A New Algorithm for Compressing Massive Region-of-Interest Location Information in Videos

Mingliang Chen[1], Weiyao Lin[1], Xiaozhen Zheng[2], Xu Chen[2]

[1]Department of Electronic Engineering, Shanghai Jiao Tong University, China
[2]Research Department of Hisilicon Semiconductor and Component Business Department, Huawei Technologies, China

Abstract

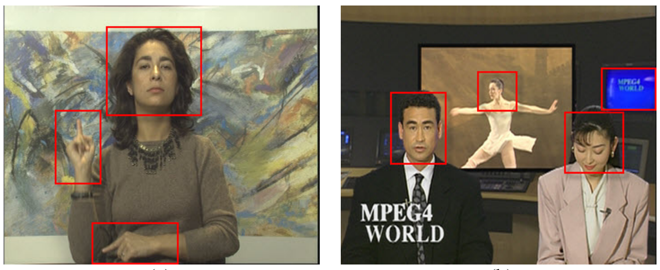
Region-of-Interest (ROI) location information in videos is of increasing importance in many applications including user interest analysis and user experience improvement. Although ROI-based coding has been studied by many researchers, most of the works only focus on using the ROI information for improving coding efficiency while the ROI location information itself is seldom coded or transmitted. Though some methods can recover the ROI locations at the decoder by some parameters such as the quantization parameter, they will fail to work when the number of ROIs becomes large or the ROIs become overlapping. In this paper, we propose a new algorithm to compress this massive ROI location information in videos. The proposed algorithm introduces the region position information extracted from the reconstructed frame as the reference to reduce the ROI location data. Furthermore, the temporal correlations among ROIs in neighboring frames are also utilized for compressing the ROI locations. By suitably integrating the extracted region position as well as the temporal correlation, the proposed algorithm can reduce the data by about 20% for videos with 30 ROI regions. Experimental results demonstrate the effectiveness of the proposed algorithm.

I. Introduction

Region-of-Interest (ROI) location information in videos is of increasing importance in many applications [1-4, 11]. For example, the ROI locations estimated from user eye gaze can be used to analyze the changes of user interests when viewing a video. In video conference applications, the identified ROIs can be used to guarantee the video qualities of the interest regions in order for improving the user experiences. Furthermore, the ROI locations of cars and pedestrians in surveillance videos can also be used for event analysis and abnormality detection [11].

Various ROI-based coding methods have been developed [2-5]. Chen et al. [2] use robust skin-color detection to obtain ROI position information for efficient coding. Menser et al. [3] analyze the video contents by face detection & tracking and apply them into the application of ROI coding. Hu et al. [4] divide one frame into three different region types and then perform rate control by allocating more bits to those interest region types. Arachchi et al. [5] also separate the video frame into different interest regions for achieving unequal error protection in video streams. However, most of these methods only focus on using the extracted ROIs for improving the coding efficiency while few works try to code and transmit the ROI location information itself. Although some methods [4] can recover the ROI locations at the decoder by some parameters such as the quantization parameters (e.g., macroblocks (MBs) with different quantization parameters can be classified into different ROIs in Hu's method [4]), they will fail to work when the number of ROIs becomes large or the ROIs become overlapping. Moreover, although some object-based coding schemes such as MPEG-4 [10] have the functionality for encoding the object information in videos, the problem of coding the massive ROI location information is still seldom addressed. Therefore, it is important to develop new algorithms to encode these ROI location data.

We assume that ROIs are circled by rectangles as shown in Fig 1. And our task is to encode and transmit the sizes and locations of these rectangles to the decoder. One straightforward way to do this is to directly transmit the rectangle's height, width, and (x, y) coordinates. This simple method can work when there are only a couple of ROIs in the video. However, when the number of ROIs becomes large (e.g., the number of interested objects will become large for surveillance or conference videos), these ROI location data will become huge and non-negligible. For example, according to our experiments, the ROI location data will take about 15% of the total bits for a video with about 15 ROIs. Therefore, new algorithms are required to efficiently compress these huge ROI location data.



(a) (b)
Fig. 1 Some example ROIs in videos (The areas in red rectangles are ROIs).

In this paper, we propose a new algorithm to compress the massive ROI location information in videos. The proposed algorithm introduces the region position information extracted from the reconstructed frame as the reference to reduce the ROI location data. Furthermore, the temporal correlations among ROIs in neighboring frames are also utilized for compressing the ROI location data. By suitably integrating the extracted region position as well as the temporal correlation, the proposed algorithm can efficiently reduce the ROI location data while the original location data values are kept exactly the same.

The rest of the paper is organized as follows: Section II describes the framework of our proposed ROI location data compressing algorithm. Section III describes the details of the key components of our proposed algorithm. The experimental results are given in Section IV. And Section V concludes the paper.

II. The Framework of the Proposed Algorithm

The framework of our proposed ROI location data compression algorithm is shown in Fig. 2.

