

Chapter 2

Prediction Techniques for Image and Video Coding

Prediction techniques have been widely investigated over the last decades for image and video compression. Currently, most of the existing image and video compression solutions incorporate some sort of predictive coding in their algorithms. For example, the well-known, 25-year-old, Joint Photographic Experts Group (JPEG) image coding standard, predictively encodes the quantised DC transform coefficients. Also in the case of video sequences, compression systems typically exploit the temporal redundancy through the use of motion-compensated prediction.

This chapter describes some of the most important predictive coding techniques used in this book, which have been reported in literature and employed in the state-of-the-art image and video encoders. Section 2.1 provides some basic concepts about digital video representation. Section 2.2 presents a general overview on predictive coding for image compression. In Sect. 2.3, the most important prediction techniques used in the current state-of-the-art video coding standards are described. Section 2.4 presents various linear prediction methods based on least-squares optimisation, while Sect. 2.5 describes some predictive methods inspired on sparse representation and dimensionality reduction.

2.1 Digital Video Representation

A digital video sequence is a temporal succession of digital images, also referred to as pictures or frames, which usually presents a significant amount of temporal and spatial correlations. Typically, each image has a rectangular dimension, also called spatial resolution, and it is organised as a matrix of numerical values, denominated by pixels, which means picture elements.

An important aspect related to digital images is the colour representation. In general, image sensing systems generate colour images based on the RGB (Red, Green and Blue) model. In this model, image colours are generated through a linear

combination of red (R), green (G) and blue (B) primary colours. The colour image is thus formed by three equally sized pixel matrices, which represent the amount of each primary colour at each pixel position. This contrasts to the monochromatic image representation, which only uses one image channel, whose pixels correspond to the brightness intensity at each position.

One of the main issues of RGB system is the spectral redundancy that usually exists between pixels of the three primary colour channels. In order to provide a more efficient representation of colour images, the YCbCr (Luminance, Blue Chrominance and Red Chrominance) colour system, also known as YUV, has been proposed in ITU-R BT.601 Recommendation [33]. The idea of the YCbCr system is to concentrate the image spatial information in the luminance (Y) component and use a reduced bandwidth for the chrominance (Cb and Cr) components. This colour system exploits the fact that the human visual system (HVS) is much more sensitive to luminance (or luma) information than to colour information, represented by chrominance (or chroma) components.

A common approach used to achieve reduced bandwidth for the chrominance components is through sub-sampling. Chrominance sub-sampling reduces the spatial resolution of colour components, providing a more compact representation of image or video data. Although YCbCr sub-sampling causes unrecoverable loss of colour information, this should not be noticeable by the HVS since it is less sensitive to colour information. The most common sub-sampling patterns are the 4:2:2 and the 4:2:0. The 4:2:2 pattern uses two Cb and two Cr samples for each group of four luma samples, while the 4:2:0 pattern uses one Cb and one Cr samples for each group of four luma samples. In this book, we will assume the 4:2:0 sub-sampling pattern, whenever we refer to the YUV colour format.

Due to its losses, the YCbCr colour system can be viewed as a compression technique that removes the irrelevancy associated to the colour information. In order to better represent image and video data, other compression methods that exploit data irrelevancy and also redundancy can be used. While the irrelevancy is related to the information that can be removed without being perceived by the HVS, the redundancy has a statistical nature and its removal does not cause any loss in the original signal. Video compression methods that only exploit data redundancy are denominated lossless compression algorithms, while the coding approaches that exploit both redundancy and irrelevancy are the so-called lossy encoders.

In lossless image and video coding, the compressed signals can be completely recovered after the decoding process, i.e. the original and reconstructed signals are exactly the same. Generic lossless video compression algorithms not only exploit the spatial and temporal redundancies of video signals, but also the entropic redundancy. The spatial or intra-frame redundancy of one image is associated to the correlation between neighbouring pixels in the same image, tending to increase for higher spatial resolutions. Temporal or inter-frame redundancy is related to the existing similarities between adjacent temporal frames. Video sequences present higher temporal correlations when their contents are constant in time or the motion in the scene is slow. Such correlations also tend to increase for sequences with higher temporal frame rate, because the motion effect between two consecutive frames is

smoother. Entropic redundancy is related to the statistics of the symbols transmitted by the image and video compression algorithms. Symbols with higher probability of occurrence may be more efficiently represented using shorter binary codes.

In addition to redundancy coding, the lossy compression algorithms also exploit data irrelevancy, in order to further compress image and video signals. Contrary to the redundancy coding, the removal of data irrelevancy is not a reversible process, being responsible for the coding losses associated to lossy image and video compression algorithms. Such loss of information is accepted in favour of a much higher compression ratio, as long as the video quality is not significantly degraded. The efficiency of lossy video coding algorithms depends on their ability to remove both the redundant and irrelevant information present in image and video signals, while maintaining an acceptable video quality. This is measured by considering both the compression ratio and the quality of the reconstructed signal.

2.2 Image Prediction Overview

The prediction process consists of a statistical estimation of future random variables from past and present observable random variables. In image compression scenario, the prediction of an image pixel or a group of pixels may be derived from the previously transmitted pixels. The predicted pixels are subtracted from the original pixels and a residual signal is obtained. In general, the prediction is considered successful if the energy of the residual signal is lower than the energy of the original signal. As a consequence, the residual pixels might be more efficiently encoded by entropy coding algorithms, being transmitted with fewer bits.

Since the early days of image compression, predictive coding has showed to be an effective technique, being widely used for both lossless and lossy image coding. A generic compression approach commonly used to describe predictive coding is known as Differential Pulse Code Modulation (DPCM) [20]. This method consists in transmitting the difference between a pixel and its prediction instead of directly transmitting the pixel's value. The main motivation for DPCM is the fact that most source signals, e.g. image and video signals, present a high correlation between adjacent pixels.

A representative block diagram of DPCM approach for lossy image compression is outlined in Fig. 2.1. The basic principle of DPCM is to predict the current pixel from the previous pixel (or multiple pixels) and to encode the prediction error (or residual) resulting from the difference between the original and predicted pixels. In a lossy coding approach, the residue signal (represented in Fig. 2.1 by e) is encoded by means of a scalar quantiser which converts a predefined range of values into a single quantised value. In a final stage the entropy coding is used to efficiently represent the quantised residual (e') into fewer bits, by exploiting the existing statistical redundancy [96]. The compressed quantised residual is transmitted by the encoder and received by the decoder. After entropy decoding the residue signal, the reconstructed signal is obtained by summing the decoded quantised residue signal (e') with the prediction signal (\hat{s}), estimated from previous decoded pixels. All the encoding and decoding processes are performed on a pixel-by-pixel basis.

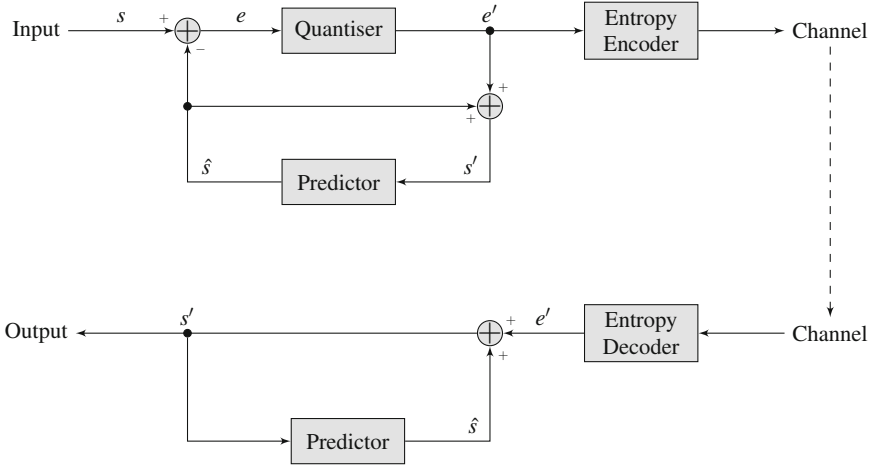
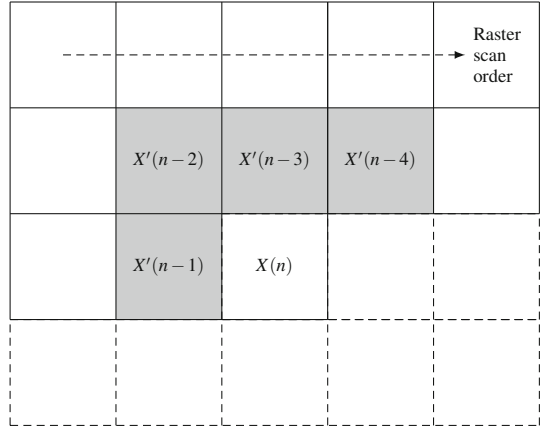


