 4:3 conversion (32.4 MHz) and (e) line quincunx subsampling (16.2 MHz).

area of the stationary parts of the image is increased by using one global motion vector to compensate for camera panning. Next the frame is prefiltered using a diamond-shaped filter and subsampled according to the field quincunx sampling lattice (Figure 5.6(b)). The sampling lattice is shifted for the following frame, so this is the first sub-Nyquist sampling stage. The subsampling causes the horizontal high-frequency components to fold back. Next, an interpolation filter is applied in the horizontal direction, hereby converting the image back

to the original sampling lattice (Figure 5.6(c)). Next the sampling lattice is modified by changing the sampling frequency in the horizontal direction with a factor 3/4 (Figure 5.6(d)). On this new sampling lattice line quincunx subsampling is applied which is also shifted for the following frame, folding back the spectrum another time (Figure 5.6(e)). This is the second sub-Nyquist sampling stage. Now it can be seen that the 3/4 sampling structure conversion and the initial oversampling was intended to keep the frequency region around the DC frequency free of any aliasing. At the decoder, this part of the spectrum is used for motion detection.

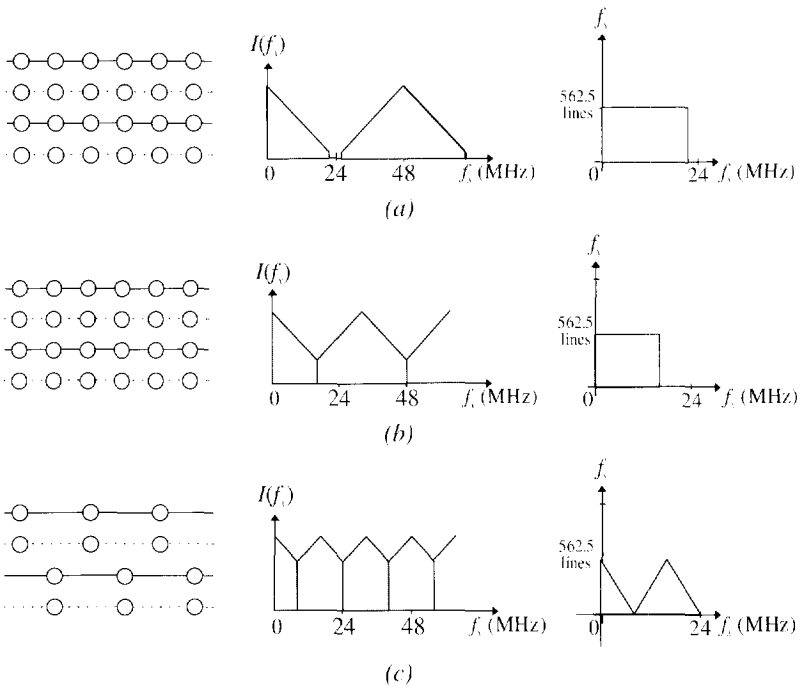


Figure 5.7: Encoding of the non-stationary parts in the MUSE system (the number between the parenthesis indicates the horizontal sampling frequency): (a) original (48.6 MHz), 4:3 conversion (32.4 MHz), (c) line quincunx subsampling (16.2 MHz).

The encoding of the non-stationary parts is illustrated in Figure 5.7. Because of the non-stationarity, each field has to be processed individually because successive fields belong to a different instance in time. A field has half the resolution of an entire frame and the corresponding original lattice and spectrum are shown in Figure 5.7(a). Starting from this image, the sampling lattice is converted to the sampling lattice used for the stationary image in Figure 5.6(d). The conversion is done by changing the sampling frequency in the horizontal direction with a factor 3/4 (Figure 5.7(b)). To avoid aliasing, prefiltering is necessary. Next, after another prefiltering operation, line quincunx subsampling is applied (Figure 5.7(c)), giving the same sampling structure as the stationary branch.

At the receiver, motion is detected based on the low-frequency part of the signal spectrum which does not contain aliasing. Because the detection is based on a different signal then the detection performed at the transmitter, the detection result may differ. At the receiver there are again two branches. One branch uses a filter to interpolate the image spatially. The second branch uses temporal filtering to undo the different sub-Nyquist subsampling stages.

5.2.2 HD-MAC

The second system is the HD-MAC (High Definition - Multiplexed Analog Components) system which is also originally intended for transmission through direct broadcasting satellites. The scanning format of the input signal is 1250 lines/1440 pixels/50 fields/2:1 interlace with an aspect ratio of 16:9. The HD-MAC system is designed to be downward compatible with D2-MAC [Anne86] [Bern90], which is the reason for some of the choices made in the algorithm design. A D2-MAC receiver simply displays the samples which are transmitted for the HD-MAC system without any interpolation.

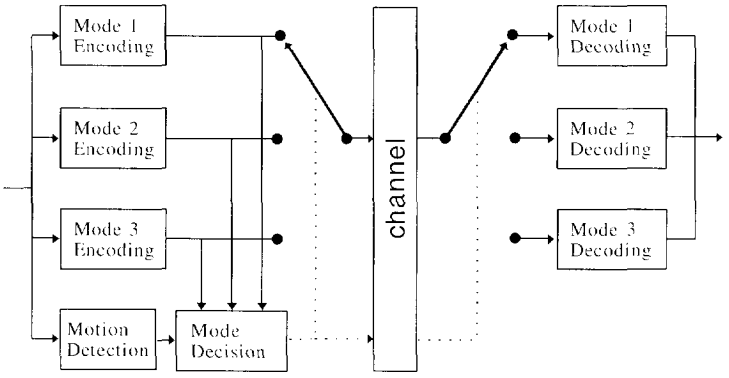


Figure 5.8: Simplified version of the HD-MAC encoder and decoder.

A simplified version of the encoding and decoding scheme is shown is Figure 5.8. HD-MAC distinguishes between three different transmission modes instead of the two modes used in the MUSE system. These modes vary with the amount of motion in the image sequence. Each frame is divided into blocks of 8 by 8 pixels. Every block is subsampled according to one of the three modes. The choice as to which mode is used for each block is transmitted as side information in the DATV (Digitally Assisted TeleVision) channel, which uses the time during the vertical blanking interval. At the decoder, the different modes are interpolated and combined to form the complete, reconstructed image. As we see from this simple discussion, HD-MAC results in a simple decoder because no motion detection is necessary at the decoder. However, with the MUSE system, the detection is done on a pixel basis which makes it easier to adapt to variations in the local displacement at the expense of a more complex decoder.

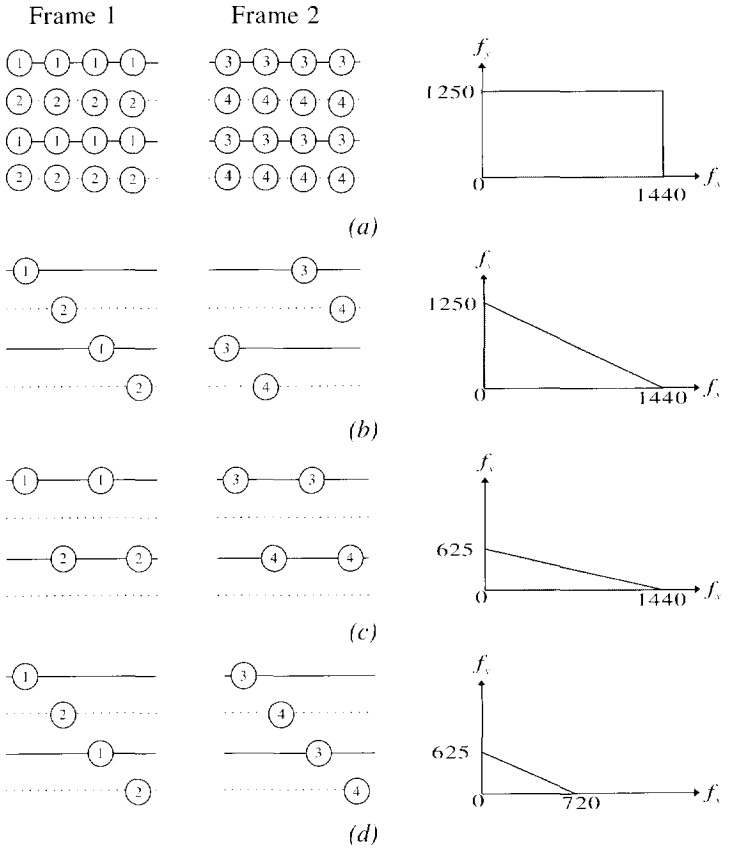


Figure 5.9: HD-MAC subsampling: (a) original, (b) mode 1, (c) mode 2 and (d) mode 3.

An illustration of the different modes is given in Figure 5.9. The figure shows the sampling structures over a period of two frames (80 ms) for a quarter part of a block. The sampling structure for the entire block is formed by tilling this part over the entire block. The resolution in the spatial frequency plane is also depicted. The original image is shown in Figure 5.9(a). The different modes are used in the following situations:

- Mode 1 (Figure 5.9(b)) is used for stationary parts and has a periodicity of 80 ms. The input images in this branch are processed on a frame basis. The diagonal frequency components are sacrificed by using a quincunx subsampling lattice. The remaining samples are transmitted using sub-Nyquist sampling. The transmission of the necessary samples is divided into four parts. The numbers in Figure 5.9(b) indicate the order in which the samples are transmitted.
- Mode 2 (Figure 5.9(c)) is used for slowly moving parts and has a periodicity of 40 ms. In practice, this mode is used for parts of the image for which a proper motion estimate can be made. This branch processes the sequence on a field basis. Again the diagonal frequency components are discarded. Only pixels from one field are transmitted over a period of two

fields. The missing field is interpolated at the receiver using motion compensated interpolation. Therefore this mode requires the transmission of a motion vector.

- Mode 3 (Figure 5.9(d)) is used for moving parts for which no proper motion estimate can be made. Now each subsampled field has to be transmitted without any delay. The periodicity is therefore equal to 20 ms. Compared to mode 2, the horizontal resolution is halved.

5.3 Critical velocities

If an image sequence contains little or no motion, then after subsampling the sequence can be reconstructed using sub-Nyquist sampling. If the image sequence contains motion, direct combination of the pixels from the different subsampled images is not possible. This problem can be partially solved by using motion information for the interpolation. A motion compensated interpolation filter is used instead of a fixed temporal interpolation filter.

Unfortunately, perfect reconstruction using motion compensated interpolation cannot be achieved in all situations. For a given *spatial* subsampling lattice, there are certain object velocities, called *critical velocities* [Giro85], which do not allow for a perfect reconstruction of an object moving at that velocity. An example using the sampling lattice from Figure 5.3 is given in Figure 5.10. In Figure 5.10(a) the horizontal velocity is equal to 1 pixel/image. We see that after subsampling the replicas overlap with the original spectrum. It is now no longer possible to recover the original spectrum. To show that this effect depends on the velocity, in Figure 5.10(b) a velocity of 2 pixel/image is assumed. Now the replicas do not overlap with the original spectrum and the image can be reconstructed with a motion compensated interpolation filter. The conditions under which these critical velocities occur are discussed in the next section.

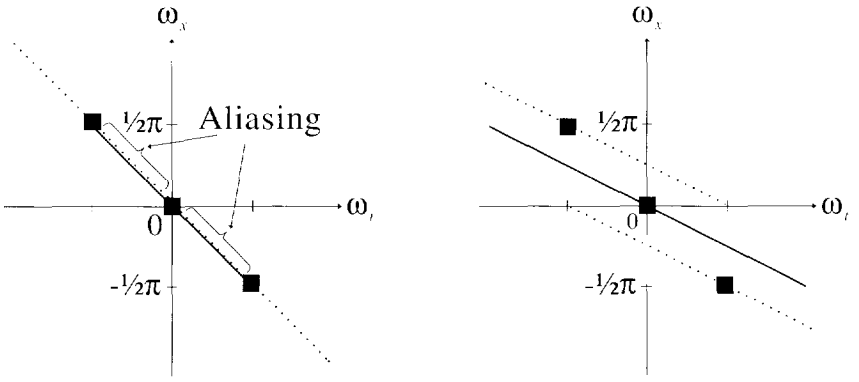


Figure 5.10: Critical velocities shown in the frequency domain: (a) v_x = 1, (b) v_x = 2. For clarity only the relevant replicas are shown.

5.3.1. Relation between critical velocities and the sampling lattice

In this section, an expression is derived for the critical velocities given a certain subsampling lattice [Belf91]. It is assumed that the original sequence was sampled on an orthogonal sampling lattice and that the subsampling factor is equal to 2. The lattice points $(x,y,t)^T$ after subsampling can always be described as follows:

$$\begin{pmatrix} s_{11} & s_{12} & s_{13} \\ 0 & s_{22} & s_{23} \\ 0 & 0 & s_{33} \end{pmatrix} \begin{pmatrix} k \\ l \\ m \end{pmatrix} = \begin{pmatrix} x \\ y \\ t \end{pmatrix}, \quad k,l,m \in \mathbb{Z} \tag{5.10}$$

The upper diagonal matrix **S** with elements s_{ij} determines the structure of the subsampling lattice. At $t = 0$ and $t = 1$ all possible sampling positions $\mathbf{x}_0$ and $\mathbf{x}_1$ are given by

$$\mathbf{x}_0 = \begin{pmatrix} s_{11}k_0 + s_{12}l_0 \\ s_{22}l_0 \end{pmatrix}, \mathbf{x}_1 = \begin{pmatrix} s_{11}k_1 + s_{12}l_1 + \dfrac{s_{13}}{s_{33}} \\ s_{22}l_1 + \dfrac{s_{23}}{s_{33}} \end{pmatrix}, k_0,k_1,l_0,l_1 \in \mathbb{Z} \tag{5.11}$$

This result is arrived at by substituting the values for t and eliminating the variable m. If the velocity **v** is defined as the displacement of a pixel between two consecutive frames, then **v** is equal to

$$\mathbf{v} = \mathbf{x}_1 - \mathbf{x}_0 = \begin{pmatrix} s_{11}(k_1 - k_0) + s_{12}(l_1 - l_o) + \dfrac{s_{13}}{s_{33}} \\ s_{22}(l_1 - l_o) + \dfrac{s_{23}}{s_{33}} \end{pmatrix} = \begin{pmatrix} s_{11}p + s_{12}q + \dfrac{s_{13}}{s_{33}} \\ s_{22}q + \dfrac{s_{23}}{s_{33}} \end{pmatrix}, p,q \in \mathbb{Z} \tag{5.12}$$

Two conclusions follow from Equation (5.12). In the first place, if a pixel moves at a velocity **v** that satisfies Equation (5.12), and if the pixel also satisfies Equation (5.10), i.e. its position is an element of the subsampled lattice, then we can always associate a certain pixel on the same subsampling lattice in any other image with the pixel considered. Secondly, if, however, a pixel is moving with a velocity **v** and is *not* an element of the subsampled grid, then we can *never* associate any other pixel on the same subsampling grid in any other image with this pixel, otherwise contradicting the first conclusion. The latter conclusion implies that if a pixel (or object) moves with a velocity satisfying Equation (5.12), it can never be perfectly recovered using a motion compensated interpolation filter, unless such a pixel does not exist. If the matrix **S** described a lattice with only temporal subsampling, then the individual images do not contain discarded pixels and the above argument no longer applies. Hence Equation (5.12) gives an expression for critical velocities given a certain spatial subsampling lattice.

The conclusion that can now be drawn is that every regular sampling lattice using spatial subsampling possesses the problem of critical velocities. In the next section different solutions to this problem are introduced. It should be noted that the method described in this section does not necessarily give *all* the critical velocities for an arbitrary subsampling factor. If a higher subsampling factor is used, it may be possible to go from one sampling point to another sampling point while skipping some intermediate images. The method described in

this section can then be extended straightforwardly by taking more than just pairs of image into account.

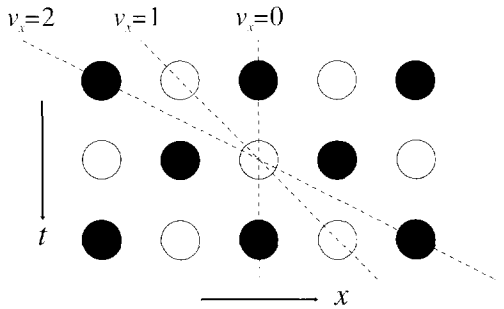


Figure 5.11: Illustration of critical velocities. A cross-section in the x t plane is shown. In this figure $v_x = 1$ is a critical velocity.