At the encoder side (i.e., the left of Fig. 2), given the

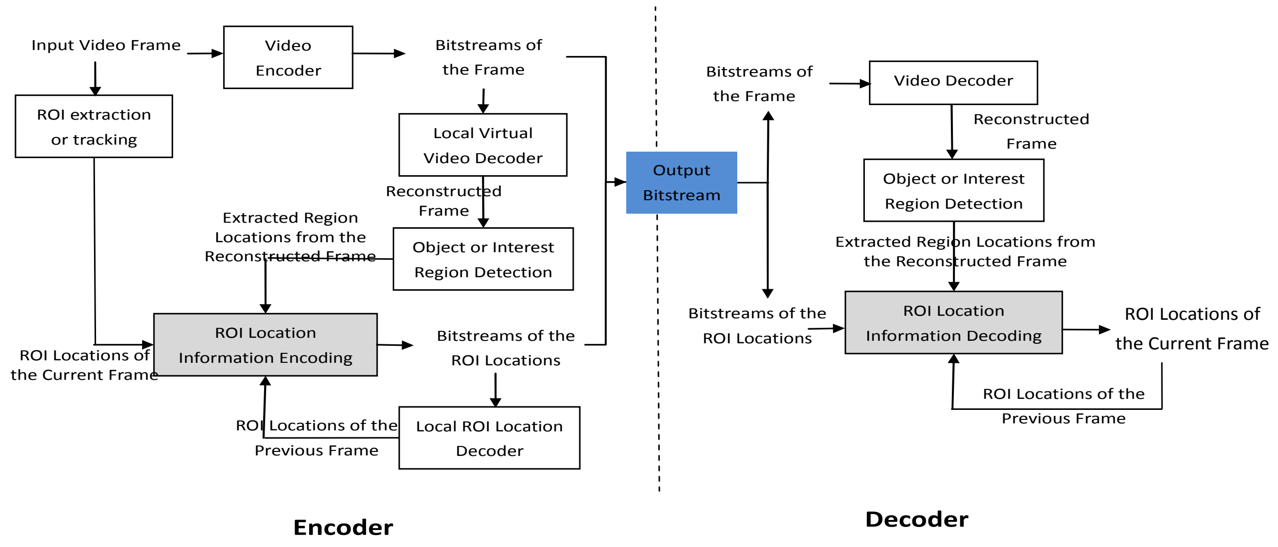

Input Video Frame
Video Encoder
Bitstreams of the Frame
ROI extraction or tracking
Local Virtual Video Decoder
Reconstructed Frame
Extracted Region Locations from the Reconstructed Frame
Object or Interest Region Detection
ROI Location Information Encoding
Bitstreams of the ROI Locations
ROI Locations of the Current Frame
ROI Locations of the Previous Frame
Local ROI Location Decoder
Output Bitstream
Bitstreams of the Frame
Video Decoder
Reconstructed Frame
Object or Interest Region Detection
Extracted Region Locations from the Reconstructed Frame
Bitstreams of the ROI Locations
ROI Location Information Decoding
ROI Locations of the Current Frame
ROI Locations of the Previous Frame
Encoder
Decoder

Fig. 2 The framework of the proposed ROI location data compression algorithm.

current input video frame, the ROI locations in the current frame are first extracted by the "ROI extraction or tracking" step. At the same time, the original video frame will be encoded by the video encoder such as H.264. The bitstreams of the video encoder will be further decoded by the "local virtual video decoder" to create the reconstructed frame. Then, an "object or interest region detection" process will be applied to achieve the possible interest objects and regions in the reconstructed frame. Since many of the interest regions extracted from the reconstructed frame may be the same as the ROIs being coded, they can be used as the reference to reduce the ROI location data. Furthermore, the ROI locations from the previous frame can also be recovered by the "local ROI location decoder" and these previous ROI locations will also be used as another reference to reduce the ROI locations of the current frame. Finally, the "ROI Location Information Encoding" module will take "the interest regions from the reconstructed frame" as well as "the ROI locations from the previous frame" as the reference to encode the ROI location data of the current frame. And the resulting ROI location bitstream will be added together with the bitstream of the frame as the final output bitstream.

Similarly, at the decoder side (i.e., the right of Fig. 2), the interest regions from the reconstructed frame and the ROI locations of the previous frame are first extracted. Then, based on these extracted information, the ROI locations of the current frame can be recovered from the ROI location bitstream.

Several things need to be mentioned about Fig. 2. They are described in the following:

(1) In our proposed algorithm, we introduce two reference information to reduce the ROI location data of the current frame. Firstly, as many of the ROIs are extracted by object detection or tracking methods, we propose to apply the same methods on the reconstructed frame to extract the possible interest regions or interest objects. Since the reconstructed frames are visually similar to the original frames, many of the interest regions or objects from the reconstructed frame are the same as the ROIs from the original frame. Thus, by using these interest regions in the reconstructed frame, the ROI location data can be reduced. Secondly, since the ROI locations have high correlation in neighboring frames (e.g., the locations of the person-head ROI in Fig. 1 (a) are similar in different frames), we also use the previous frame's ROI locations to help reduce the ROI location data in the current frame.