Fig. 2.1 DPCM block diagram for lossy image compression

Fig. 2.2 Spatial prediction in a raster scan order encoder, e.g. DPCM



The compression performance of DPCM depends on the accuracy of the used prediction technique. An example of a spatial prediction technique for the pixel-based DPCM approach processed in raster scan order is illustrated in Fig. 2.2, where the pixel $X(n)$ is predicted based on a linear combination of the neighbouring pixels indicated on the gray squares. This kind of prediction, based on the linear combination of the previously decoded pixels, is known as linear prediction [65, 74]. Since the coding procedure is done in raster scan order, both encoder and decoder can use the pixels $X'(n-1)$, $X'(n-2)$, $X'(n-3)$ and $X'(n-4)$, available in current and previous pixel rows, in order to generate the prediction for the current pixel, $X(n)$. The decoded (or reconstructed) pixels should be used instead of the original ones, in both encoder and decoder sides, because the decoded pixels may differ from the original pixels due to the quantisation step. Since the original pixels are not available

in the decoder side, the use of decoded pixels guarantees that the prediction signal is identical in both encoder and decoder sides. Otherwise, if the encoder used the original pixels for prediction, a cumulative mismatching error could be observed between the decoded pixels at the encoder and decoder, leading to an increased distortion in the decoder reconstructed image.

Linear Predictive Coding (LPC) techniques have been extensively studied in literature and successfully used in image, speech and audio processing applications. For the case of image compression, several algorithms based on DPCM, using an adaptive approach of LPC have been presented since the early days of digital image compression [65]. The use of adaptive prediction schemes for image compression has shown to provide better prediction accuracy than the fixed LPC approach. This is mainly because the adaptive methods try to adapt the predictor parameters to the changing local statistics, that characterise the image signal. The main challenge of adaptive prediction is to keep an affordable computational complexity.

For example, adaptive prediction can be achieved by testing a set of different predictors in the encoder and signalling the chosen predictor to the decoder, by using extra bits. However, the trade-off between the prediction efficiency gains and the extra bits required to signal the chosen predictor must be carefully evaluated. Alternatively, implicit approaches, which do not require extra signalling bits, also can be used for adaptive prediction. A discussion on implicit linear prediction techniques for lossy image compression using least-squares optimisation is presented in Sect. 2.4.

Pixel-based compression schemes using adaptive linear predictive coding techniques have also been successfully applied in lossless image compression. A basic lossless DPCM system can be designed by discarding the quantisation step in the block diagram of Fig. 2.1. In such lossless scenario, the prediction stage may directly receive the causal original pixels, since they are equal to the decoded pixels. Some examples of popular pixel-based lossless compressors using adaptive linear and non-linear predictors are the JPEG-LS standard [119], Context-based Adaptive Lossless Image Codec (CALIC) [123], Minimum Rate Predictor (MRP) [68] and Edge-Directed Prediction (EDP) [51].

Although DPCM compression is commonly associated to spatial or intra-frame coding, it can be designed to exploit inter-frame similarities, such as temporal redundancy in the case of video signals. While intra-frame coding usually uses the causal neighbouring pixels belonging to the same frame, the inter-frame coding can be conducted using the causal pixels belonging to another frame, e.g. the co-located pixel in the previous coded frame.

Alternatively to the pixel-based approach, DPCM can operate with vectors. In a vector-based DPCM, image blocks can be used as input vectors and the prediction can be performed using block-based methods, such as directional prediction or motion-compensated prediction. These prediction methods, used in the state-of-the-art image and video coding standards, are described in Sect. 2.3. In fact, the vector-based formulation of DPCM works quite similar to the typical transform-based image and video compression standards. The main difference refers to the residue quantiser that is replaced by transform coding and quantisation methods.

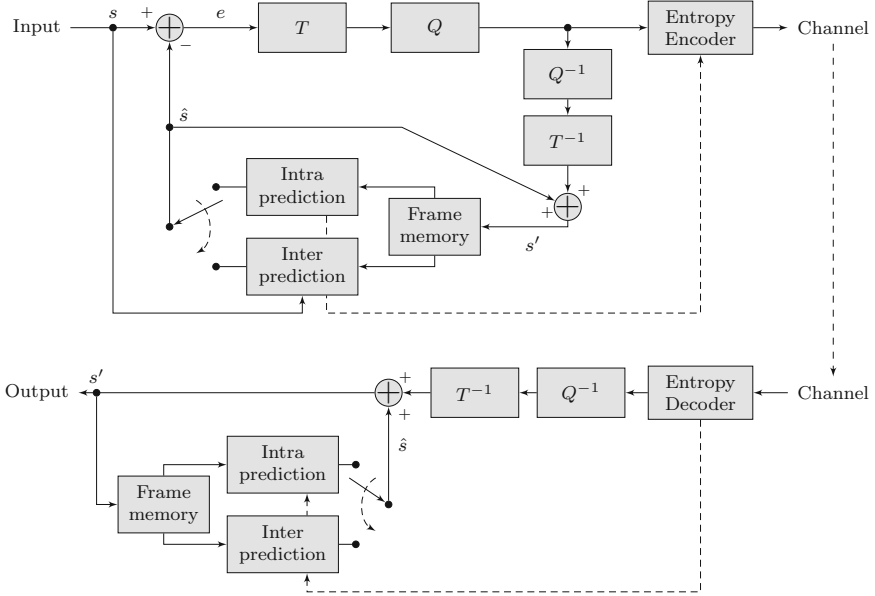


Fig. 2.3 Block diagram for a generic transform-based hybrid video compression system

Figure 2.3 presents the block diagram for a generic block-based hybrid video compression system using predictive coding and residue coding based on transform (T) and quantisation (Q) methods. Apart from the residue coding module, which may be based on a different coding paradigm, the illustrated block diagram represents the architecture of most image and video compression algorithms investigated in this book.

The advantage of reducing spatial and temporal redundancies using prediction methods is directly related to the probability distribution of the residual signal. Unlike the original signal, the residue typically presents a highly peaked probability distribution, centred around zero, which is favourable for statistical coding techniques. Regarding the transform coding methods, such probability function is advantageous because the transformed residual presents a lower energy, resulting in less non-null coefficients and producing a more compact representation in the frequency domain. This transform-based representation tends to present a lower entropy being more efficiently encoded with fewer bits.

Besides the prediction of original image pixels, most of the image and video compression algorithms use predictive coding to efficiently encode other symbols generated during the encoding process. For example, the motion vectors generated by motion-compensated prediction can be differentially transmitted as a lossless DPCM system. Such predictive coding approach is advantageous because motion vectors typically present high correlation between neighbouring blocks, specially in situations of camera panning.

2.3 State-of-the-Art Prediction Methods

The prediction methods play an important role in image and video coding standards, due to their ability to reduce the signal redundancy based on the previously encoded samples. With the evolution of compression standards, more complex and sophisticated predictive coding techniques have been investigated and adopted. The most recent state-of-the-art H.264/AVC [36, 121] and H.265/HEVC [37, 103] standards use both intra and inter prediction methods. Although these algorithms have been primarily conceived for video compression applications, they also present efficient results for still image coding [83]. In the image coding scenario, only intra prediction methods can be used.

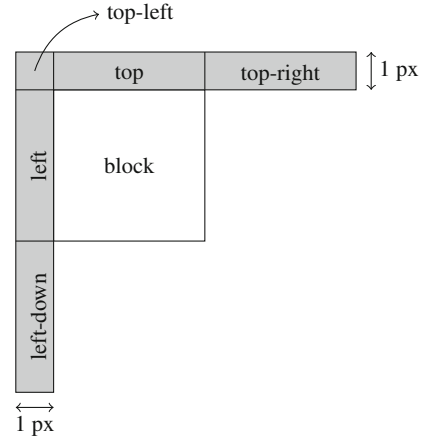
This section describes the main prediction techniques used in the state-of-the-art H.265/HEVC standard, and highlights some differences in relation to the previous H.264/AVC encoder. These techniques include the intra directional prediction for efficient spatial redundancy coding and the motion-compensated prediction for inter-frame temporal redundancy coding. The advantage of these techniques is not only associated to their prediction efficiency but also to their reasonable computational complexity, which make them appropriate for practical applications.

2.3.1 *Intra-Frame Prediction*

In recent video coding standards, block-based intra prediction techniques are used for efficient spatial redundancy exploitation. Intra predicted blocks are generated based on the pixels of the previously encoded and reconstructed blocks. Directional prediction plus Planar and DC modes were adopted in HEVC standard for luma component prediction. These prediction modes are defined for square block sizes from 4×4 up to 32×32 . The encoder tests all the available intra prediction modes and signals the best mode to the decoder. The decoder must generate an identical prediction block, based on the reconstructed pixels from the previously encoded neighbouring blocks.