As an example of critical velocities consider the following matrix **S**:

$$\mathbf{S} = \begin{pmatrix} 2 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.13}$$

The subsampling lattice associated with this matrix is the discarding of every other column in the image, and shifting this pattern for every image (Figure 5.11). According to Equation (5.12) the critical velocities $\mathbf{v}_c$ are:

$$\mathbf{v}_c = \begin{pmatrix} 2p+1 \\ q \end{pmatrix}, \quad p,q \in \mathbb{Z} \tag{5.14}$$

Figure 5.11 illustrates that it is indeed impossible to interpolate the shaded pixel for the velocity $v_x = 1$ irrespective of v_y, because in this direction only discarded pixels are encountered.

5.3.2 Possible solutions for critical velocities

In the previous section the problem of critical velocities was examined. In this section some possible solutions for this problem are argued. These solutions are illustrated in the spectral domain in Figure 5.12. The subsampling structure used for the example in Figure 5.11 is used again. It was shown in the previous section that not all the velocities are critical velocities. Therefore, in Figure 5.12, velocities of 1 pixel/image in the horizontal direction and 2 pixels/image in the horizontal direction are used in order to illustrate the implications of the different solutions for different velocities. The dashed lines are the replicas introduced by subsampling the image sequence.

The following solutions are possible:

- In Figures 5.12(a) and 5.12(b), the spatial resolution of all the moving regions is reduced in order to avoid critical velocities. This is done by low-pass filtering the image prior to

subsampling. As we saw in Section 2.4, this is objectionable in the case of slowly moving objects because of the loss of resolution.

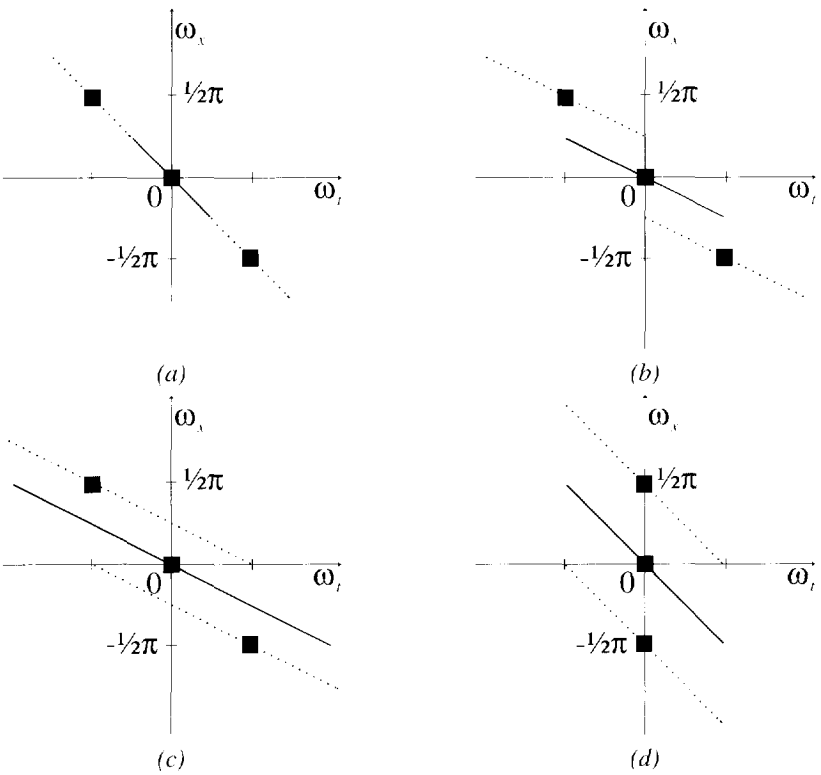


Figure 5.12: Possible solutions for critical velocities shown in the frequency domain: in (a) and (d) the velocity is 1 pixel/image. In (b) and (c) the velocity is 2 pixels/image.

- An alternative solution is to reduce the spatial resolution only if objects are moving with a critical velocity. Figures 5.12(a) and 5.12(c) apply to this situation. A similar solution is generally used in the case of motion compensated de-interlacing [Giro85] [Erns88], where for critical velocities a spatial interpolation filter is used. For a subsampling system, this possibility is rejected using the same argument as applied to the first solution.
- The cause of the critical velocities is the fact that in the interpolation stage only discarded pixels are encountered in the direction of the motion. To avoid this situation, we can use a spatially adaptive subsampling lattice [Belf92a]. The idea is to shift the subsampling structure depending on the amount of motion. This is illustrated in Figure 5.13. Different subsampling lattices are used for different velocities. The Figures 5.12(d) and 5.12(c) show this solution in the frequency domain. In Figure 5.12(d), we see that because of the horizontal shift in the subsampling lattice for a velocity of 1 pixel/image, the replicas introduced by the subsampling are positioned at different locations, causing no interference with the other replica. No shifting is required for a velocity of 2 pixels/image as Figure

5.12(c) shows. The advantage of this solution is that the image can be subsampled without reducing any spatial or temporal resolution.

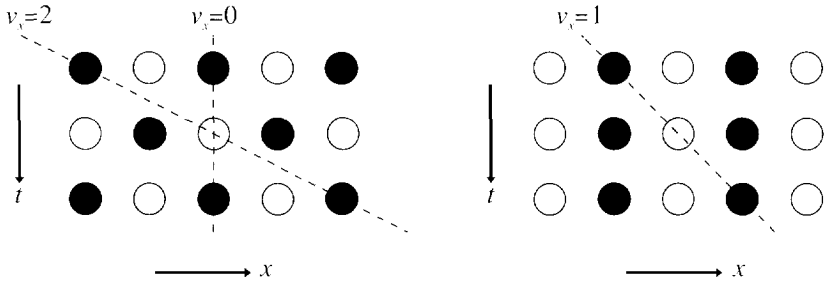


Figure 5.13: Subsampling structures (a) v_x is even. (b) v_x is odd.

5.4 Motion compensated sub-Nyquist sampling

In the previous section we saw that critical velocities can be avoided by adapting the subsampling structure. This idea of a motion adaptive subsampling lattice is further explored in this section.

5.4.1. Principle

In a motion compensated sub-Nyquist sampling scheme, the sub-Nyquist principle is modified so that it can be applied to non-stationary regions. The pixels which are transmitted are (using the same notation as in Section 5.1.2):

$$L_k = \{\mathbf{x} \in L_o \mid \mathbf{x} - \mathbf{d}_b(\mathbf{x},k) \notin L_{k-1}\} \qquad (5.15)$$

The vector $\mathbf{d}_b(\mathbf{x},k)$ is the estimated displacement vector for the pixel $(\mathbf{x},k)$. After motion compensated interpolation, the union of L_k and L_{k-1} form L_o. This is because no discarded pixel is present in the direction of the motion so it is always possible to reconstruct the current image based on the previous image and the subsampled current image using motion compensated interpolation. Note that Equation (5.15) degenerates to Equation (5.8) in the absence of motion. Therefore this method can be called motion compensated sub-Nyquist sampling. Equation (5.15) however does not guarantee a constant data reduction as is always the case with non-adaptive sub-Nyquist sampling. Further, $\mathbf{d}_b(\mathbf{x},k)$ has to be transmitted as additional side information.

Starting from a fixed subsampling grid, a recursive update is made of the subsampling structure. This principle is illustrated in Figure 5.14. The subsampling structure of image k is based on the subsampling structure of image k-1 and the displacement vector $\mathbf{d}_b(\mathbf{x},k)$. If the pixel at $(\mathbf{x}\text{-}\mathbf{d}_b(\mathbf{x},k),k\text{-}1)$ of the previous image was not an element of L_{k-1}, then the corresponding pixel $(\mathbf{x},k)$ in the current image is transmitted. If the pixel was not discarded in the previous image, then the corresponding pixel along the motion trajectory in the current image is not transmitted.

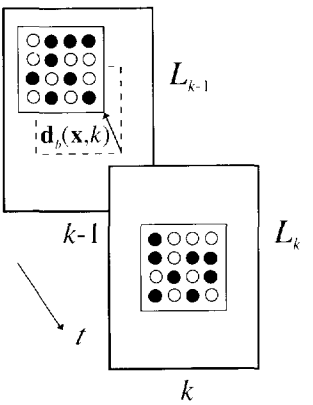


Figure 5.14: Motion adaptive sub-Nyquist sampling.

5.4.2 Motion compensated sub-Nyquist sampling and fixed lattice subsampling

In a subsampling system it may be desirable to combine motion compensated sub-Nyquist sampling with fixed lattice subsampling [Belf92b]. There are two reasons for this:

- The fixed lattice subsampling stage can be used as a first step to reduce the number of pixels.
- In the next section we see that it enables the use of fractional motion estimation accuracy.

A simple scheme to combine these two data compression methods is shown in Figure 5.15(a). Without loss of generality we only consider the x-t dimensions and assume a constant velocity of 1 pixel/image in the horizontal direction. What is shown in each step of the figure is *one* line of two consecutive images.

The image is first prefiltered in the horizontal direction, reducing the horizontal bandwidth W_x from π to $\frac{1}{2}\pi$. Because of this prefiltering the image may be subsampled by discarding half the columns of the image. These two operations are illustrated in the second step of Figures 5.15(a). By substituting $W_x = \frac{1}{2}\pi$ in Equation (3.20) and again assuming a perfect interpolation filter with a cut-off frequency at $\frac{1}{2}\pi$ the nominal velocities in the horizontal direction v_{nx} are now

$$v_{nx} = 2k\,, \quad k \in \mathbb{Z} \qquad \text{(pixels/image)} \tag{5.16}$$

Therefore a displacement of 1 pixel/image has to be truncated to either 0 pixel/image or 2 pixels/image. In the third step of Figure 5.15(a), the displacement of 1 pixel/image is assigned to a region covered by the nominal velocity of 0 pixel/image. Using this nominal velocity, motion compensated sub-Nyquist sampling is applied on the subsampled grid, resulting in an overall data reduction with a factor of 4. This is shown in the third step of Figure 5.15(a). At the receiver, the image is interpolated with a motion compensated filter (step four in Figure 5.15(a)), followed by a spatial interpolation filter.

A major disadvantage of this scheme is that a velocity of 1 pixel/image falls in the transition band of the temporal interpolation filter, resulting in a poor attenuation of the replicas and a

distortion of the baseband. Another disadvantage is that no benefit is taken from the extra available accuracy in the displacement estimate, which is lost because of the necessary truncation.

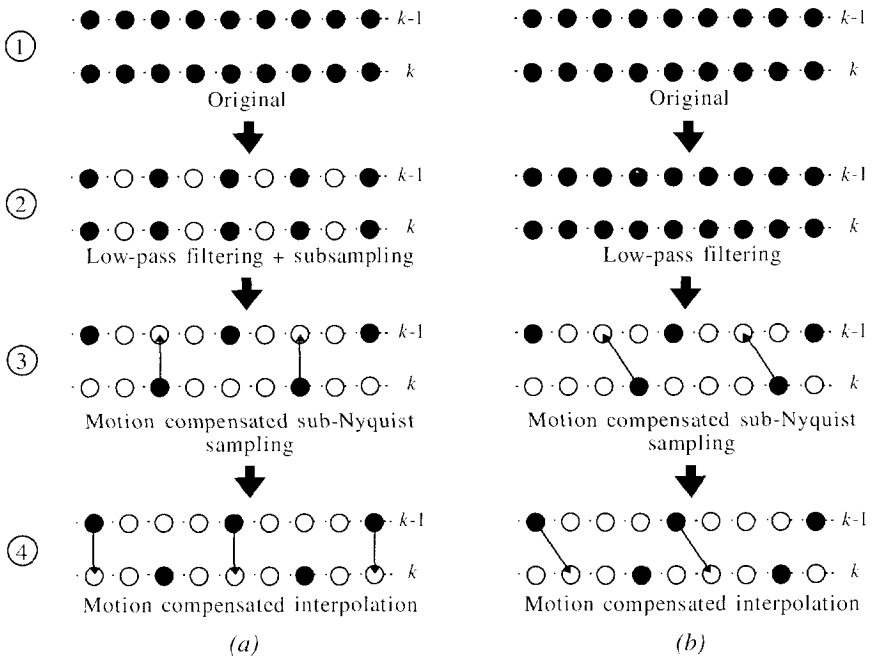


Figure 5.15: Two possible solutions to combine motion compensated sub-Nyquist sampling with fixed subsampling (v_x= 1 pixel/image): (a) with loss of the resolution of the motion vector and (b) without loss of the resolution of the motion vector.

A more attractive alternative is shown in Figure 5.15(b). It is based on the discussion in Section 3.1.3, which showed that a decrease in distance between the nominal velocities improves the result of the motion compensated interpolation. After the spatial low-pass filtering the images are *not* immediately subsampled. On this sampling structure a velocity of 1 pixel/image is still defined, and the nominal velocities in the horizontal direction are given by

$$v_{nx} = k\,, \quad k \in \mathbb{Z} \qquad \text{(pixels/image)} \tag{5.17}$$

so truncation of the velocity is not necessary. Now the subsampling following the spatial low-pass filter is combined with the motion compensated sub-Nyquist sampling. The algorithm is modified in such a way that after spatial interpolation a sequence of three consecutive discarded pixels may not exist, thus satisfying the sampling theorem. This is illustrated in the third step of Figure 5.15(b). Because the images are not yet subsampled, a velocity of 1 pixel/image can be used, exploiting the full accuracy of the displacement estimate. If now motion compensated interpolation is used at the receiver (step four in Figure 15(a)), a velocity of 1 pixel/image will fall in the middle of the passband of interpolation filter at the receiver, achieving a better suppression of the replicas.

This scheme can be extended to two dimensions straightforwardly. Only the shape and the size of the region which should not contain discarded pixels must be adapted to the sampling lattice used.

5.4.3 Fractional motion accuracy

The above scheme also presents a way to use motion compensated sub-Nyquist sampling using motion vectors estimated with a higher accuracy. The modifications which have to be made to the original algorithm are illustrated in Figure 5.16.

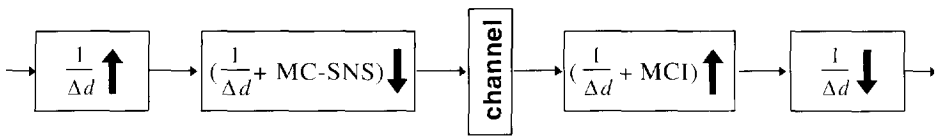


Figure 5.16: Motion compensated sub-Nyquist sampling combined with fractional accuracy of the motion estimate.

If the accuracy of the motion estimate is Δd pixel, then first the original sequence is spatially upsampled with a factor $1/\Delta d$. On this sampling lattice, the motion accuracy is defined on a pixel basis. This concept was previously discussed in Section 3.2. At this higher spatial resolution, the motion compensated sub-Nyquist sampling (MC-SNS) is combined with a downsampling of a factor $1/\Delta d$. The downsampling with a factor $1/\Delta d$ cancels the upsampling with a factor $1/\Delta d$, so the net result of these operations is the subsampling factor of the motion adaptive sub-Nyquist sampling. At the receiver these operations are inverted, replacing the subsampling with motion compensated interpolation (MCI) and an upsampling.

5.4.4 Motion compensated sub-Nyquist sampling coding system

A system incorporating motion compensated sub-Nyquist sampling is described in this section. In Chapter 3, we have seen that large interpolation errors in the motion adaptive branch were caused by an inaccurate motion vector, giving noticeable artifacts in the interpolated image. Therefore when the motion vector is inaccurate, special precautions have to be taken to provide an acceptable image quality while still realizing the same data reduction.

This leads to a system with two branches (Figure 5.17):

- The lower branch is used if a correct motion estimate (ME) is made, using motion compensated sub-Nyquist sampling (MC-SNS) to subsample the image. At the receiver, motion compensated interpolation (MCI) is used in the reconstruction process.
- The upper branch is the fall-back method which is used in the case of an incorrect motion vector. In this case low-pass filtering (LPF1) and a fixed quincunx subsampling lattice is used (QNX-SS). At the receiver spatial interpolation (LPF2) is used to reconstruct the image.