(2) Note that both the interest regions from the reconstructed frame and the ROI locations from the previous frame can be achieved at the decoder side. Thus, our proposed algorithm can be effectively recovered at the decoder without any data loss (i.e., the recovered ROI locations are exactly the same as the original ones from the encoder).

(3) As for ROI extraction and tracking, in our experiments, we use CT-based method [7] and Adaboost-based method [8] to extract ROIs. Similarly, we also use the same CT-based and Adaboost-based methods [7-8] to detect the possible interest regions in the reconstructed frame. However, note that the framework of our proposed algorithm is general and various other object detection methods can be used to achieve the ROIs and interest regions.

(4) When combining the ROI location bitstreams and frame bitstreams, we view the ROI location bits as the header bit and put them together with the other header bits in the frame bitstream. By this way, the ROI location information can be flexibly compatible with the existing bitstream formats.

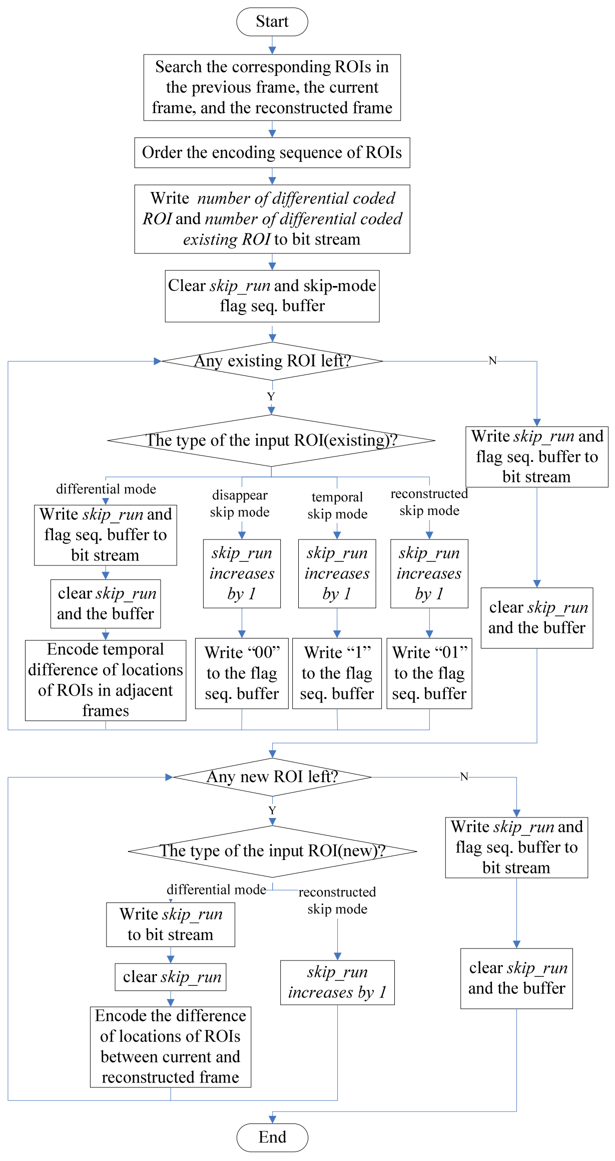
(5) In Fig. 2, the "ROI location information encoding" and the "ROI location information decoding" modules are the key contributions in this paper. In these modules, we introduce various modes such as the differential mode and the skip mode to reduce the input ROI location data. And the details of these modules will be described in the following section. More specifically, since the decoding process can be easily derived from the encoding process, we will only focus on discussing the "ROI location information encoding" module in the following section.

III. The Process of the ROI Location Information Encoding

In this paper, we define the position information of one ROI as:

$$R_i = \{x_i, y_i, w_i, h_i\} \tag{1}$$

where R_i is the i-th ROI in the frame. x_i and y_i are the horizontal and vertical coordinates of the top-left corner pixel of ROI R_i. And w_i and h_i are the width and height of the



Start
Search the corresponding ROIs in the previous frame, the current frame, and the reconstructed frame
Order the encoding sequence of ROIs
Write number of differential coded ROI and number of differential coded existing ROI to bit stream
Clear skip_run and skip-mode flag seq. buffer
Any existing ROI left?
Y
N
The type of the input ROI(existing)?
Write skip_run and flag seq. buffer to bit stream
differential mode
disappear skip mode
temporal skip mode
reconstructed skip mode
Write skip_run and flag seq. buffer to bit stream
skip_run increases by 1
skip_run increases by 1
skip_run increases by 1
clear skip_run and the buffer
clear skip_run and the buffer
Encode temporal difference of locations of ROIs in adjacent frames
Write "00" to the flag seq. buffer
Write "1" to the flag seq. buffer
Write "01" to the flag seq. buffer
Any new ROI left?
Y
N
The type of the input ROI(new)?
Write skip_run and flag seq. buffer to bit stream
differential mode
reconstructed skip mode
Write skip_run to bit stream
clear skip_run
skip_run increases by 1
clear skip_run and the buffer
Encode the difference of locations of ROIs between current and reconstructed frame
End

Fig. 3 The process of the ROI location information encoding module.