The reconstructed reference samples used in the prediction process belong to the neighbouring blocks at left-down, left, top-left, top and top-right positions, as shown in Fig. 2.4. For an $N \times N$ block, intra prediction requires a row of $2N$ top neighbouring samples, a column of $2N$ neighbouring samples and the top-left corner neighbouring sample, totalling $4N + 1$ reference samples. For some blocks, the left-down or top-right neighbouring blocks may not be available. This also can occur for the top or left neighbouring blocks, e.g. at slice or tile boundaries. In order to solve the unavailability issue, HEVC substitutes the unavailable reference samples with the closer available reference sample values, so that all intra prediction modes can be used.

Fig. 2.4 Available neighbouring regions used by intra prediction modes in HEVC standard



2.3.1.1 Directional Prediction Modes

Directional intra prediction was first introduced in the H.264/AVC standard [36]. The idea of directional prediction is to estimate regions with a structured texture or directional edges. In its process, directional prediction projects the reconstructed samples in the block neighbourhood along a specific direction. While H.264/AVC defines 8 possible directional modes, HEVC uses a more sophisticated scheme based on 33 directions, denominated as angular modes [37]. The 33 angular modes in HEVC offer an increased number of possible prediction directions and are able to better predict the larger block sizes available in HEVC. Figure 2.5 illustrates the available prediction directions in the HEVC standard, numbered from 2 to 34.

Angular prediction modes were designed to better cover the angles near the horizontal and vertical directions, being sparser near the diagonal directions. Such angle distribution was observed to be more frequent and useful for prediction. When the projected samples need to be extrapolated, HEVC uses bilinear interpolation of the two nearest samples at integer location with $1/32$ sample accuracy.

2.3.1.2 Planar and DC Prediction Modes

In addition to the directional modes, HEVC implements the Planar and DC prediction modes, which were also used in H.264/AVC standard. These modes were designed to efficiently predict smooth regions, by using constant or linear planes.

DC mode uses the average value of the neighbouring reference samples to generate a constant prediction for the block. The planar mode works by predicting the block through an amplitude surface with vertical and horizontal slopes derived from the neighbouring reference samples.

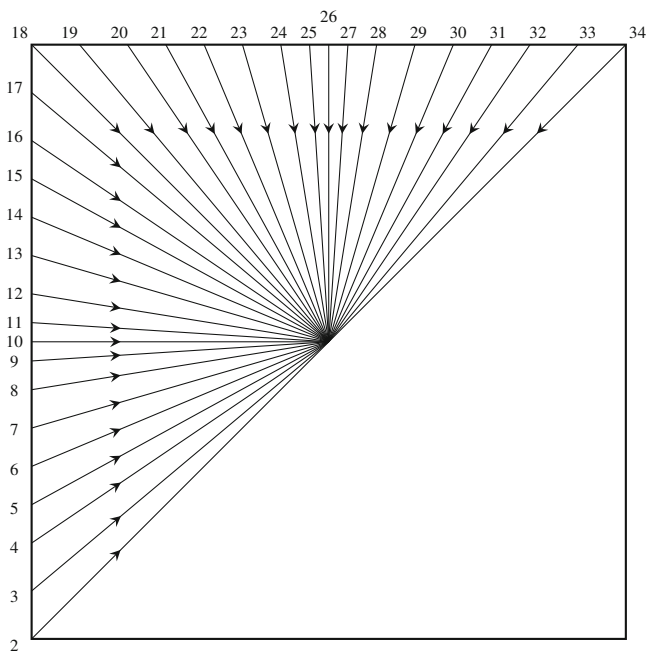


Fig. 2.5 Intra prediction angular modes used in HEVC standard

Table 2.1 Reference sample smoothing in HEVC standard based on block size and intra prediction modes

Block size	Intra prediction mode
4 × 4	No smoothing
8 × 8	Planar, Angular 2, 18 and 34 smoothing
16 × 16	Planar and all Angular except 9, 10, 11, 25, 26 and 27 smoothing
32 × 32	Planar and all Angular except 10 and 26 smoothing

2.3.1.3 Sample Smoothing

HEVC applies sample smoothing in two distinct situations to improve intra prediction results, namely the filtering of reference samples and the filtering of the predicted samples. In H.264/AVC standard, only the reference samples are filtered. In HEVC, this filtering is done in an adaptive manner, depending on the block size and intra prediction mode. Table 2.1 presents the cases in which reference sample smoothing is applied. HEVC uses the same three-tap $[1 \ 2 \ 1]/4$ filter, as defined in H.264/AVC standard.

The other filtering proposed in HEVC is used to smooth the discontinuities generated along block boundaries due to intra prediction. This filter is applied over the boundary predicted samples for the DC, Angular 10 and Angular 26 modes. In

case of DC mode, when the block is smaller than 32×32 , the samples of both the first row and first column of the intra predicted block are filtered. A two-tap $[3 \ 1]/4$ filter fed by the original value and the adjacent reference sample is used. In the case of Angular 10, only the samples of the first column are changed by adding half of the difference between the adjacent reference sample and the top-left reference sample. A similar filtering is done for Angular 26 mode, but only for the first row of predicted samples.

2.3.1.4 Prediction Mode Coding

An important aspect in the design of a predictive method is the need for signalling of additional information in the bitstream. In the case of the presented intra predictive scheme for luma prediction, the encoder should choose the best mode and signal it to the decoder.

To better encode the prediction modes, the recent video coding standards estimate the Most Probable Modes (MPMs), which allow to use fewer bits when selected. In the case of H.264/AVC, only one MPM is used, being adaptively initialised from the neighbouring blocks for each predicted block. Due to the increased number of prediction modes, HEVC selects three MPMs to be more efficiently represented. The MPMs are chosen according to the used prediction modes in the top and left neighbouring blocks.

The initialisation of MPMs in HEVC is shown in Table 2.2. It is possible to observe that prediction modes from left and top neighbouring prediction blocks are represented as modes A and B, respectively. When a neighbouring prediction block is unavailable, the DC mode is used instead. If modes A and B are different, they become the two first MPMs, otherwise only the first MPM is assigned. The remaining MPMs, specifically the third one (and also the second one when $A = B$), are defined in such a way that the same mode is not used twice in MPMs list. Table 2.2 explicitly defines a small set of mode candidates that can be used for the third MPM (and also the second MPM when $A = B$). In these cases, the MPM is assigned to the first candidate that does not result in a duplicated MPM.

For encoding purposes, HEVC transmits one flag indicating whether the chosen prediction mode matches any of the MPMs or not. If the matching occurs, the index of the matched MPM (1 out of 3 possible values) is entropy coded and transmitted. Otherwise, a 5-bit fixed length code is used to signal one of the remaining 32 modes (35 modes minus 3 MPMs) to the decoder.

Table 2.2 Assignment of the most probable prediction modes in HEVC

Block neighbouring modes: A and B	Most probable modes		
	1st	2nd	3rd
$A \neq B$	A	B	DC, Ang.10, Ang.26
$A = B = \text{DC, Planar}$	A	Planar, DC, Ang.26	Planar, DC, Ang.26
$A = B = \text{Angular}$	A	Angular closest to A	Angular closest to A

2.3.1.5 Chroma Intra Prediction

For chroma component intra prediction, HEVC considers five prediction modes: Planar, Angular 26 (vertical), Angular 10 (horizontal), DC and another mode denominated as Derived mode. The objective of the Derived mode is to use for chroma prediction the same mode as the co-located luma block. Despite allowing only 5 modes, the Derived mode gives a chance to predict the chroma block using any of the 35 previously presented modes for luma prediction.

Derived mode relies on the assumption that corresponding luma and chroma blocks are highly correlated. This is so because some image features, like edges or textures, can be observed in both components. Thus, the Derived mode provides an efficient solution to use the directional modes in chroma component prediction, exploiting some inter-component correlation and using less bits to signal the prediction mode.

For prediction mode signalling, HEVC does not use the MPM approach for chroma component. One flag is transmitted indicating whether or not the chosen chroma prediction mode is the Derived mode. When the chosen mode is not the Derived mode, HEVC uses a 2-bit fixed length code to signal one of the four remaining modes. If any of these remaining modes matches the Derived mode, HEVC replaces it by Angular 34 mode.

2.3.2 Inter-Frame Prediction

The efficiency of video coding algorithms highly relies on inter-frame predictive techniques used to reduce temporal redundancy. The underlying idea of inter-frame prediction is to estimate the target frame from one or more previously encoded reference frames using block-based Motion Compensated Prediction (MCP). The most common technique used for motion estimation in H.265/HEVC, as well as in H.264/AVC, is the Block-Matching Algorithm (BMA).

