2
Literature Survey

In this chapter, a detailed review of the existing work in the field of motion estimation and predictive coding is provided. First, attempts are made to provide the basics of data compression with primary emphasis on lossless predictive coding and lossy video coding. Secondly, motion estimation, which plays a crucial role in the lossy video coding framework, is introduced in detail. Later, typical computational challenges in motion estimation processes are described along with various computationally efficient solutions for motion estimation. Advanced algorithms used for motion estimation in surveillance videos are also described in detail. Lastly, a special case for human skeleton information coding is explored. To this end, various challenges in the existing motion estimation approaches and skeleton predictive coding frameworks are explored.

The remainder of the chapter is organized as follows. Section 2.1 describes the basics of data compression. The motion estimation process, which plays a vital role in video compression, is described in Section 2.2. Skeleton sequence coding methods are presented in Section 2.3. Section 2.4 summarizes the chapter.

2.1 BASICS OF DATA COMPRESSION

The compression of information is a reduction in the total number of bits needed to represent data. Compression is useful because it reduces data storage and transmission resources. In the compression process and usually in the process reversal (decompression), computational resources are consumed. The compression of data is subject to a trade-off between space and time complexity. Data compression can save storage space significantly, speed up data transmission and, in effect, reduce data storage capacity and network bandwidth costs. The idea of data compression dates back to 1838 when [Barrett, 1875] invented Morse codes. Morse code is a character encoding scheme that provides shorter Morse codes for the more common letters in the English language such as 'a', 'e', and 't'. Modern work on data compression began in the late 1940s when [Shannon, 1948] and [Fano, 1949] presented systematic approaches for information theory and coding.

At the heart of data compression, lies information theory concepts, which act as the blood pumping the heart. In information theory, it is of utmost importance to understand the probabilistic behavior of the source data that needs to be compressed. This probabilistic behavior representing the distribution of the occurrence of symbols from the source data could provide theoretical bound on bit-rate. An information-theoretic bit-rate bound is computed using entropy. Information entropy is the average bit-rate at which a stochastic source of data produces information. An information entropy indicates the amount of information particular source data carries. To this end, a mathematical representation was developed such that, a data-source with low-probability value carries more information than the data-source with high-probability. [Shannon, 1948] presented a mathematical representation for an information entropy, and it is computed as:

$$Entropy = \sum_i P(x_i) \times log_b(1/P(x_i)) \tag{2.1}$$

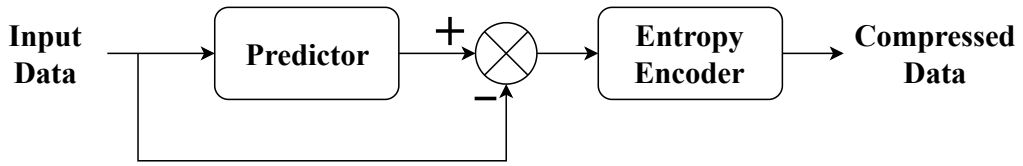

Figure 2.1 : Illustration of a differential encoding scheme.

where x_i is the i^{th} symbol in the data source, $P(x_i)$ is the probability of occurrence of i^{th} symbol, and b is the base of the logarithm used. A typical value of b is 2 for binary representation. This representation is considered as the information-theoretic bound on bit-rate in the data compression process. In later years, numerous works on data compression specific to the type of data were presented. In general, the data compression methods can be broadly categorized into two types: (1) lossless compression, and (2) lossy compression. The details of these categories are provided in the next sections.

2.1.1 Lossless Compression

Lossless compression techniques do not allow any loss of data, as their name implies. When data has been compressed losslessly, it is possible to recover the original data from the compressed data exactly. The core principle of the lossless compression is to minimize the storage capacity required to represent the original input data without losing any information. Hence, lossless compression is also termed as a reversible process. Lossless compression is generally used in applications where the original and reconstructed data can not tolerate any difference. Lossless compression is possible because there is statistical redundancy in most real-world data. The statistical redundancy mostly comes from the probability distribution of symbols in the source data. In entropy based encoding schemes, the symbols which are highly probable are assigned lower bit-length codes, and symbols that are less probable are assigned higher bit-length codes. By doing so, we aim to get as close as possible to the theoretic entropy bound. Hence, the statistical redundancy helps in reducing the bit-rate requirement.

In practice, various entropy coding methods are employed for lossless data compression. In entropy coding, each symbol is assigned a unique prefix-free code. Then, each symbol in the source data is encoded using the corresponding prefix-free code, such that it is exactly recovered at the decoder. The Lempel-Ziv compression methods are among most popular lossless compression algorithms [Ziv and Lempel, 1977, 1978; Wyner and Ziv, 1976; Lempel and Ziv, 1986]. However, it was Huffman encoding and Arithmetic encoding schemes that significantly contributed to lossless data compression. These algorithms are simple to understand and easy to implement. The detailed overview of these entropy coding schemes is out of the scope of this Thesis. However, discussion on various approaches for the reduction in entropy for source data is of key importance. To this end, the simplest entropy reduction technique with data prediction is explored in this survey.

