5
Skeleton Sequence Coding

As discussed in Chapter 4, surveillance videos are crucial in home-care, public safety and security, and traffic management applications. Surveillance videos mainly contain static regions, and the remaining active region typically consists of human action movements. The skeleton information plays an important role in various human action recognition, event detection, surveillance feature analysis, and health-care monitoring applications. Many emerging edge-computing applications tend to extract skeleton information at the sensor or transmitter side and directly transmit the extracted skeleton information together with the original video data to the receiver side. Thus, it becomes a new but non-trivial problem to encode skeleton data efficiently. The higher accuracy demands original and reliable skeleton information. With the significant increase in the skeleton data, the storage of skeletons in the original form imposes space constraints. It is desired to perform lossless compression of skeleton sequences to preserve the naturalness.

In this pursuit, we propose a novel lossless compression scheme for video skeleton sequences, which can efficiently compress skeleton data while maintaining exactly the same skeleton quality as the original ones at the decoder side. Intuitively, a skeleton sequence contains a considerable amount of spatial, temporal, and coding redundancies. For example, the relative position between a skeleton's body joints offers spatial redundancy. A person's skeletons between consecutive frames offer significant similarities and result in temporal redundancy. Moreover, the prediction residuals tend to repeat and hence possess the coding redundancy. In some cases where residuals do not repeat, they mostly tend to be closer to the previously occurred residual values. If we can adequately model and exploit these redundancies, the size of skeleton data can be reduced.

Although the traditional video coding methods also exploit redundancies in video data, the redundancy models and algorithms in video coding cannot be directly applied to skeleton sequences due to the large characteristic difference between the two data types. For example, in video coding, the coding blocks are ordered while the context of each block has an arbitrary manner. By contrast, the skeletons are located at random positions, but the body points in a skeleton have constrained structure. Moreover, skeleton data also include some additional problems such as the movement of each body joint in different directions, occlusion of some of the body joints, and the complex movements of skeleton sequences. Therefore, new approaches need to be developed to model and address the redundancy of skeleton data. In this chapter, we focus on exploiting and removing redundancies in two parts: 1) spatial and temporal prediction models and 2) entropy coding model. Our three major contributions in this chapter are:

- We propose a novel framework that can losslessly compress the skeleton sequences by exploiting the spatial, temporal, and coding redundancies. To the best of our knowledge, this is the first attempt to address the problem of compressing skeleton sequence data.
- Under the framework, we introduce a set of prediction modes for skeleton prediction to exploit the spatial and temporal redundancies inherent in the skeleton sequences. Furthermore, a multimodal method that dynamically integrates these modes is presented to achieve robust coding performance on skeleton sequences.

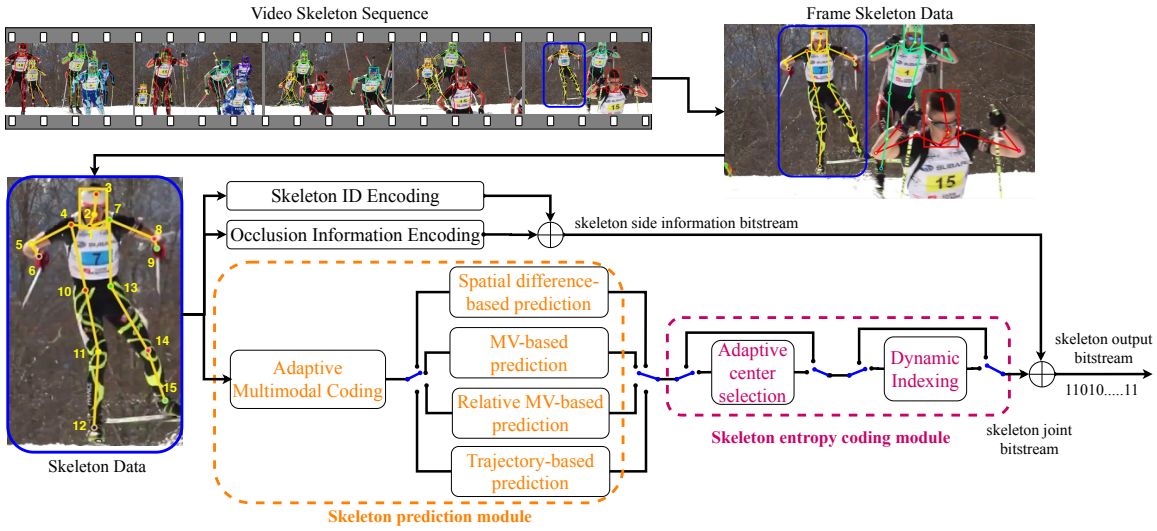

Figure 5.1 : Overview of proposed skeleton coding method.

- We also introduce a mode-based entropy coding mechanism to exploit residual coding redundancy further. To this end, we developed two different entropy-coding methods based on residual statistics. Lastly, we integrate both prediction modes and mode-based entropy coding schemes for lossless compression of skeleton sequences.

It is worth mentioning that our method can be applied to any skeleton sequences as long as we define a metric to determine the center skeleton point and the reference relation among skeleton points. For example, our method can be applied to face key point sequences, animal skeleton sequences, or vehicle key point sequences.

The rest of the chapter is organized as follows. Section 5.1 provides the framework of the proposed skeleton coding scheme and skeleton representation. The detailed description of proposed prediction modes and our entropy coding method is presented in Section 5.2 and Section 5.3, respectively. The skeleton side information coding mechanism is presented in Section 5.4. Section 5.5 presents experimental settings and performance results for the proposed study. Section 5.6 summarizes the chapter.

5.1 FRAMEWORK AND SKELETON REPRESENTATION

5.1.1 Framework of Proposed Skeleton Sequence Coding Scheme

The overview of the proposed skeleton coding method is shown in Figure 5.1, where each individual frame of the skeleton sequence is processed sequentially. For each skeleton in a frame, it mainly contains two types of information: 1) body joint information, which indicates the location of each joint, and 2) skeleton side information such as ID and occlusion flag information. In our work, we independently encode this information and concatenate them together to yield the final output bit-stream.

Since the skeleton body joint contains the bulk of the proportion in the final bit-stream, our work is mainly focused on the compression of this information. To this end, each body joint is first fed into the *skeleton prediction module* to reduce its spatial and temporal redundancy. Specifically, four prediction modes will be checked in this module, and the best mode will be used to perform

encoding. In order to avoid transmitting the bits of coding prediction modes, we further propose an adaptive multimodal coding scheme that uses the previously encoded skeletons in the previous frames to automatically decide the prediction mode of the body joint in the current frame. After the skeleton prediction module, the residual data of the skeleton will go through the *skeleton entropy coding* module to further reduce the coding redundancies through the *adaptive center selection* and *dynamic indexing* mechanisms. At the same time, the skeleton side information will also be encoded by the *skeleton ID encoding* and *occlusion information encoding* mechanisms, yielding the final compressed bit-stream.

5.1.2 Skeleton Representation

In this section, we describe the representation of the skeleton sequences. The study on the human skeleton sequence is generally associated with the study of movements of the fixed number of body joints present in the skeleton.

Human skeleton. We represent the 2D human skeleton as a set of N_J joints.

$$J = \{j_1, j_2, j_3, ..., j_{N_J}\} \tag{5.1}$$

where $j_i = (x_i, y_i)$ represents the horizontal and vertical coordinates of the body joint j_i. In our study, we consider total fifteen ordered body joints ($N_J = 15$); namely: neck, nose, head-top, left-shoulder, left-elbow, left-wrist, right-shoulder, right-elbow, right-wrist, left-hip, left-knee, left-ankle, right-hip, right-knee, and right-ankle. The illustration of typical skeleton body joints is shown in Figure 5.1. Note that our method is general. Besides this skeleton structure, our method can also be applied to the three-dimensional (3D) skeletons, bounding boxes, circular shapes, and arbitrary shapes like animal skeletons.

Occlusion flag. It should be noted that sometimes, the skeleton might go under occlusion resulting in reduced body joint information. The illustration of the occluded person is shown in Figure 5.1. It can be observed that in the last frame of video skeleton data, some of the body joints of the rightmost, two skeletons are occluded. The occlusion may happen due to the movement of different persons or objects in the video frame or when the person is moving outside or inside the camera frame. For better analysis, the occlusion information of the body joints in the skeleton is also stored. Traditionally, a one-bit occlusion flag is used to represent if the body joint is occluded or not. The typical occlusion information contains ($N_J = 15$) bits such that:

$$O = \{o_1 o_2 o_3 .....o_{N_J}\} \tag{5.2}$$

where $o_i = 1$ represents the body joint j_i is occluded, whereas $o_i = 0$ represents the body joint j_i is not occluded.

Skeleton ID. Now, the skeleton side information associated with s^{th} skeleton in the t^{th} video frame can be represented as:

$$S_s^t = \left\{SID_s^t, O_s^t, J_s^t\right\} \tag{5.3}$$

where SID_s^t is the skeleton ID of the s^{th} skeleton in the t^{th} frame. O_s^t and J_s^t represents the corresponding occlusion information and body joint coordinate information, respectively. The skeleton ID is assigned to each person based on its first appearance in the video. It should be noted that a unique skeleton ID

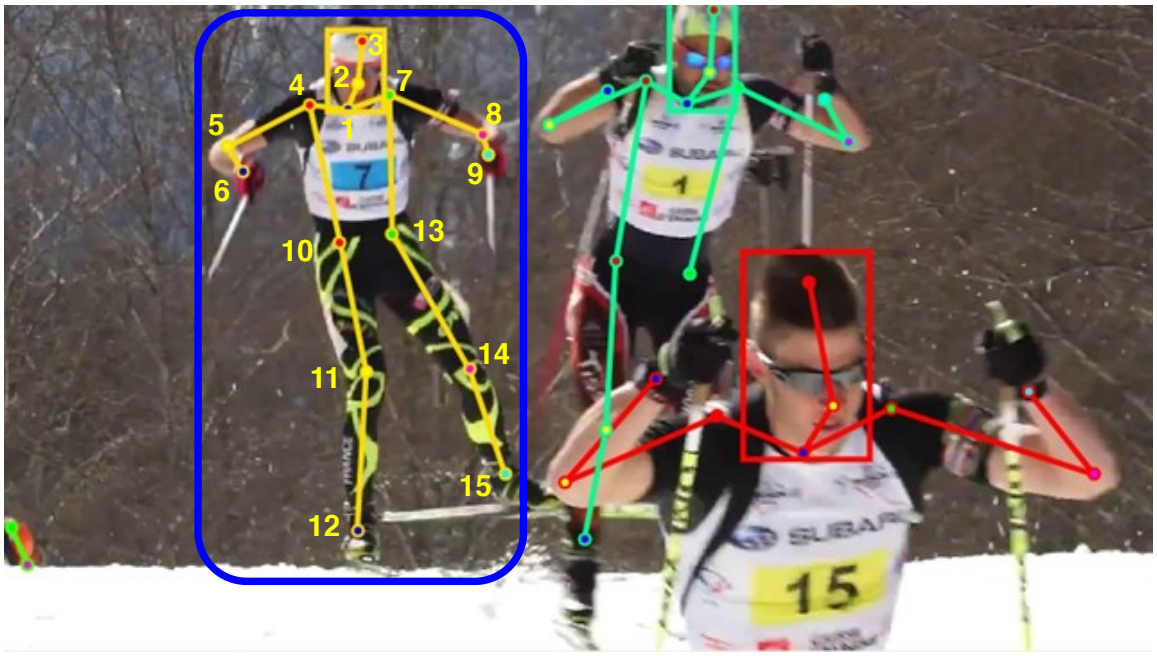

Figure 5.2 : Example of skeleton representation in a frame.

is assigned to each person over the entire length of the video. That means a person reappearing after few frames would not be assigned new ID, rather, the previous ID is retained for further processing.