Inter-frame prediction is also used for disparity compensation in stereo and multiview video coding applications [81, 117]. In addition to temporal redundancy, the stereo and multiview video systems present inter-view redundancy among the multiple views. Similar inter prediction techniques, typically based on BMA, are used for both disparity and motion estimation. Despite being based on BMA, inter-frame prediction methods have received significant improvements, mainly in the most recent video coding standards. In the following, the state-of-the-art techniques proposed for MCP in HEVC standard are discussed.

2.3.2.1 Motion Compensation Using Block-Matching Algorithm

Motion estimation consists in searching for the causal block with highest similarity with the target block to be predicted, using BMA over the previously encoded

frames, i.e. the reference frames. Typically, reference frames correspond to past or future temporal frames. The frame encoding order determines which reference frames are selected and whether future frames are available.

In its procedure, BMA subtracts the target block from each equally sized block that exists in a search window defined in the reference frame and computes the resulting difference errors. The reference frame block that produces the lowest matching error is used as prediction to the target block. The motion vector is given by the offset between the best matched block position and the co-located position of the target block. This vector is signalled to the decoder, in order to be able to use the same predictor.

HEVC allows either one or two motion vectors to be chosen to compensate the target block, in a process denominated uni-predictive or bi-predictive coding, respectively. In the case of bi-predictive coding, the average of two motion compensated blocks is performed. These vectors are obtained from frames of the two available reference frame lists. In addition to the motion vectors, HEVC transmits the indices of the used reference frames and, in some cases, the reference picture list associated to each index. Additionally, HEVC allows an enhanced prediction technique, called weighted prediction, which is also present in H.264/AVC standard. This technique consists in applying a scaling and offset to the predictor block. In HEVC, only the explicit mode is used, in which the scale and offset parameters are transmitted to the decoder.

BMA has been evolving over the various video coding standard generations. While first block-based MCP approaches used a fixed block size, more recent algorithms introduced the adaptive block size. The H.264/AVC standard uses adaptive block size with seven possibilities from 4×4 up to 16×16 pixels for MCP, including square and rectangular blocks generated by symmetric block partitioning. A significant evolution is visible on HEVC standard, that uses much more block sizes from 4×4 up to 64×64 pixels, including square and rectangular blocks generated by symmetric and asymmetric block partitioning (see Sect. 3.2.1).

By using larger prediction blocks, HEVC is able to save more bits in the signalling of the chosen motion vector, because more pixels can be predicted using a single translational motion vector. At more complex textured regions, large blocks may not produce efficient prediction results and a more precise motion representation may be required. For these cases, the HEVC block partitioning gives rise to a large number of possible block sizes that may use different motion vectors, leading to better prediction results. However, such flexible block partitioning has the cost of an additional computational complexity and increased bitrate requirements to signal the chosen partitioning tree and motion information associated to each sub-block.

Besides the use of adaptive block size, MCP algorithms adopted in most video coding standards have received some improvements in the motion vector accuracy. Fractional interpolation has been used to generate non-integer sample positions at the reference frames used for motion estimation. Both H.264/AVC and HEVC standards use motion vectors up to quarter-pixel accuracy for luma component. For chroma component, motion vector accuracy depends on the sampling format, being eighth-pixel accuracy for the 4:2:0 sampling format.

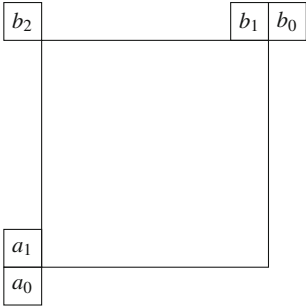
In order to generate the luma interpolated pixels, HEVC uses an eight-tap filter for half-pixel positions and a seven-tap filter for quarter-pixel positions, in contrast to the H.264/AVC interpolation based on a six-tap filter for half-pixel positions and two-pixel averaging for quarter-pixel positions. For chroma pixel interpolation, HEVC uses four-tap filters, while H.264/AVC uses two-tap bilinear filtering. By using larger tap filters, HEVC interpolation accuracy is improved. In contrast to the two-stage interpolation process used by H.264/AVC standard, HEVC interpolation uses a simpler architecture based on a single and separable process to interpolate all fractional positions, without requiring any intermediate rounding operations. Detailed information about HEVC interpolation filter can be found in [37, 103].

2.3.2.2 Merge Mode

The Merge mode was introduced in HEVC standard, as a new technique to derive motion information from spatial and temporal neighbouring blocks. This technique extends the concepts of the direct and skip modes used in H.264/AVC, using a more sophisticated algorithm. It provides an alternative solution for motion representation, which selects the motion vector from an adaptive set of candidate vectors and signals it using an index number, that can be more efficiently represented.

A set of five spatial motion vector candidates, with the positions illustrated in Fig. 2.6, is considered in Merge mode. Unavailable candidates are not accounted and duplicated candidates (i.e. with the same motion information) are removed. According to the order a_1, b_1, b_0, a_0, b_2 , after the proper exclusions, the $C - 1$ spatial candidates are retained, where C is the maximum number of Merge candidates defined in the slide header. One temporal candidate is also added to fulfil the C Merge candidates. This candidate is obtained from the right bottom pixel outside the co-located block of one reference picture, whose index should be transmitted. If no candidate is available in right bottom position, the middle position of the block is used instead. HEVC Merge mode also considers some mechanisms to derive additional candidates, when the number of available spatial and temporal candidates is inferior to C .

Fig. 2.6 Spatial candidates of motion information for Merge mode



2.3.2.3 Motion Vector Prediction

In order to encode motion vectors, in those cases when the Merge mode is not chosen, HEVC uses a differential encoding approach for more efficient representation. This predictive approach for motion vector coding works similarly to a lossless DPCM system, where no loss is admitted. Its advantage is the possibility to encode a residual signal (difference motion vectors) that presents a concentrated probability density function around zero, providing more efficient entropy coding results.

HEVC considers the same spatial candidate pixels of the Merge mode, illustrated in Fig. 2.6, to derive the motion vector predictors from which the difference vector is obtained. Among these candidates, only two are chosen to be used as possible predictors. HEVC limits the number of predictors to avoid further increasing of the computational complexity associated to the motion estimation process. The first motion candidate is chosen by picking up the first available candidate in the set of left candidates arranged as a_0, a_1 , while the second one is the first available candidate among the set of above candidates arranged as b_0, b_1, b_2 .

When the number of available spatial motion vector predictors is less than two, HEVC includes the temporal candidate as predictor. This is one of the mechanisms used by HEVC to guarantee the existence of two motion vector predictors. For the signalling, only a single binary flag is required to indicate which motion vector predictor should be used.

2.4 Least-Squares Prediction Methods

Least-squares methods have been used for linear prediction in many signal processing applications. The modelling and compression of audio and speech signals are successful applications of linear prediction for one-dimensional signals. Currently, linear prediction is used in most speech coding algorithms.

In regard to image signals, several successful applications of linear prediction for lossless image coding using Least-Squares Prediction (LSP) methods have been presented in literature [41, 51, 120, 124, 126]. Due to the abrupt changes in image's local statistics, adaptive approaches of linear prediction, such as the context-based adaptive least-squares prediction [51], have shown to perform better than the non-adaptive methods. For the case of lossy image and video compression, there are some research works that demonstrate the advantage of the context-based adaptive prediction for the recent state-of-the-art image and video coding algorithms [10, 26, 27, 49, 50].

In this section, some methods based on LSP for efficient image and video compression are described. Initially, the advantage of LSP for edge prediction in both spatial and temporal directions is explained. Then, the LSP algorithm proposed for lossless image coding is described. Finally, this section presents some variants of LSP for block-based prediction in lossy image coding applications.

2.4.1 Linear Prediction of Images and Video Using LSP

The principle of LSP is to linearly combine a set of causal neighbouring pixels using coefficients previously estimated in a local causal training window. A common implementation of LSP estimates the filter coefficients on a pixel-by-pixel basis, i.e. a new set of coefficients is estimated for each pixel to be predicted. Such procedure allows LSP to adaptively embed the changing local texture characteristics in the linear prediction coefficients. Since the coefficients are estimated in a causal training window, LSP adaptation is implicit and does not require the transmission of the coefficient values.

In [51], it has been demonstrated that context-based adaptive LSP provides an effective modelling of the edges present in natural images. This fact led to a new interpretation of LSP, which is referred to as edge-directed property. The reasonable match of the linear predictor to the edge direction is justified by the higher influence of the pixels around the edges in the least-squares optimisation process.