The data sources, such as images and videos, have a significant correlation from sample to sample. This fact can be used to predict the sample value based on its adjacent sample values. The strong correlation between samples contributes to the optimal sample prediction. Hence, only the difference between the original sample value and predicted sample value could be encoded, resulting in a reduction in entropy. This technique is termed as differential encoding. A simple illustration of a differential encoding scheme is shown in Figure 2.1. In real-world data, sample values do not change drastically from one sample to the next sample. Hence, this means that the entropy of a sequence of differences between consecutive sample values is significantly smaller than the original data source. The difference between consecutive sample values could be computed as:

$$d_i = x_i - x_{i-1} \tag{2.2}$$

where x_i denotes the i^{th} sample in the data source sequence. For highly correlated data sources, the probability distribution of d_i is highly peaked at zero. This kind of distribution makes a critical case for entropy coding. Hence, the difference sequence d_i is more suitable for data compression than the original data source sequence x_i. As discussed in the previous chapter, the lossless compression has higher bit-rate requirements as compared to lossy compression techniques. Next, our emphasis and discussion are on lossy compression techniques, which contributes to better compression performance.

2.1.2 Lossy Compression

Lossy compression techniques involve some information loss, and data compressed using lossy techniques can not necessarily be recovered or accurately restored. We can generally get much higher compression ratios than it is possible with lossless compression in return for accepting this distortion in the reconstruction. Hence, lossy data compression permanently removes some inherent redundancies that are unimportant or imperceptible. To this end, the entire focus is on achieving a better trade-off between preserving information and reducing storage capacity requirements. In practice, only a small loss in information could provide a significant reduction in storage space requirements. It is to bring to readers kind notice that, this small loss in information is permanent. Hence, lossy compression is also termed as an irreversible process. This compression is used only in the applications where perfect reconstruction of the original data is not of critical importance. In such cases, our target is to find any possible repetitive patterns which can help to identify redundant information in the given data. These data redundancies could lead to better compression.

In the best possible scenario, we would like a data compression system to incur a minimum loss in information while reducing the minimum possible bit-rate. Hence, for obvious reasons, there is a trade-off to be achieved between minimizing the loss in information or distortion and minimizing the bit-rate. The study of this trade-off is termed as rate-distortion theory in the field of data compression. Rate is defined as the average number of bits used to represent each sample value. On the other hand, distortion is defined as the amount of similarity or fidelity of a reconstructed sample sequence to the original data source sequence. The simplest way to compute fidelity is to evaluate the differences between the reconstructed sequence and the original sequence. The popular distortion metrics are the sum of absolute differences (SAD), mean of absolute differences (MAD), and mean squared error (MSE). If x_i is the original data sequence and y_i is the reconstructed data sequence, then SAD, MAD and MSE is computed as:

$$SAD = \sum_i |x_i - y_i| \tag{2.3}$$

$$MAD = \frac{1}{N}\sum_i |x_i - y_i| \tag{2.4}$$

$$MSE = \frac{1}{N}\sum_i (x_i - y_i)^2 \tag{2.5}$$

where N is the length of the data sequence, it can be understood that when the reconstructed sequence is highly similar to the original sequence, then, resulting distortion value would be small. The rate-distortion theory plays a vital role in limited bandwidth systems, especially in video compression systems.

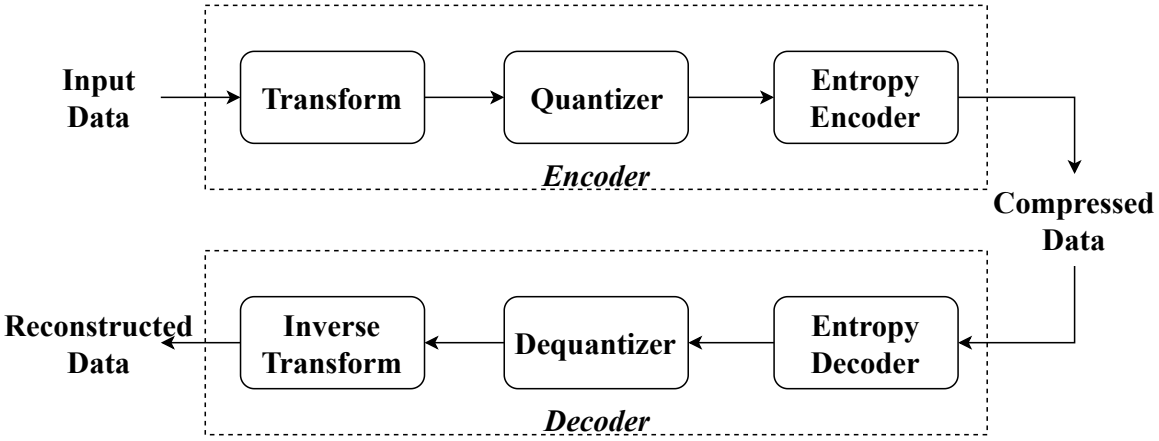

Figure 2.2 : Illustration of a lossy encoding and decoding scheme.