Complete Skeleton Information. Hence, the complete skeleton information in a single video frame can be represented as:

$$F^t = \left\{ N_S^t, S_1^t, S_2^t, S_3^t, ..., S_{N_S^t}^t \right\} \tag{5.4}$$

where N_S^t represents the number of skeletons present in the t^{th} frame. An illustration shown in Figure 5.2 consists of three skeletons and the corresponding skeleton side information includes number of skeletons, skeleton ID, and occlusion flag bits information. It can be observed that all the body joints are visible or not-occluded for the first skeleton, whereas few body joints are occluded for remaining two skeletons. This information needs to be transmitted as occlusion flag bits. The overall skeleton data structure is also highlighted in Figure 5.2.

It should be noted that the framework of our skeleton compression approach is general. In practice, besides the representation in Eq. (5.4), our approach can also flexibly handle other data formats, such as 3D skeleton sequences [Tagliasacchi *et al.*, 2016; Wang *et al.*, 2013; Vemulapalli *et al.*, 2014] or non-skeleton 3D bounding boxes [Geiger *et al.*, 2013].

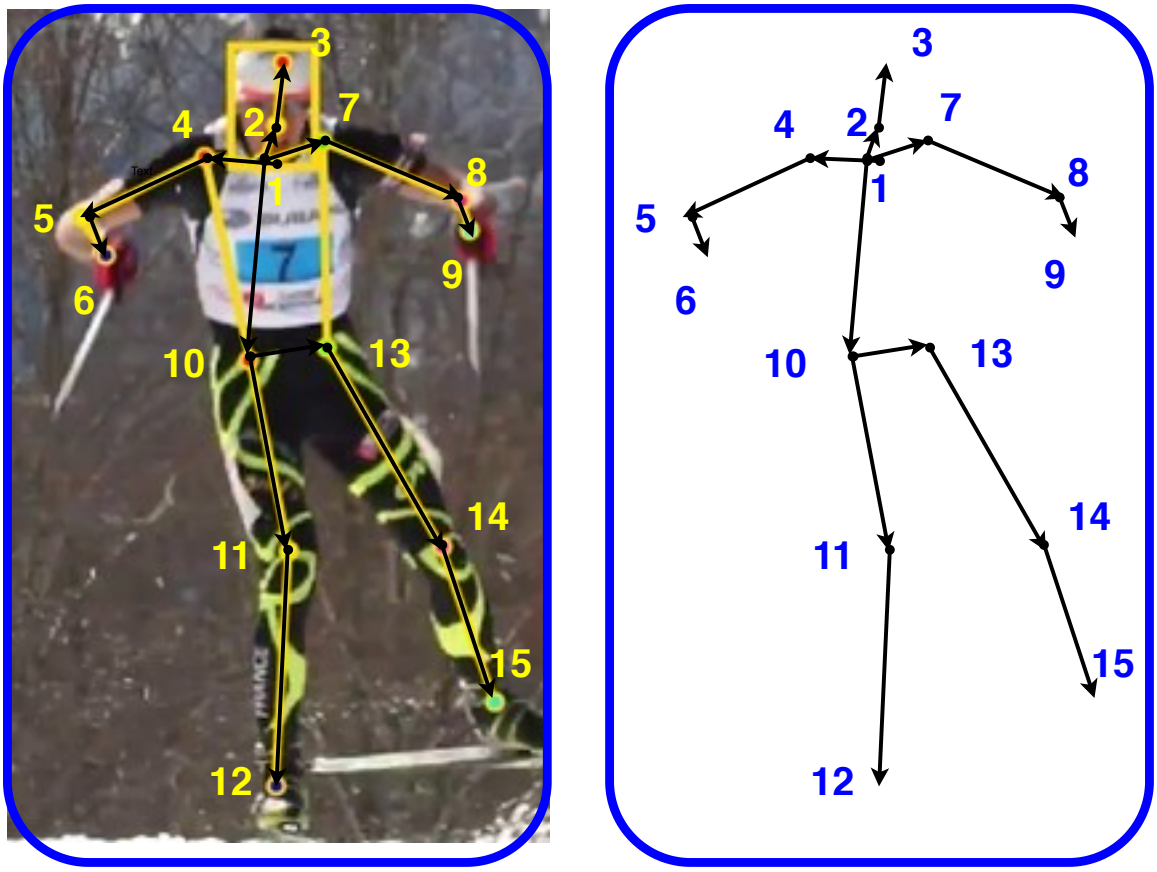

Figure 5.3 : Illustration of spatial differential skeleton coding method. (The direction arrow indicates the parent-child relationship)

5.2 SKELETON PREDICTION MODES

The redundancies present in the video skeleton sequence are exploited to introduce novel skeleton prediction modes are in this work. To exploit the spatial and temporal correlation between skeletons, we propose four prediction modes and then provide our adaptive mode switching scheme.

5.2.1 Spatial Differential Coding

To exploit the spatial correlations between skeleton body joints, we have developed a spatial differential coding scheme. To this end, the spatial difference between adjacent body joints is computed. The illustration of spatial differential coding is shown in Figure 5.3. The arrow direction in Figure 5.3 indicates parent-child relation between body joints. The spatial difference for each parent-child relation in the skeleton is computed as:

$$Diff_{j_i} = j_i - j_i^p \tag{5.5}$$

where j_i^p and j_i form a parent-child relation for spatial encoding. The spatial parent-child relation for body joint in our study is: {0-j_1, j_1-j_2, j_2-j_3, j_1-j_4, j_4-j_5, j_5-j_6, j_1-j_7, j_7-j_8, j_8-j_9, j_1-j_{10}, j_{10}-j_{11}, j_{11}-j_{12}, j_{10}-j_{13}, j_{13}-j_{14}, j_{14}-j_{15}}. It should be noted that body joint j_1 do not have any spatial parent, hence it is directly encoded. For better efficiency, the body joints are encoded in numeric order.

5.2.2 MV-based Prediction Mode

The assumption that skeletons of the same person in the consecutive frames are highly correlated. This has motivated us to exploit temporal redundancy by computing the motion vector (MV) between skeleton in the consecutive frames. The straightforward way to compute MV could be to consider average motion of all body joint and transmit the obtained MV. However, this process suffers from two problems: 1) the average MV could be non-integer value and may require more number of bits for encoding MV, and 2) the average MV could not correctly indicate person motion since the large motion in any of the body parts for a stationary human would also raise motion flag. To counter this, we select human neck (joint j_1) as skeleton center. This would not only greatly improve the MV accuracy, but also limit number of symbols to be encoded. With this understanding, the MV-based prediction for s^{th} skeleton in the t^{th} frame is carried out by first computing MV using:

$$MV(S_s^t) = j_1^{S_s^t} - j_1^{S_s^{t-1}} = \left(x_1^{S_s^t} - x_1^{S_s^{t-1}}, y_1^{S_s^t} - y_1^{S_s^{t-1}}\right) \tag{5.6}$$

where $(t-1)^{th}$ frame is considered as reference frame. An illustration for typical MV-based skeleton prediction mode is shown in Figure 5.4. Later, both the skeletons are aligned to each-other using the obtained MV as shown in Figure 5.4 (c). The MV-based skeleton prediction (SP) is obtained as:

$$SP_{MV-based}^{S_s^t} = j_i^{S_s^t} + MV(S_s^t), i = 2,3,...,N_J \tag{5.7}$$

Finally, the skeleton prediction error (SPE) is computed as:

$$SPE_{MV-based}^{S_s^t} = S_s^t - SP_{MV-based}^{S_s^t} \tag{5.8}$$

5.2.3 Relative MV-based Prediction Mode

Although the MV-based skeleton prediction provided significant coding gain over the direct coding method (i.e., coding each body joint with fixed length bits), we can still exploit the spatiotemporal relation between skeletons. The anatomy of the human skeleton provides us additional knowledge about skeleton motion. It is observed that the motion in the child body joint is proportional to its parent body joint. It means that there are high chances that the child will move in the same direction and sometimes by the same amount as the parent joint. To exploit these characteristics, we present a relative motion vector (RMV)-based skeleton prediction method. The skeleton prediction error obtained at parent body joint is used to fine-tune the skeleton prediction for the child body joint. The RMV-based skeleton prediction is obtained as:

$$SP_{RMV-based}^{S_s^t} = SP_{MV-based}^{S_s^t}(j_i) + SPE_{MV-based}^{S_s^t}(j_i^p) \tag{5.9}$$

where j_i^p and j_i form a spatial parent-child relation. An illustration of typical RMV-based skeleton prediction mode is shown in Figure 5.6. Finally, the skeleton prediction error is computed as:

$$SPE_{RMV-based}^{S_s^t} = S_s^t - SP_{RMV-based}^{S_s^t} \tag{5.10}$$

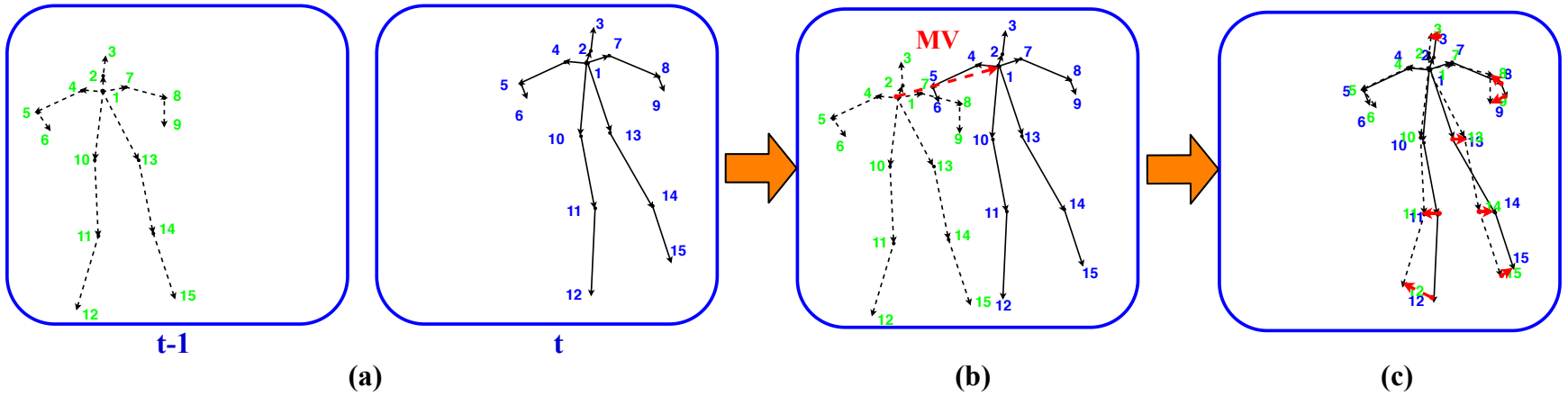

Figure 5.4 : Illustration of MV-based skeleton coding method: (a) skeletons in the $(t-1)^{th}$ reference frame and current (t^{th}) frame, (b) demonstration of MV computation, and (c) demonstration of skeleton alignment and MV-based skeleton prediction. (Best viewed in color)