ROI R_i, respectively. By the above definition, we can describe the process of our ROI location information encoding process by Fig. 3.

From Fig. 3, there are mainly five steps in the ROI location encoding process. (1) For all the ROIs in the current frame, their corresponding ROIs in the previous frame and the reconstructed frame are first searched and matched. (2) Then these ROIs are organized in a proper order to be written into the bitstreams. (3) When writing the bitstreams, we first write the number of all differential-coded ROIs (NDC) and the number of all "existing" differential-coded ROIs (NEDC, i.e., the ROIs that do not newly appear in the current frame and whose locations are differential-coded) to indicate the ROI numbers. (4) Then, the existing ROI locations in the current frame are written into the bitstream where four modes are used to encode the existing ROIs: differential mode, disappear skip mode, temporal skip mode, and reconstructed skip mode. (5) Finally, the newly-appeared ROIs are written into the bitstream where two modes are used to encode these new ROIs: the differential mode and the reconstructed skip mode.

In the following, we will describe these five steps in detail.

A. Searching the corresponding ROIs in the previous and reconstructed frames

In the first step, we need to find the corresponding ROIs in the previous frame and the reconstructed frame for each ROI in the current frame. In this paper, we use different ways to find the corresponding ROIs for previous frame and reconstructed frame, respectively.

For the previous frames, the tracking method [9] is used to match the ROIs in the current frame with the ones in the previous frame. Note that various other methods can also be used to match the ROIs between the neighboring frames. Furthermore, the temporal matching step will also handle the ROI appear and disappear cases (i.e., an ROI will appear when no corresponding ROI is found in the previous frame and vice versa).

For the reconstructed frames, we first use the object detection methods [8-9] to detect the possible interest regions. Then, the one which has the largest overlapping areas with the current ROI (i.e., the ROI in the current frame) will be decided as the corresponding region of this current ROI. This process can be described by Eqn. (2):

$$R_i^* = \begin{cases} \arg\max\limits_{R^*(k)} \left(OP\left(R^*(k), R_i\right) \right) & if\ \ R_i^* \geq T_{OP} \\ Null & if\ \ R_i^* < T_{OP} \end{cases} \tag{2}$$

where R_i is the ROI in the current frame, R_i^* is the decided R_i's corresponding region in the reconstructed frame. $R^*(k)$ is the k-th possible interest region in the reconstructed frame. T_{OP} is a threshold. $OP(\cdot)$ is the overlap area between two regions and it can be calculated by:

$$\begin{aligned} OP\left(R^*(k), R_i\right) = & \left(min\left(x^*(k)+w^*(k), x_i+w_i\right) - max\left(x^*(k), x_i\right)\right)^2 \\ & + \left(min\left(y^*(k)+h^*(k), y_i+h_i\right) - max\left(y^*(k), y_i\right)\right)^2 \end{aligned} \tag{3}$$

where the definitions of x, y, w, h are the same as in Eqn. (1).

From Eqn. (2), we can see that if a possible interest region $R^*(k)$ has the largest overlap area with the current ROI R_i and the overlap area is larger than a threshold, it will be decided as R_i's corresponding region in the reconstructed frame. Otherwise, no corresponding region will be decided for R_i. And example is shown in Fig. 4.

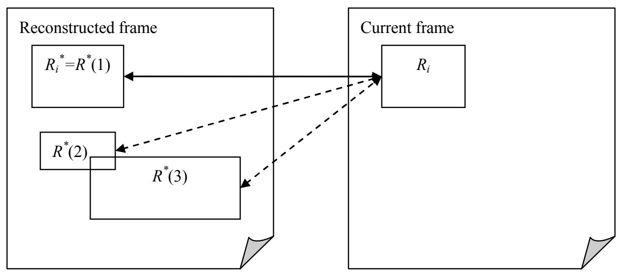


Fig. 4 An example of finding the corresponding regions in the reconstructed frame for ROI R_i.

B. Order the encoding sequence of ROIs

After the correspondences are established for the ROIs in the current frame, the encoding order for these ROIs needs to be organized. In this paper, we first categorize the ROIs into two types: the existing ROIs (i.e., the ROIs that have corresponding ROIs in the previous frame) and the new ROIs (i.e., the ROIs that do not have corresponding ROIs in the previous frame and are the newly-appeared ROIs). Then, these ROIs are organized by putting the existing ROIs first, followed by the new ROIs. Furthermore, the order within the existing ROIs will be the same as the one in the previous frame. And the order within the new ROIs is organized by an zig-zag order according their locations in the current frame. To better illustrate this point, Fig. 5 gives an example.