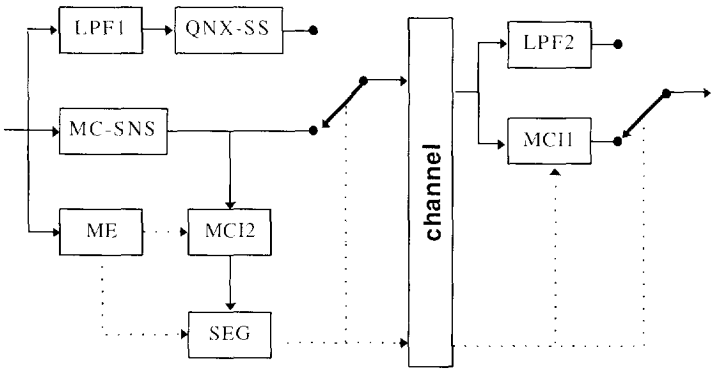


Figure 5.17: System overview.

The interpolation quality of the motion adaptive branch controls the switching process between these two modes. This guarantees that the highest possible resolution is always chosen, regardless of the spectrum of the current image. The error analysis in Chapter 3 validates this principle. In order to be able to make a decision, the image is also interpolated at the transmitter side (MCI2). The decision is made on a block basis using the same blocks as the motion estimation. Smaller blocks improve the interpolation quality but also increase the amount of side information. Now there are two streams of side information which can be multiplexed into each other: the motion vectors necessary for the upper branch and the segmentation information to determine in which state the switch between the two branches is.

5.5 Experiment Results

The experiments consist of a comparison between the different subsampling schemes discussed in this chapter and a more detailed investigation of motion compensated sub-Nyquist sampling. Fixed lattice subsampling, sub-Nyquist sampling and motion compensated sub-Nyquist sampling are investigated and compared with each other. Also motion compensated sub-Nyquist sampling with fractional motion estimation accuracy is considered. The experiments are done using the *MOBILE* sequence.

5.5.1 Implementation details

Three different schemes are used for the experiments. Several details of these systems are described in this section. The first system uses spatial fixed lattice subsampling. A block diagram of this system was given in Chapter 2. A quincunx subsampling lattice is used with $Q_x = Q_y = 1$, yielding a data reduction factor of 2. This system is used as the fall-back mode in all the systems described in this section. The purpose of the system is also to serve as a reference for the other systems.

The second system is a sub-Nyquist sampling scheme. Again a quincunx subsampling lattice is used which is shifted for each image, so all the cosets of the lattice are used. Instead of spatial filtering, a temporal interpolation filter with two filter taps is used. With this system

also a data reduction of two is achieved. The input image sequence is compensated for the global pan in order to make the image sequence more stationary. This is especially useful if the entire sequence is moving which is the case for the *MOBILE* sequence.

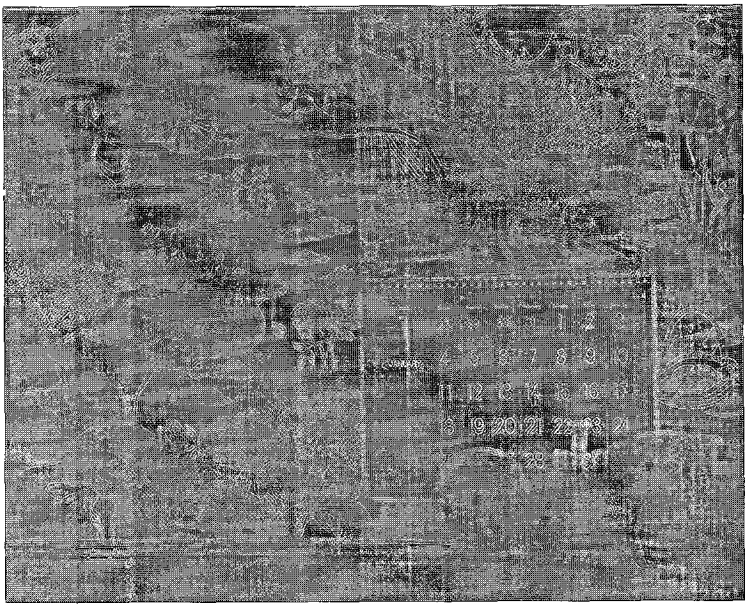
The third system employs motion compensated sub-Nyquist sampling. Two variants of this system are considered. One uses motion vectors with pixel accuracy, whereas the other one uses motion vectors with half pixel accuracy. In Section 5.4.3 it was shown that to utilize fractional accuracy, spatial upsampling is necessary. For this purpose 7-taps maximally-flat filters were used, so the original input samples are not changed by the filtering process. For the interpolation, a motion compensated interpolation filter with two taps was used.

5.5.2 Comparison between different subsampling schemes

In Figure 5.18, the stretched differences between the second image of the *MOBILE* sequence and the different interpolated images are shown. To evaluate the performance under all conditions, no fall-back mode is used. In Figure 5.18(a) the result for fixed lattice subsampling is shown. As can be expected, the difference image shows that the errors are mainly concentrated around the object edges thus causing a noticeable loss of resolution. This has already been observed in Chapter 2. In Figure 5.18(b) the difference in the case of sub-Nyquist sampling is shown. As can be seen sub-Nyquist sampling works well for a large region of the image but fails if the actual motion deviates from the global pan.

The results of the motion compensated sub-Nyquist sampling system are shown in the Figures 5.18(c) and 5.18(d). In Figure 5.18(c) integer accuracy is used for the motion estimate and in Figure 5.18(d) half pixel accuracy is used. The results show that motion compensated sub-Nyquist sampling works better than non-adaptive sub-Nyquist sampling. The area with low error values now consists of the regions which move according to the global pan and regions for which an accurate motion estimate is possible. In the case of a good motion estimate only the noise component is present, whereas in the case of a bad motion estimate more clearly visible artifacts are introduced. The results also show that if half pixel accuracy is used for the motion estimate the result improves. An image detail is shown for the pixel motion estimation accuracy case in Figure 5.19. This detail also shows the improvement in visual quality compared to fixed lattice subsampling because of the preservation of the high-frequency contents.

The SNR and the MSE for the different coding methods are shown in Table 5.1. A distinction is made between the result achieved with and without a fall-back mode. As can be expected, motion adaptive sub-Nyquist sampling with half pixel motion estimation accuracy results in the smallest error value because the temporal correlation can be exploited more efficiently. This has also a positive influence on the subjective quality because the area subsampled with a fixed subsampling lattice is decreased. We see that the fall-back mode improves the numerical results for all the systems and avoids clearly visible artifacts in the interpolated image. A remaining artifact is aliasing caused by the noise which is folded back to the baseband and which is not properly attenuated by the motion compensated interpolation filter.

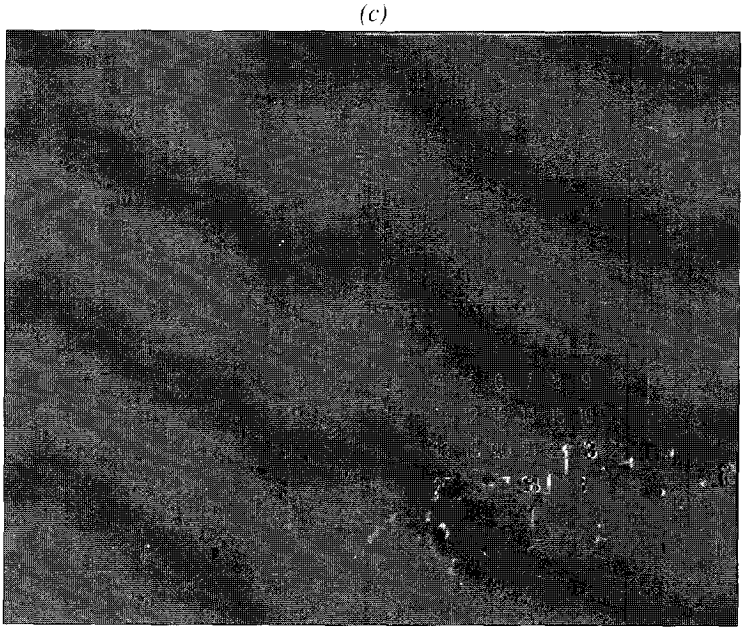


(a)

(b)

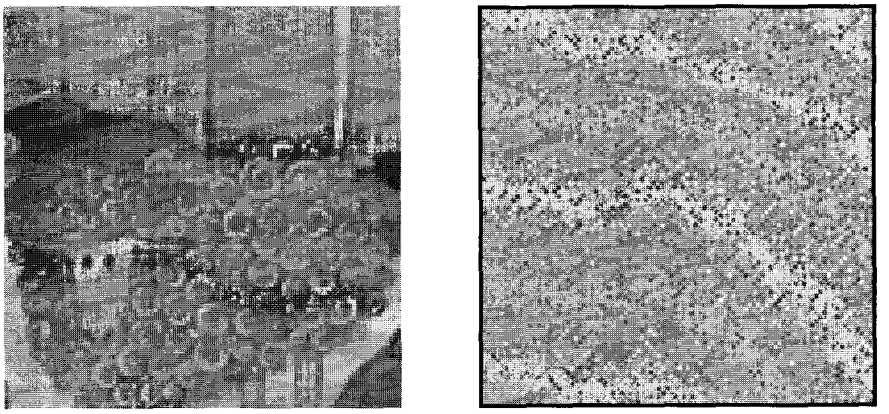
Figure 5.18: Stretched error using: (a) quincunx subsampling, (b) sub-Nyquist sampling.

(c)



(d)

Figure 5.18 (continued): Stretched error using: (c) motion compensated sub-Nyquist sampling with integer motion vector accuracy (d) motion compensated sub-Nyquist sampling with half pixel motion vector accuracy.



(a) (b)

Figure 5.19: Detail of image after using motion compensated sub-Nyquist sampling: (a) coded image (b) stretched difference.

Table 5.1 Average SNR and MSE of the different coding methods using the MOBILE sequence.

Coding method	*Without fall-back:*		*With fall-back:*	
	SNR (dB)	*MSE*	*SNR(dB)*	*MSE*
Quincunx subsampling	18.9	21.9	-	-
Sub-Nyquist sampling	15.5	49.2	19.9	17.8
Motion compensated sub-Nyquist sampling (pixel accuracy)	19.5	19.2	20.4	15.5
Motion compensated sub-Nyquist sampling (half-pixel accuracy)	21.8	11.3	22.7	9.4

In Section 5.4.1 we argued that the compression factor of motion compensated sub-Nyquist sampling is not constant. However, in practice, the compression factor deviates very little around the desired value of two. For motion compensated sub-Nyquist sampling with pixel accuracy, the average compression factor was exactly equal to 2 and with half-pixel accuracy the average compression factor was equal to 2.02.

5.5.3 Motion compensated sub-Nyquist sampling

In this section the performance of motion compensated sub-Nyquist sampling is investigated in more detail. We use motion compensated sub-Nyquist sampling with integer motion vector accuracy combined with fixed lattice quincunx subsampling as a fall-back mode.

First we investigate the behavior of the switch which either decides to use motion compensated sub-Nyquist sampling (MC-SNS) or fixed lattice quincunx subsampling (QNX-SS). The division between the two branches is illustrated in Figure 5.20. Also shown is the prediction error of the motion estimate. The prediction error, defined as the difference between the original image and the motion compensated previous image, is used as a reference for the quality of the motion estimate. The correlation between the classification

type of the blocks and the MSE of the motion estimate is equal to 0.94. Therefore, as the correlation between these values is high, we can state that a high prediction error causes an increase in the use of the fall-back mode, as was to be expected.

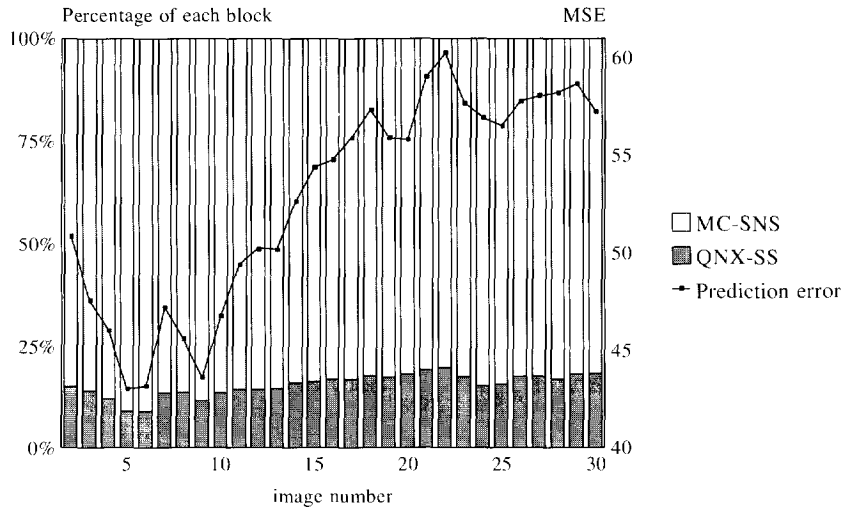


Figure 5.20: The division of the two modes as a function of the prediction error of the motion estimate.

The SNR of each branch is shown in Figure 5.21. The overall SNR is increased by adaptively combining both branches. The SNR increases because, for the regions contributing the most to the interpolation error, the non-adaptive branch is used. The SNR is also less sensitive to errors introduced by the motion estimator. The overall SNR does not fluctuate as much as the SNR of the motion adaptive branch.

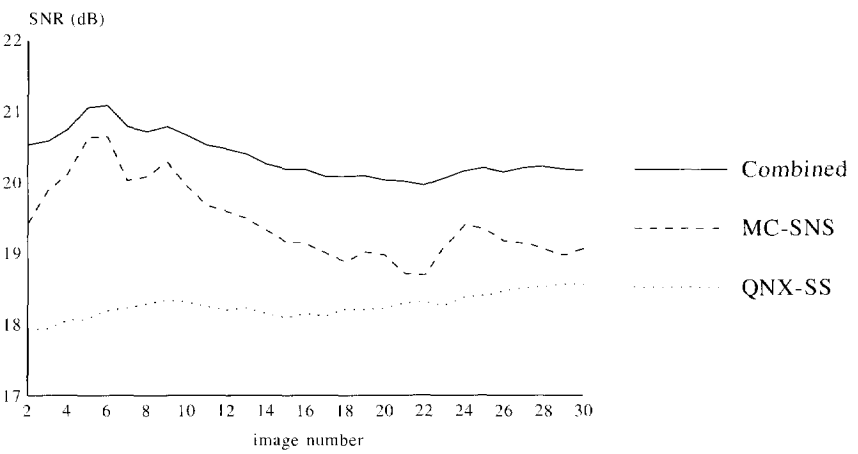


Figure 5.21: SNR of each branch and the combination of both branches.

Chapter 6

SUBSAMPLING AND TRANSFORM CODING

In this chapter we investigate the possibilities of combining subsampling with transform coding. Some of the benefits of combining the two approaches are the adaptation to the HVS, the incorporation of motion adaptivity into the transform coding system and the simplification of the coding system. In [Scha90] a combined subsampling-transform coding system is proposed based on the discrete cosine transform. In [Vos92] a different approach is taken, and subband coding is used. The discussion in this chapter is on frequency transform coding in general. However, the implementation aspects of both the discrete cosine transform coding and subband coding are discussed.

The chapter starts with a brief introduction of the basics of transform coding. The issue of the bit allocation is discussed in a separate section where we also consider the fundamental differences between subsampling and frequency transform coding. For the purpose of completeness, an overview is given of subband coding and discrete cosine transform coding. Next, some general aspects of combining subsampling with frequency transform coding are covered. Then we focus on two cases, namely the combination with either spatial subsampling or spatio-temporal subsampling. The chapter is concluded with the results of the experiments carried out with subband coding.

6.1 Transform coding

6.1.1 Principle

In a transform coding scheme, a signal is represented by a set of components which are more or less uncorrelated with each other. This is done by applying a (nearly) orthogonal transform. Uncorrelated components can be coded more efficiently than correlated components because the statistical redundancy is removed and the signal's energy is concentrated into fewer components. A generic transform coding scheme is shown in Figure 6.1. After the transform each component is coded independently and then transmitted across the channel. At the decoder the information is decoded and the inverse transform is applied, forming the reconstructed image. Data compression is achieved because the components containing much energy are generally coded with more bits than the components containing less energy.

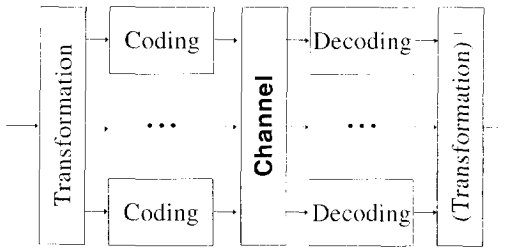


Figure 6.1: Generic transform coding scheme.