2.1.3 Lossy Compression of Video Data

It is a well-known fact that video data sources generate probably the most substantial amount of data. Hence, video compression is of vital importance in limited bandwidth scenarios. Video compression can be treated as compression of images in the temporal dimension. The temporal correlation among consecutive frames in the video is remarkably high and contributes to the considerably high temporal redundancy. The temporal redundancy is widely explored in the video compression standards.

To this end, an illustration of the coding principle, which has been adopted in the state-of-the-art lossy compression techniques, is shown in Figure 2.2. This simple illustration contains both the encoding and decoding processes. In the encoder, first, the input data is transformed into a more compressible form. For compression, Discrete Cosine Transform (DCT) is the widely adopted transformation. Secondly, the transformation coefficients can be quantized to remove redundant information. The quantizer introduces additional distortion to achieve better compression. Lastly, the quantized coefficients are encoded into the binary code-words using entropy encoder. An entropy encoder ensures the variation in the length of code words such that the length of code-words is inversely proportional to the frequency of occurrence. The entire process is termed as encoding.

On the other hand, reverse operations are performed at the decoder to reconstruct the original signal. It should be noted that the original signal can only be reconstructed exactly in the lossless encoding and decoding process. In this case, the quantization step in the encoder is skipped, and transformation coefficients are directly entropy coded. However, for most of the application scenarios, where real-time encoding and transmission is of paramount importance, the lossy encoder is employed. The best example of the lossy encoder is H.264 [Marpe *et al.*, 2006] and HEVC [Sullivan *et al.*, 2012], which is used for video compression.

The standard video encoder is built using two key components, namely: (1) Discrete Cosine Transform and (2) Motion compensation. DCT is a basis in image and video compression standards, and it is widely adopted for intra-frame coding. On the other hand, motion compensation helps in improved inter-frame coding. The typical video encoder is shown in Figure 2.3. The encoder can judiciously choose between intra-frame or inter-frame coding schemes. For intra-frame coding, a frame is divided into blocks of size 8×8. Then, the 2D-DCT is employed on each block, and the obtained coefficients are entropy coded for better compression. On the other hand, for inter-frame coding, a motion-compensated frame is obtained. The motion-compensated frame is expected to be very similar to the current frame. In turn, this would result in lower differences between the motion-compensated

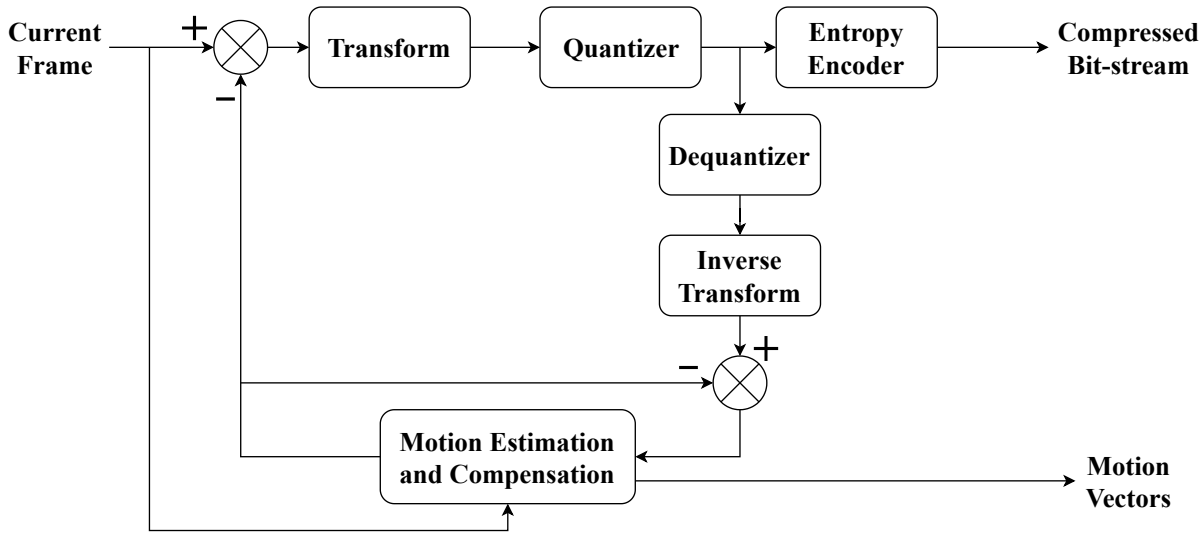

Figure 2.3 : Illustration of a video encoding process.

frame and the current frame. The difference between these two frames is termed as a prediction error. The distribution of prediction error is highly peaked at zero. Hence, the prediction error is entropy coded to achieve better compression.