The problem of edge modelling is of particular interest, due to the large amount of information carried out by edges present in natural images. Typically, smooth regions are easily predicted and compressed. However, the same does not apply to edge areas or complex textured areas. The current state-of-the-art standards use the directional intra prediction to explicitly model image edges, based on a predefined number of fixed directions. Due to the edge-directed property, LSP is able to provide a reasonable prediction of arbitrarily oriented edges using an implicit methodology. The performance of LSP has been evaluated in [51], based on the Edge-Directed Prediction algorithm for lossless image coding. The experiments have shown a superior performance than other state-of-the-art lossless image coding standards.

Modified approaches of LSP algorithm have been also investigated in literature for lossy image coding. The main challenge of these methods is to adapt LSP for block-based image coding algorithms, because pixel-based coding approaches are not efficient for lossy compression. Some of these LSP-based methods have been implemented and evaluated using the state-of-the-art transform-based H.264/AVC standard [26], as well as alternative image coding algorithms based on pattern matching [27].

Regarding video coding applications, proposals of LSP have been investigated for implicit motion compensation. An interesting solution is to use a spatio-temporal prediction approach, in which temporal samples are included in the LSP filter context in order to implicitly learn the motion information. This kind of LSP-based motion compensation approach has been proposed in [49] as an alternative to the explicit block-matching algorithm, widely used for MCP in current video coding standards.

2.4.2 Context-Based Adaptive LSP

In this section, the context-based adaptive LSP algorithm is described as proposed in [51] for lossless intra image coding. The main challenge of LSP is to develop an efficient predictive model, which fully exploits the information contained in its context.

Let $X(\mathbf{n})$ denote the image pixel to be linearly predicted, where \mathbf{n} is a two-dimensional vector with the spatial coordinates in the image. By using the N nearest spatial causal neighbours, according to an N th order Markovian model, the predicted pixel is computed as:

$$\hat{X}(\mathbf{n}) = \sum_{i=1}^N a_i X(\mathbf{n} - \mathbf{g}(i)), \quad (2.1)$$

where $\mathbf{g}(i)$ gives the relative position of each pixel in the filter context (or support), and a_i are the filter coefficients. An example of the filter context, as proposed in [51], is illustrated in Fig. 2.7 for $N = 12$.

In order to avoid the transmission of side information, LSP locally estimates the prediction coefficients using a causal training window, which is available in both the encoder and the decoder. This approach provides an implicit way to adaptively learn the orientation of the edges contained in the training window. In [51], a rectangular training window that contains $M = 2T(T + 1)$ elements is proposed, as shown in Fig. 2.7. The training window pixels can be arranged in a column vector

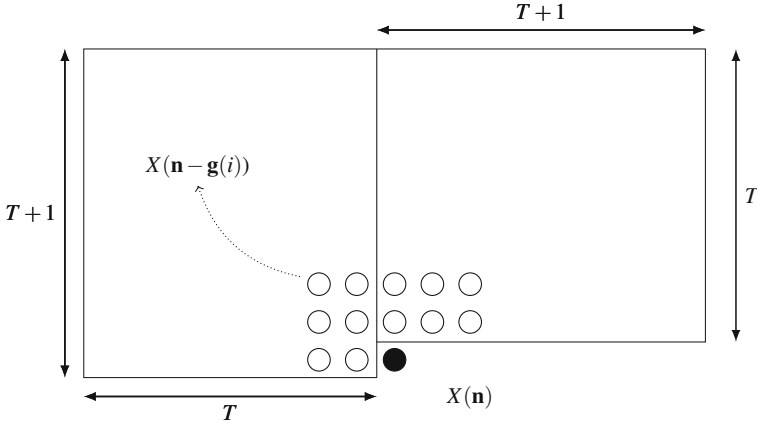


Fig. 2.7 LSP filter context (pixels represented by white circles) and associated training window

$\mathbf{y} = [X(\mathbf{n} - \mathbf{h}(1)) \dots X(\mathbf{n} - \mathbf{h}(M))]^T$, where $\mathbf{h}(j)$ represents the relative position of each pixel. With the filter context pixels, the following $M \times N$ matrix can be formed:

$$\mathbf{C} = \begin{bmatrix} X(\mathbf{n} - \mathbf{h}(1) - \mathbf{g}(1)) \dots X(\mathbf{n} - \mathbf{h}(1) - \mathbf{g}(N)) \\ \vdots \\ X(\mathbf{n} - \mathbf{h}(M) - \mathbf{g}(1)) \dots X(\mathbf{n} - \mathbf{h}(M) - \mathbf{g}(N)) \end{bmatrix}.$$

Note that $X(\mathbf{n} - \mathbf{h}(j) - \mathbf{g}(i))$ represents the i th filter context sample associated to the training window pixel, given by $X(\mathbf{n} - \mathbf{h}(j))$. The filter coefficients, $\mathbf{a} = [a_1 \dots a_N]^T$, can be determined by least-squares optimisation, finding the solution for $\min_{\mathbf{a}} (\|\mathbf{y} - \mathbf{C}\mathbf{a}\|_2^2)$. A well-known closed-form solution for LS problem is given by:

$$\mathbf{a} = (\mathbf{C}^T \mathbf{C})^{-1} (\mathbf{C}^T \mathbf{y}). \quad (2.2)$$

As suggested by the edge-directed property, the LSP estimated predictor coefficients tend to adapt to the orientation of the edges contained in the training window. Because of such adaptation, LSP is able to provide a reasonable prediction of $\hat{X}(\mathbf{n})$ along the estimated edge's orientation.

2.4.3 Block-Based LSP

Block-based implementations of LSP have been proposed specifically for lossy image encoders. In [27], an LSP mode based on the previously described algorithm was presented for block-based intra prediction using the Multidimensional Multiscale Parser encoder. In it, LSP estimates the weighting coefficients on a pixel-by-pixel basis, using a slightly different training window for each pixel of the block to predict. The training and linear prediction procedures of such lossy-based LSP use not only the reconstructed pixels from previously encoded blocks, but also the available predicted pixels from the current block.

The recursive use of predicted samples in LSP algorithm has some drawbacks, because predicted samples inherently incorporate an error not present in the reconstructed samples. This yields a sort of error propagation. In order to reduce such error propagation, the method in [10] presents a line-based linear prediction scheme for the H.264/AVC standard. Unlike the pixel-based training procedure of [27], the line-based model is fixed for the whole line of the block, being updated after each encoded line.

An alternative LSP method for block-based intra prediction in the H.264/AVC standard has been proposed in [26]. In it, an adaptive training window and filter context are used, depending on the number of available neighbouring blocks. Similarly to the LSP proposal in [27], this approach uses the predicted pixels from the current block in addition to the reconstructed ones of the previously encoded

blocks. However, the proposal in [26] performs the training procedure on a block-by-block basis. By performing a single training procedure for each block, the same set of estimated linear coefficients is used to predict the entire block. Furthermore, due to the reduced amount of training procedures, this solution presents a lower computational complexity.

Some applications of LSP for linear prediction between colour image components have been also presented using a single least-squares training procedure in the neighbourhood of the block [48, 54, 128]. The work of [48] exploits the fact that the same filter coefficients are used to predict the whole block and investigates an explicit approach that quantises and transmits the linear coefficients.

An LSP prediction mode for inter prediction between stereo pair views has been also proposed in [55, 61]. One interesting feature of this proposal is the usage of multiple filter contexts of different orders and shapes. At the encoder side, eight possible contexts are tested and the one that produces the minimum block prediction error is signalled to the decoder, as side information. A combination of spatial and inter view prediction is used, by including pixels from both the left and right views in the filter context.

2.4.4 Spatio-Temporal LSP

The idea of spatio-temporal prediction has been investigated for implicit motion compensation in video coding applications. This approach exploits the LSP edge-directed property for prediction of the motion trajectory along the temporal axis, based on the duality between 2D image contour edges and 3D video motion trajectory [49, 50]. Such duality is given by the similarity between image edges and flow-like patterns that are observed in transversal sectioning of successive video frames along the time direction.

In order to learn the edge information associated to the motion flow pattern along temporal direction, an appropriate filter context that includes causal samples from previous temporal frames is used. Such context configuration allows LSP to implicitly embed motion trajectory in the linear predictor coefficients, based on the training procedure in a causal window. Figure 2.8 illustrates an example of the referred spatio-temporal context for the LSP algorithm. Spatio-temporal context provides some flexibility to exploit motion information from the temporal portion of the context or to employ spatial prediction by assigning more weights to the spatial neighbours.

The filter context of LSP should cover the flow-like patterns observed in 3D video along temporal axis, in order to exploit the existing motion trajectories. Typically, filter contexts as the one illustrated in Fig. 2.8 tend to be more effective to predict slow motion, in a variety of types, such as panning, rotations, zoom, and jittering. In order to estimate LSP coefficients, a cube shaped training area that contains the local causal pixels, considered stationary for natural slow motion video, is suggested in [49].