The optimal transform in the sense of a maximum decorrelation of the components is the Karhunen-Loève transform (KLT) (e.g. [Jaya84]). The KLT is signal dependent and requires correlation measurements, therefore several sub-optimal transforms which decompose the signal into different "frequency" bands have been proposed (Figure 6.2). The frequency bands are now the different transform components of the signal, so in this situation we speak of frequency transform coding. Examples of frequency transform coding are the discrete Fourier transform (DFT), the discrete cosine transform (DCT) and subband coding (SBC). In the rest of this chapter we focus only on frequency transform coding systems.

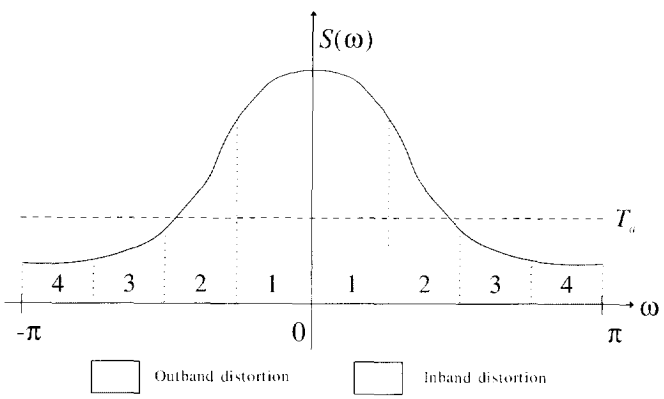


Figure 6.2: Power spectral density function of a signal divided into 4 frequency bands.

6.1.2 Bit allocation in a frequency transform coding system

In a frequency transform coding system, the frequency bands are quantized with non-uniform quantizers. The number of quantization levels is coupled with the bit rate and can be varied for each frequency band. During the bit allocation, quantizers have to be assigned to the different frequency bands under the constraints of a minimum mean square error and a constant bit rate. The convex hull allocation algorithm discussed in Chapter 4 was originally designed for this purpose [West88a]. It optimally assigns the quantizers over the frequency bands, based on the variance and the distribution of the transform coefficients in each frequency band. The variance is an indicator of the energy contribution of each frequency band to the total signal. More bits are assigned to frequency bands with a high variance because these frequency bands contain more energy and are therefore more significant in the

overall signal. The distribution of the transform coefficients is an indicator of the redundancy within each frequency band. If the distribution is peaked around a single value then the frequency bands can be coded more efficiently than in the case where the distribution is flat.

The division of the bits over the different frequency bands is a fundamental difference between fixed lattice subsampling and transform coding. In a fixed lattice subsampling scheme, a part of the spectrum is discarded without taking the power spectral density function into account. We saw in Chapter 2 that this approach is only optimal if the power spectral density function is monotonically decreasing. In a frequency transform coding scheme, a frequency band is quantized based on the activity in the frequency band. Effectively this means that an *analysis* is made of the power spectral density function and it is no longer necessary for this function to be monotonically decreasing.

The area under the power spectral density function in each frequency band is proportional to the variance in that frequency band. Therefore the variance σ_k^2 of the kth frequency band, ranging from ω_k^s to ω_k^e, is equal to

$$\sigma_k^2 = \frac{1}{\pi} \int_{\omega_k^s}^{\omega_k^e} S(\omega) d\omega \tag{6.1}$$

where we have taken into account the symmetry of $S(\omega)$ around the origin. We have already seen that the variance of the frequency bands is one of the inputs of the bit allocation algorithm. From the rate-distortion theory [Berg71] [Jaya84] we know that in the optimal case the bit allocation acts as a threshold operation on the power spectral density function. If the power spectral density function lies below the threshold T_a in a particular frequency band, then the variance in that frequency band is too low, and no bits are assigned to that frequency band. As the bit rate decreases, the value of T_a increases and more frequency bands are assigned no bits. We see that in Figure 6.2 no bits are assigned to the frequency bands 3 and 4 because they lie entirely below the threshold. In the two-dimensional case, the threshold does not describe a line but a plane acting on the two-dimensional power spectral density function. The underlying principle, however, remains the same.

The total distortion D_{ftc} introduced during the quantization stage in a frequency transform coding scheme is equal to ([Jaya84]):

$$D_{ftc} = \frac{1}{2\pi} \int_{-\pi}^{+\pi} \min\left[S(\omega), T_a\right] d\omega \tag{6.2}$$

The area under the power spectral density function in the frequency bands 3 and 4 in Figure 6.2 is called the outband distortion as defined in Chapter 2. The frequency bands 1 and 2 are coded, and the distortion in these frequency bands is the inband distortion. We see that the distortion is as far as possible evenly distributed over the entire frequency range. In [Berg71] it is shown that this is the optimal solution in the sense of the rate-distortion theory. This is again in contrast to fixed lattice subsampling. We showed in Chapter 2 that in a fixed lattice subsampling scheme the inband and the outband distortion depends on the prefilter and the interpolation filter. If these filters have an ideal frequency response, the inband distortion is

equal to zero. As a consequence, the distortion is not evenly distributed over the entire frequency range.

6.1.3 Subband coding and DCT coding

Two popular frequency transform coding systems are discussed in this section, namely subband coding (e.g. [Wood86]) and DCT coding. In a subband coding system, the signal is globally divided into different frequency bands, and each band-pass filtered and downsampled frequency band is called a subband. In an image coding system, each image is divided into subbands of different spatial frequencies. A required property of the filters used for the subdivision into the different subbands is that aliasing introduced during the subdivision into subbands is (almost) entirely canceled out when the subbands are combined to form the reconstructed signal. A special class of FIR filters satisfying this criterion are the *quadrature mirror* filters (QMF filters) [John80]. Attractive properties of these filters are that the filtering can be implemented efficiently and that the same filters can be used for decomposition and reconstruction.

The spatial correlation in the subbands is usually low, except for the lowest subband which contains a low-pass version of the input signal. Therefore the subbands are typically PCM coded with non-uniform quantizers, except for the lowest subband which is DPCM encoded. The bit allocation assigns the quantizers to the different subbands where the variance in each subband determines the number of quantization levels. The redundancy after quantization is removed by applying a variable length coding on the quantizer output symbols. At the decoder, first the inverse variable length coding and the inverse quantizer are applied. After that the subbands are interpolated and combined, forming the reconstructed signal. The interpolation is again done with QMF filters. It should be noted that here we discussed only one of the possible implementations of a subband coding system. Numerous other possibilities have been proposed in the past (e.g. [Wood91], [West88b] (vector quantization), [Diab90] (block bit allocation), [Zafa93] (wavelets)).

In a discrete cosine transform coding system, the different frequency transform coefficients are obtained by convoluting the input signal with sampled versions of cosine functions. The frequency band associated with a coefficient is determined by the frequency of the cosine function. The convolution is done on a block basis: this in contrast to subband coding where the entire image is used. The discrete cosine transform can therefore be implemented more efficiently. A popular approach taken in DCT coding schemes is to assign the bits over the different transform coefficients for each block without taking the other blocks into consideration [Penn93]. This does not guarantee that the minimum mean square error measured over the entire image is achieved. The method chosen here is to combine the corresponding transform coefficients from each block into frequency bands. These frequency bands are treated in the same way as the subbands in subband coding, and the transform coefficients can be coded using a global bit allocation which takes all the blocks into consideration.

6.2 Subsampling combined with frequency transform coding

Subsampling is combined with frequency transform coding in this section. Distinction is made between the combination of transform coding with spatial subsampling and the combination with spatio-temporal subsampling. First we consider some general aspects relevant for both situations. We here call a frequency transform coding system without any subsampling a *standard* frequency transform coding system.

An advantage of combining subsampling with frequency transform coding is that the subsampling can be used to selectively exclude a part of the spectrum. The remaining part of the spectrum can now be coded more efficiently. For example, diagonal frequency components can be discarded because of the reduced sensitivity of the HVS to these frequency components. As a result, the distortion of the horizontal and vertical frequency components is smaller than in a standard frequency transform coding system. As already pointed out in Chapter 2, subsampling the chrominance components prior to coding is already common practice in a lot of applications [MPEG92] [Penn93]. Combining subsampling with frequency transform coding can also serve to adapt the image dimensions at the encoder side to the dimensions of the display device at the decoder side. If the image is too big for the display device, then the image can be subsampled to match the dimensions of the display device. Another advantage is that motion adaptivity can be brought into a frequency transform coding system when it is combined with spatio-temporal subsampling. The main motivation given in [Scha90] is the reduced complexity of the frequency transform coding system, because the frequency transform is done on a smaller image.

6.2.1 Implementation aspects

There are two possible configurations for combining subsampling with frequency transform coding. The first configuration (Figure 6.3(a)) is to place the subsampling stage after the frequency transform coding stage and to subsample the different frequency bands. This is only useful for the lowest frequency band which contains a low-pass version of the image. It was pointed out in Section 6.1.3 that there is still some spatial correlation left in the lowest frequency band. The spatial correlation in the other subbands is usually low and subsampling is not useful. This configuration is therefore rejected because of its limited possibilities. The second configuration (Figure 6.3(b)) is to consider the subsampling stage as a front end of the frequency transform coding stage. The output of the subsampling is a low-pass version of the input image. Because there is still some spatial correlation left, additional coding is appropriate. In the rest of this chapter we confine ourselves to this configuration.

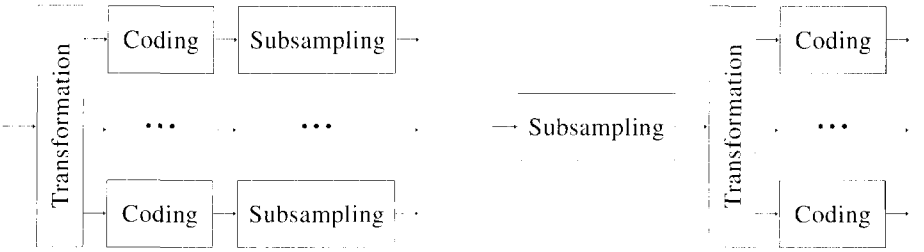


Figure 6.3: System configurations: (a) Frequency transform coding before subsampling and (b) Subsampling before frequency transform coding.

Another implementation aspect is the sampling lattice used as input for the frequency transform coding stage. In the previous chapters we saw that there are two possibilities: a fixed sampling lattice or an adaptive subsampling lattice. In this particular application the principal advantage of a fixed sampling lattice over an adaptive subsampling lattice is the ease of implementation. It is much easier to define a frequency transform on a fixed sampling lattice than on an adaptive sampling lattice. Another advantage in this situation of a fixed subsampling lattice is the homogenous correlation structure. If an image is sampled on an adaptive sampling lattice, the geometric position of the samples is not constant and neighboring pixels are less likely to be correlated with each other. Another problem of adaptive sampling lattices in this context is that it is complicated to divide the available bits over the different stages. If fixed lattice is used, then most of the coding effort is left to the frequency transform coding stage. The conclusion is that a fixed sampling lattice is better suited for this particular situation than an adaptive subsampling lattice.

We now consider how to divide the available bit rate over the two coding stages. First the one-dimensional case is investigated. If a signal is prefiltered and subsampled to a bandwidth of W_{ss} then the relative bit rate is reduced to W_{ss}/π (See Section 2.3). The total relative bit rate R_t of the entire system is equal to the product of the relative bit rate in the subsampling stage and the relative bit rate in the frequency transform coding stage:

$$R_t = \frac{W_{ss}}{\pi} \cdot R_{ftc} \tag{6.3}$$

where R_{ftc} is the relative bit rate of the frequency transform coding stage. We see that for a fixed R_t there are π/W_{ss} more bits available for the frequency transform coding stage than in a standard frequency transform coding system. Therefore the remaining part of the spectrum is coded more accurately. Therefore Equation (6.3) is a quantification of the main motivation behind combining subsampling with frequency transform coding. If an image defined on a two-dimensional sampling lattice L is subsampled to the lattice L_{ss}, then Equation (6.3) becomes

$$R_t = \frac{\text{Area}(U_{L_{ss}^R})}{\text{Area}(U_{L^R})} \cdot R_{ftc} \tag{6.4}$$

where L^R and L_{ss}^R are the reciprocal lattices of L and L_{ss}. The ratio between the areas of the unity cells in the frequency domain is the relative bit rate of the subsampling stage (see Section 2.3).

6.2.2 Spatial subsampling combined with frequency transform coding

In this section, a fixed lattice spatial subsampling system is combined with a frequency transform coding system. First, a qualitative relation between the mean square error distortion $D_{ss+ftc}(R_t)$ of the combined system and the mean square error distortion $D_{ftc}(R_t)$ of a standard frequency transform coding system is examined for the one-dimensional case, using Figure 6.4. We assume a perfect overall response of the fixed lattice subsampling system, so the subsampling system only has an outband distortion component. This component is equal to two times the area under the power spectral density function in the interval $[W_{ss},\pi]$ (see Section 2.3). If the rest of the spectrum is coded without any distortion then this is the only component contributing to the total distortion. Therefore the distortion of the subsampling stage imposes a lower bound on the distortion in the total system.

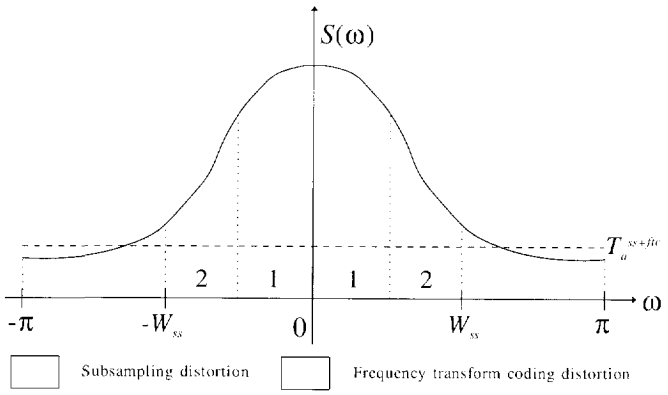


Figure 6.4: Power spectral density function of a signal coded with subsampling and subband coding.

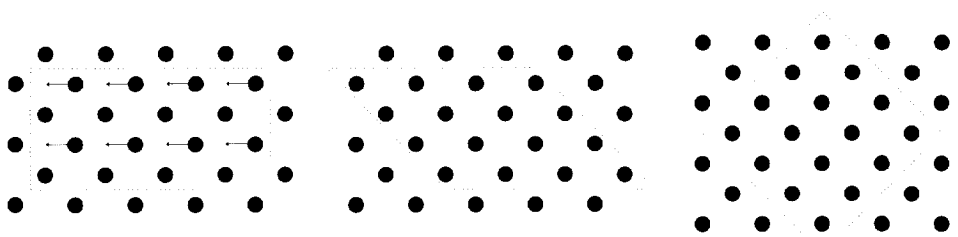
First we define the threshold T_a^{ss+ftc} as the allocation threshold of the combined system. If in the interval $[W_{ss},\pi]$ part of the power spectral density function is above T_a^{ss+ftc}, a part of the spectrum is discarded which would otherwise be coded in a standard frequency transform coding system. The distortion of the combined system is then higher than the distortion of a standard subband system, otherwise contradicting the fact that the bit allocation minimizes the total distortion. If the bit rate is lowered, then the allocation threshold increases. At some point the part of the power spectral density function discarded in the subsampling stage is completely below T_a^{ss+ftc}. The part discarded in the subsampling stage would now also be discarded in a standard frequency transform coding system. In this situation the distortion of the combined system is equal to the distortion of the standard frequency transform coding system. As the allocation threshold is coupled to the bit rate, the conclusion is that theoretically for the same bit rate R_t, the relation between the different distortions is

$$D_{ss+ftc}(R_t) \geq D_{ftc}(R_t) \tag{6.5}$$

where the equality holds for low bit rates. In practice, the filters used in a subsampling scheme have a better response than the filters used for the splitting into frequency bands. A prefilter is designed to minimize the amount of aliasing, whereas QMF filters allow for some aliasing. Therefore for low bit rates $D_{ss+ftc}(R_t)$ is a fraction smaller than $D_{ftc}(R_t)$.

The question of whether the relative increase of the mean square error distortion in the combined system is objectionable depends on the application. If the combination of subsampling and frequency transform coding is used to suppress certain frequency components, then a deliberate choice is made and therefore the objective distortion is no longer relevant. The chrominance components usually do not contain many high-frequency components. Therefore the part of spectrum that is discarded in the subsampling stage is usually not coded in a standard frequency transform coding scheme. In this case, the equality in Equation (6.5) usually holds and combining subsampling with frequency transform coding gives a simpler system with the same distortion.