The compression capability strongly depends on the quantizer employed to transform coefficients. In the absence of quantizer, it is theoretically possible to achieve lossless compression. That means we can reconstruct the frame that is the same as the original frame. On the other hand, the introduction of quantizer incurs the distortion in the reconstructed frame, which in turn results in lossy compression. The quantization is used to remove the least important transform coefficients and roughly approximate the remaining non-zero transform coefficients. This mechanism intends to remove the least important data, which is not perceptible by the human visual system. To this end, low-frequency coefficients are lightly quantized, whereas high-frequency coefficients are heavily quantized. Since the human visual system is more concerned about low-frequency changes as compared to the high-frequency changes, this type of adaptive quantization is more suitable to achieve trade-off compression performance. [Jamali and Coulombe, 2019] reported rate-distortion trade-off in HEVC. The value of the quantizer step depends on the quantization parameter (QP) value, which may vary from 0 to 51. It should be emphasized that higher QP value results in higher quantization, which contributes not only to the better compression but also to the higher distortion. Hence a rate-distortion trade-off needs to be established during the video compression process. This combination results in an efficient video compression process. The key component in the video compression process is motion estimation, which is discussed in detail in the next section.

2.2 MOTION ESTIMATION: REVIEW AND CHALLENGES

Motion estimation, which used to reduce temporal redundancies through successive frame matching, plays a vital role in video compression. In this section, firstly, the typical block-based motion search process is explained, followed by a discussion on computational complexity associated with these algorithms. The discussion on existing efficient motion search solutions is presented to understand the research gaps. This discussion will be further extended from motion search for generic video sequences to motion search for a specific type of video sequences. Hence, it is expected that at the end of this section, the readers get all necessary knowledge about the motion estimation process

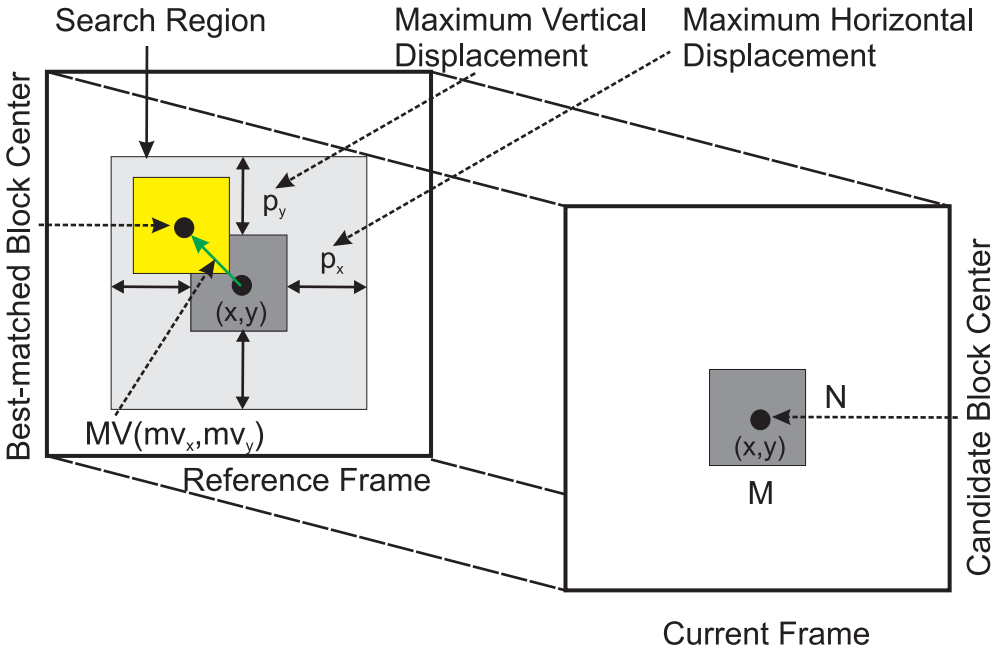

Figure 2.4 : Traditional block matching process for $M \times N$ candidate block.

with special emphasis on challenges associated with traditional approaches.

2.2.1 Block-based Motion Search Algorithms

For motion estimation, the current frame is divided into non-overlapping blocks of size $M \times N$ referred to as candidate blocks. For each candidate block, the most matched block is searched in the reference frame, typically the previous frame. This process is termed as block matching. The basic block matching process is illustrated in Figure 2.4. The block matching for each candidate block is carried out within the predefined search area. The size of the search area can be kept limited. Video sequences acquired at a high frame rate will have significant similarity in its consecutive frames. Hence, a limited search area, termed as search region (SR), in the reference frame is required in the motion search process for the candidate blocks. The size of SR is based on search displacement parameter $p = (p_x, p_y)$, where p_x and p_y denotes maximum displacement in horizontal and vertical directions, respectively.

For computing matching between two blocks, various metrics, based on distortion error function, are reported in the literature. [Chen *et al.*, 1995] presented a overview of these criteria in detail. The list of block matching criteria includes SAD, MAD, and MSE, among others. In general, SAD metric is preferred over other distortion error functions. SAD is a popularly used measure of similarity between two frame blocks: candidate block (C) and reference block (R). For the block of size $M \times N$, SAD is computed as:

$$SAD(x,y,m_x,m_y) = \sum_{i=0}^{M-1}\sum_{j=0}^{N-1} \left| \begin{matrix} C_{(x+i,y+j)} - \\ R_{(m_x+x+i,m_y+y+j)} \end{matrix} \right| \tag{2.6}$$

where, (x,y) represents position of the candidate block and (m_x,m_y) denotes relative displacement from the candidate block location. The displacement between the candidate block and the best-matched block in SR is termed as the motion vector (MV), as shown in Figure 2.4. The FS method is reported to provide accurate MV since it computes the SAD values at all possible locations

Table 2.1 : Overview of popular block matching algorithms (BMA).