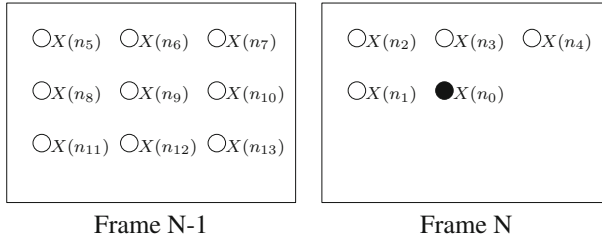


Fig. 2.8 Example of a 13th order spatio-temporal filter context using causal neighbours from the current frame (spatial neighbours) and previous encoded frame (temporal neighbours)

The prediction efficiency of LSP can be further improved by means of an adaptive update of the filter context. In [49], it is proposed to change LSP context configuration according to the motion characteristics. The contexts are designed and chosen based on the statistics of the causal data, not requiring the transmission of auxiliary information. By using different contexts for zooming, jittering or panning, LSP can better learn local motion characteristics and predict unknown samples.

When the video content is characterised by fast motion, the prediction performance of LSP tends to reduce. The inconsistency between the training data and the actual motion trajectory is the main reason for the performance degradation. A solution proposed in [49] for a fast camera panning uses temporal frame warping to compensate it. To minimise the effect of other types of unpredictable events, which reduce the stationary statistics of the signal, an adaptive training window size is also proposed. The main idea is to change the number of temporal frames included in the training window, being updated on a frame-by-frame basis and explicitly transmitted.

2.5 Sparse Representation for Image Prediction

Another important class of algorithms, which has been increasingly considered in literature for intra image coding, is based on sparse representation [67, 111, 112]. These approaches are motivated by the assumption that natural image signals are formed by few structural primitives or representative features. Sparse prediction tries to approximate the input signal using a linear combination of a small number of these primitives, selected from a large and redundant basis, known as dictionary. Typically, these methods provide efficient prediction of highly textured areas, where the traditional directional prediction techniques present some issues.

In the context of sparse representation, dimensionality reduction methods have been also recently proposed for image prediction [16, 113]. The main idea of dimensionality reduction techniques is to approximate the input unknown block as a linear combination of the k nearest neighbours (k -NN) in terms of the Euclidean distance, determined from an adaptive local dictionary. Such dictionary is formed by

elements derived from texture patches present in a causal reconstructed area in the neighbourhood of the unknown block. This kind of prediction can be viewed as a k -sparse representation of the block. In order to avoid the transmission of the nearest neighbours, an approximation to a causal region defined in the neighbourhood of the block (also called template region) is firstly determined. The optimal coefficients estimated for the template are then used to predict the unknown block, by combining the block patches adjacent to the k -NN templates. Since the coefficients for the template can be implicitly determined in the decoder side, no information about the k -NN needs to be transmitted.

In this section, prediction algorithms for image and video coding based on sparse representation are presented. These techniques include Matching Pursuit-based algorithms (MP) and the Template Matching (TM) algorithm. Furthermore, dimensionality reduction methods based on Non-negative Matrix Factorisation (NMF) and Locally Linear Embedding (LLE) methods are described.

2.5.1 Sparse Prediction Problem Formulation

Consider the N -pixel region \mathbf{S} , the union of the N_p -pixel block \mathbf{P} to be predicted, and the N_c -pixel approximation support (or template) \mathbf{C} , as illustrated in Fig. 2.9. Note that the pixels from the block \mathbf{P} to be predicted are unknown, and the ones of the template \mathbf{C} are the previously reconstructed pixels. By using an appropriate dictionary, the sparse prediction method estimates the best linear approximation for the known template \mathbf{C} and keeps the same model to approximate the corresponding unknown block \mathbf{P} .

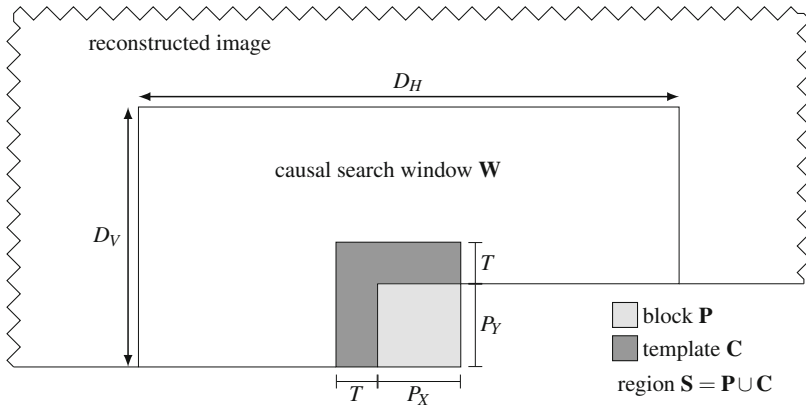


Fig. 2.9 Approximation support (template) and search window used in sparse coding and neighbour-embedding methods

Let vector \mathbf{b} be composed by the N pixel values of region \mathbf{S} , stacked in a column (assuming zero value for unknown pixels of block \mathbf{P}). Also, let an $N \times M$ matrix \mathbf{A} represent the dictionary matrix, a horizontal concatenation of M basis functions represented as column vectors, \mathbf{a}_m , with $m = 0, \dots, M - 1$. An over-complete dictionary is used, i.e. the number of atoms (or basis functions) M is greater than the size N of each function.

The dictionary \mathbf{A} can be built in an adaptive way or using fixed basis functions. In [67], fixed dictionaries based on Discrete Cosine Transform (DCT) or Discrete Fourier Transform (DFT) waveforms are used. Adaptive dictionaries are frequently formed by atoms derived from patches present in a local window in the reconstructed causal neighbourhood [111]. However, dictionary learning methods have also been used for sparse prediction [114]. In this description, adaptive dictionaries formed based on the reconstructed causal texture patches are considered. These dictionaries are built by stacking all the patches similar to region \mathbf{S} , which exist in a predefined causal search window \mathbf{W} , as shown in Fig. 2.9.

The dictionary matrix \mathbf{A} (and vector \mathbf{b}) can be separated into two vertically concatenated sub-matrices \mathbf{A}_c and \mathbf{A}_p (and two vectors \mathbf{b}_c and \mathbf{b}_p), corresponding to the pixels in the spatial location of template \mathbf{C} and predicting block \mathbf{P} , respectively. Sparse representation algorithms aim at approximating the template \mathbf{C} (vector \mathbf{b}_c), by solving the following optimisation problem:

$$\min_{\mathbf{x}} \|\mathbf{b}_c - \mathbf{A}_c \mathbf{x}\|_2^2 \quad \text{subject to} \quad \|\mathbf{x}\|_0 \leq t, \quad (2.3)$$

where $\|\mathbf{x}\|_0$ denotes the L_0 -norm of \mathbf{x} , i.e. the number of non-zero components in \mathbf{x} , and t is a bound on sparsity.

Since searching for the sparsest solution for this problem is NP-hard, approximated solutions are usually estimated. The Matching Pursuit (MP) [63] and Orthogonal Matching Pursuit (OMP) [85] algorithms are iterative greedy methods commonly used to find approximate solutions with tractable computational complexity. Once the optimal coefficients are computed, sparse prediction estimates the predicted signal through $\hat{\mathbf{b}}_p = \mathbf{A}_p \mathbf{x}_{\text{opt}}$, where \mathbf{x}_{opt} is the sparsest solution of Eq. (2.3).

The advantage of the presented approach, based on a template area, is to provide an implicit way to derive the solution of the sparse prediction problem. By solving the problem expressed in Eq. (2.3) exclusively based on the causal reconstructed neighbourhood, specifically through \mathbf{b}_c and \mathbf{A}_c data structures, the decoder can implicitly derive the same sparsest solution \mathbf{x}_{opt} . This approach avoids the bitrate cost associated to the explicit transmission of the sparse solution. However, since the solution \mathbf{x}_{opt} was optimised for sparse representation of the template \mathbf{C} , its prediction performance for the block \mathbf{P} may be reduced. This is so because the solution for the template \mathbf{C} may significantly differ from the optimal sparse solution for the block \mathbf{P} . To obtain a minimal difference to the optimal solution, the assumption of stationary statistics among the template \mathbf{C} and block \mathbf{P} should apply.

2.5.2 Matching Pursuit Methods

The MP algorithm can be used to compute sparse signal representations by iteratively selecting the dictionary atoms. This method provides a possible approximate solution to the problem in Eq. (2.3), which can be rewritten as:

$$\min \{ \|\mathbf{x}\|_0 : \|\mathbf{b}_c - \mathbf{A}_c \mathbf{x}\|_2^2 \leq \rho \} , \quad (2.4)$$

with some admissible approximation error $\rho \geq 0$.