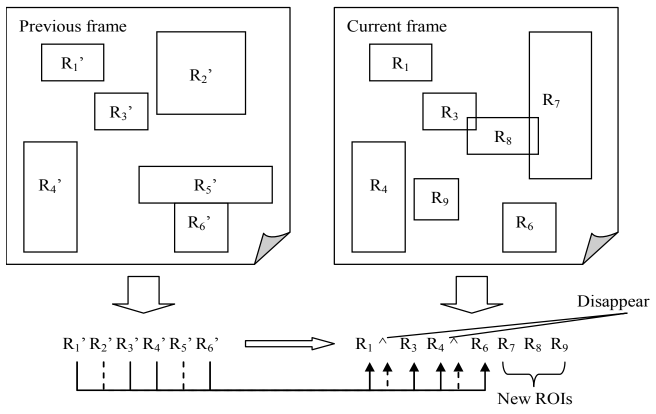


Fig. 5 An example of the ordering of ROIs in the current frame.

C. Write the number of all differential-coded ROIs and the number of all existing differential-coded ROIs

As mentioned, when writing the bitstreams, we first write the number of all differential-coded ROIs (NDC) and the number of all "existing" differential-coded ROIs (NEDC). More specifically, NDC is the total number of ROIs whose location data are coded by the differential mode. And NEDC is the total number of existing ROIs (ROIs that have corresponding ROIs in the previous frame) whose locations are coded by the differential mode. Since our algorithm has various skip modes, by transmitting these two values, the distribution of different ROI coding modes can be clearly indicated and the skip/differential modes for each ROI can be effectively identified at the decoder. This point will be further discussed in the example of Figs 7-8.

D. Encode the existing ROI locations

After the NDC and NEDC are written into the bitstream, the location data of the existing ROIs will be written into the bitstream. In this paper, given the references from the previous and the reconstructed frames, we develop four modes for coding the existing ROI locations. They are described in the following:

(1) The disappear skip mode. When an ROI disappears in the current frame (i.e., there is ROI in the previous frame but no corresponding ROI in the current frame, such as R_2' and R_5' in Fig. 5), we need a mode to indicate this situation. In this paper, we use a disappear skip mode which writes "00" flag bits into the bitstream. At the same time, the indicator *skip_run* will be increased by one to indicate that one skip mode is coded in the bitstream.

(2) The reconstructed skip mode. When the current ROI locations are exactly the same as its corresponding region in the reconstructed frame, we will not transmit the locations of this ROI and simply use a reconstructed skip mode (flag bits "01") to indicate this. Similarly, the indicator *skip_run* will be increased by one to indicate that one skip mode is coded in the bitstream.

(3) The temporal skip mode. Similar to the reconstructed skip mode, when the current ROI locations are exactly the same as its corresponding region in the previous frame, we will not transmit this ROI's locations and simply use a temporal skip mode (flag bit "1") to indicate this. And *skip_run* will be increased by one to indicate the usage of one skip mode.

(4) The differential mode. If the current ROI locations are neither the same as the previous frame nor as the reconstructed frame, we will utilize a differential mode to code the location differences between current ROI and its corresponding ROI in the previous frame, as in Eqn. (4).

$$D_{i,j} = R_i - R_i' = \{x_i - x_i', y_i - y_i', w_i - w_i', h_i - h_i'\} \quad (4)$$

where P_i is the current ROI and P_i' is the corresponding ROI in the previous frame. In our paper, the difference values are coded by the Variable Length Coding (VLC) scheme [1].

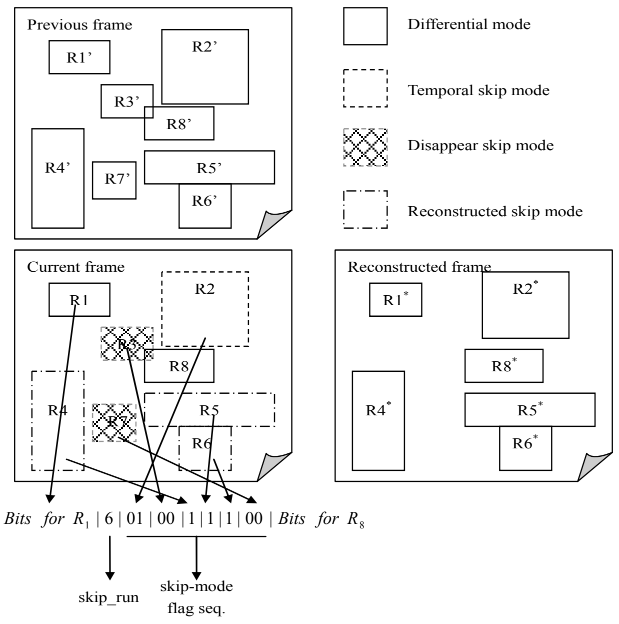


Fig. 6 An example of coding the existing ROIs.