If the spatial output lattice from the fixed lattice subsampling stage is an orthogonal lattice then the DCT transform can be applied straightforwardly. In the previous chapters we saw that the quincunx lattice is a frequently used sampling lattice. If this lattice is used, the implementation of the DCT transform is no longer trivial. There are several possible ways to implement the DCT transform on a quincunx lattice [Scha90]. Figure 6.5 shows some possibilities for a block size of 4×4 pixels. In Figure 6.5(a) the pixels are treated as if they were defined on an orthogonal lattice, and the odd lines are shifted in the horizontal direction. The shifting introduces artificial high-valued vertical transform coefficient for vertical edges in the image and therefore reduces the coding efficiency. No shifting is required for the configurations shown in the Figures 6.5(b) and (c). However, there are two main drawbacks for the configuration from Figure 6.5(c). The first drawback is that the correlation between the pixels is lower than the correlation of the configuration from Figure 6.5(b). The other disadvantage is that the memory requirements are larger because more image lines are involved.



(a) *(b)* *(c)*

Figure 6.5: Configuration of DCT blocks: (a) rectangular blocks, (b) diagonal blocks and (c) rotated blocks.

The division of an image sampled with a quincunx lattice into different subbands is also not straightforward. The problem is that QMF filters are designed to cancel the aliasing of images sampled on an orthogonal lattice and not on a quincunx lattice. In [Bamb90] an algorithm is described to transform a one-dimensional QMF filter designed for an orthogonal lattice to a one-dimensional directional filter suitable for a quincunx lattice. This new filter is again a QMF filter so the aliasing is still canceled. Because the filter is separable it can be implemented in an efficient way.

6.2.3 Spatio-temporal subsampling combined with frequency transform coding

In this section we extend the system as described in the previous section by using spatio-temporal subsampling instead of spatial subsampling. Two possibilities for spatio-temporal subsampling were discussed in Chapter 5: sub-Nyquist sampling and motion compensated sub-Nyquist sampling. The choice between these two methods is based on the discussion from Section 6.2.1 were it was argued that the input lattice of the frequency transform coding stage should be a fixed sampling lattice. A sub-Nyquist sampling can be designed in such a way that the output lattice is a fixed sampling lattice (e.g. the MUSE system). However in a motion compensated sub-Nyquist sampling system the sampling lattice is adapted to the motion, causing the sampling lattice to be non-uniform. In conclusion, motion compensated sub-Nyquist sampling is not suited to this application and only standard sub-Nyquist sampling is used.

The sub-Nyquist subsampling stage is placed before the frequency transform coding stage, and the signal is first subsampled without any prefiltering. After that the signal is divided into the different frequency bands and each frequency band is coded. Equations (6.3) and (6.4) are also valid in this situation. Therefore the number of available bits for the coding of the different frequency bands is increased. At the decoder, the signal is reconstructed from the different frequency bands. A temporal interpolation filter is used afterwards to cancel the alias caused by the absence of a prefilter.

First we consider the relative performance of the combined system compared with a standard frequency transform coding system. In this discussion we ignore the possible aliasing errors introduced in the frequency transform coding stage, as the aliasing introduced in the sub-Nyquist sampling stage has a greater magnitude. In Figure 6.6 we see that after the subsampling the high frequency components fold back into the baseband spectrum ranging from $[0, W_{ss}]$ because no prefilter was used. Some of the high-frequency components are present in frequency band #1 whereas some are folded into frequency band #2. The corresponding parts of the power spectral density function are added to each other so it is no longer possible to make a distinction between the aliasing components and the low-frequency components. As a result the exact location of the allocation threshold T_a^{sn+ftc} is not as precisely known as in Figure 6.4. If all the frequency bands are coded, the aliasing introduced in the sub-Nyquist sampling stage is canceled out by the temporal interpolation filter at the decoder. The distortion is lower than the distortion of a standard frequency transform coding system, because effectively a larger part of the spectrum is coded. However, if a frequency band is not coded, the corresponding aliasing is not canceled. Now the distortion is higher

than a standard frequency transform coding system. The magnitude of the quality gained by combining the two systems depends on the magnitude of the aliasing. If the region from $[W_{ss}, \pi]$ does not contain much energy then the gain is low.

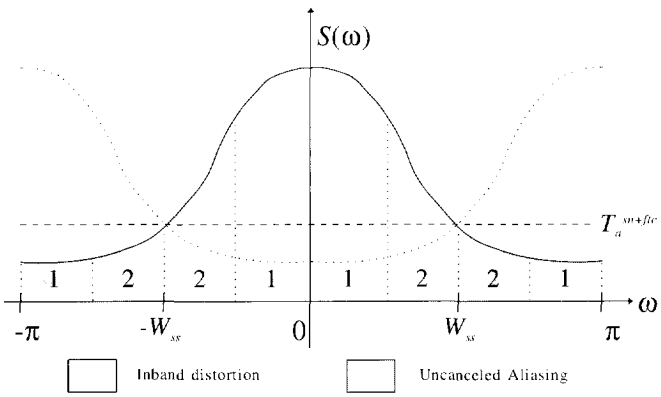


Figure 6.6: Power spectral density function of a signal coded with sub-Nyquist sampling and subband coding.

We saw in Chapter 5 that in a sub-Nyquist sampling system there should always be a fall-back mechanism for the non-stationary parts of the image sequence. In this case, spatial subsampling combined with frequency transform coding as discussed in Section 6.2.2 can be used as a fall-back mode. In [Scha90] an *a priori* mechanism is used to detect whether the fall-back mode is necessary or whether sub-Nyquist sampling can be used. The motion is detected by taking the difference between successive images on a pixel basis and averaging the values over a block. When the average value exceeds a certain threshold, spatial subsampling is used.

An alternative method is to use an *a posteriori* mechanism instead. First the average of two successive images is computed thus forming one single image. For stationary parts the combined image contains no artifacts, but for non-stationary parts the combined image is distorted. Now the average distortion between the combined image and the original images is computed on a block basis. Next the original images are prefiltered and the average distortion is computed. Based on these distortion values, a choice is made between sub-Nyquist sampling and spatial subsampling.

In a DCT coding system the image is already divided into blocks and the division between sub-Nyquist sampling and the fall-back can be based on these blocks. No information from neighboring blocks is necessary at the encoder and the decoder. If subband coding is used, the transform filters are applied to the whole image. This complicates the division into blocks because information from neighboring blocks is required. At the encoder the blocks from the two coder branches are multiplexed into on single image. After that the combined image is decomposed into different subbands. If the neighboring blocks do not belong to the same branch of the coding scheme, incorrect information is used. The consequence is that some

artificial frequency components are introduced. The same problem arises at the decoder where the subbands have to be interpolated.

6.3 Experiment results

In the previous sections both DCT coding and subband coding were discussed in combination with subsampling. In this section experiments are done only for subband coding. The reason for this is that although there are some implementation differences, no significant qualitative difference is expected between the two systems. This is because both systems as implemented in this case use a global bit allocation to assign the different quantizers to the frequency bands.

6.3.1 Spatial subsampling combined with subband coding

In this section fixed lattice spatial subsampling is combined with subband coding. The fixed lattice subsampling stage consists of a quincunx subsampling lattice with $L = 1$ and $M = 1$. Hence the data reduction factor of this stage is equal to 2. The 21-taps prefilter and 7-taps interpolation filter described in Chapter 2 are used. For the subband splitting, the scheme from Figure 6.7 is used *without* subband #9, because this part of the spectrum is not present in the quincunx subsampled image. The splitting is done with a 16-taps one-dimensional QMF filter denoted as the filter 16C in [John80]. Max-Lloyd quantizers are used for the quantization of the subbands. The coding scheme is compared with a standard subband coding scheme using the splitting scheme from Figure 6.7 with subband #9 included.

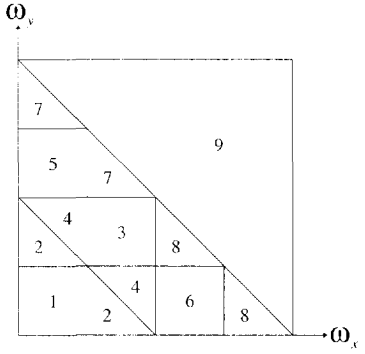


Figure 6.7: Subband splitting scheme. Subband #9 is not present in the combined system.

The results obtained with the *LENA* test image are given in Figure 6.8 for both the combined system and standard subband coding. In Figure 6.8(a) the SNR is shown for both systems and in Figure 6.8(b) and 6.8(c) the number of bits assigned to each subband for the different bit rates. As was expected from the theoretical analysis, the two SNR curves converge as the bit rate decreases. For high bit rates the combined system converges to an upper bound which is equal to the distortion of a coding scheme without subband coding. We see in the Figures 6.8(b) and 6.8(c) that for high bit rates the number of bits assigned to the subbands in the combined system is higher than in a standard subband coding system. Hence more emphasis

subband coding system. Hence more emphasis is placed on the remaining subbands. As the bit rate is lowered, we see that at a bit rate of 1.4 bits per pixel the difference between the combined system and standard subband coding decreases abruptly. In Figure 6.8(b) we see that this is the point where no more bits are assigned to subband #9. If we look at the assignment of the bits to the different subbands then we see that from this point on approximately the same number of bits is assigned to each subband. The quality difference between the systems is influenced by the different filter characteristics. At very low bit rates, the combined system is slightly better than standard subband coding. In Figure 6.9 the detail images and difference images are shown for standard subband coding and the combined system at a bit rate of 1 BPP. We see that the errors of the combined system are more concentrated around the diagonal frequency components.

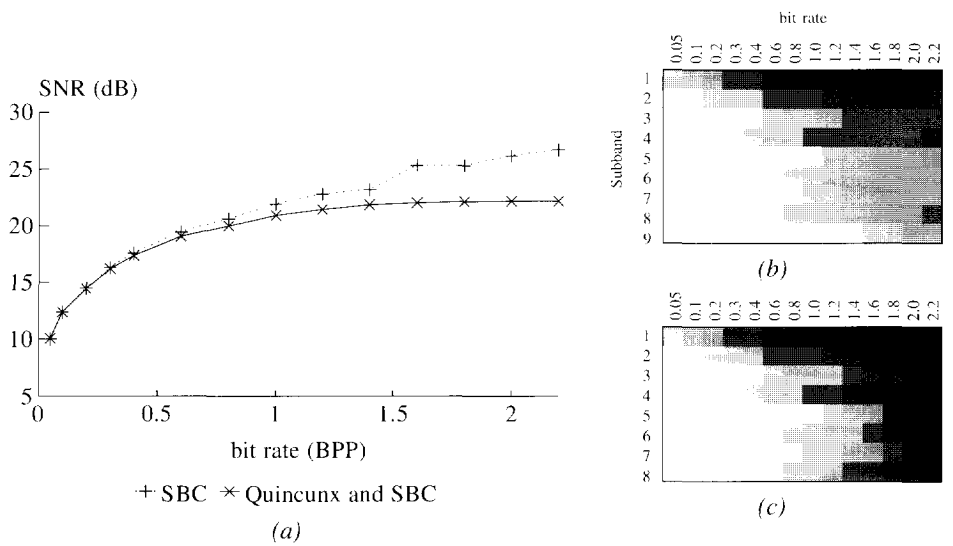


Figure 6.8: (a) SNR of quincunx subsampling combined with subband coding compared with standard subband coding. (b,c) Bit allocation with the bit rate along the horizontal axis and the subband number along the vertical axis for: (b) standard subband coding, (c) quincunx subsampling combined with subband coding. The gray value corresponds with the bit rate in each subband (white = 0 bits per pixel).

The conclusion from these experiments is that the behavior of the combined system corresponds with the expectations from the theoretical analysis. Experiments done with DCT coding gave similar results. Based on a mean square error criterion, a standard subband coding system performs better than the combined system. However, as have already been pointed out, the motivation behind the combined system is that the evaluation of the reconstructed image is not always based on a mean square error criterion. In that case the experiments show that the penalty in a mean square error sense is acceptable for a broad range of bit rates.

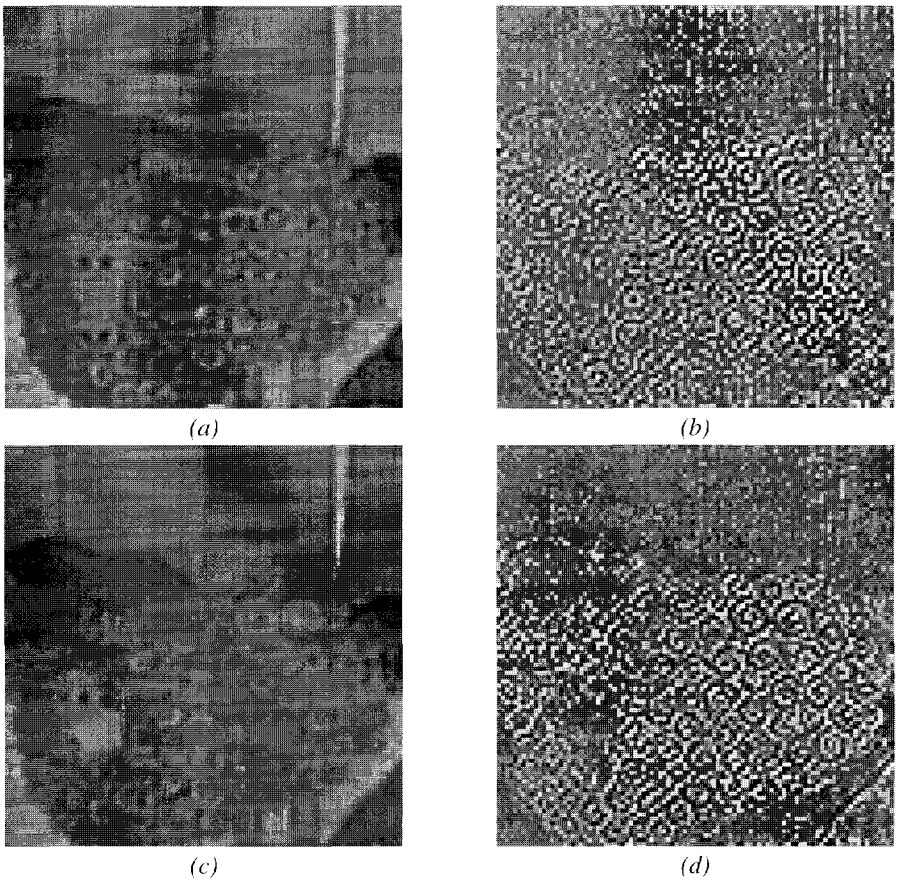


Figure 6.9: Coding result and difference image at a bit rate of 1 BPP for: (a) standard subband coding and (b) the subband coding combined with quincunx subsampling.

6.3.2 Spatio-temporal subsampling combined with subband coding

In this section sub-Nyquist sampling is combined with subband coding. A quincunx lattice is used together with its coset, so the entire image is coded over a period of two images. The same subband coder settings as in the previous paragraph are used.

The first experiment illustrates what happens to the stationary parts of an image sequence. These are the regions for which sub-Nyquist sampling is appropriate. For this we assume that the *LENA* test image is an image sequence consisting of two identical images. Thus this experiment gives the theoretical lower bound for the distortion of the coding system. The simulation results are shown in Figure 6.10. For high bit rates, sub-Nyquist sampling combined with subband coding gives better results than standard subband coding. This is the range of the bit rate for which the aliasing is canceled. Below a bit rate of 0.6 bits per pixel the two curves coincide. This coincides with the point where the highest subbands containing the aliasing information are no longer coded. The conclusion from this experiment is that

based on theoretical grounds combining spatio-temporal subsampling with frequency transform coding is meaningful.

Next, experiments are done with a complete system with a fall-back mode. For the fall-back mode a quincunx subsampling lattice is used with spatial interpolation at the decoder. The choice between sub-Nyquist sampling and the fall-back mode is based on the a posteriori method. The distortion in each branch determines which branch is used. The input image sequence is compensated for the global pan.