MP procedure iteratively generates a sequence of possible solutions \mathbf{x}_k , which present an increasing number of non-zero components. In first iteration, $\mathbf{x}_0 = \mathbf{0}$ and the initial residual vector is given by $\mathbf{r}_0 = \mathbf{b}_c - \mathbf{A}_c \mathbf{x}_0 = \mathbf{b}_c$. At iteration k , MP selects the basis function, $\mathbf{a}_{c_{m_k}}$ (with m_k referring to the column index number), which has the highest correlation with the residual $\mathbf{r}_{k-1} = \mathbf{b}_c - \mathbf{A}_c \mathbf{x}_{k-1}$. The optimal dictionary atom $\mathbf{a}_{c_{m_k}}$ can be found by

$$m_k = \arg \max_m \frac{|\mathbf{a}_{c_m}^T \mathbf{r}_{k-1}|}{\|\mathbf{a}_{c_m}\|_2}. \quad (2.5)$$

The residual vector at iteration k is then computed as

$$\mathbf{r}_k = \mathbf{r}_{k-1} - \frac{\mathbf{a}_{c_{m_k}}^T \mathbf{r}_{k-1}}{\mathbf{a}_{c_{m_k}}^T \mathbf{a}_{c_{m_k}}} \mathbf{a}_{c_{m_k}} = \mathbf{r}_{k-1} - x_{m_k} \mathbf{a}_{c_{m_k}} \quad (2.6)$$

where x_{m_k} is the weight of the new atom, which is added to \mathbf{x}_{k-1} in order to generate the new sparse solution vector \mathbf{x}_k . MP proceeds with this algorithm until $\|\mathbf{b}_c - \mathbf{A}_c \mathbf{x}_k\|_2^2 \leq \rho$ is satisfied.

As previously stated, the optimal solution estimated for the template area \mathbf{C} may not be the best one for the block \mathbf{P} . For a more efficient prediction of block \mathbf{P} , some methods propose to explicitly select the solution that provides the best prediction results for block \mathbf{P} , from a set of possible solutions. For example, in [67] all the intermediate solutions \mathbf{x}_k , for $k = 1, \dots, k_{\max}$, where k_{\max} is the value of k that satisfied the maximum error ρ , are saved and tested to predict the block \mathbf{P} . The optimum solution \mathbf{x}_{opt} that minimises the block prediction error, given by $\|\mathbf{b}_p - \mathbf{A}_p \mathbf{x}_k\|_2^2$, is selected.

In order to reproduce the prediction process in the decoder side, the iteration k^* associated to the optimum solution \mathbf{x}_{opt} should be explicitly transmitted. The prediction block is calculated as $\hat{\mathbf{b}}_p = \mathbf{A}_p \mathbf{x}_{\text{opt}}$. Algorithm 2.1 summarises the described sparse image prediction procedure using MP algorithm.

Algorithm 2.1: Matching Pursuit algorithm for sparse image prediction**Input:** $\mathbf{A}_c, \mathbf{b}_c, \mathbf{A}_p, \mathbf{b}_p, \rho$ **Output:** $\hat{\mathbf{b}}_p, k^*$

- 1: initialisation: $k = 0, \mathbf{x}_0 = \mathbf{0}, \mathbf{r}_0 = \mathbf{b}_c - \mathbf{A}_c \mathbf{x}_0 = \mathbf{b}_c$
- 2: **repeat**
- 3: $k = k + 1$
- 4: $m_k = \arg \max_m \frac{|\mathbf{a}_{cm}^T \mathbf{r}_{k-1}|}{\|\mathbf{a}_{cm}\|_2}$
- 5: $x_{m_k} = \frac{\mathbf{a}_{cm_k}^T \mathbf{r}_{k-1}}{\mathbf{a}_{cm_k}^T \mathbf{a}_{cm_k}}$
- 6: $\mathbf{r}_k = \mathbf{r}_{k-1} - x_{m_k} \mathbf{a}_{cm_k}$
- 7: $\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{u}_k$, where \mathbf{u}_k is 1-sparse vector with x_{m_k} on position m_k and remaining entries null
- 8: **until** $\|\mathbf{b}_c - \mathbf{A}_c \mathbf{x}_k\|^2 \leq \rho$
- 9: $k_{max} = k$
- 10: $k^* = \arg \min_{k \in [1, k_{max}]} \|\mathbf{b}_p - \mathbf{A}_p \mathbf{x}_k\|_2^2$
- 11: $\mathbf{x}_{opt} = \mathbf{x}_{k^*}$
- 12: $\hat{\mathbf{b}}_p = \mathbf{A}_p \mathbf{x}_{opt}$

2.5.2.1 Orthogonal Matching Pursuit

OMP is a popular extension to the MP algorithm, which is able to provide better results, at the cost of an additional computational complexity [85]. The main difference of OMP relative to MP is that all non-zero coefficients chosen so far are updated at each iteration. In the context of the presented sparse prediction problem, this consists of an orthogonal projection of the template signal \mathbf{b}_c onto the subspace spanned by the dictionary atoms selected so far.

A sparse image prediction method based on OMP is summarised in Algorithm 2.2. Note that \mathbf{A}_c^k refers to a compacted matrix, which contains a sub-set of the atoms of matrix \mathbf{A}_c , specifically all the atoms selected until iteration k . As for stopping condition, alternative approaches that limit the maximum number of iterations can be used.

2.5.3 Template Matching Algorithm

The TM algorithm [105] can be interpreted as a particular case of the presented sparse prediction method using MP, in which only one iteration is performed and the weighting coefficient is equal to 1. In practice, TM algorithm can be formulated as a search procedure which compares the reference template with all equally shaped candidate templates existing in a causal search window (the atoms of the dictionary). The prediction of the unknown block is given by the reconstructed block associated (or adjacent) to the candidate template, which resulted in the lowest matching error. Typically, the reconstructed pixels belonging to the neighbourhood of the block to be predicted are used as reference template, equivalently to the area C, of Fig. 2.9.

Algorithm 2.2: Orthogonal Matching Pursuit algorithm for sparse image prediction

Input: $\mathbf{A}_c, \mathbf{b}_c, \mathbf{A}_p, \mathbf{b}_p, \rho$

Output: $\hat{\mathbf{b}}_p, k^*$

1: initialisation: $k = 0, \mathbf{x}_0 = \mathbf{0}, \mathbf{r}_0 = \mathbf{b}_c, \mathbf{A}_c^0 = [\], \mathbf{A}_p^0 = [\]$

2: **repeat**

3: $k = k + 1$

4: $m_k = \arg \max_m \frac{|\mathbf{a}_{c_m}^T \mathbf{r}_{k-1}|}{\|\mathbf{a}_{c_m}\|_2}$

5: $\mathbf{A}_c^k = \mathbf{A}_c^{k-1} \cup \{\mathbf{a}_{c_{m_k}}\}$ and $\mathbf{A}_p^k = \mathbf{A}_p^{k-1} \cup \{\mathbf{a}_{p_{m_k}}\}$

6: $\mathbf{x}_k = (\mathbf{A}_c^{kT} \mathbf{A}_c^k)^{-1} \mathbf{A}_c^{kT} \mathbf{b}_c = \mathbf{A}_c^{k+} \mathbf{b}_c$

7: $\mathbf{r}_k = \mathbf{b}_c - \mathbf{A}_c^k \mathbf{x}_k$

8: **until** $\|\mathbf{b}_c - \mathbf{A}_c^k \mathbf{x}_k\|^2 \leq \rho$

9: $k_{max} = k$

10: $k^* = \arg \min_{k \in [1, k_{max}]} \|\mathbf{b}_p - \mathbf{A}_p \mathbf{x}_k\|_2^2$

11: $\mathbf{x}_{opt} = \mathbf{x}_{k^*}$

12: $\hat{\mathbf{b}}_p = \mathbf{A}_p \mathbf{x}_{opt}$

Improved variations of TM algorithm have been proposed and implemented in H.264/AVC standard, e.g. TM based on the averaging of multiple predictors [106] and TM using adaptive illumination compensation methods [129]. The use of TM algorithm in addition to the traditional intra prediction modes has shown to be advantageous to predict textured areas. Applications of BMA for intra image prediction have also shown to provide effective prediction results [127]. The principle of both methods is to reuse repeated patterns along the image. However, BMA requires some kind of signalling to indicate the optimal matched block in the causal reconstructed area, while TM provides an implicit way to find the matching block in the decoder. Hybrid approaches combining both BMA and TM for intra prediction have also been investigated in the past [15].