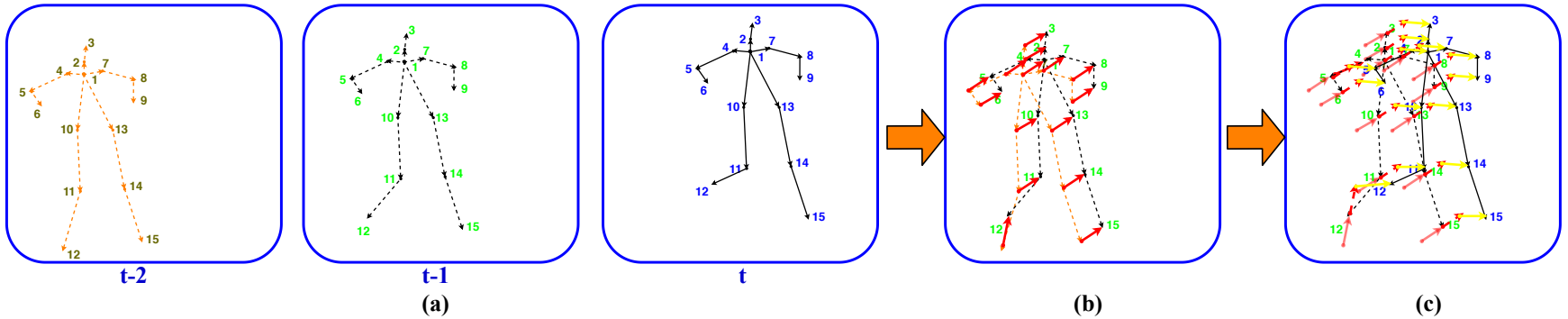

Figure 5.5 : Illustration of Trajectory-based skeleton coding method: (a) skeletons in the $(t-2)^{th}$, $(t-1)^{th}$ reference frames, and current (t^{th}) frame, (b) trajectory difference between two reference frames highlighted in red arrow, and (c) skeleton trajectory prediction highlighted in red arrow and skeleton prediction error highlighted by yellow arrow. (Best viewed in color)

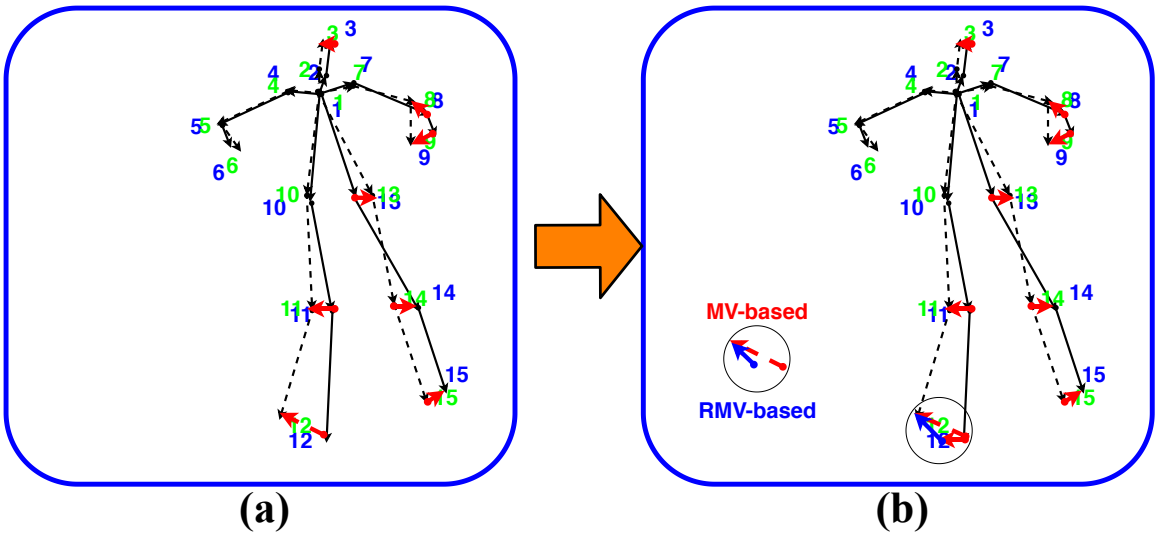

Figure 5.6 : Illustration of Relative MV (RMV)-based skeleton coding method: (a) MV-based skeleton prediction method, (b) RMV-based skeleton encoding for body joint 12 (right-ankle), the comparison of prediction MV-based and RMV-based prediction error is highlighted in circle. (Best viewed in color)

5.2.4 Trajectory-based Prediction Mode

The motion vector-based skeleton prediction methods are optimal when we only deal with translation motions. However, the human skeleton is not rigid, and the body part can move differently. This behavioral change demands a method to incorporate these motion changes for more accurate skeleton prediction. To this end, we use the trajectory prediction method based on previous skeleton frames. We consider the human skeletons with complex motions in our study, and hence we use only two previous frames for better trajectory prediction. It means that for S_s^t skeleton prediction in the t^{th} frame, the corresponding skeletons S_s^{t-1} and S_s^{t-2} in $(t-1)^{th}$ and $(t-2)^{th}$ frame are used such that:

$$SP_{T-based}^{S_s^t} = 2 \times S_s^{t-1} - S_s^{t-2} \tag{5.11}$$

Each body joint is predicted individually using the trajectory prediction method. An illustration of the typical trajectory-based skeleton prediction mode is shown in Figure 5.5. Note that in this work, we only use two previous frames to have a simple but effective trajectory prediction. In practice, we can also use more sophisticated trajectory prediction methods [Qi *et al.*, 2017; Quintero Mínguez *et al.*, 2019; Butepage *et al.*, 2017; Liu *et al.*, 2018] to obtain more accurate prediction results. For example, we can apply bi-direction trajectory prediction with adaptive importance to each frame, thus can work in synchronization with the state-of-the-art bi-directional video coding schemes [Park and Kim, 2019; Kong *et al.*, 2018; Zhao *et al.*, 2018].

Finally the skeleton prediction error is computed as:

$$SPE_{T-based}^{S_s^t} = S_s^t - SP_{T-based}^{S_s^t} \tag{5.12}$$

5.2.5 Adaptive Multimodal Coding Scheme

The proposed four prediction modes provide varying performance due to their inherent prediction structures. The spatial difference-based method performs well for the skeletons where no temporal correlation exists. On the other hand, the remaining three modes could exploit temporal correlations. The MV-based method could perform well for rigid and small motion cases, whereas the RMV-based method performs better at child body joint movements. The trajectory prediction could be useful in cases where skeleton size is large, and different body parts are performing the different or complex motion. Thus, the prediction model should be chosen to provide a lower prediction error for better compression. The use of the most suitable prediction mode is desired for optimizing skeleton coding efficiency. The simplest switching way could be to use the best mode at any given body joint for best skeleton body joint coding efficiency. However, it should be noted that we also need to encode the selected mode information as an overhead. This process involves an additional two-bit overhead for encoding, and most of the time, it will hamper the coding efficiency.

To avoid this overhead, we propose an adaptive multimodal skeleton coding method to dynamically switch the prediction modes based on the already encoded causal skeleton data. To this end, we use the prediction mode information of the already encoded causal skeleton data in the current and reference frames to estimate the best prediction mode of the current body joint. It should be explicitly noted that our proposed mode estimation is highly accurate, although the estimated prediction mode might be different from the actual prediction mode. The mode estimation for the current body joint is computed based on prediction mode information of its spatial and temporal parents. In this way, we can skip the bits overhead bits for prediction mode, thus enables each body joint to choose its own prediction mode flexibly.

More specifically, we use the bit-information for already encoded spatial parent (j_i^p) and collocated temporal body joints of previous frames to estimate the bit-requirement for current body joint in the selected prediction mode.

$$\widehat{b}^t_{mode}(j_i) = w_{j_i^p} \times b^t_{mode}(j_i^p) + \sum_{f=1}^{N_f} w_f \times b^{t-f}_{mode}(j_i) \tag{5.13}$$

where $w_{j_i^p}$ is weight corresponding to the j_i's parent body joint j_i^p (which has already been coded before j_i), w_f is the weight corresponding to the $(t-f)^{th}$ reference frame, and N_f is the number of temporal frames used for bit-requirement estimation. The weights are chosen based on the spatial or temporal distance between current body joint and reference body joint. As expected, it was observed that the effect of reference body joint on current body joint reduces, when their distance increases. Thus, we used proportional weight selection criteria to parameterize this idea in our experiments. For example, when the weight for spatial parent is set to 1, the weights for collocated temporal body joints of previous frames are chosen as $1/f$. Note that the weights are further normalized to unit sum. However, different combinations of optimal weights could be found with additional computational complexity. The bit-requirement for body joint j_i is estimated using each of the prediction modes, resulting into three different estimates. The prediction mode with lowest estimated bits is chosen as estimated prediction mode for current body joint j_i. The mode selection is done as:

$$\min\left(\widehat{b}^t_{MV-based}(j_i), \widehat{b}^t_{RMV-based}(j_i), \widehat{b}^t_{T-based}(j_i)\right) \tag{5.14}$$