There are several things that need to be mentioned about our existing ROI location coding step.

(1) In our algorithm, we introduce two skip modes (reconstructed skip mode and temporal skip mode) to skip transmitting the ROI locations when they are the same as the reconstructed or previous frames. Furthermore, even when the ROI locations are not the same to their reference frame, the difference values are transmitted. By this way, the ROI location data can be greatly reduced.

(2) Note that when coding each ROI, the modes are searched in a specific order for finding a suitable mode. That is, for an input ROI, we first check whether it is disappearing in the current frame (disappear skip mode), then check whether it's locations are the same as the reconstructed frame (reconstructed skip mode) and the previous frame (temporal skip mode), respectively. Finally, when none of the above modes are coded, the differential mode will be utilized to code the ROI locations.

(3) Also, note that the skip mode bits are not written into the stream directly. Rather, the skip mode bits as well as the *skip_run* indicator will first be saved and updated in a buffer and then written into the bitstream before the next differential mode is coded. By this way, the *skip_run* is able to accumulate the total number of skip ROIs between two differential-coded ROIs.

Fig. 6 shows an example of coding the existing ROIs. In Fig. 6, the first ROI R_1 is differential coded. Then the following ROIs (R_2 - R_7) are coded by different skip modes (note that the crossed blocks for R_3 and R_7 mean that these two ROIs do not appear in the current frame), and the bits for these skipped ROIs will not written into the bitstream until the next differential-coded ROI R_8 is coded.

E. Encode the new ROI locations

The encoding of the new ROIs is similar to the existing ROIs. In this paper, two modes are developed for coding the new ROIs. They are described in the following.

(1) **The reconstructed skip mode.** When the new ROI locations are exactly the same as its corresponding region in the reconstructed frame, we will use reconstructed skip mode to skip coding its locations. However, since there is only one skip mode for new ROIs, we even do not transmit the skip mode flag and only increase the *skip_run* indicator by 1 to indicate the usage of one skip mode.

(2) **The differential mode.** If the new ROI locations are not the same as the reconstructed frame, we will use differential mode to code the location differences between the new ROI and its corresponding region in the reconstructed frame. The difference values will also be coded by the VLC scheme.

To summarize the entire process of our ROI information encoding module, a detailed example is given by Fig. 7.

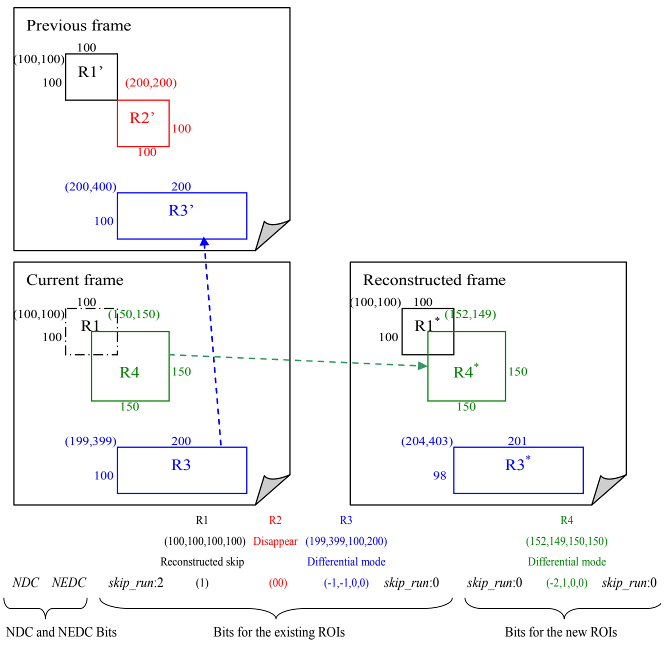


Fig. 7 An example of the entire ROI information encoding process.

In Fig. 7, the previous frame has three ROIs: $R_1^{'}$, $R_2^{'}$, and $R_3^{'}$. The current frame has three ROIs R_1, R_3, and R_4 where R_4 is the newly-appeared ROI and R_2 disappears in the current frame. The corresponding regions in the reconstructed frame are R_1^{*}, R_3^{*}, and R_4^{*}, respectively. As mentioned, when coding the current ROI locations, the NDC and NEDC are first written into the bitstream, and then the bits of the existing ROIs are written, followed by the bits of the new ROIs. Thus, the final output bits of Fig. 7 can be shown in Fig. 8.