BMA	Description	Computational Complexity
FS	FS [Lin and Tai, 1997] exhaustively search at each possible location in the search window and hence it is the computationally extensive method.	$(2p+1)^2$
DS	DS [Zhu and Ma, 2000] uses two search patterns: small (SDSP), and large (LDSP) diamond-shaped patterns. DS provides PSNR as similar to FS with significantly reduced computations.	$9+5n+4$
HS	HS [Zhu *et al.*, 2002] uses two search patterns: small (SHSP), and large (LHSP) hexagon-shaped patterns. HS slightly compromise on PSNR as compared to DS, but with significantly fewer computations.	$7+3n+4$
CDHS	CDHS [Cheung and Po, 2005] firstly uses two cross-shaped search patterns: small (SCSP), and large (LCSP) cross-shaped patterns. Then it employs two diamond-shaped patterns: small (SDSP), and large (LDSP). The computations are further reduced by using LHSP instead of LDSP. CDHS provides PSNR as similar to HS with reduced computations.	$5+4+2+3n+4$
ARPS	ARPS [Nie and Ma, 2002] rely on the fact that the motion of the candidate block is usually coherent, and hence it uses MV of the neighboring block to predict its MV. In general, ARPS outperforms other algorithms due to its ability to provide higher PSNR at a very lower number of computations.	$1+5+4+3n$
TZS	TZS [JCT-VC, 2013] uses MV prediction technique comprising of five predictors: median, left, up, up-right, and zero predicted MV. Then it combines diamond search and raster search methods to outperform FS. Sometimes, square shaped search patterns are employed instead of diamond search. In TZS, *iRaster* is raster scan sampling factor for the search window, *uiBD* is *uiBestDistance* where $uiBD = \{iRaster, \frac{iRaster}{2}, \frac{iRaster}{4}, ..., 0\}$.	$1 + 4 + 8(\lfloor \log_2(p) \rfloor) + \left(\left\lceil \frac{2p+1}{iRaster} \right\rceil\right)^2 + 8 \sum_{uiBD} \lfloor \log_2 \left(\frac{uiBD}{2}\right) \rfloor$

where p is search displacement parameter, and n is number of intermediate search steps.

in SR. The block with the minimum SAD value is considered as the best-matched block. Since the FS technique exhaustively searches all possible locations, it is considered as a computationally expensive method.

2.2.2 Computational Complexity and Efficient Motion Search Algorithms

In practice, motion search complexity for the FS algorithm is maximum. For example, motion search complexity for a candidate block of size 16×16 with SR $p = \pm 8$ would require SAD computations at all $(2p+1)^2 = 289$ search points. It should be noted that a single frame contains multiple candidate blocks, and hence, the computational complexity is high and does not suit for practical applications. In end-to-end video compression, almost 60-70% of the computational time is consumed by the motion estimation process, alone.

For this problem, several efficient motion search approaches were presented in the literature. In general, computational complexity can be reduced by (1) reducing the total number of search points, and (2) reducing the total number of computations required to compute distortion measure. On this basis, these efficient motion search approaches are broadly categorized into two classes: (1) fast block matching algorithms, and (2) partial distortion measure based block matching algorithms.

Firstly, various approaches to reducing the total number of search points are discussed. To understand the importance of fast block matching algorithms, it is also important to understand the Full Search (FS) algorithm. [Lin and Tai, 1997] reported that the FS technique provided the best block matching performance due to the use of all possible locations in the search region. However, FS requires a significantly high computational cost. Several computationally efficient block matching algorithms based on various search patterns are reported in literature such as: cross-search (CS) [Ghanbari, 1990], three-step search (TSS) [Koga, 1981], new three-step search (NTSS) [Li *et al.*, 1994], four-step search (FSS) [Po and Ma, 1996], diamond search (DS) [Zhu and Ma, 2000], hexagon-shaped search (HS) [Zhu *et al.*, 2002], cross diamond search (CDS) [Lam *et al.*, 2003], kite cross diamond search (KCDS) [Lam *et al.*, 2004], cross diamond hexagon-shaped search (CDHS) [Cheung and Po, 2005], modified hexagon grid search (MHGS) [Singh and Ahamed, 2018], and adaptive rood pattern search (ARPS) [Nie and Ma, 2002], among others. Overview of these state-of-the-art algorithms is summarized in Table 2.1.

DS and its variants are widely used block matching algorithms. [Zhu and Ma, 2000] presented the DS algorithm, which uses two diamond-shaped search patterns. The first one referred to as a large diamond search pattern (LDSP), and the second one being a small diamond search pattern (SDSP), which consists of nine and five checking points respectively in the diamond shape. The motion search in DS is straightforward and easy to implement. LDSP is continuously used in the DS algorithm until the minimum SAD block is found in the search center. If the best block is in the center, then SDSP will be used to find the optimum motion vector. DS works well with wide variations in motion content for various types of motion sequences. It can provide better performance than both TSS and NTSS. This algorithm can accurately find a global minimum as it has block matching accuracy similar to that of FS with a significantly lower computational cost. However, DS suffers from local minimum and early search termination problems. These problems might arise issues in our aim for faster motion search convergence without compromise in block matching accuracy. For videos containing directional motion, the traditional DS algorithm is not suitable since it still requires several search points for convergence. In a similar way, other fast block matching algorithms fail to provide optimal performance.