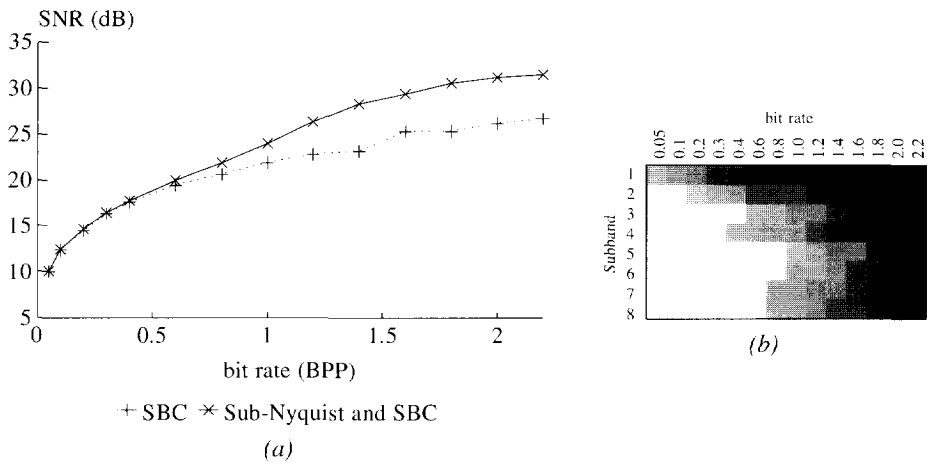


Figure 6.10: Sub-Nyquist sampling combined with subband coding compared with standard subband coding: (a) SNR, (b) Bit allocation with the bit rate along the horizontal axis and the subband number along the vertical axis (see also Figure 6.8).

The results are shown in Figure 6.11 for the first two images of the *MOBILE* sequence. We observe that sub-Nyquist sampling combined with subband coding is only better than standard subband coding for a small range of bit rates. This was expected for the low bit rates where the results suffer from uncanceled aliasing. For the high bit rates, there are several aspects which limit the quality of the combined system. First of all some parts of the image are coded with the fall-back mode. The distortions in these parts influence the overall distortion in a negative way. In Chapter 3 and in Chapter 5 we saw that the noise is the remaining error part after temporal interpolation of a stationary image. This error imposes a lower bound on the overall distortion.

Because of the interpolation stage after the frequency transform decoding stage, the artifacts introduced during the frequency transform coding are magnified. The artifacts introduced by the uncanceled aliasing are therefore clearly noticeable. The high frequency noise components are folded back into the low frequencies and are also clearly noticeable. The conclusion which can be drawn from these experiments is that although the theoretical analysis was promising, in practice sub-Nyquist sampling combined with subband coding is not worth pursuing further.

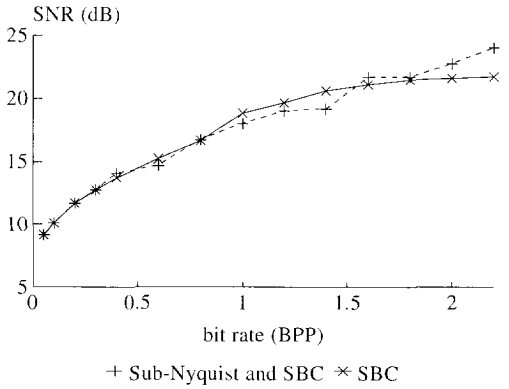


Figure 6.11: Result of sub-Nyquist sampling combined with subband coding for the first two images of the MOBILE sequence. Also shown are the results for a standard subband coding scheme.

Chapter 7

CONCLUSIONS

In this chapter we compare the different subsampling systems presented in this thesis. We base the comparison on the set of consistent experiments described in Chapters 2 to 5. The experiment results are also compared with the results obtained with another sequence, namely the *KIEL* sequence. The first image of this sequence, which has a length of 20 images, is shown in Figure 7.1. The image dimensions are the same as for the *MOBILE* sequence. The sequence describes a camera zoom on a highly detailed scene with a significant number of edges. This is the worst-case situation for most of the subsampling methods discussed in this thesis as the assumption of only translational motion is violated. Therefore this sequence is an indicator for the robustness of the different methods.

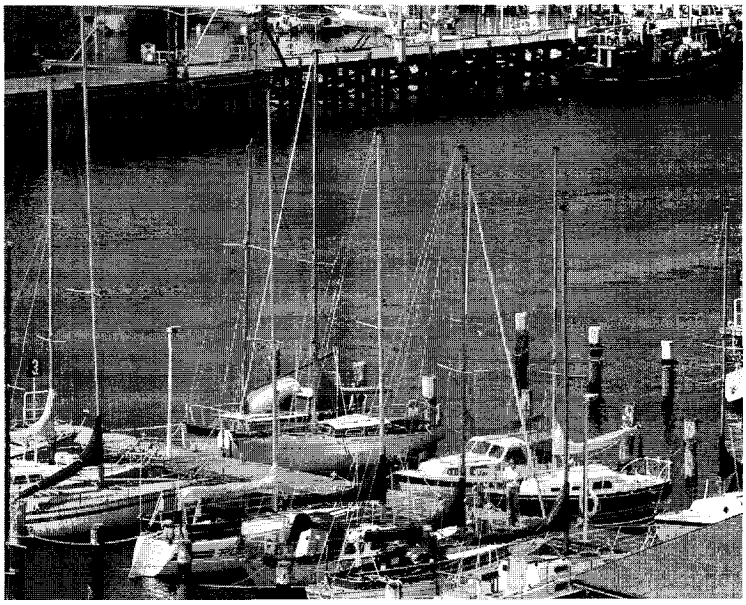


Figure 7.1: First image of KIEL sequence.

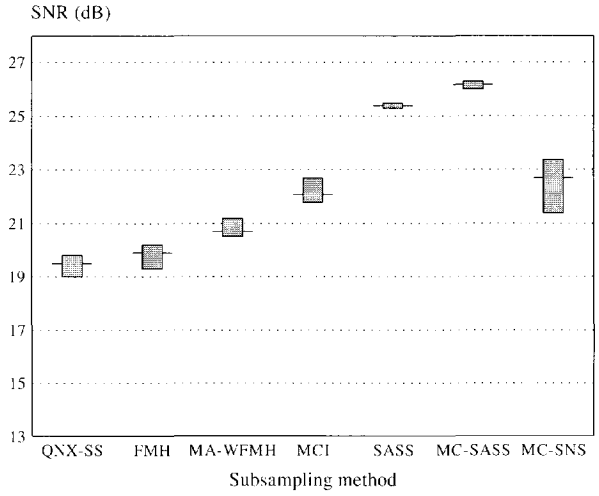
In Figure 7.2(a), the average SNR of the different subsampling systems from the previous chapters are compared using the *MOBILE* sequence. To give an indication of the usefulness of the average value, given also are the maximum and minimum SNR values. The data reduction factor is equal to two. Each time, the best representative from a class of methods is chosen. A reference is given as to in which chapter a description of the specific system can be found.

The first three systems are fixed lattice subsampling systems with quincunx subsampling. The worst performance is obtained for the simple system with linear interpolation (*QNX-SS*). The result is marginally improved by using a spatially adaptive interpolation filter such as the FMH filter (*FMH*). A further improvement is obtained when motion information is taken into account, which results in the use of the motion adaptive weighted FMH filter (*MA-WFMH*). The fourth system (*MCI*) uses temporal subsampling and relies completely on motion information for the interpolation. This system gives the best result among all the fixed lattice subsampling systems. Even in regions where the spatial correlation is low, the temporal correlation can still be high. A motion compensated system can benefit from this property.

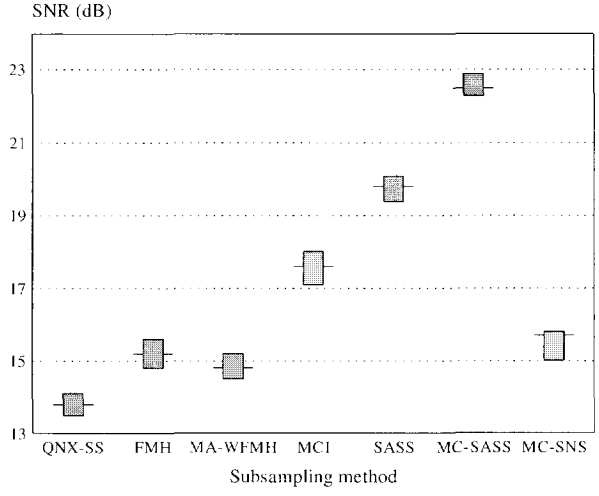
The last three systems use adaptive subsampling lattices instead of a fixed lattice. These systems all outperform the fixed lattice subsampling systems, because better use can be made of the local signal structure. Motion compensated sub-Nyquist sampling (*MC-SNS*) does not differ much from the system with temporal subsampling and motion compensated interpolation as both systems rely on the motion vector. The additional benefit of the motion compensated sub-Nyquist sampling system is that because the images are not entirely discarded also some spatial information is available to the decoder.

A significant improvement is obtained if spatially adaptive subsampling (*SASS*) is used, as the difference in the SNR between the spatially adaptive subsampling systems and the other subsampling systems is quite large. This is because an analysis is made of the local image structure, whereas, for example, the *MC-SNS* system implicitly assumes that the all the parts of the image are equally important. On the basis of spatial analysis of the local image structure, a better trade-off between quality and the compression factor can be made. If we extent the *SA-SS* system with the use of motion information to a motion compensated spatially adaptive subsampling system (*MC-SASS*), the overall result improves even further. The sampling lattice is now based on a spatio-temporal analysis of the local image structure.

The results of the experiments with the *KIEL* sequence are given in Figure 7.2(b). Because of the fine details, the quality after quincunx subsampling combined with linear interpolation is much lower than the *MOBILE* sequence. The results are somewhat improved if an FMH filter is used. The *MA-WFMH* method, which relies partly on the motion estimate, does not give a further improvement. The first substantial improvement is obtained if motion compensated interpolation is used. Especially with a camera zoom, the use of fractional accuracy and a 3-taps interpolation filter are advantageous. The *SASS* and *MC-SASS* systems give the best results. The *MC-SASS* can still benefit from the motion information, as is indicated by the difference between the two spatially adaptive systems. However, there is still some difference if we compare the results with those obtained with the *MOBILE* sequence. Motion compensated sub-Nyquist sampling suffers the most from the complex motion and the large amount of detail information. This system was for the *MOBILE* sequence better than the *MCI* system, but for the *KIEL* sequence temporal subsampling is favorable, because of the longer temporal filters.



(a)



(b)

QNX-SS : *Quincunx subsampling with linear interpolation (Ch. 2).*
FMH : *FIR/median hybrid filter (Ch. 3).*
MA-WFMH : *Motion adaptive weighted FIR/median hybrid filter (Ch. 3).*
MCI : *Motion compensated interpolation (Ch. 3).*
SASS : *Spatially adaptive subsampling (Ch. 4).*
MC-SASS : *Motion compensated spatially adaptive subsampling (Ch. 4).*
MC-SNS : *Motion compensated sub-Nyquist sampling (Ch. 5).*

Figure 7.2: Comparison of the SNR of the different subsampling methods for: (a) the MOBILE sequence and (b) the KIEL sequence. The lines indicate the mean values and the bars the maximum and minimum values.

The purpose of all the subsampling systems discussed was to preserve as much as possible the high-frequency detail information. The images in Chapter 2 show that in the *QNX-SS* system the high-frequency information was not recovered. In Chapter 3 we saw that the systems with a fixed subsampling lattice in combination with a nonlinear interpolation filter (*FMH, MA-WFMH*) did not totally succeed in preserving the high frequency information. The other systems (*MCI, SASS, MC-SS, MC-SN*) all performed equally well.

Because of the absence of a prefilter (*FMH, MA-WFMH, MC-SN, MCI*) or the use of a weak prefilter (*SASS, MC-SASS*), the noise plays an important role if we look at the visual artifacts of the different systems. The relative small variance of the noise compared to the image variance makes it difficult to detect high frequency noise and therefore no extra precautions can be taken. The consequence of this is that high-frequency noise folds back into the low-frequency components and becomes more clearly visible. The visibility is maximal in areas of the image with a constant luminance. The fixed lattice systems with nonlinear interpolation (*FMH, MA-WFMH*) suffer the most from this problem. In the other systems, except for the *MCI* system, the noise which is folded back is also the most dominating visual artifact but the magnitude of the distortion is smaller.

We have seen throughout this thesis that subsampling methods in general are not suitable for applications which requires a high compression factor. In most of the experiments, a compression factor of two was used. Only for the systems with spatially adaptive subsampling were higher compression factors possible. If, for example, we look at the distortion of the *MCI* and the *MC-SN* system for a compression factor of two, then the same distortion is reached in the *MC-SASS* system for a compression factor of approximately 5.

The required compression factor can also not be adjusted flexibly. In most systems, changing the compression factor requires the design of a new system. For example in sub-Nyquist sampling systems, a decrease of the compression factor means different subsampling lattices and longer temporal interpolation filters. The spatially adaptive subsampling systems are an exception. The compression factor can be varied freely because of the presence of a mode allocation.

BIBLIOGRAPHY

[Anne86] M.J.J.C. Annegarn et al., "HD-MAC: a step forward in the evolution of television", Philips Technical Journal, Vol. 43, No. 8, pp. 213-229, December, 1986

[Ashi88] M. Ashibe, K. Mitsushi and S. Tsuruta, "A Study on Adaptive Subsampling Methods for HDTV Signal Compression", Signal Processing of HDTV, L. Chiariglione (ed.), pp. 145-152, Elseviers Science Publishers, 1988

[Bamb90] R.H. Bamberger and M.J.T Smith, "Efficient 2-D Analysis/Synthesis Filter Banks for Directional Image Component Representation", Proceedings of the IEEE International Symposium on Circuits and System, pp. 2009-2012, May, 1990

[Belf91] R.A.F. Belfor, R.L Lagendijk and J. Biemond, "Motion Compensated Subsampling of HDTV", Proceedings of SPIE conference on Visual Communications and Image Processing '91: Visual Communication, Vol. 1601, Kou-Hu Tzou and Toshio Koga (ed.), pp. 274-284, November, 1991

[Belf92a] R.A.F. Belfor, R.L. Lagendijk and J. Biemond, "Motion Adaptive Sub-Nyquist Sampling of HDTV", Signal processing VI: theories and applications: Proceedings of EUSIPCO-92, Vol. I, J. Vandewalle (ed.), pp. 291-294, August, 1992

[Belf92b] R.A.F. Belfor, "Spatialy Adaptive Subsampling of HDTV using Motion Information", Proceedings of the SPIE conference on Visual Communications and Image Processing, Vol. 1818, pp. 585-593, November, 1992

[Belf93a] R.A.F. Belfor, R.L. Lagendijk and J. Biemond, "Subsampling of HDTV using Motion Information", in: Motion Analysis and Image Sequence Processing, M.I. Sezan and R.L. Lagendijk (ed.), Kluwer Academic Publishers, 1993

[Belf93b] R.A.F. Belfor, M.P.A. Hesp, R.L. Lagendijk and J. Biemond, "Motion Compensated Spatially Adaptive Subsampling", Proceedings of the 8th Workshop on Image and Multidimensional Signal Processing, pp. 134-135, September, 1993

[Belf94] R.A.F. Belfor, M.P.A. Hesp, R.L. Lagendijk and J. Biemond, "Spatially Adaptive Subsampling of Image Sequences", IEEE Transactions of Signal Processing, September, 1994, (to appear)

[Berg71] T. Berger, Rate Distortion Theory: A Mathematical Basis for Data Compression, Prentice-Hall, 1971

[Bern90] P. Bernard, C. Blaize M.-J. Colaitis, "HDMAC Coding Scheme and Compatibility", Proceeding of the Third International Workshop on HDTV: Signal Processing of HDTV, Vol. II, L. Chiariglione (ed.), pp. 261-271, Elseviers Science Publishers, 1990

[Bier88] M. Bierling, "Displacement Estimation by Hierarchical Block-Matching", Proceedings of SPIE Conference on Visual Communications and Image Processing '88, Vol. 1001, T. Russell Hsing (ed.), pp. 942-951, November, 1988

[CCIR82] CCIR Recommendations 601, Encoding parameters of digital television for studios, CCIR Recommendations, 1982

[Cort93] G. Cortelazzo and R. Maduchi, "On the Determination of All the Sublattices of Preassigned Index and Its Application to Multidimensional Subsampling", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 3, No. 4, pp. 318-320, August, 1993

[Croc75] R.E. Crochiere and L.R. Rabiner, "Optimum FIR Digital Filter Implementations for Decimation, Interpolation, and Narrow-Band Filtering", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-23, No. 5, pp. 444-456, October, 1975