From Fig. 8, we can see that (1) Since there are totally two ROIs using differential coding mode (R_3 and R_4), NDC equals to 2. Furthermore, since there is only one existing ROI using the differential coding mode (R_3), NEDC equals to 1. (2) Note that the differential coding mode uses different references for the existing and new ROIs. For the existing ROIs such as R_3, the ROIs in the previous frame is used while the ROIs in the reconstructed frame are used for the new ROIs such as R_4, as shown in Fig. 6. (3) Also, note that a "0" will be added at the ends of the existing ROI bits and the new ROI bits respectively to indicate that there is no further skip mode left.

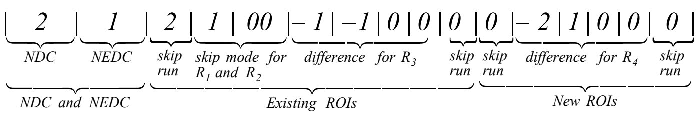


Fig. 8 The final output bits for the example of Fig. 7.

IV. Experimental Results

In this section, we show experimental results for our proposed algorithm. The algorithms are implemented on the H.264/MPEG-4 AVC reference software JM10.0 version [6]. For each of the sequences, the picture coding structure was IPPP…. However, note that our proposed algorithm can also be easily extended to other coding structures including B frames.

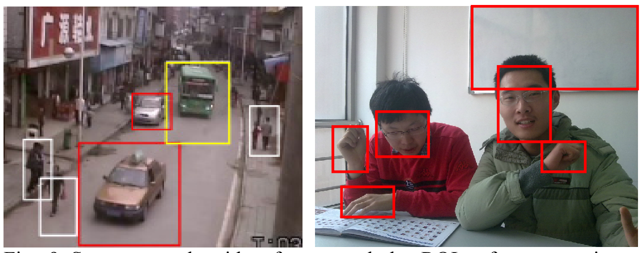


Fig. 9 Some example video frames and the ROIs of our experimental videos.

We collect several videos from the surveillance and the conferencing videos. Some example video frames and the corresponding ROIs are shown in Fig. 9. In our experiments, four methods are compared.

(1) **Direct.** Directly write the absolute location values for the ROIs into the bitstream.

(2) **Differential.** Write the difference of the ROI locations in the current frame and the corresponding ROI locations in the previous frame.

(3) **Proposed.** Using our proposed algorithm to encode the ROI locations in videos.

Furthermore, in order to show the relative importance of the ROI location bits in the videos, we also include the bit rates for the original videos (i.e., the bit rates by using H.264 to encode the videos and not including the bits for ROI location, named as "**Null**" in Tables 1 and 2). In the following, two experiments are carried out to demonstrate the effectiveness of our proposed algorithm.

A. Results of Coding the Same Video with Different QP

In this experiment, we encode one QCIF-sized video with different Quatization Parameters (QPs) ranging from 16 to 36. The number of frames to be encoded is 50, the frame rate is 30 fps, and the average number of ROI per frame is 5. The results are shown in Table 1.

In Table 1, the left part shows the total bit rates while the right part shows the bit rates for the ROI locations only. Note

Table 1 The Bit Rates of a Video with Different QPs

QP	Total Bit Rates (kbit/s) @30Hz				Bit Rates for the ROI Locations (kbit/s) @30Hz		
	Null	Direct	Differential	Proposed	Direct	Differential	Proposed
16	915.48	920.52	917.64	916.70	5.04	2.16	1.22
20	497.84	502.88	500.00	499.08	5.04	2.16	1.24
24	255.46	260.50	257.63	257.03	5.04	2.16	1.57
28	124.54	129.58	126.69	126.22	5.04	2.16	1.68
32	61.73	66.77	63.89	63.51	5.04	2.16	1.78
36	36.19	41.23	38.36	38.03	5.04	2.16	1.84

Table 2 Bit Rates for Videos with Different Resolutions (QP = 28)

seq.	resolution	ROI/ frame	Bit Rate (kbit/s) @30Hz				Bit Rate for the ROI Locations (kbit/s) @30Hz		
			Null	Direct	Differential	Proposed	Direct	Differential	Proposed
I	176×144	4~6	124.54	129.58(+4.05%)	126.69(+1.73%)	126.22(+1.35%)	5.04	2.15	1.68
II	352×288	5~7	108.92	115.64(+6.17%)	111.57(+2.43%)	111.08(+1.98%)	6.72	2.65	2.16
III	176×144	5~7	97.67	103.67(+6.14%)	99.72(+2.10%)	99.47(+1.84%)	6.00	2.05	1.80
IV	864×480	14~17	132.76	152.20(+14.64%)	142.35(+7.22%)	141.71(+6.74%)	19.44	9.59	8.95
V	640×480	27~30	122.69	156.79(+27.79%)	137.84(+12.35%)	133.44(+8.76%)	34.10	15.15	10.75
VI	640×480	55~63	285.69	357.24(+25.04%)	313.75(9.82%)	305.09(+6.79%)	71.55	28.06	19.40

that since we use the same QP for comparison, the PSNR values are the same for different methods and thus are not in this table. From Fig. 1, we can see that if we use the direct method to encode the videos, the bit rate for the ROI locations is high. By using the differential method, the ROI location rate can be reduced by more than half. Comparatively, by using our proposed algorithm, the bit rate can be further reduced by utilizing the reconstructed frame information as well as the various skip modes. Also, note that there are only ROIs in the video of Table 1 and the improvement by our proposed method will be more obvious when the number of ROIs becomes larger, as will be shown in Table 2.