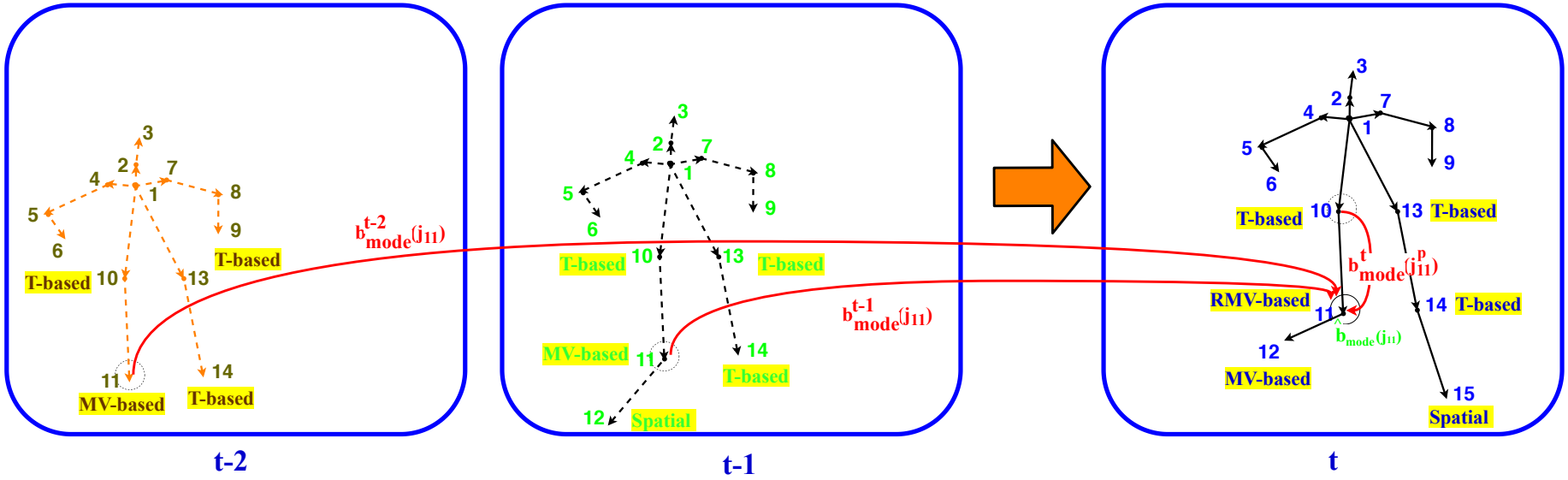

Figure 5.7 : Adaptive prediction mode selection based on the causal skeleton body joint mode information. (Best viewed in color)

The selected mode is used to predict the location of the current body joint. The typical illustration of mode selection is shown in Figure 5.7. It should be noted that the mode selection uses only the previous two reference frames for illustration purposes only, whereas, in reality, the mode selection can use any number of reference frames.

Overall, the adaptive multimodal coding method follows the below rules:

1. The spatial differential coding method is used for the newly appeared skeleton or skeleton body joint in the current frame. Besides, the first frame in the sequence also uses a spatial differential coding method. For example, as shown in Figure 5.7, the left-ankle body joint j_15 in the skeleton uses spatial mode since the left-ankle body joint has appeared for the first time in the current frame.

2. One of the MV-based and RMV-based skeleton coding method is chosen if skeleton in the current (t^{th}) frame exists in the $(t-1)^{th}$ frame but does not exists in the $(t-2)^{th}$ frame. For example, as shown in Figure 5.7, the right-ankle body joint j_12 in the skeleton uses one of the MV-based and RMV-based modes since the right-ankle body joint exists only in the current and adjacent previous frame.

3. One of the MV-based, RMV-based, and trajectory-based skeleton coding method is chosen if skeleton in the current (t^{th}) frame exists in the both $(t-1)^{th}$ frame and $(t-2)^{th}$ frame. For example, as shown in Figure 5.7, all the body joints except the left-ankle and right-ankle can use any one of the existing modes since all these joints exist in the current and both previous frames.

5.3 ENTROPY CODING

The adaptive prediction mode mechanism for skeleton prediction has considerably exploited the spatiotemporal correlations existing in the skeleton sequences. However, the skeleton prediction residuals are observed to follow some peculiar statistics. We can use these special statistics that still present in the prediction residuals for better entropy coding. In traditional entropy coding methods, the prediction residuals are directly encoded using variable-length encoding techniques. However, it is empirically observed that the prediction residuals obtained after skeleton prediction still contain notable redundancies. Consequently, our entropy coding method exploited these redundancies to improve further compression efficiency, where the resulting residuals are encoded using specifically designed variable-length encoding techniques. Hence, our approach is expected to provide better performance than the traditional entropy coding methods.

To this end, we propose two methods: 1) adaptive center selection, and 2) dynamic indexing.

5.3.1 Adaptive Center Selection (ACS)

We first observe that the prediction residuals in the consecutive frames tend to be similar. To exploit this behavior, we consider the prediction residual obtained in the reference frame as a reference error, as shown in Figure 5.8. Then the difference vector between current prediction residual is computed such that:

$$(d_x, d_y) = (c_x, c_y) - (r_x, r_y) \tag{5.15}$$

where (c_x, c_y), (r_x, r_y), and (d_x, d_y) represents the current prediction residual, reference prediction residual, and the difference vector, respectively. In cases where collocated prediction errors tend to be similar, the difference vector will tend to be close to zero. Hence, this scheme has the ability to encode larger error values into a very small number of bits due to reference center selection.

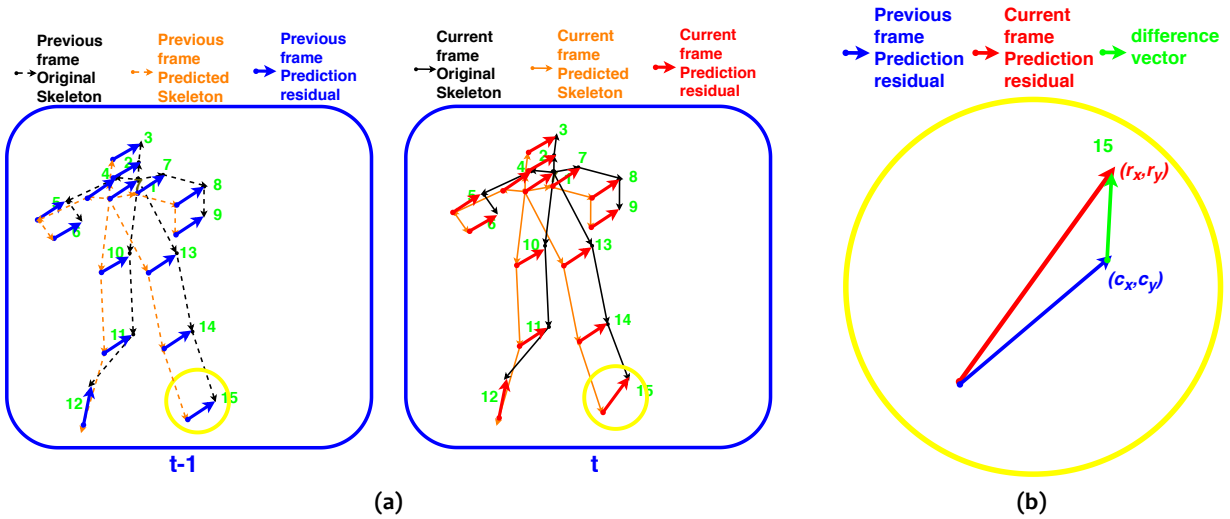

Figure 5.8 : (a) Illustration of prediction residual computation for the current frame and the reference frame, (b) Illustration of difference vector computation. (Best viewed in color)

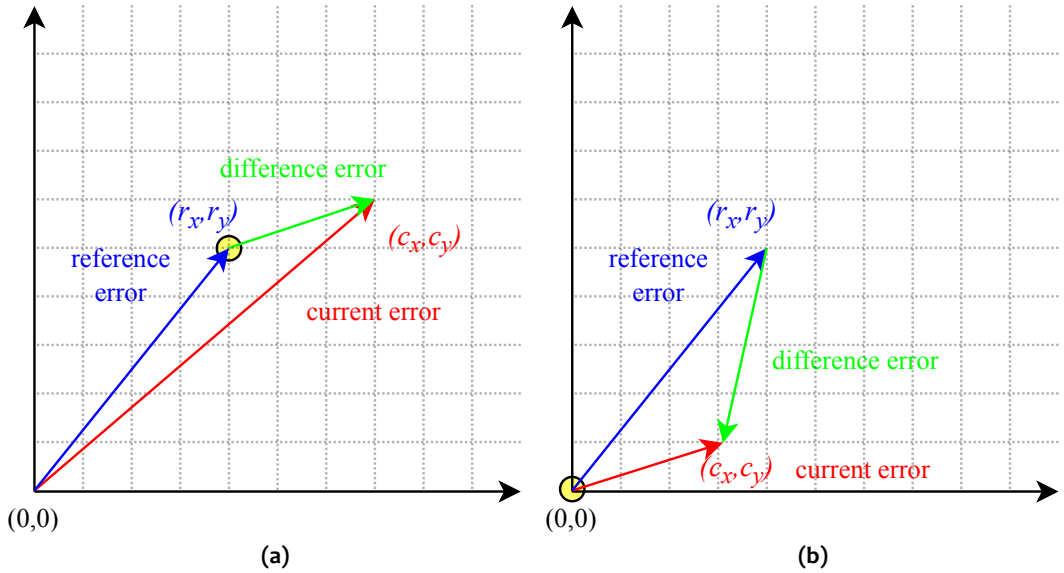

Figure 5.9 : Illustration of adaptive center selection based on the length of current error vector and difference error vector: (a) (r_x, r_y) is considered as adaptive center, and (b) $(0,0)$ is considered as adaptive center. The adaptive center is shown in yellow circle.

However, this mechanism might be counterproductive in some cases where our assumption on the similarity between collocated residuals fails. In Figure 5.9 (b) the difference vector has a larger magnitude than the current residual vector. This eventually will result in incorrect center selection. To address this problem, we propose to select the encoding center among 1) traditional center $(0,0)$, and 2) reference frame prediction residual (r_x, r_y). The selection is based on a number of bits required for encoding (d_x, d_y) and (r_x, r_y). The one with a lower magnitude is considered for encoding. However, this mechanism comes at the cost of one-bit overhead. To reduce the bits of coding the overhead, we employ block-based encoding where all body joints for the same skeleton-part is represented by one overhead bit. This mechanism can properly balance overhead and coding performance.

5.3.2 Dynamic Indexing (DI)

We also observe that the prediction residuals in the consecutive frames of the typical skeleton sequences tend to be similar. This means the prediction residuals exhibit considerable repetitiveness. To exploit this behavior, we propose to use a frequency lookup table of causal prediction residuals. The terms residual and symbol are interchangeably used in the further discussion. Like variable-length encoding, the more frequently occurring symbols would be assigned a lower number of bits as compared to the rarely occurring symbols. This idea can properly reduce the bit requirements.

To this end, we create a new frequency lookup table for each skeleton. Initially, the frequency table is empty, and it is updated with obtained prediction residuals. The frequency lookup table can be seen as a list of expected residuals with a different number of previous occurrences. The illustration of lookup table creation and dynamic indexing is shown in Figure 5.10. The current prediction residuals are matched to the symbols present in the lookup table. Then, for encoding current body joint in a particular skeleton, we use the frequency table corresponding to the particular skeleton and check whether the current prediction residual is present in the frequency lookup table. If all the prediction residuals of a particular skeleton exist in the frequency lookup table, then the indexes corresponding to the respective residuals are encoded in the bit-stream. Otherwise, the residuals are directly encoded by a variable length coding method [Chen *et al.*, 2006; Sugiura *et al.*, 2018]. Similar to the Adaptive Center Selection scheme, a flag is transmitted for each skeleton to indicate whether the Dynamic Indexing scheme is applied or not.