[DeHa93] G. de Haan, P.W.A.C. Biezen, H. Huijgen and O.A. Ojo, "True-Motion Estimation with 3-D Recursive Search Block Matching", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 3, No. 5, pp. 368-379, October, 1993

[Diab90] C. Diab, R. Prost and R. Goutte, "Block-adaptive subband coding of images", Proceeding of the International Conference on Acoustics, Speech and Signal Processing, pp. 2093-2096, April, 1990

[Dubo85] E. Dubois, "The Sampling and Reconstruction of Time-Varying Imagery with Application in Video Systems", Proceedings of the IEEE, Vol. 73, No. 4, pp. 502-522, April, 1985

[Dubo92] E. Dubois, "Motion-Compensated Filtering of Time-Varying Images",

Multidimensional Systems and Signal Processing, Vol. 3, No. 2/3, pp. 211-239, May, 1992

[Elme88] P.M. Elmer, "Variable Compression Coding of HDTV Signals", Signal Processing of HDTV: Proceeding of the Second International Workshop on Signal Processing of HDTV, L. Chiariglione (ed.), pp. 279-285, Elsevier Science Publishers, 1988

[Erns88] M. Ernst and T. Reuter, "Adaptive Filtering for Improved Standards Conversion", Proceedings Second International Workshop on HDTV, pp. 449-458, Elsevier Science Publishers, February/March, 1988

[Erns91] M. Ernst, "Motion Compensated Interpolation for Advanced Standard Conversion and Noise reduction", Proceedings Fourth International Workshop on HDTV, September, 1991

[Gaar72] N.T. Gaarder, "A Note on the Multidimensional Sampling Theorem", Proceedings of the IEEE, pp. 247-248, February, 1972

[Gall68] R.G. Gallager, Information Theory and Reliable Communication, pp. 85-89, John Wiley & Sons, 1968

[Gall81] N.C. Gallagher and J.L. Wise, "A Theoretical Ananlysis of the Properties of Median Filters", IEEE Transactions on Acoustics, Speech and Signal Processing, Vol. ASSP-29, pp. 1136-1141, December, 1981

[Gers92] A. Gersho and R.M. Gray, Vector Quantization and Signal Compression, Kluwer Academic Publishers, 1992

[Giro85] B. Girod and R. Thoma, "Motion-compensated Field Interpolation from Interlaced and Non-Interlaced Grids", Proceedings of SPIE conference on Image Coding, Vol. 594, M. Kunt and T.S. Huang (ed.), pp. 186-193, 1985

[Giro88] B. Girod, "Eye Movements and Coding of Video Sequences", Proceeding of the SPIE conference on Visual Communications and Image Processing, Vol. 1001, pp. 398-405, 1988

[Giro89] B. Girod and W. Geuen, "Vertical Sampling Rate Decimation and Line-Offset Decimation of Colour Difference Signals", Signal Processing, Vol. 16, No. 2, pp. 109-127, February, 1989

[Giro93] B. Girod, "Motion-Compensating Prediction with Fractional-Pel Accuracy", IEEE Transactions on Image Communications, Vol. 41, No. 4, pp. 604-611, April, 1993

[Golz90] U. Gölz and R. Schäfer, "Considerations on the Possibility to Exchange Temporal against Spatial Resolution in Image Coding", Image Communication, Vol. 2, No. 1, pp. 39-51, May, 1990

[Guma78] C.Gumacos, "Weighting Coefficients for Certain Maximally Flat Nonrecursive Digital Filters", IEEE Transactions on Circuits and Systems, Vol. CAS-25, No. 4, pp. 234-235, April, 1978

[Haav92] P. Haavisto, Y. Neuvo and J. Juhola, "Motion Adaptive Scan Rate Up-converion", Multidimensional Systems and Signal Processing, Vol. 3, No. 2/3, pp. 113-130, May, 1992

[Hagh88] M.R. Haghiri and F. Fonsalas, "Motion Compensated Interpolation applied to HD-MAC Pictures Encoding and Decoding", Signal Processing of HDTV: Proceedings of the Second International Workshop on HDTV, L. Chiariglione (ed.), pp. 375-381, Elsevier Science Publishers, 1988

[Hagh90] M.R. Haghiri and F.W.P. Vreeswijk, "HDMAC Coding for MAC Compatible Broadcasting of HDTV Signals", IEEE Transactions on Broadcasting, Vol. BC-36, No. 4, pp. 284-287, December, 1990

[Hein87] P. Hienonen and Y. Neuvo, "FIR-Median Hybrid Filters", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-35, No. 6, pp. 832-838, June, 1987

[Hent89] C. Hentschel, "Comparison between Median Filtering and Vertical Edge Controlled Interpolation for Flicker Reduction", IEEE Transactions on Consumer Electronics, Vol. CE-35, No. 3, pp. 279-289, August, 1989

[Hesp93] M.P.A. Hesp, Optimal Spatial-Adaptive Subsampling of Images, M.Sc. thesis, TU Delft, January, 1993, (in Dutch)

[Huan63] J.J.Y. Huang and P.M. Schultheiss, "Block Quantization of Correlated Gaussian Random Variables", IEEE Transactions on Communcations Systems, Vol. 11, No. 3, pp. 289-296, September, 1963

[Huuh92] T. Huuhtanen, "Median-Based Quincunx Interpolation Filtering", Proceedings of the Fifth Workshop on HDTV, pp. 7.1-7.6, 1992

[Huuh93] T. Huuhtanen, "Median-Based Quincunx Interpolation Filtering", Proceeding of the IEEE Winter Workshop on Nonlinear Digital Signal Processing, pp. 7.1-1.1-6, 1993

[Iu92] S.-L. Iu, "Comparison of Motion Compensation using different degrees of sub-pixel accuracy for Interfield/interframe hybrid coding of HDTV Image

Sequences", Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Vol. 3, pp. 465-468, March, 1992

[Jain89] A.K. Jain, Fundamentals of Digital Image Processing, Prentice Hall, 1989

[Jaya84] N.S. Jayant and P. Noll, Digital Coding of Waveforms, A.V. Oppenheim (ed.), Prentice-Hall Signal Processing Series, 1984

[Jerr77] A.J. Jerri, "The Shannon Sampling Theorem - Its Various Extensions and Applications: A Tutorial Review", Proceedings of the IEEE, Vol. 65, No. 11, pp. 1565-1596, November, 1977

[John80] J.D. Johnston, "A Filter Family Designed for Use in Quadrature Mirror Filter Banks", Proceedings of the International Conference on Acoustics, Speech and Signal Processing, pp. 291-294, April, 1980

[Kell79] D.H. Kelly, "Motion and vision. II. Stabilized spatio-temporal threshold surface", Journal Optical Society of America, Vol. 69, No. 10, pp. 1340-1349, October, 1979

[Kish88] R. Kishimoto and N. Sakurai, "A 'High-Efficiency TCM' Bandwidth Reduction for High-Definition TV", , L. Chiarglione (ed.), pp. 129-136, Elseviers Science Publishers, 1988

[LeGa92] D.J. Le Gall, "The MPEG video compression algorithm", Signal Processing: Image Communication, Vol. 4, pp. 129-140, Elsevier Science Publishers, 1992

[Leht90] A. Lehtonen and M. Renfors, "Nonlinear Quincunx Interpolation Filtering", Proceeding of the SPIE conference on Visual Communications and Image Processing, Vol. 1360, pp. 135-142, 1990

[Mers83] R.M. Mersereau and T.C. Speake, "The Processing of Periodically Sampled Multidimensional Signals", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-31, No. 1, pp. 188-194, February, 1983

[MPEG92] ISO-IEC JTC1\SC2\WG11, Test Model 2 for MPEG 2, July, 1992, edition MPEG 92/245

[Musm85] H.G. Musmann, P. Pirsch and H.-J. Grallert, "Advances in Picture Coding", Proceedings of the IEEE, Vol. 73, No. 4, pp. 523-547, April, 1985

[Nino82] Y. Ninomiya and Y. Ohtsuka, "A Motion-Compensated Interframe Coding Scheme for Televsion Pictures", IEEE Transactions on Communications,

Vol. COM-30, No. 1, pp. 201-211, January, 1982

[Nino87] Y. Ninomiya, "An HDTV Broadcasting System Utilizing a Bandwidth Compression Technique-MUSE", IEEE Transactions on Broadcasting, Vol. BC-33, No. 4, pp. 130-160, December, 1987

[Nyqu28] H. Nyquist, "Certain topics in Telegraph Transmission Theory", AIEE Transactions, Vol. 47, pp. 617-644, 1928

[Pear91] W.A. Pearlman, "Performance Bounds for Subband Coding", in: Subband Image Coding, J.W. Woods (ed.), pp. 1-41, Kluwer Academic Publishers, 1991

[Penn93] W.B. Pennebaker and J.L. Mitchell, Joan, JPEG still image data compression standard, Van Nostrand Reinhold, New York, 1993

[Pirs83] P. Pirsh and M. Bierling, "Changing the Sampling Rate of Video Signal by Rational Factors", Signal Procecessing II: Theories and Applications, H.W. Schussler (ed.), pp. 171-14, Elseviers Science Publishers, 1983

[Renf90] M. Renfors, T. Huuhtanen, A. Nieminen and T. Koivunen, "Linear and Nonlinear Filter for Sampling Structure Conversion of Two-Dimensional Sequences", Signal Processing of HDTV, II: Proceedings of the Third International Workshop on HDTV, Vol. II, L. Chiariglione (ed.), pp. 685-694, Elseviers Science Publishers, 1990

[Reut85] T. Reuter, "Motion Adaptive Downsampling of High Definition Television Signals", Proceeding of SPIE conference on Image Coding, Vol. 594, pp. 30-40, 1985

[Reut86] T. Reuter, "Mehrdimensionale Abtastratenumsetzung", AEÜ, Vol. 40, No. 4, pp. 219-224, 1986, (in German)

[Reut89] T. Reuter, "Standard Conversion using Motion Compensation", Signal Processing, Vol. 16, No. 1, pp. 73-82, January, 1989

[Saku90] N. Sakurai and R. Kishimoto, "Mode Partition of High-Definition Television Signals Using Adaptive Subsampling", Electronics and Communciations in Japan, Vol. 73, pp. 52-62, 1990

[Scha87] G.Schamel, "Pre- and Postfiltering of HDTV Signals for Sampling Rate Reduction and Display Up-Conversion", IEEE Transactions on Circuits and Systems, Vol. CAS-34, No. 11, pp. 1432-1439, November, 1987

[Scha90] G. Schamel, "Spatio-Temporal Subsampling and Transform Coding of

HDTV Signals", Image Comunnication, Vol. 2, No. 3, pp. 305-318, October, 1990

[Sioh91] P. Siohan, "2-D FIR Filter Design for Sampling Structure Conversion", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 1, No. 4, pp. 337-350, December, 1991

[Stra76] G. Strang, Linear Algebra and its Applications, Academic Press, 1976

[Tana90] Y. Tanaka and T. Nishizawa, Compatible MUSE Systems for Terrestrial Broadcasting of HDTV Signals, NHK Science and Technical Research Laboratories, March, 1990

[Tani88] M. Tanimoto, N. Chiba, H.Yasui and M. Murakami, "TAT (Time-Axis Transform) Bandwidth Compression System of Picture Signals", IEEE Transactions on Communications, Vol. 36, No. 3, pp. 347-354, March, 1988

[Thom87] G.A. Thomas, Television Motion Measurement for DATV and Other Applications, Research Department, Engineering Division, BBC (BBC RD 1987/11), September, 1987

[Tong81] G.J. Tonge, The Sampling of Television Images, No. 112/81, May, 1981

[Tong87] G.J. Tonge, "Image Processing for Higher Definition Television", IEEE Transactions on Circuits and Systems, Vol. CAS-34, No. 11, pp. 1385-1398, November, 1987

[Vos92] W. Vos, A Combination of Spatio-Temporal Subsampling and Subband Coding of Image Sequences, M.Sc. thesis, TU Delft, April, 1992, (in Dutch)

[Vree89] F.W.P. Vreeswijk and M.R. Hagiri, "HDMAC Coding for MAC Compatible Broadcasting of HDTV Signals", Signal Processing of HDTV, II: Proceedings of the third International Workshop on HDTV, pp. 187-194, Elsevier Science Publishers, September, 1989

[Wang89] F.-M. Wang, D. Anastassiou and A.N. Netraval, "Time-Recursive Motion Compensated Deinterlacing", Proceedings of the Third International Workshop on HDTV, September, 1989

[Wang90] F.-M. Wang and D. Anastassiou, "Time-Recursive Motion Compensated Deinterlacing", Signal Processing of HDTV: Proceedings of the Third International Workshop on HDTV, Vol. II, L. Chiariglione (ed.), pp. 635-642, Elsevier Science Publishers, 1990

[Wang92] Q. Wang and R.J. Clarke, "Motion Estimation and Compensation for Image Sequence Coding", Image Coding, Vol. 4, pp. 161-174, 1992

[West88a] P.H. Westerink, J.Biemond and D.E. Boekee, "An optimal Bit Allocation Algorithm for Sub-band Coding", Proceedings of the International Conference on Acoustics, Speech and Signal Processing, pp. 757-760, April, 1988

[West88b] P.H. Westerink, D.E. Boekee, J. Biemond and J.W. Woods, "Subband Coding of Images using Vector Quantization", IEEE Transactions on Communications, Vol. COM-36, pp. 713-719, June, 1988

[Will91] P. Willemin, T.R. Reed and M. Kunt, " Image Sequence Coding by Split and Merge", IEEE Transactions on Communications, Vol. COM-39, No. 12, pp. 1845-1855, December, 1991

[Wood86] J.W. Woods and S.D. O'Neil, "Subband Coding of Images", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-34, No. 5, pp. 1278-1288, October, 1986

[Wood91] J.W. Woods, Subband Image Coding, Kluwer Academic Publisher, 1991

[Yli91] O. Yli-Harja, J. Astola and Y. Neuvo, "Analysis of the Properties of Median and Weighted Median Filters using Threshold Logic and Stack Filter Representation", IEEE Transactions on Signal Processing, Vol. SP-39, No. 2, pp. 395-410, February, 1991

[Zafa93] S. Zafar, Y.Q. Zhang and B. Jabbari, "Multiscale video representation using multiresolution motion compensation and wavelet decomposition", IEEE journal on selected areas in communications, Vol. 11, No. 1, pp. 24-35, 1993

LIST OF SYMBOLS AND ABBREVIATIONS

β	Cut-off frequency of temporal interpolation filter
ρ	Correlation coefficient
$\mathbf{A}$	Polynomial matrix in least squares estimation
B	Bit rate in bits/pixel
B_O	Bit rate necessary to code the original image in bits/pixel
$\mathbf{c}$	Polynomial coefficients vector with elements c_{ij}
D	Mean square error distortion
$D(R)$	Mean square error rate-distortion function
$\mathbf{d}_b(\mathbf{x},t)$	Displacement vector of the object from an image at time t to an Image at t-1
Δd_x	Accuracy of the motion estimate in the horizontal direction
Δd_y	Accuracy of the motion estimate in the vertical direction
$\mathbf{d}_f(\mathbf{x},t)$	Displacement vector of the object from an image at time t to an Image at t+1
$D_i(R_i)$	Mean square error distortion rate function of region i
$D_t^A(R_t)$	Mean square error distortion rate function using spatially adaptive subsampling
$e(\mathbf{x},t)$	Interpolation error signal
$h(.)$	Impulse response of a filter
$H(.)$	Frequency response of a filter
HVS	Human visual system
$I(\omega_x,\omega_y,\omega_t)$	Three-dimensional spectrum of an image sequence
$i(\mathbf{x})$	Discrete two-dimensional image intensity function
$i(\mathbf{x},t)$	Discrete three-dimensional image intensity function
$i_c(\mathbf{x},t)$	Analog three-dimensional image intensity function
$i_{int}(\mathbf{x},t)$	Interpolated image sequence
$I_O(\omega_x,\omega_y)$	Two-dimensional spectrum of an image at $t = 0$
L	Lattice
L_o	Original lattice
L^R	Reciprocal lattice
L_s	Sublattice
L_{ss}	Subsampling lattice
M	Number of modes
MSE	Mean square error
$n(\mathbf{x},t)$	Noise signal

N_b	Number of blocks in an image
N_r	Number of regions
P	Number of rows in a block
$\mathbf{p}$	Vector in which each element p_i represents the fraction of the contribution of each region i to the total average bit rate
PCM	Pulse code modulation
Q	Number of columns in a block
θ	Maximum value of derivative of rate-distortion function at the point of the optimal bit allocation
Q_x, Q_y	Parameters of a quincunx sampling lattice
R	Relative bit rate
$\mathbf{R}$	Matrix which describes a reciprocal lattice
R_i	Relative bitrate used to code a region i
R_t	Total relative bitrate
$\mathbf{S}$	Matrix which describes a lattice
$S(\omega)$	One-dimensional power spectral density function
σ^2	Variance
$S_i(\omega)$	One-dimensional power spectral density function of region i
SNR	Signal to noise ratio
T	Sampling period
$u(x)$	Generic discrete signal
$u_c(x)$	Generic continuous signal
U_L	Unity cell of lattice L
$\mathbf{v}$	Velocity in pixels/image
$\mathbf{v}_n$	Nominal velocity
W	One-dimensional bandwidth of a signal
W_{ss}	Bandwidth after prefiltering
W_x	Horizontal bandwidth
W_y	Vertical bandwidth
$\mathbf{x}$	Spatial coordinate vector $(x,y)^T$
$\mathbf{z}$	Point in a multi-dimensional space

SAMENVATTING

De opkomst van breedbandige digitale communicatiekanalen en recente ontwikkelingen op het gebied van de digitale signaalverwerkingsapparatuur hebben het mogelijk gemaakt dat een groot aantal nieuwe audio-visuele diensten kan worden geboden aan de consument. Deze diensten hebben een grote invloed op allerlei dagelijkse activiteiten, zowel zakelijk en educatief als in de ontspanningssfeer. De "High Definition Television" (HDTV) geeft vergeleken met de huidige televisie een betere beeld- en geluidskwaliteit. Ander nieuwe diensten zijn onder andere de beeldtelefoon, interactieve televisie en de digitale videorecorder. Al deze nieuwe diensten vereisen het verzenden van beeld-, geluid- en digitale informatie. Datacompressie richt zich op het efficiënt overzenden van de informatie. Door de toepassing van datacompressie komt de informatie sneller bij de consument en worden de kosten die verbonden zijn aan de nieuwe diensten verminderd.