Furthermore, it can also be observed from Table 1 that for the direct and differential methods, the ROI location rates are the same for different QPs. This is because the ROI locations and the ROIs' temporal correlations are fixed in the original videos and are not affected by the coding parameters. Comparatively, since our proposed algorithm introduces the reconstructed frame as references, the ROI location rates of our algorithm are affected by the QP values. More specifically, when the QP increases, the ROI location rate will increase because the reconstructed video qualities will decrease with larger QPs. And this will make the detection results on the reconstructed frame less coherent with the original ROIs, thus decreasing the number of ROIs being skipped by the reconstructed skip mode. However, note that even if our algorithm is affected by the QP parameters, our method can still effectively reduce the ROI location bit under large QPs. And the effect from QP parameters can be reduced by utilizing more sophisticated detection algorithms.

B. Results for Different Videos with Different Resolution

In this sub-section, we conduct experiments on various videos with different resolutions and different number of ROIs. The QP is set to 28, the frame rate is 30 fps, and the number of frames to be encoded is 50. The results are shown in Table 2.

From the table 2, we can see that our proposed method can obviously improve the coding efficiency. For example, since the sequence IV has a large number of ROIs, if we use the direct method, the bit rate for the ROI locations will take about 15% of the total output bits. This is a heavy overhead for video transmission. However, by using our proposed method, the ROI location bits can be effectively reduced to about 6.7%. Furthermore, comparing our method with the differential method, we can see that the improvement of our method is more obvious when the number of ROIs is large. This implies that our proposed method is most effective when coding the videos with massive ROIs, such as coding the people trajectory information in a crowded scene and the people head/hand movements in a remote education class. Such scenarios have become popular and important in recent applications.

V. Conclusion

In this paper, a new algorithm is proposed for compressing the massive ROI position information in videos. The proposed algorithm introduces the region position information extracted from the reconstructed frame as the reference to reduce the ROI location data. Furthermore, the temporal correlations for ROIs in neighboring frames are also utilized for compressing the ROI location data. Experiments demonstrate the effectiveness of the algorithm.

Acknowledgements

This work was supported in part by the following grants: National Science Foundation of China grants (61001146, 61103124), Huawei Innovation Program grant (YB2012120150), the Open Project Program of the National Laboratory of Pattern Recognition (NLPR), the SMC grant of SJTU, Shanghai Pujiang Program (12PJ1404300).

References

[1] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans.CSVT*, 2003.
[2] M.-J.. Chen, M.-C. Chi, C.-T. Hsu and J.-W. Chen, "ROI Video Coding Based on H.263+ with Robust Skin-color Detection Technique," *IEEE Trans. Consumer Electronics*, vol. 49, pp. 724-730, 2003.
[3] B. Menser and M. Brunig, "Face Detection and Tracking for Video Coding Applications," *IEEE Int'l Conf. Signal, Systems and Computers*, vol. 1, pp. 49-53, 2000.
[4] H.-M. Hu, B. Li, W. Lin, W. Li and M.-T. Sun, "Region-Based Rate Control for H.264/AVC for Low Bit-Rate Applications," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 22, pp. 1564-1576, 2012.
[5] H. Arachchi, W. Fernando, S. Panchadcharam, and W. Weerakkody, "Unequal Error Protection Technique for ROI Based H.264 Video Coding," *Int'l Conf. Electrical and Computer Engineering*, pp. 2033–2036, 2006.
[6] JM 10.0, http://iphome.hhi.de/suehring/tml/download/old_jm/.
[7] J. Wu, C. Geyer, and J. M. Rehg, "Real-Time Human Detection Using Contour Cues," *IEEE Int'l Conference on Robotics and Automation*, 2011.
[8] P. Viola, M. Jones, "Robust real-time object detection," *Int'l Journal of Computer Vision*, 2001.
[9] G. Wu, Y. Xu, X. Yang, Q. Yan, K. Gu, "Robust object tracking with bidirectional corner matching and trajectory smoothness algorithm," *Int'l Workshop on Multimedia Signal Processing*, pp. 294-298, 2012.
[10] T Sikora, "The MPEG-4 video standard verification model," *IEEE Trans. Circuits and Systems for Video Technology*, 1997.
[11] W. Lin, M.-T. Sun, R. Poovendran, Z. Zhang, "Activity recognition using a combination of category components and local models for video surveillance," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 18, pp. 1128-1139, 2008.