The switching mechanism for this method is shown in Figure 5.11. There are four possible switching combinations of the proposed entropy coding scheme. First, both the ACS and DI can be skipped, and the prediction residual could be directly encoded using a variable length coding method. Second, only ACS is employed, third, only DI is employed, and fourth, both ACS and DI schemes are employed in tandem to exploit the coding redundancies. We use exponential Golomb codes (EGC) [Sugiura *et al.*, 2018] to encode the final residual information and concatenate the obtained EGC code in the output bit-stream.

5.4 SKELETON SIDE INFORMATION CODING

In addition to skeleton body joint data, the skeleton sequence in a frame contains three types of side information: 1) number of skeletons in the frame (N_S^t), 2) skeleton ID information (SID_s^t), and 3) occlusion information (O_s^t) for each skeleton in the t^{th} frame. This side information may affect overall coding performance if not properly encoded. We address all three skeleton side information independently for better performance.

Number of skeletons in the frame: The number of skeletons in the frame usually remains constant. N_S^t changes only when some of the skeletons appear or disappear from the frame. Since this value changes slowly and smoothly, we encode the difference between the values corresponding to the consecutive frames. The difference is mostly zero and requires minimal bits for representation.

Skeleton ID: The skeleton ID's are assigned in arithmetic sequence. This is helpful in computing difference between two skeleton IDs. For better coding performance we always encode IDs in ascending order. The skeleton ID difference is computed as:

$$diff_{SID_s^t} = SID_s^t - (SID_{s-1}^t + 1) \tag{5.16}$$

For ordered continuous skeleton ID sequence the difference is always zero, and hence need very limited bits for skeleton ID encoding.

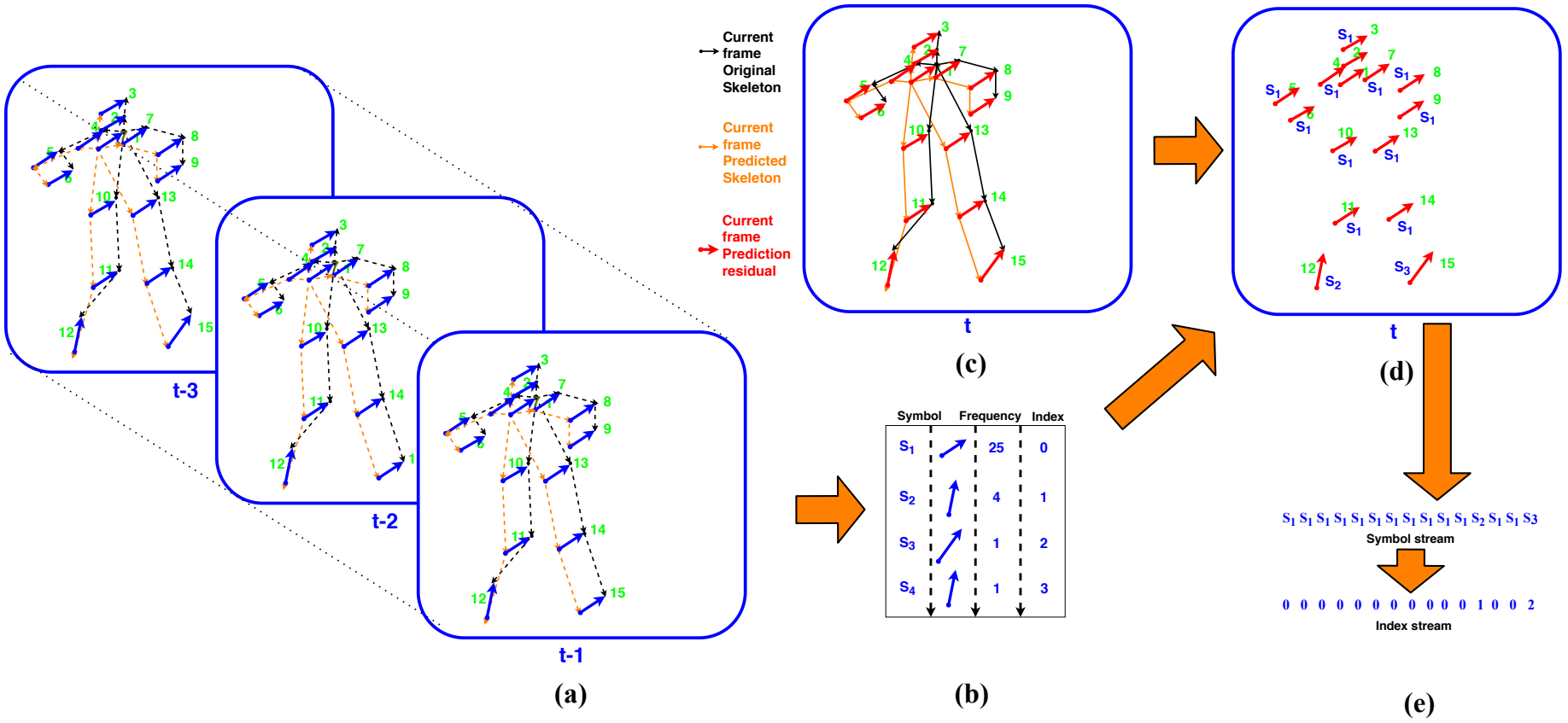

Figure 5.10 : (a) Illustration of prediction residual in reference frames, (b) symbol frequency lookup table creation based on prediction residual in reference frames, (c) illustration of prediction residual in current frame, (d) mapping current prediction residual to the lookup table, and (d) generating index stream for current frame prediction residual symbols. A more frequent residual value is assigned with a smaller index number. (Best viewed in color)

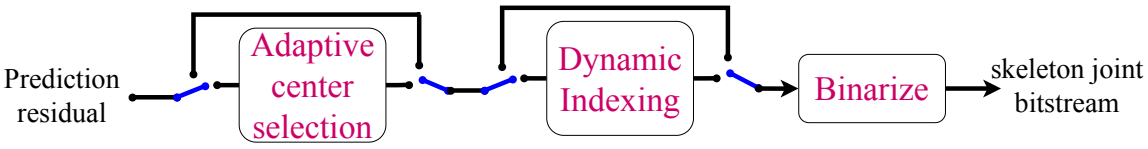

Figure 5.11 : Entropy mode switching scheme.

Table 5.1 : Test skeleton sequences used in the study.

Sequence Name	Frame resolution	Total frames	Skeletons per frame	Occlusion (Yes/No)	Description	Dataset
Indoor	1920 × 1080	31	11 - 11	Y	Indoor dance practice	Posetrack [Andriluka *et al.*, 2018]
Ice stick	1280 × 720	31	11 - 12	Y	Ice stick ball	
Karate	1280 × 720	31	5 - 6	Y	Karate training	
Workers	1280 × 720	31	7 - 7	Y	Workers reparing road	
Street	1920 × 1080	31	6 - 8	Y	Street view from camera	
Action	1280 × 720	31	5 - 6	Y	Medium motion action sequence	
Basketball	1280 × 720	31	10 - 10	Y	Basketball match	
Hurdle-race	1920 × 1080	31	7 - 9	Y	Persons running hurdle-race	
Football	1440 × 1080	31	5 - 7	Y	Boys playing football	
Football2	1440 × 1080	31	8 - 10	Y	Persons playing football (side-view)	
Rugby	1280 × 720	31	3 - 6	Y	Persons playing rugby	
Musical	1280 × 720	31	8 - 9	Y	Musical dance performance	
Womens	1280 × 720	31	4 - 4	Y	Womens dancing	
Ice skating	1280 × 720	31	6 - 9	Y	People enjoying ice skating	
Volleyball	640 × 480	31	7 - 10	Y	Volleyball match	
Hotel	1920 × 1080	31	17 - 21	Y	People eating at hotel	Surveillance
Childrens	1280 × 720	31	12 - 13	Y	Childrens running (side-view)	
Beach	1280 × 720	31	5 - 6	Y	Beach running	
Stadium	1280 × 720	31	15 - 18	Y	People entering stadium	
Police	654 × 480	31	4 - 5	Y	Police captures criminal	
Baby	1920 × 1080	31	12 - 13	Y	Parents enjoying babysitting	
Workshop	1280 × 720	31	5 - 6	Y	Persons working in workshop	
Restaurant	1920 × 1080	31	13 - 18	Y	People eating in restaurant	
Piano	1280 × 720	31	9 - 12	Y	Childrens playing piano	
Footpath	1008 × 672	50	8 - 10	N	People walking on footpath (top-view)	
Road	800 × 608	46	18 - 22	N	People walking on the road	
Luggage	800 × 608	40	18 - 19	N	People walking with luggage	
Square	1280 × 720	40	28 - 33	N	People walking on the square	
Crossing	1280 × 720	40	23 - 29	N	People crossing the road	
Night	1920 × 1080	45	34 - 35	N	People walking at night	

Occlusion flag: The occlusion flag information for each skeleton indicates the visibility of particular body joints in the skeleton. The occlusion can occur in two cases: 1) when a body joint in the skeleton goes behind another skeleton or object in the field of view, and 2) when skeletons are located at frame boundary and move outside or inside the field of view of the camera. These cases do occur on complex motion video skeleton sequence and sparingly occur in slow motion surveillance sequences. To this end, we perform temporal block-coding on the occlusion information belonging to the skeleton of the same person in the consecutive frames. We represent a one-bit flag to indicate if the occlusion information in the consecutive frames is exactly the same or not. This simple block-coding method resulted in a reduction of fourteen (93.33%) occlusion bits when occlusion information matches temporally. On the other hand, we have one (6.67%) extra bits, when occlusion information does not match temporally. On average, the block coding has provided significant improvement in coding performance.

5.5 EXPERIMENTS AND ANALYSIS

In this section, we first provide a description of datasets, experimental parameter settings, and various evaluation metrics used for comparative analysis. Then, we demonstrate the effectiveness of the proposed approach against traditional coding approaches and finally conduct an ablation study on different components of the proposed method.

5.5.1 Datasets

We evaluate the proposed method on two datasets: 1) Posetrack dataset [Andriluka *et al.*, 2018], and 2) Surveillance dataset created by ourselves. The details of the test skeleton sequences used in our study are given in Table 5.1.

Posetrack dataset: The Posetrack dataset presented [Andriluka *et al.*, 2018] that contains large number of test sequences. To demonstrate the effectiveness of our method, we choose 15 challenging test skeleton sequences from the Posetrack dataset. The Posetrack sequences contain fast and complex motion cases with different occlusion scenarios. The skeleton size tends to vary from large to very large in the given frame. Some example skeleton sequences are shown in Figure 5.12 (a) and 5.12 (b).