In fast block matching algorithms, the improvement in computational efficiency is at the expense of some loss in matching performance. [Purwar and Rajpal, 2013] reported that, for fast motion content video sequences, almost all of the previous methods provide unsatisfactory performance due to their fixed search patterns. On the other hand, ARPS uses predicted MV to form an adaptive search pattern and provides matching performance similar to that of FS, with enormously alleviated computations. The test zone search (TZS) algorithm is employed in the reference software of the latest HEVC standard called HM software [JCT-VC, 2013], which is claimed to provide superior matching performance as compared to other fast block matching algorithms, but at a much higher computational cost.

Secondly, various approaches in reducing the total number of computations required to compute distortion measure are discussed. [Cheung and Po, 1997] presented hierarchical block matching techniques and [Lee, 2010] reported partial distortion measure calculations, which provided a significant reduction in computations. [Cheung and Po, 2003a] described that the partial distortion measure is based on the assumption that all the pixels of the candidate block belong to the homogeneous region and hence have the same distortion behaviour. However, [Cheung and Po, 2000] reported that these methods provide sub-optimal performance compared to the full distortion measure. SAD is a popularly used measure of similarity between two frame blocks: candidate block (C) and reference block (R). [Yi and Ling, 2007] presented sub-sampling in pixel domain, which substantially speeds-up the calculation of SAD, due to reduction in the total number of pixels used for this purpose. In literature, [Xia *et al.*, 2015] reported that $1:2$, and $1:4$ sub-sampled SAD is mostly used, which provides 50% and 75% reduction in computations, respectively. The $1:2$, and $1:4$ sub-sampled SAD is computed as follows:

$$SAD_{1:2} = \sum_{i=0}^{\lfloor \frac{M-1}{2} \rfloor} \sum_{j=0}^{\lfloor \frac{N-1}{2} \rfloor} \left[\left| \begin{matrix} C_{(x+2i,y+2j)} - \\ R_{(m_x+x+2i,m_y+y+2j)} \end{matrix} \right| + \left| \begin{matrix} C_{(x+2i+1,y+2j+1)} - \\ R_{(m_x+x+2i+1,m_y+y+2j+1)} \end{matrix} \right| \right] \tag{2.7}$$

$$SAD_{1:4} = \sum_{i=0}^{\lfloor \frac{M-1}{2} \rfloor} \sum_{j=0}^{\lfloor \frac{N-1}{2} \rfloor} \left| \begin{matrix} C_{(x+2i,y+2j)} - \\ R_{(m_x+x+2i,m_y+y+2j)} \end{matrix} \right| \tag{2.8}$$

However, due to sub-sampling, the accuracy of the MV prediction is compromised. The accuracy of the MV prediction is observed to be superior in a $1:2$ sub-sampled pixel structure as compared to a $1:4$ sub-sampled pixel structure [Seidel *et al.*, 2015]. The search for MV is terminated at a search

point whenever the SAD value is lower than the pre-set threshold value (T). Traditionally, $T = 512$ is used for blocks of size 16×16 and the threshold value is modified by a scaling factor of $1/\beta$ where $1 : \beta$ sub-sampling is employed, i.e. $T_{1:\beta}^{new} = T/\beta$. Both the techniques discussed earlier contribute to computational complexity reduction. However, the motion search process convergence still requires multiple iterations. These methods are highly likely to fail for specific directional motion sequences. In the pursuit of faster motion search convergence, a highly likely motion search center could be obtained. Then, the motion search could start directly from the newly obtained search center instead of the traditionally employed zero search center.

For exploiting the inherent characteristics of blocks with direction-oriented motion, motion vectors of adjacent blocks can be explored. Motion vectors of adjacent blocks are used for the prediction of the initial search center of the candidate block. [Chen, 2000] and [Luo *et al.*, 2015] reported an initial search center prediction scheme, which provided a notable reduction in computational cost and also improved matching efficiency. [Nisar and Choi, 2009] claimed that these methods are biased with the assumption that the candidate block follows the same motion trajectory as their neighboring blocks. This assumption severely hits the matching accuracy for blocks located at the object boundaries, as these blocks may follow different motion trajectory than its neighboring blocks. [Lin *et al.*, 2009] described that, although these algorithms can rapidly converge to the distortion minimal location, they suffer from becoming trapped in the local minimum. Although the approaches discussed above provide faster convergence and considerable reduction in computational complexity, they still need to reduce computational requirements for real-time processing and analysis. For real-time motion search, the primary focus is on reducing computational complexity without any compromise in matching accuracy. Surveillance video is the best example of real-time processing and analysis. Today, surveillance video is almost everywhere, mainly for security purposes. Hence, it is of utmost importance to develop computationally efficient motion search algorithms for motion search in surveillance videos.