2.5.4 Neighbour Embedding Methods

Neighbour embedding methods form a subset of data dimensionality reduction methods, which have been successfully applied for intra image prediction. The idea of these methods is to search for the NNs (in terms of the Euclidean distance) of the input data point and then to compute the linear combination of the NNs that best approximate the input data point.

Data dimensionality reduction methods rely on the assumption that the real world is sampled from a non-linear low dimensional manifold, which is embedded in the high dimensional space. When applied to image prediction, these methods learn the neighbour embedding of input data, but they do not proceed to the computation of the low dimensional space. Typical proposals find the linear combination of the

NN patches which best approximates some known data, e.g. the causal template in the block neighbourhood, and then extrapolate that linear relationship to the NNs of the unknown block to be predicted. This kind of operation makes neighbour embedding methods very similar to sparse prediction techniques based on MP methods. Actually, neighbour embedding methods can be interpreted as sparse image representations in which the selected dictionary atoms (i.e. the elements that participate in sparse solution) are given by k -NN patches. The investigation of two neighbour embedding methods for image prediction was performed in [16, 113], namely the Non-negative Matrix Factorisation (NMF) and Locally Linear Embedding (LLE) techniques. The main difference between these methods is related to the method used to compute the linear coefficients associated to the NN patches.

The NMF method consists in a low rank approximation of the input data, given by the product of two matrices of lower dimension whose elements are non-negative. One of these matrices typically contains basis functions (in its columns) that provide a good linear approximation of the input data. The non-negative constraint only allows additive linear combinations, which in some applications can be useful for physical interpretation of the results. In the case of image data, which is made of non-negative values, this constraint ensures a non-negative prediction result. Similarly to the sparse prediction problem given by Eq. (2.3), the NMF image prediction method is formulated as a constrained least-squares problem, which imposes non-negative coefficients. A sparsity constraint is also imposed by using this method over a compacted matrix formed by k -NN patches.

The LLE has been designed to learn the underlying non-linear manifolds embedded in the high dimensional data. The method exploits the local linear characteristics of the high dimensional data, in order to generate a lower dimensional representation which preserves those characteristics. Applications of LLE to intra image prediction have shown better results than NMF method [113]. Given its importance for the research work described in this book, the description of the LLE algorithm for image prediction as proposed in [113] is presented in what follows.

2.5.4.1 Image Prediction Based on LLE

The LLE method first searches a representation of the template \mathbf{C} (refer to Fig. 2.9) using a linear combination of the k -NN patches defined in the matrix \mathbf{A}_c , built from the causal search window \mathbf{W} , as previously defined in Sect. 2.5.1. Then, the same estimated coefficients are used to predict the block \mathbf{P} , by linearly combining the corresponding k -NN patches defined in matrix \mathbf{A}_b .

The k -NN method works like a sparsity constraint by choosing the k closest patches (represented by the columns of matrix \mathbf{A}_c) to the template \mathbf{C} (vector \mathbf{b}_c), in terms of Euclidean distance. The sub-matrix of \mathbf{A}_c containing the selected k -NN patches is denoted by \mathbf{A}_c^k . In order to compute the linear weights, LLE problem solves a least-squares problem, which imposes a sum-to-one constraint on the linear weights. This constraint is part of LLE algorithm forcing the approximation of each data point to lie in the linear subspace spanned by its nearest neighbours.

The enunciated problem can be written as

$$\min_{\mathbf{x}_k} \|\mathbf{b}_c - \mathbf{A}_c^k \mathbf{x}_k\|_2^2 \quad \text{subject to} \quad \sum_m \mathbf{x}_{k_m} = 1. \quad (2.7)$$

where \mathbf{x}_k is the solution vector containing the k optimal linear coefficients. Note that, although sparsity constraint is not explicit in problem formulation, the selection of a limited number of columns of matrix \mathbf{A}_c by k -NN method imposes the sparse representation.

The solution to this problem can be obtained based on a covariance matrix \mathbf{D}_k , computed for the k -NN templates of sub-matrix \mathbf{A}_c^k in reference to the template \mathbf{b}_c , by solving the linear system $\mathbf{D}_k \mathbf{x}_k = \mathbf{1}$ (where $\mathbf{1}$ is the column vector of ones), and then rescaling the weights so that they sum to one. The predicted block is then computed using $\hat{\mathbf{b}}_p = \mathbf{A}_p \mathbf{x}_k$.

For an improved performance, the proposal of [113] suggests to vary the sparsity constraint by testing several k -NNs, e.g. with $k = 1, \dots, K$. The number of k -NNs can be either constant for the whole encoding process or explicitly signalled to the decoder. In the second case, the optimal number of NNs is selected by testing the block \mathbf{P} prediction accuracy for each k value. The Algorithm 2.3 describes the presented LLE method for intra image prediction, assuming the explicit selection and signalling of the optimal sparsity value k^* .

Experiments performed in [113] using H.264/AVC standard demonstrate the advantage of the described LLE-based prediction method. The reported results show that LLE provides superior prediction quality and compression efficiency than directional prediction, TM algorithm, as well as other sparse prediction algorithms based on MP method.

Algorithm 2.3: LLE-based algorithm for intra image prediction

Input: $\mathbf{A}_c, \mathbf{b}_c, \mathbf{A}_p, \mathbf{b}_p, K$

Output: $\hat{\mathbf{b}}_p, k^*$

- 1: initialisation: $k = 0, \mathbf{A}_c^0 = [], \mathbf{A}_p^0 = [], \mathbf{A}_c'^0 = \mathbf{A}_c$
 - 2: **repeat**
 - 3: $k = k + 1$
 - 4: $m_k = \arg \min_{m | a_{cm} \in \mathbf{A}_c^{k-1}} \{d_m\}$ where $d_m = \|\mathbf{b}_c - \mathbf{a}_{cm}\|_2^2$
 - 5: $\mathbf{A}_c^k = \mathbf{A}_c^{k-1} \cup \{\mathbf{a}_{cm_k}\}$ and $\mathbf{A}_p^k = \mathbf{A}_p^{k-1} \cup \{\mathbf{a}_{pm_k}\}$
 - 6: $\mathbf{A}_c'^k \leftarrow \mathbf{A}_c \setminus \{\mathbf{A}_c^k\}$
 - 7: calculate covariance matrix \mathbf{D}_k of \mathbf{A}_c^k in reference to \mathbf{b}_c
 - 8: solve $\mathbf{D}_k \mathbf{x}_k = \mathbf{1}$ for \mathbf{x}_k
 - 9: $\mathbf{x}_k = \frac{\mathbf{x}_k}{\sum \mathbf{x}_k}$
 - 10: **until** $k = K$
 - 11: $k^* = \arg \min_{k \in [1, K]} \|\mathbf{b}_p - \mathbf{A}_p^k \mathbf{x}_k\|_2^2$
 - 12: $\mathbf{x}_{opt} = \mathbf{x}_{k^*}$
 - 13: $\hat{\mathbf{b}}_p = \mathbf{A}_p^{k^*} \mathbf{x}_{opt}$
-

In a more recent research work, modified variants of the neighbour embedding algorithms, denominated as correspondence map-aided neighbour embedding methods, were proposed [16]. The idea of these methods is to enable alternative procedures to select the k -NNs, which make use of auxiliary information transmitted to the decoder. These methods solve some limitations of implicit NNs selection, namely when the template and the block to be predicted are not correlated.

2.6 Conclusions

In this chapter we presented an overview of several prediction techniques proposed for image and video compression. Predictive techniques are of great importance in the design of current image and video coding algorithms, being the main focus of the investigation described in this book.

Section 2.2 presented the basis of predictive image coding, describing the legacy DPCM encoder and its relations with the current lossless and lossy image and video encoders. In Sect. 2.3, the state-of-the-art on image and video prediction, namely the directional intra prediction and motion-compensated prediction methods were described, as proposed in the recent HEVC standard. In this book, the directional intra prediction has been investigated for efficient depth map coding in Chap. 4.

Two important families of prediction techniques were also presented in this chapter, namely the linear prediction methods, based on least-squares optimisation described in Sect. 2.4, and the sparse representation-based prediction methods described in Sect. 2.5. Due to the importance of linear prediction and sparse representation, alternative techniques based on these principles are discussed in this book for the compression of generic images. For instance, an extended version of the LLE-based prediction method, described in Sect. 2.5.4, is presented in Chap. 5 for the HEVC encoder.

Efficient Predictive Algorithms for Image Compression

Rosário Lucas, L.F.; Barros da Silva, E.A.; Maciel de
Faria, S.M.; Moraes Rodrigues, N.M.; Liberal Pagliari, C.
2017, XIX, 169 p. 66 illus., 24 illus. in color., Hardcover
ISBN: 978-3-319-51179-5