Een methode die gebruikt kan worden voor het comprimeren van digitale beeldsequenties is onderbemonsteren. Een digitaal beeld bestaat uit een verzameling van beeldelementen ("pixels"). In een onderbemonsteringssysteem wordt de hoeveelheid te verzenden informatie verminderd door een deel van de pixels niet te verzenden. Aan de ontvanger moeten vervolgens op basis van de pixels die wel verzonden zijn de ontbrekende pixels worden teruggewonnen (interpolatie). Het is nu belangrijk dat de verzonden pixels voldoende informatie bevatten om een goede interpolatie mogelijk te maken. Hierbij moet een afruil worden gemaakt tussen de vereiste kwaliteit, de compressie factor en de complexiteit van de methode. In dit proefschrift worden verschillende mogelijkheden voor onderbemonsteringssystemen besproken.

Een belangrijke keuze in een onderbemonsteringssysteem is het bemonsteringsraster. Dit is een (regelmatige) verzameling van discrete punten in de drie-dimensionale ruimte. Het raster definieert de positie van de pixels in zowel de originele als de onderbemonsterde beeldsequentie. Door de vorm van het raster te variëren kunnen bepaalde frequentiecomponenten worden benadrukt of juist worden onderdrukt. De dichtheid van het raster bepaalt de compressiefactor. Uit het raster kunnen ook de posities in het frequentiedomein van de herhaal spectra, die onstaan door het bemonsteren van de beeldsequentie, worden afgeleid. Om vouwvervorming te voorkomen, mogen de herhaal spectra elkaar niet overlappen. Daarom moeten de frequentiecomponenten die vouwvervorming kunnen veroorzaken vóór het onderbemonsteren worden onderdrukt met een bandbegrenzend laagdoorlaat filter. Als er geen frequentiecomponenten zijn die aanleiding geven tot vouwvervorming dan is het bandbegrenzend filter overbodig. Bij de ontvanger worden de ontbrekende pixels weer teruggewonnen met behulp van een interpolatie filter, wat in essentie weer een laagdoorlaat filter is.

Een eerste groep van spatio-temporele onderbemonsteringssystemem, zijn systemen die gebruik maken van een vast bemonsteringsraster. In een eenvoudig onderbemonsterings-

systeem met een vast bemonsteringsraster worden lineaire niet-adaptieve bandbegrenzende en interpolatie filters gebruikt. Het eenvoudig systeem maakt gebruik van de eigenschap dat het menselijk visueel systeem minder gevoelig is voor diagonale spatiële frequentiecomponenten. Bij gebruik van een zogenaamd quincunx bemonsteringsraster worden deze frequentiecomponenten niet overgezonden. Een andere toepassing van het eenvoudige systeem is het onderbemonsteren van de kleurinformatie. De energiebijdrage van de kleurcomponenten is vaak laag, waardoor een raster met een lage bemonsteringsdichtheid kan worden gebruikt.

De kwaliteit van het eenvoudige systeem kan worden verbeterd door adaptieve interpolatie filters te gebruiken in plaats van vaste (d.w.z. niet-adaptieve) interpolatie filters. Het doel hierbij is om een deel van de resolutie die verloren is gegaan bij het onderbemonsteren terug te winnen. Om dit mogelijk te maken mag er geen bandbegrenzend filter worden gebruikt, en kan er dus vouwvervorming optreden. De interpolatie kan adaptief gemaakt worden aan de beweging in de beeldsequentie door bewegingsadaptieve filters te gebruiken. Bij bewegingsadaptieve filters wordt de spatio-temporele doorlaatband van het interpolatie filter aangepast aan de lokale bewegingsvector, en wordt er ook bij de interpolatie geen spatiële informatie weggefilterd. De resolutie van de geïnterpoleerde beeldsequentie is daarom hoger dan in het eenvoudige onderbemonsteringssysteem. Voor een goed interpolatieresultaat is het noodzakelijk dat de geschatte bewegingsvectoren voldoende nauwkeurig zijn en goed overeenkomen met de echte beweging. Omdat bewegingsschatting niet altijd correct verloopt, moet er een mechanisme zijn dat foute bewegingsvectoren detecteert en in dat geval een alternatieve interpolatie uitvoert. Foutdetectie kan worden gedaan door bij de zender al te kijken naar de grootte van de interpolatiefout. Het eenvoudige onderbemonsteringssysteem met niet-adaptieve interpolatie kan dienen als een alternatieve terugval mogelijkheid.

Een andere klasse van adaptieve filters omvat de niet-lineaire interpolatie filters welke gebaseerd zijn op het mediaan filter. Het voordeel van deze filters is dat het interpolatie resultaat niet bepaald wordt door het gewogen gemiddelde van de pixelwaarden in een spatio-temporeel venster om het te interpoleren pixel, maar dat de meerderheid van de pixelwaarden bepaalt wat het uiteindelijke interpolatie resultaat is. Als het te interpoleren pixel tot de meerderheid behoort, is het interpolatie resultaat beter dan wanneer lineaire interpolatie filters worden gebruikt. Het interpolatieresultaat kan nog verder worden verbeterd door een hybride filter te gebruiken, bestaand uit een combinatie van lineaire en niet-lineaire filters. Hierbij kan ook het bewegingsadaptieve interpolatie filter gebruikt worden. Experimenten tonen aan dat de prestaties van de niet-lineaire interpolatie filters geen grote verbetering geven ten opzichte van lineaire interpolatie filters. Vouwvervorming blijkt hierbij de beperkende factor te zijn.

Een tweede groep van spatio-temporele onderbemonsteringssysteem gebruikt voor het verkrijgen van beter resultaten een adaptief bemonsteringsraster in plaats van een vast bemonsteringsraster. In een systeem met een spatiëel adaptief raster wordt de vorm en de dichtheid van het lokale bemonsteringsraster aangepast aan de beeldinhoud. Het beeld wordt in blokken opgedeeld en voor elk blok wordt bepaald wat het optimale bemonsteringsraster is. Met behulp van de produktie-vervormingstheorie kan worden aangetoond dat deze aanpak altijd een lagere vervorming geeft dan een systeem met een vast bemonsteringsraster. Deze theorie kan ook worden gebruikt voor het toewijzen van de lokale bemonsteringsrasters aan

de verschillende blokken, onder de randvoorwaarde van een vooraf vastgestelde data-compressie factor.

Het gevolg van spatiëel adaptief onderbemonsteren is wel dat aan de ontvangstzijde de interpolatie uitgevoerd moet worden op een onregelmatig bemonsteringsraster. Binnen één blok is er wel een regelmatig bemonsteringsraster. Gebaseerd op de pixel binnen één blok kunnen met behulp van de kleinste-kwadraten-methode polynomen van verschillende orde worden geschat, welke gebruikt worden om de ontbrekende pixels te berekenen. Een ander interpolatiemethode is het opbouwen van een hiërarchische piramide, waarbij het beeld vanuit de bemonsteringsrasters met een lage bemonsteringsdichtheid steeds wordt uitgebreid naar een hogere bemonsteringsdichtheid. Een voorwaarde hierbij is dat elk lokaal bemonsteringsraster steeds een deelverzameling is van de rasters met een hogere bemonsteringsdichtheid.

Het systeem kan ook adaptief in de temporele richting gemaakt worden door gebruik te maken van bewegingsinformatie. Als een blok met behulp van de bewegingsvector voorspeld kan worden uit het vorig beeld, dan hoeft dit blok niet nog een keer te worden verzonden. Op deze manier wordt de temporele bemonsteringsfrequentie aangepast aan de aktiviteit in de temporele richting. De experimenten tonen aan dat het systeem betere resultaten oplevert dan een systeem met een vast bemonsteringsraster, en dat de kwaliteit toeneemt naarmate er meer verschillende lokale bemonsteringsrasters zijn waaruit gekozen kan worden.

Een andere manier om een adaptief bemonsteringsraster te verkrijgen is door het raster aan te passen aan de aanwezigheid van beweging. Een niet-bewegend gebied in een beeld verandert niet in de tijd, waardoor de bemonstering van het beeld verdeeld kan worden over een aantal beelden. Elk beeld wordt onderbemonsterd met een bepaald onderbemonsteringsraster, waarbij het raster voor elk beeld zodanig verschoven wordt dat alle pixels van het stilstaande beeld over een periode van een aantal beelden overgezonden worden. Bij de ontvanger worden de verschillende beelden gecombineerd in één beeld met een temporeel interpolatie filter. Dit principe heet "sub-Nyquist" bemonsteren, omdat de bemonsteringsfrequentie lager is dan de voorgeschreven Nyquist-frequentie.

Sub-Nyquist bemonsteren wordt gebruikt in verschillende compressiesystemen voor HDTV, zoals het Japanse MUSE systeem en het Europese HD-MAC systeem. In een praktisch systeem moet er altijd een alternatieve onderbemonsteringsmogelijkheid zijn voor het geval dat er wel sprake is van beweging. Bij beide systemen wordt in dit geval gekozen voor een vast onderbemonsteringsraster in combinatie met spatiële interpolatie. Bij het HD-MAC systeem wordt de onderverdeling in bewegende en niet-bewegende gebieden gedaan op blok-basis en in het MUSE systeem op pixel-basis. In het HD-MAC systeem wordt deze beslissingsinformatie als zij-informatie verzonden, terwijl het MUSE systeem de beslissing apart herhaalt bij de ontvanger. De complexiteit is daardoor in het HD-MAC systeem geconcentreerd bij de zender, terwijl in het MUSE systeem zowel de zender als de ontvanger dezelfde complexiteit hebben.

Het principe van sub-Nyquist bemonsteren kan met behulp van bewegingsinformatie worden uitgebreid naar bewegende gebieden. Dan treedt echter wel het probleem van de kritische snelheden op. Als een vast bemonsteringsraster wordt gebruikt is het voor sommige gebieden die bewegen met een bepaalde snelheid niet langer mogelijk om door combinatie van de verschillende onderbemonsterde beelden het oorspronkelijk beeld terug te winnen. Dit probleem wordt opgelost door een adaptief bemonsteringsraster te gebruiken, zodanig dat

combinatie van de verschillen onderbemonsterde beelden wel mogelijk wordt. Ook hier geldt dat een toename van de nauwkeurigheid van de bewegingsvectoren een positieve invloed heeft op het eindresultaat. Uit de experimenten blijkt dat het bewegingsadaptieve sub-Nyquist systeem betere resultaten oplevert dan het niet-adaptieve systeem omdat het ook kan worden toegepast op bewegende gebieden in de beeldsequentie.

Onderbemonsteren kan gecombineerd worden met transformatiecodering met als doel de complexiteit van het transformatiecoderingssysteem te verlagen. Een andere motivatie is de aanpassing van het transformatiecoderingssysteem aan het menselijk visueel systeem. Het overgebleven spectrum na onderbemonstering kan nauwkeuriger worden gecodeerd. Verder kan het transformatiecoderingssysteem bewegingsadaptief gemaakt worden door combinatie met sub-Nyquist bemonstering. In de praktijk komen alleen onderbemonstering systemen met een vast bemonsteringsraster in aanmerking voor combinatie met transformatiecodering. De gemiddelde kwadratische vervorming van een systeem bestaande uit een vast bemonsteringsraster gevolgd door transformatiecodering is altijd groter of gelijk aan de vervorming van een systeem met alleen transformatiecodering, maar de keuze voor het gecombineerde systeem is in de praktijk niet altijd gebaseerd op dit objectieve kwaliteitscriterium. Wanneer een transformatiecoderingssysteem gecombineerd wordt met sub-Nyquist bemonstering dan valt met theoretische argumenten aan te tonen dat de gemiddelde kwadratische fout altijd kleiner is dan in een standaard transformatiecoderingssysteem. In de experimentele evaluatie blijkt dit theoretische resultaat echter niet op te gaan.

Wanneer de verschillende besproken onderbemonsteringsmethodes met elkaar worden vergeleken, dan blijkt dat de systemen die gebaseerd zijn op een vast onderbemonsteringsraster slechtere resultaten oplevert vergeleken met de systemen die gebruik maken van een adaptief onderbemonsteringsraster. De uitzondering hierop is het systeem met een bewegingsadaptieve interpolatie filter, dat ook een goed resultaat oplevert. Het beste resultaat wordt verkregen met een spatio-temporeel adaptief bemonsteringsraster, omdat in dit geval de lokale beeldinhoud goed wordt benut.

ACKNOWLEDGMENTS

I would like to thank all the people who contributed to this thesis. This also holds for the people who are not explicitly listed below. My first thanks go to my family who supported and encouraged me throughout my years of study. I would especially want to mention my parents, my sisters, my uncle Orlando and my aunt Muriël.

I would also like to thank all my colleagues at the Information Theory Group for supporting me and providing me with a pleasant working environment. Special thanks goes to my mentor Inald Lagendijk who carefully and critically read everything I wrote. I also thank Jan Biemond for the encouragement and for giving me the opportunity to do my research.

My gratitude extends also to Frank Bosveld for joining me in the discovery of the mysterious world of computer programming. The software packages which we developed together helped me considerably during my own research. I also thank my other successive roommates, namely Otto Rompelman, Hans Driessen, Ruggero Franich and Stefan Westen.

Further I would like to thank the master's degree students Willem Vos who contributed to Chapter 6 of this thesis and Marc Hesp whose work contributed to Chapter 4.

CURRICULUM VITAE

Ricardo Alexander Frederik Belfor was born in Zeist, The Netherlands, on April 30, 1966. In July 1984 he obtained his VWO diploma from the Mr. Dr. J.C de Miranda Lyceum, in Paramaribo, Surinam. He took his M.Sc. in Electrical Engineering from the Delft University of Technology, Delft, The Netherlands, in November 1989. His Master's project was carried out at the Information Theory Group at the Department of Electrical Engineering and his thesis was entitled: "Predictive Image Sequence Coding using Pixel-Recursive Motion Compensation".

After receiving his M.Sc., he stayed with the Information Theory Group to do research towards a Ph.D. in the area of subsampling methods for image sequence coding. In that area he is the author of several publications in international scientific journals and conference proceedings. As a Ph.D. student, he has been responsible for the formulation of Master's and pre-Master's projects as well as for the supervision of the students who were working on these projects.

His professional interests include image coding, motion estimation and multi-dimensional digital signal processing.