2.2.3 Efficient Motion Search Algorithms for Surveillance Videos

Surveillance video plays a vital role in today's security and navigation applications. The need for fine-tuned motion search algorithms to address computational efficiency challenges in surveillance videos is prevalent. To understand this, let us first review the motion search solutions for natural videos, followed by a brief discussion on motion search algorithms tailored specifically for surveillance videos. In literature, the H.264 standard uses different variants of DS and HS patterns, whereas the HEVC employed the TZS pattern for better motion estimation accuracy. [JCT-VC, 2013] analyzed the flow of the TZS algorithm. Various attempts were made to reduce the computational complexity of TZS [Nalluri *et al.*, 2015; Singh and Ahamed, 2018; Guarda *et al.*, 2017; Jia *et al.*, 2019; Oh, 2018; Zhang *et al.*, 2015]. Still, TZS suffers from very high computational requirements. In another approach, hexagon pattern based motion search reported providing faster convergence without significant compromise in matching performance. [Singh and Ahamed, 2018] presented the MHGS algorithm, which uses different mechanisms of complexity reduction in addition to hexagon patterns. [Poularakis *et al.*, 2017; Manafifard *et al.*, 2017] reported a reduction in computational complexity for fast activity recognition applications. These algorithms use different search patterns for reducing the total number of search points. [Zhu and Ser, 2005] described that different early search termination methods also reduce the number of search points. [Yang and Chen, 2002] reported the zero-motion termination method, which compared the distortion measure: the SAD value with a threshold at the zero MV point. The search process is terminated if the SAD value is less than the threshold. In another approach, [Cheung and Po, 2003b] reported that the partial distortion measures (partial SAD) could be employed to reduce the total number of computations involved at each search point. [Shinde and Tiwari, 2018] proposed the work on an efficient direction oriented motion search, which uses both a reduction in search points and partial distortion measure.

Although these fast motion search algorithms demonstrated the significant computational gain,

they do not consider specific properties of surveillance videos. In general, the surveillance videos are captured from fixed-angle or fixed-view cameras. For better compression, [Ma *et al.*, 2019] presented a pre-built dictionary-based coding scheme for traffic surveillance videos. These videos always have considerable static background regions, unless the camera is moving. Traditionally surveillance video contains two regions: background and foreground. For the foreground region, a motion search could use already discussed efficient block matching algorithms. However, the main question arises, if the traditional motion search approaches hold the similar importance for motion search at the background or static regions. To address this issue, we need to understand that the background region in the surveillance video is mostly static and does not exhibit any motion. Hence, there is no compulsory need to use motion search algorithms. This could significantly reduce the overall computational requirements in the motion search process for surveillance videos.

[Zhao *et al.*, 2014] reported a motion search algorithm on a similar understanding. The background-foreground-division-based search (BFDS) method exploited these surveillance-specific characteristics for fast motion estimation. The BFDS algorithm accelerated the motion search process for surveillance videos. The basic idea of BFDS is to classify a candidate block into two classes: foreground block and background block. Then different search strategies are employed for each category. That is, a zero motion vector biased search strategy is applied for background blocks to reduce the search complexity. On the other hand, a precise global search strategy is applied for the foreground block to achieve higher coding performance. This approach helped to significantly reduce overall search complexity since the proportion of static regions is generally high in surveillance videos. The BFDS outperformed TZS in search complexity reduction. The computational complexity is reduced in BFDS while maintaining a similar block matching accuracy. Although BFDS searched only zero-MV points for background areas and used variants of TZS in the foreground areas for fast and efficient motion estimation, it failed to exploit inherent directional motion characteristics. Besides, BFDS needs additional computations for generating and updating the background frame. Moreover, only the bi-level classification of each block is done: either background or foreground. However, multilevel classification is desired for better matching accuracy. Hence, motion search for surveillance videos poses a stiff challenge for real-time processing.

Surveillance videos are not only crucial for video analysis but also human action analysis and recognition tasks for security purposes. Human action or body part movement in the video can be captured using various body-joint recognition techniques. This body-joint information results in human skeleton information. Independent storage of the skeleton information can play a vital role in various applications. In this pursuit, lossless compression of skeleton information is surveyed in the next section.

2.3 FROM VIDEO AND MOTION CAPTURE CODING TO SKELETON SEQUENCE CODING

The human body joint information carries great importance in security applications. To this end, the body-joint information for a human is termed as a skeleton. The motion of objects or humans from the frame-to-frame result in the motion of skeletons. The moving nature of skeletons in videos results in a skeleton sequence. The skeleton sequence needs to be stored as semantic information for further analysis. However, the storage of the skeleton sequence is not straightforward and rarely investigated in the literature. In this section, firstly, the existing video coding schemes are explored to establish a correlation between videos and skeleton sequences, if any. Then, the closest possible literature to the skeleton sequence coding mechanism is discussed and analyzed.