Surveillance dataset: The surveillance sequences mostly contain a large number of skeletons in the video sequence. To demonstrate the effectiveness of our method on these sequences, we use 15 surveillance test skeleton sequences. The surveillance sequences are collected and labeled by ourselves. In the surveillance sequences, the skeleton sizes tend to comparatively small compared to the frame size. Some example skeleton sequences are shown in Figure 5.12 (c) and 5.12 (d).

5.5.2 Evaluation Metrics

Since our method performs skeleton data compression in a lossless manner, we can recover exactly the same skeleton sequence as the original input. Therefore, we only need to evaluate the bit rate performances while no need to evaluate the quality of the decoded skeleton sequences. Here, we illustrate two bit-rate metrics: 1) bits required per skeleton joint, and 2) bit-savings compared with a direct coding method. The average bits per skeleton joint is the ratio between total number of bits required to encode all skeleton information and the total number of skeleton joints present in a frame. It can be computed as:

$$\overline{b^{t}_{joint}} = b^{t}_{total} \Big/ \sum_{s=1}^{N^{t}_{S}} \sum_{i=1}^{N_J} (1 - O^{t}_{s,i}) \tag{5.17}$$

where $O^{t}_{s,i}$ is the occlusion flag corresponding to the i^{th} body joint of the s^{th} skeleton in the t^{th} frame.

The bits-saving metric is computed by evaluating the bit reductions compared with a traditional direct coding method (i.e., using fixed bit length for each body joint):

$$bits\ saving = \frac{\overline{b^{Proposed}_{joint}} - \overline{b^{Direct}_{joint}}}{\overline{b^{Direct}_{joint}}} \times 100\% \tag{5.18}$$

5.5.3 Experimental Settings

In this work, the MV-based and RMV-based prediction modes use only one previous frame as the reference, whereas T-based prediction mode uses the previous two frames as the reference. In the

(a)

(b)

(c)

(d)

Figure 5.12 : Example skeleton sequences: (a) Posetrack sequence- Ice skating, (b) Posetrack sequence- Volleyball, (c) Surveillance sequence- Night, and (d) Surveillance sequence- Footpath.

multimodal prediction mode switching scheme, the previous N_f frames are used for mode selection, as illustrated in Eq. (5.13). We have empirically found that $N_f = 5$ provided better results. In a dynamic

indexing entropy coding scheme, the symbols in the current frame tend to be similar to the symbols of the nearby frames. Moreover, the dictionary size could increase dramatically with the increase in the number of previous frames. Hence, we have empirically found that using only 5 previous frames for dictionary generation would suffice.

5.5.4 Methods used for Comparisons

Since skeleton data compression is newly studied in this chapter, a few existing methods can directly be applied to perform skeleton compression. Therefore, we choose to select two most relevant methods to compare with our approach: the direct coding method and the temporal differential coding method [Chen *et al.*, 2014].

Direct Coding: The direct coding method aims to use a fixed bit length to code all skeleton information [Chen *et al.*, 2014]. In this work, we employ the direct coding method on each component of skeleton information mentioned in Eq. (5.4). Firstly, a fixed number of bits are assigned to encode N_S^t. The number of bits is directly dependent on the maximum number of skeletons in any given frame.

$$b_{N_S^t}^t = \left\lceil \log_2(\max_{\forall t} N_S^t) \right\rceil \tag{5.19}$$

Secondly, the skeleton ID (SID_s^t) is encoded into $b_{SID_s^t}^t$ bits using (5.19). Thirdly, $b_{O_s^t}^t = 15$ bits are used to encode the occlusion information corresponding to the s^{th} skeleton in the t^{th} frame. Lastly, the body joint information is encoded based on the frame resolution of the video.

$$b_{j_i}^t = \lceil \log_2(H) \rceil + \lceil \log_2(W) \rceil \tag{5.20}$$

where H is height and W is width of the video frame. Hence, total number of bits required to encode complete skeleton information corresponding to the t^{th} frame is:

$$b_{total}^t = b_{N_S^t}^t + b_{N_S^t}^t \times \left(b_{SID_s^t}^t + b_{O_s^t}^t + N_J \times b_{j_i}^t \right) \tag{5.21}$$

Temporal Differential Coding: The temporal differential coding method aims to use the direct temporal difference between corresponding skeleton body joints for data compression [Chen *et al.*, 2014]. To this end, the location movement for each body joint is computed against the collocated body joints in the reference frame. It should be noted that the body joint appearing for the first time does not have any temporal reference, and hence it is encoded using the same way as our approach for a fair comparison.

5.5.5 Results for the Proposed Prediction Modes

In this section, we compare the performance of our prediction modes (i.e., the four prediction modes of 'Spatial differential coding', MV-based coding', 'RMV-based coding', 'T-based coding' and the combined 'Multimodal coding' mode) against the direct coding and temporal differential coding methods [Chen *et al.*, 2014]. In order to exclude the effect of entropy coding and skeleton side information, we simply use exponential Golomb codes to encode all residuals and also exclude the bits for coding side information. The performances of average bits per skeleton joint and bit-savings against the direct coding method are shown in Table 5.2.

Table 5.2 : Bits per skeleton points and Bit-savings for different prediction modes (EGC is used to encode all residuals and the bits for coding side information are excluded).

	Sequence Name	Direct coding	Temporal differential coding	Spatial differential coding	MV-based coding	RMV-based coding	T-based coding	Multimodal coding
Posetrack [Andriluka *et al.*, 2018]	Indoor	22.00	7.72 (-64.91%)	24.28 (10.36%)	8.56 (-61.10%)	8.00 (-63.65%)	**6.62 (-69.92%)**	6.73 (-69.39%)
	Ice stick	21.00	9.15 (-56.44%)	18.75 (-10.69%)	6.88 (-67.25%)	6.62 (-68.47%)	5.76 (-72.56%)	**5.74 (-72.66%)**
	Karate	21.00	7.41 (-64.73%)	20.68 (-1.55%)	6.75 (-67.85%)	6.73 (-67.94%)	**5.86 (-72.11%)**	5.87 (-72.05%)
	Workers	21.00	6.21 (-70.42%)	23.13 (10.14%)	6.12 (-70.88%)	5.34 (-74.57%)	**4.29 (-79.55%)**	4.34 (-79.33%)
	Street	22.00	11.24 (-48.93%)	22.74 (3.36%)	8.70 (-60.43%)	8.19 (-62.78%)	**6.13 (-72.12%)**	6.36 (-71.07%)
	Action	21.00	13.15 (-37.39%)	20.96 (-0.21%)	7.30 (-65.22%)	6.81 (-67.59%)	6.30 (-69.98%)	**6.15 (-70.73%)**
	Basketball	21.00	11.27 (-46.31%)	18.98 (-9.63%)	8.99 (-57.20%)	8.61 (-59.01%)	7.63 (-63.67%)	**7.51 (-64.23%)**
	Hurdle-race	22.00	13.46 (-38.84%)	20.71 (-5.85%)	12.48 (-43.26%)	11.99 (-45.48%)	11.44 (-48.01%)	**11.26 (-48.81%)**
	Football	22.00	14.41 (-34.49%)	21.25 (-3.40%)	10.97 (-50.16%)	10.70 (-51.37%)	11.59 (-47.32%)	**10.06 (-54.27%)**
	Football2	22.00	11.69 (-46.87%)	21.16 (-3.80%)	10.75 (-51.14%)	10.13 (-53.97%)	10.37 (-52.85%)	**9.65 (-56.15%)**
	Rugby	21.00	13.42 (-36.11%)	20.47 (-2.52%)	10.83 (-48.44%)	10.43 (-50.34%)	10.91 (-48.03%)	**10.21 (-51.39%)**
	Musical	21.00	12.13 (-42.26%)	22.81 (8.61%)	9.74 (-53.60%)	9.76 (-53.53%)	9.24 (-56.00%)	**8.88 (-57.70%)**
	Womens	21.00	7.26 (-65.44%)	21.02 (0.10%)	7.08 (-66.28%)	6.79 (-67.68%)	5.82 (-72.29%)	**5.66 (-73.05%)**
	Ice scating	21.00	14.52 (-30.88%)	20.09 (-4.35%)	10.48 (-50.10%)	10.02 (-52.28%)	9.56 (-54.47%)	**8.93 (-57.49%)**
	Volleyball	19.00	12.47 (-34.38%)	18.38 (-3.26%)	9.48 (-50.12%)	9.10 (-52.08%)	9.98 (-47.49%)	**8.87 (-53.32%)**
	Average	21.20	11.03 (-47.96%)	21.03 (-0.82%)	9.01 (-57.51%)	8.61 (-59.37%)	8.10 (-61.79%)	**7.75 (-63.45%)**
Surveillance	Hotel	22.00	12.44 (-43.45%)	24.35 (10.67%)	9.41 (-57.21%)	9.27 (-57.88%)	8.44 (-61.64%)	**8.03 (-63.52%)**
	Childrens	21.00	8.59 (-59.11%)	20.02 (-4.68%)	7.87 (-62.51%)	7.65 (-63.56%)	**6.69 (-68.15%)**	6.78 (-67.72%)
	Beach	21.00	10.63 (-49.40%)	21.80 (3.81%)	8.71 (-58.50%)	8.54 (-59.32%)	7.60 (-63.81%)	**7.50 (-64.30%)**
	Stadium	21.00	8.42 (-59.90%)	24.02 (14.38%)	8.44 (-59.80%)	8.21 (-60.89%)	7.30 (-65.25%)	**7.20 (-65.71%)**
	Police	19.00	11.31 (-40.47%)	23.04 (21.28%)	9.34 (-50.86%)	9.06 (-52.29%)	**7.84 (-58.74%)**	7.87 (-58.58%)
	Baby	22.00	11.60 (-47.28%)	24.99 (13.60%)	9.07 (-58.76%)	9.22 (-58.10%)	**7.82 (-64.44%)**	7.85 (-64.33%)
	Workshop	21.00	11.49 (-45.29%)	24.57 (17.00%)	10.15 (-51.68%)	9.76 (-53.51%)	**7.48 (-64.40%)**	7.62 (-63.70%)
	Restaurant	22.00	15.53 (-29.39%)	27.07 (23.07%)	10.09 (-54.12%)	10.12 (-54.02%)	**7.66 (-65.20%)**	7.90 (-64.11%)
	Piano	21.00	9.17 (-56.33%)	24.50 (16.68%)	9.03 (-57.00%)	8.54 (-59.35%)	**6.71 (-68.04%)**	6.72 (-67.98%)
	Footpath	20.00	11.95 (-40.26%)	17.16 (-14.18%)	6.64 (-66.82%)	6.43 (-67.84%)	11.41 (-42.95%)	**6.41 (-67.96%)**
	Road	20.00	8.47 (-57.65%)	15.09 (-24.57%)	4.87 (-75.65%)	**4.76 (-76.22%)**	8.95 (-55.25%)	4.78 (-76.10%)
	Luggage	20.00	8.93 (-55.36%)	15.62 (-21.90%)	4.70 (-76.52%)	**4.68 (-76.59%)**	9.58 (-52.11%)	4.69 (-76.53%)
	Square	21.00	7.89 (-62.42%)	15.40 (-26.68%)	3.61 (-82.79%)	**3.57 (-83.00%)**	8.86 (-57.80%)	3.62 (-82.78%)
	Crossing	21.00	8.09 (-61.47%)	15.71 (-25.19%)	3.37 (-83.96%)	**3.36 (-84.00%)**	8.95 (-57.36%)	3.39 (-83.88%)
	Night	22.00	4.04 (-81.63%)	17.42 (-20.81%)	**1.77 (-91.95%)**	1.78 (-91.92%)	4.49 (-79.61%)	1.91 (-91.31%)
	Average	20.93	9.90 (-52.69%)	20.72 (-1.03%)	7.14 (-65.90%)	7.00 (-66.58%)	7.98 (-61.86%)	**6.15 (-70.62%)**

From Table 5.2 we can have the following observations:

1. The spatial differential coding method has overall similar performances to the direct coding method. If taking a more detailed look at its performance, we can see that it performs obviously better than the direct coding method for some sequences (e.g., Ice stick and Square), but also performs worse for some other sequences (e.g., Indoor and Restaurant). This suggests that spatial differential coding is mainly helpful in cases where skeleton size is small, i.e., when the residual between adjacent body joints are smaller than the cost of direct coding. This also indicates the importance of our combined multimodal scheme, which can adaptively decide the proper time of selecting the spatial differential coding mode.

2. The MV-based coding method, RMV-based method, and the temporal differential coding method are all aimed at reducing the temporal redundancy of skeleton data. However, our proposed MV-based coding and RMV-based methods obviously outperform the temporal differential coding method. This indicates that our proposed temporal prediction methods can model and reduce the temporal redundancy more appropriately.

3. The trajectory-based method (T-based coding in Table 5.2 can also achieve obvious bit reductions. Comparatively, it performs extremely well on sequences with a large number of easily predictable skeleton motions (e.g., workers and piano). At the same time, its improvements will be comparatively smaller on sequences with less predictable motions. More importantly, the improvements from the trajectory-based method are complementary to the MV-based coding and RMV-based methods (i.e., the trajectory-based method often has good performance when the MV-based coding and RMV-based methods do not). This allows us to combine them to have a better compression approach.

4. Our multimodal-based method, which adaptively combines the four prediction modes, obtains the best overall performance by properly combine the advantages of all prediction modes. Even when the multimodal-based method does not achieve the best performance for some sequences (e.g., piano) due to the imperfect selection of the best modes (cf. Eqs. 5.13-5.14), its performance is still very close to the best-performed mode. This also demonstrates that our adaptive multimodal coding scheme can reliably select proper modes to guarantee the performance of the combined result.

5.5.6 Results for Proposed Entropy Coding Schemes

In this section, we evaluate the performance improvement provided by our mode-based entropy coding schemes. We consider, four different cases for better analysis: 1) Do not use our proposed entropy coding schemes and only use exponential Golomb codes to encode the prediction residuals ('EGC' in Tables 5.3 and 5.4); 2) Use our adaptive center selection (ACS) scheme together with EGC to encode prediction residuals ('EGC+ACS' in Tables 5.3 and 5.4); 3) Use our dynamic indexing (DI) scheme together with EGC ('EGC+DI' in Tables 5.3 and 5.4); 4) Use both ACS and DI schemes together with EGC to do entropy encoding ('EGC+ACS+DI' in Tables 5.3 and 5.4).

Table 5.3 shows the performance of different entropy coding methods when using the multimodal prediction method to create prediction residuals. Table 5.4 further shows the average performances of the entropy coding methods under different prediction modes. Tables 5.3 and 5.4 provide following observations:

1. In Table 5.3, both the EGC+ACS and EGC+DI methods can obtain improved performance over the EGC method. This indicates the usefulness of our proposed ACS and DI strategy. Moreover, the EGC+ACS+DI method can obtain further improved results by properly combine the strategies of ACS and DI.

Table 5.3 : Statistics of bits per body joint and bit-savings for different entropy coding schemes (the adaptive multimodal prediction method is used as the prediction mode).

	Seq. Name	EGC	EGC+ACS	EGC+DI	EGC+ACS+DI
Posetrack [Andriluka *et al.*, 2018]	Indoor	6.73 (-69.39%)	6.08 (-72.38%)	6.58 (-70.10%)	**6.04 (-72.54%)**
	Ice stick	5.74 (-72.66%)	5.21 (-75.21%)	5.64 (-73.15%)	**5.15 (-75.45%)**
	Karate	5.87 (-72.05%)	5.40 (-74.27%)	5.76 (-72.57%)	**5.34 (-74.55%)**
	Workers	4.34 (-79.33%)	4.36 (-79.25%)	4.33 (-79.40 %)	**4.29 (-79.59%)**
	Street	6.36 (-71.07%)	6.06 (-72.46%)	6.12 (-72.16%)	**5.89 (-73.25%)**
	Action	6.15 (-70.73%)	5.54 (-73.61%)	6.11 (-70.88%)	**5.48 (-73.88%)**
	Basketball	7.51 (-64.23%)	6.87 (-67.28%)	7.47 (-64.45%)	**6.83 (-67.46%)**
	Hurdle-race	11.26 (-48.81%)	10.25 (-53.41%)	11.05 (-49.76%)	**10.20 (-53.63%)**
	Football	10.06 (-54.27%)	9.34 (-57.54%)	9.86 (-55.20%)	**9.27 (-57.88%)**
	Football2	9.65 (-56.15%)	8.96 (-59.25%)	9.49 (-56.84%)	**8.90 (-59.54%)**
	Rugby	10.21 (-51.39%)	9.45 (-55.02%)	10.01 (-52.32%)	**9.40 (-55.25%)**
	Musical	8.88 (-57.70%)	8.31 (-60.44%)	8.73 (-58.45%)	**8.24 (-60.75%)**
	Womens	5.66 (-73.05%)	5.27 (-74.89%)	5.58 (-73.42%)	**5.20 (-75.26%)**
	Ice skating	8.93 (-57.49%)	8.31 (-60.43%)	8.90 (-57.62%)	**8.25 (-60.75%)**
	Volleyball	8.87 (-53.32%)	8.25 (-56.59%)	8.64 (-54.52%)	**8.14 (-57.14%)**
	Average	7.75 (-63.45%)	7.18 (-66.14%)	7.62 (-64.07%)	**7.11 (-66.47%)**
Surveillance	Hotel	8.03 (-63.52%)	7.23 (-67.16%)	7.99 (-63.69%)	**7.18 (-67.36%)**
	Childrens	6.78 (-67.72%)	6.04 (-71.24%)	6.63 (-68.45%)	**6.00 (-71.42%)**
	Beach	7.50 (-64.30%)	6.85 (-67.38%)	7.45 (-64.55%)	**6.79 (-67.66%)**
	Stadium	7.20 (-65.71%)	6.53 (-68.92%)	7.08 (-66.29%)	**6.47 (-69.21%)**
	Police	7.87 (-58.58%)	7.27 (-61.73%)	7.61 (-59.92%)	**7.13 (-62.49%)**
	Baby	7.85 (-64.33%)	7.09 (-67.79%)	7.73 (-64.84%)	**7.05 (-67.96%)**
	Workshop	7.62 (-63.70%)	7.24 (-65.53%)	7.45 (-64.53%)	**7.12 (-66.11%)**
	Restaurant	7.90 (-64.11%)	7.33 (-66.68%)	7.65 (-65.22%)	**7.19 (-67.31%)**
	Piano	6.72 (-67.98%)	6.13 (-70.81%)	6.58 (-68.69%)	**6.07 (-71.11%)**
	Footpath	6.41 (-67.96%)	6.12 (-69.41%)	6.32 (-68.38%)	**6.09 (-69.56%)**
	Road	4.78 (-76.10%)	4.75 (-76.23%)	4.74 (-76.32%)	**4.72 (-76.39%)**
	Luggage	4.69 (-76.53%)	4.68 (-76.62%)	**4.64 (-76.82%)**	4.64 (-76.79%)
	Square	3.62 (-82.78%)	3.75 (-82.13%)	**3.59 (-82.88%)**	3.72 (-82.28%)
	Crossing	3.39 (-83.88%)	3.61 (-82.79%)	**3.38 (-83.91%)**	3.58 (-82.94%)
	Night	1.91 (-91.31%)	2.11 (-90.42%)	**1.91 (-91.33%)**	2.09 (-90.52%)
	Average	6.15 (-70.62%)	5.78 (-72.38%)	6.05 (-71.10%)	**5.72 (-72.66%)**

Table 5.4 : Bits per skeleton points and Bit-savings for different prediction modes and different entropy coding schemes. (The bits for coding side information are excluded)

Entropy coding	Dataset	MV-based coding	RMV-based coding	T-based coding	Multimodal coding
EGC	Posetrack	9.01 (-57.51%)	8.61 (-59.37%)	8.10 (-61.79%)	7.75 (-63.45%)
	Surveillance	7.14 (-65.90%)	7.00 (-66.58%)	7.98 (-61.86%)	6.15 (-70.62%)
EGC+ACS	Posetrack	8.56 (-59.62%)	8.33 (-60.69%)	8.32 (-60.78%)	7.18 (-66.14%)
	Surveillance	6.86 (-67.21%)	6.95 (-66.80%)	7.83 (-62.61%)	5.78 (-72.38%)
EGC+DI	Posetrack	8.98 (-57.64%)	8.59 (-59.47%)	8.07 (-61.94%)	7.62 (-64.07%)
	Surveillance	7.07 (-66.21%)	6.94 (-66.86%)	7.95 (-62.02%)	6.05 (-71.10%)
EGC+ACS+DI	Posetrack	8.53 (-59.78%)	8.29 (-60.92%)	8.26 (-61.04%)	**7.11 (-66.47%)**
	Surveillance	6.80 (-67.53%)	6.85 (-67.29%)	7.78 (-62.86%)	**5.72 (-72.66%)**

2. The improvements from the entropy coding method in Tables 5.3 and 5.4 are smaller than the prediction methods in Table 5.2. This is mainly because of the fact that most of the redundancy has already been reduced by our prediction modes. From this point of view, it is still valuable for our proposed entropy coding schemes to obtain further improved performances from the already highly compressed residual.

3. According to Table 5.4, our ACS and DI entropy coding schemes provide improvements on all the prediction modes. Even when our entropy coding scheme is applied to the multimodal prediction method, it outperforms individual prediction modes since it can intelligently switch between

Table 5.5 : Skeleton side information coding performance comparison for different methods (in bits per skeleton joint and bit savings).