Traditional image and video coding methods have extensively exploited the spatial, temporal, and spectral redundancies [Brunello *et al.*, 2003; Sun and Shi, 2008; Memon and Sayood, 1996; Fang *et al.*, 2019; Lu *et al.*, 2019; Zhou *et al.*, 2018; Liu *et al.*, 2014]. The image compression schemes employed various Intra predictors to exploit the inherent spatial redundancies [Weinberger *et al.*, 1996]. On the

other hand, the temporal predictive block matching scheme for motion estimation is widely used in video compression schemes. [Zafar *et al.*, 1991] described the nature of predictive coding approaches used in video compression. For effective video compression, both intra-prediction and inter-prediction are important. Both predictive coding modes are judiciously chosen to achieve optimal compression performance. [Wang *et al.*, 2019] reported the use of the intra-prediction scheme for spatial redundancy removal and [Memon and Sayood, 1996; Wang *et al.*, 2019; Jiang *et al.*, 2019] reported the use of the inter-prediction scheme for temporal redundancy removal. Once the spatial and temporal redundancies are removed, the difference between original information and predicted information termed as residual, is employed to entropy coding. The entropy coding scheme further helps to reduce the residual coding redundancy. In this pursuit, arithmetic coding techniques are widely adopted to handle coding redundancy in image and video compression methods. However, since skeleton data has different characteristics from video data, the redundancy models and coding methods cannot be directly applied to the compression of skeleton sequences.

Besides video coding methods, there are some limited methods designed to encode some specific semantic data types in images or videos. [Daribo *et al.*, 2012; Park, 2015] presented object boundary coding schemes, and [Gerogiannis *et al.*, 2015; Wang *et al.*, 2016] presented shape coding schemes, among others. These coding schemes are very specific to the task or application at the desk. However, since the targeted data types are different from the skeleton data in our approach, they cannot be directly applied for handling skeleton sequences. Skeleton sequence can be considered as a close associate of motion capture (MoCap) data. The MoCap data consists of motion information for human skeletons. [Kuo *et al.*, 2010] reported that MoCap data were frequently used for movement synthesis applications in the literature. The mocap data are obtained by recording the temporal trajectories of position sensors, where the temporal trajectory of each position is represented in marker positions and joint rotations. [Hou *et al.*, 2015b, 2014, 2015a] presented many effective compression schemes to accommodate a larger MoCap data collection for higher quality motion synthesis.

[Liu and McMillan, 2006; Karni and Gotsman, 2004] presented compression methods based on transforms like principal component analysis (PCA). [Cheng *et al.*, 2015; Beaudoin *et al.*, 2007] presented discrete wavelet transform (DWT) and discrete cosine transform-based MoCap compression schemes [Hou *et al.*, 2015b]. On the other hand, [Arikan, 2006; Gu *et al.*, 2009] presented various hybrid methods for motion-based prediction, post-processing, and [Chattopadhyay *et al.*, 2007] presented resource-constrained applications. Although the MoCap data compression schemes are well established for lossy coding, their performance would be degraded for lossless coding due to their design and framework. Moreover, MoCap data compression methods commonly suffer from substantial computational complexity, as automatic motion clustering is difficult and computationally demanding. For example, PCA based clustering demands a significant amount of similar MoCap data for training. On the contrary, this Thesis develops a novel adaptive lossless compression framework for skeleton sequences. The well-designed framework exploits specific spatial and temporal motion characteristics of the skeleton data to improve the efficiency of lossless compression. The most related work to ours is the method to compress region-of-interest (RoI) location sequences [Chen *et al.*, 2014]. However, since RoI location sequences are much simpler than the skeleton sequences, only part of the redundancies are considered [Chen *et al.*, 2014]. Thus, it will still have unsatisfactory performances when directly applied to compress skeleton data.

2.4 SUMMARY

In this chapter, a detailed review, analysis, and research gaps in the field of data compression, motion estimation, and skeleton sequence coding are explored. Firstly, the basics of data compression are discussed with a primary focus on lossy compression techniques. The video compression process is described in detail, explaining the role of transformation, quantization, and rate-distortion trade-off.

Further, the motion estimation process, which consumes most computations and time in the video compression process, is described in detail. Later, typical computational challenges in motion estimation processes are described along with various computationally efficient solutions for motion estimation. The computational complexity reduction can be achieved using primarily two ways: (1) reducing the number of search points in the motion search process and (2) reducing the total number of computations in each distortion measure computation. The study is further extended to motion search in surveillance videos. An effect of two-level block classification is studied, and it is expected that multi-level block classification could help in achieving a better trade-off between computational complexity and compression performance. The existing methods for surveillance video coding are discussed in detail. Lastly, a special case for human skeleton information present in surveillance videos is addressed. In this pursuit, existing mechanisms, are discussed for skeleton sequence coding.

In a nutshell, the literature survey is primarily described to address the three objectives of this Thesis. In the next chapters, the three solutions to the research objectives are described in detail. Firstly, the proposed mechanism to reduce the computational complexity for motion search algorithms without compromise in matching accuracy is presented in Chapter 3. Secondly, the motion search complexity reduction idea is investigated for surveillance video coding in Chapter 4. Lastly, the lossless coding mechanism is developed to store skeleton information generated from human skeletons present in the surveillance videos is presented in Chapter 5.

…