Method	Direct Coding				Proposed Coding			
Dataset	N_S^t	SID_s^t	O_s^t	Total	N_S^t	SID_s^t	O_s^t	Total
Posetrack	0.055	0.285	1.178	1.518	0.014	0.128	0.264	**0.406 (73.26%)**
Surveillance	0.047	0.449	1.516	2.012	0.013	0.146	0.283	**0.442 (78.03%)**

prediction modes.

5.5.7 Results for Skeleton Side Information Coding & Overall Performance

We also show the results for skeleton side information coding. The skeleton side information consists of three parameters: the number of skeletons in the frame denoted by N_S^t, skeleton ID information for each skeleton denoted by SID_s^t, and occlusion information for each skeleton denoted by O_s^t. The comparison for direct and proposed coding method is illustrated in Table 5.5. Our skeleton side information coding method achieves 73.26% and 78.03% bit-savings on an average over direct coding method for Posetrack, and surveillance test sequences, respectively. Moreover, Figure 5.13 further shows the overall performances of our approach, which combines all the prediction, entropy coding, and side information coding schemes. According to Figure 5.13, our approach can obtain an average of 66% and 44% bit savings from the existing direct coding and temporal differential coding methods on Posetrack dataset, and an average of 73% and 48% bit savings on Surveillance dataset. This shows the obvious advantage of our approach on both the datasets.

5.5.8 Results for the Proposed Method under Different Scenarios

Finally, to further evaluate the capability of our approach in handling different scenarios, we perform experiments in the following three situations:

- **Results on different frame rates.** To investigate the response of the proposed method under different frame rates or motion degrees, we down-sample the original skeleton sequences at different rates with frameskip = {0, 1, 2}. The results for different frame skip sizes are illustrated in Table 5.6. From Table 5.6, the coding efficiency of most methods decreases when the frameskip sizes increase. This is mainly because the residual between consecutive frames become enlarged under larger skip sizes. However, even at a large skip size (e.g., frameskip = 2, meaning the video is downsampled to 1/3 of the original size), our approach can still reduce about 50% of the skeleton data.

- **Results on real-time estimated skeleton data.** We also evaluate the performance of our proposed method by creating a set of 'wild' skeleton sequences. Note that these 'wild' skeleton sequences are obtained from the same video sequences like the ones in Table 5.1. However, different from the ground-truth labeled sequences in Table 5.1, these wild skeleton sequences are extracted using a state-of-the-art real-time skeleton estimation and tracking algorithm [Xiu *et al.*, 2018]. Due to the complex nature of the videos we selected, the wild skeleton sequences include a large number of estimation biases and tracking errors, which leads to a decreased redundancy space for the wild data (cf. Figure 5.14). The compression results for the wild skeleton sequences are shown in Table 5.7. According to Table 5.7, our proposed approach can still obtain about 50% compression ratio even when the available redundancy space shrinks for the wild skeleton sequences. Moreover, since the wild skeleton sequences become less predictable, the effectiveness of the trajectory-based method (T-based coding) decreases more obviously as we only use a straightforward method for trajectory prediction (cf. Eq. 5.11). This makes it less able to complement with the other prediction methods (MV-based and RMV-based) for improving the

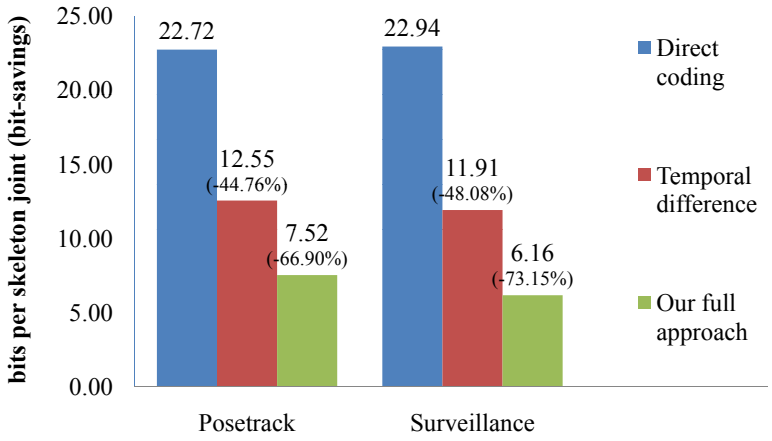

Figure 5.13 : Comparison between the overall version of our approach and the direct coding & temporal differential coding methods. Our approach includes all of the prediction, entropy coding, and side information coding schemes proposed in this chapter. The direct coding method use direct coding to code side information. The temporal differential coding method use EGC to encode prediction residuals and temporal differential coding to encode side information.

(a) (b)

Figure 5.14 : Example wild skeleton sequences: (a) Ice skating, (b) Volleyball.

combined multimodal coding method. In practice, we can utilize more sophisticated trajectory prediction methods [Li *et al.*, 2015; Lin *et al.*, 2016b; Qi *et al.*, 2017; Quintero Mínguez *et al.*, 2019] to implement our trajectory-based method, to obtain a further improved compression performance for wild skeleton data.

- **Gaussian noise.** Similar to the previous experiment, we also introduce zero-mean Gaussian noise with variance σ={0.2, 0.5} on the ground truth skeleton sequences to analyze the capability of our approach when the input skeleton sequences are interfered or become less predictable. According to the results in Table 5.8, the performances of most compared methods are decreased due to the shrunk redundancy space in the noisy skeleton sequences. However, our approach can still effectively compress more than half of the skeleton data sizes. This further demonstrates the effectiveness of our approach.

Table 5.6 : Statistics of bits per joint & bit-savings for different frame skip scenarios on Posetrack & Surveillance datasets.

Dataset	Frame skip	Direct coding	Temporal differential coding	Spatial differential coding	MV-based coding	RMV-based coding	T-based coding	Multimodal coding
Posetrack	0	22.72	12.55 (-44.76%)	22.55 (-0.75%)	8.94 (-60.65%)	8.70 (-61.71%)	8.67 (-61.84%)	**7.52 (-66.90%)**
	1	22.72	15.64 (-31.16%)	22.55 (-0.75%)	11.30 (-50.26%)	11.05 (-51.36%)	11.96 (-47.36%)	**10.08 (-55.63%)**
	2	22.72	17.47 (-23.11%)	22.54 (-0.79%)	12.96 (-42.96%)	12.58 (-44.63%)	14.27 (-37.19%)	**11.73 (-48.37%)**
Surveillance	0	22.94	11.91 (-48.08%)	22.73 (-0.92%)	7.24 (-68.44%)	7.29 (-68.22%)	8.22 (-64.17%)	**6.16 (-73.15%)**
	1	22.94	14.88 (-35.14%)	22.72 (-0.96%)	9.49 (-58.63%)	9.52 (-58.50%)	11.02 (-51.96%)	**8.48 (-63.03%)**
	2	22.94	16.82 (-26.68%)	22.73 (-0.92%)	11.04 (-51.87%)	11.01 (-52.01%)	12.87 (-43.90%)	**10.10 (-55.97%)**

Table 5.7 : Statistics of bits per joint & bit-savings for skeletons estimated [Xiu *et al.*, 2018] on Posetrack & Surveillance datasets.

Dataset	Skeleton source	Direct coding	Temporal differential coding	Spatial differential coding	MV-based coding	RMV-based coding	T-based coding	Multimodal coding
Posetrack	GT	22.72	12.55 (-44.76%)	22.55 (-0.75%)	8.94 (-60.65%)	8.70 (-61.71%)	8.67 (-61.84%)	**7.52 (-66.90%)**
	ES [Xiu *et al.*, 2018]	22.72	13.80 (-39.26%)	20.22 (-11.00%)	11.18 (-50.79%)	11.15 (-50.92%)	12.22 (-46.21%)	**10.08 (-55.63%)**
Surveillance	GT	22.94	11.91 (-48.08%)	22.73 (-0.92%)	7.24 (-68.44%)	7.29 (-68.22%)	8.22 (-64.17%)	**6.16 (-73.15%)**
	ES [Xiu *et al.*, 2018]	22.94	13.55 (-40.93%)	21.70 (-5.41%)	9.73 (-57.59%)	10.07 (-56.10%)	11.42 (-50.22%)	**9.04 (-60.59%)**

Table 5.8 : Statistics of bits per joint & bit-savings for different Gaussian noise levels on Posetrack & Surveillance datasets.

Dataset	Noise level (σ)	Direct coding	Temporal differential coding	Spatial differential coding	MV-based coding	RMV-based coding	T-based coding	Multimodal coding
Posetrack	0	22.72	12.55 (-44.76%)	22.55 (-0.75%)	8.94 (-60.65%)	8.70 (-61.71%)	8.67 (-61.84%)	**7.52 (-66.90%)**
	0.2	22.72	12.56 (-44.72%)	22.55 (-0.75%)	8.99 (-60.43%)	8.77 (-61.40%)	8.74 (-61.53%)	**7.58 (-66.64%)**
	0.5	22.72	12.86 (-43.40%)	22.55 (-0.75%)	9.90 (-56.43%)	9.86 (-56.60%)	9.74 (-57.13%)	**8.55 (-62.37%)**
Surveillance	0	22.94	11.91 (-48.08%)	22.73 (-0.92%)	7.24 (-68.44%)	7.29 (-68.22%)	8.22 (-64.17%)	**6.16 (-73.15%)**
	0.2	22.94	11.93 (-47.99%)	22.73 (-0.92%)	7.40 (-67.74%)	7.47 (-67.44%)	8.34 (-63.64%)	**6.30 (-72.54%)**
	0.5	22.94	12.36 (-46.12%)	22.76 (-0.78%)	9.02 (-60.68%)	9.10 (-60.33%)	9.43 (-58.89%)	**7.80 (-66.00%)**

5.6 SUMMARY

In this chapter, a novel lossless compression scheme for encoding skeleton sequences is presented. Our method primarily emphasized on exploiting two redundancies, namely: prediction redundancy and coding redundancy. To this end, we introduced a multimodal prediction scheme that switches between a set of prediction modes to effectively exploit spatial and temporal correlations present in the skeleton sequences. In total, four prediction modes were introduced, namely: spatial differential prediction, MV-based prediction, RMV-based prediction, and trajectory-based prediction. Then, the residual obtained after removing prediction redundancy were employed to our entropy coding schemes to remove coding redundancy further. For this, we presented two entropy coding mechanisms, namely: adaptive center selection and dynamic indexing. Lastly, we also introduced a simple approach for skeleton side information coding. Our skeleton encoding scheme is lossless in nature, and both encoder and decoder work symmetrically. The experiments were performed on two datasets containing challenging skeleton sequences. Experimental results show that our adaptive method significantly outperforms the direct coding scheme. With this scheme, the third objective of the Thesis, to develop an effective and efficient storage mechanism for Spatio-temporal skeleton sequences, is achieved.

Till now, all the three objectives set at the start of the Thesis are achieved. The conclusions drawn from the methods presented in the previous chapters will be explained in brief in the next chapter.

...