AN APPROACH TO MPEG-4 MULTI MODE CODING USING SPRITE CODING

Shigeki OKADA, Kumi JINZENJI, and Naoki KOBAYASHI

NTT Cyber Space Labs.
1-1 Hikari-no-oka Yokosuka-shi
Kanagawa, 239-0847 JAPAN
E-mail: {shige,kumi,kobayasi}@nttvdt.hil.ntt.co.jp

ABSTRACT

Sprite coding is supported by the new video coding standard MPEG-4 Version 1 Main Profile and offers strong video compression rates [1]. We use this coding technique to develop low bit rate and high quality applications suitable for narrow-band transmission links such as those provided by the Internet. Sprite coding has a problem in that it does not suit all types of video sequences. In this paper, we propose a multi mode coding method: sprite coding is paired with MPEG-4 simple profile rectangle coding mode. This paper also describes a mode switching algorithm.

1. INTRODUCTION

Sprite coding is supported by the new video coding standard MPEG-4 Version 1 Main Profile. It provides content-based functionality and low bit rate video streams by dividing video sequences into still background sprites and moving foreground objects. We focused on this coding technique in order to develop high quality and very low bit rate applications that are well supported by narrow-band transmission links (384kbps maximum) such as those provided by the Internet and mobile networks. We have already proposed an automatic VOP (Video Object Plane) generation technique that includes GME (Global Motion Estimation)[3], foreground moving object extraction, and background sprite generation[2]. Sprite coding is now very attractive because we have made it fully automatic.

Although sprite coding is expected to dramatically reduce the transmission rate, it has a problem in that it does not suit all types of video sequences. To overcome this problem, we propose the multi mode coding method; sprite coding (sprite mode) is paired with MPEG-4 simple profile rectangle coding (normal mode). With this coding method, we can realize an MPEG-4 system that strongly compresses all types of video sequences. Assessing a video sequence as to its suitability for sprite coding involves checking a few parameters. Also proposed is an automatic mode switching algorithm for multi mode coding.

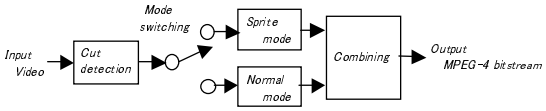

Fig. 1. Multi mode coding method

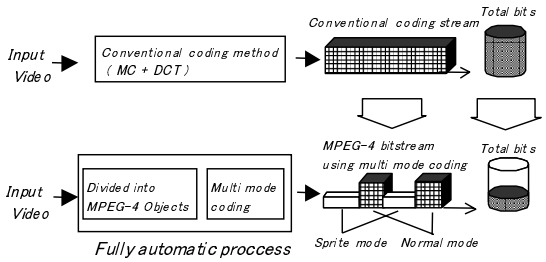

Fig. 2. Comparison of new multi mode coding method

2. MULTI MODE CODING METHOD

Fig.1 overviews the multi mode coding method. The proposed method consists of five parts: cut detection, mode switching, sprite

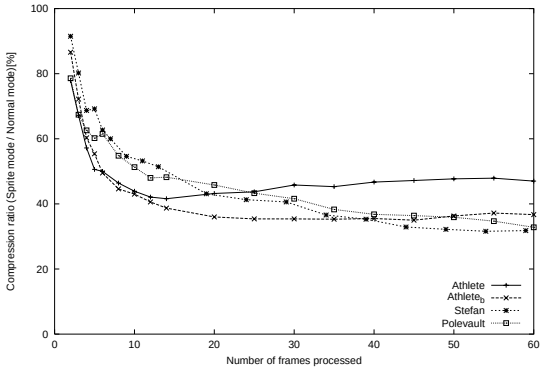

Fig. 3. Number of frames processed and coding efficiency with sprite mode coding

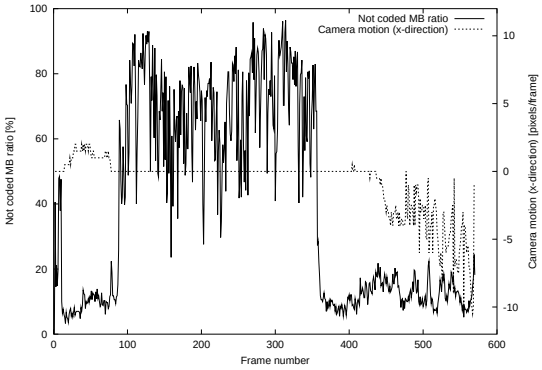

Fig. 4. Camera operation and Not Coded MB ratio with normal mode coding

mode coding and normal mode coding, and combining. At first, the input video sequence is divided by cut detection into several shots that have no cut points (scene changes). Each shot is then automatically switched and coded appropriately (sprite mode or normal mode). As a result, the system outputs a bit stream consisting of two differently coded streams. Fig.2 compares the multi mode coding method to the conventional coding method. The proposed method offers higher compression ratios than the conventional method for any type of video sequence.

3. SPRITE MODE

Our analysis of many different videos discovered that there were three features in a shot that determine if the shot should coded in sprite mode.

3.1. Number of frames

First, the shots should have more than a minimum number of frames (30) because the coding efficiency of sprite mode is poor when there are few frames. This was confirmed in experiments.

Fig.3 shows the relation between coding efficiency and number of frames processed with sprite coding mode. The compression ratio is the ratio of coded bits with sprite mode to that with normal mode.

3.2. Camera operation

Second, the shot should contain some camera operation. To check this property, we must determine the camera parameters. We have already proposed an algorithm that automatically extracts camera operation from operation vectors[3]. The parameters so determined are zooming, rotating, panning, and tilting. If the shot contains no camera operation, we can get adequate coding efficiency by using Not Coded Macroblock (macroblock type). Experiments showed that when the shot has no camera operation, the shot has a high Not Coded MB ratio and so should not be coded in sprite mode.

Fig.4 shows the relation between camera operation and Not Coded MB ratio. Although we consider four camera operations zoom, tilt, pan, and tilt, this graph shows the relationship for just pan. The Not Coded MB ratio is high when there is no operation (pan value is zero).

3.3. Foreground ratio

Third, the foreground object of the shot should occupy only a small percentage of the frame. The part of the frame other than the foreground object is the background object, which is coded as a sprite. We regenerate the frame by adding the background object extracted from the sprite to the foreground object.

Fig.5 shows the relation between coding efficiency and foreground object ratio (percentage of frame occupied by the object). If the foreground object ratio is near 10%, sprite mode coding efficiency is twice that of the ex-

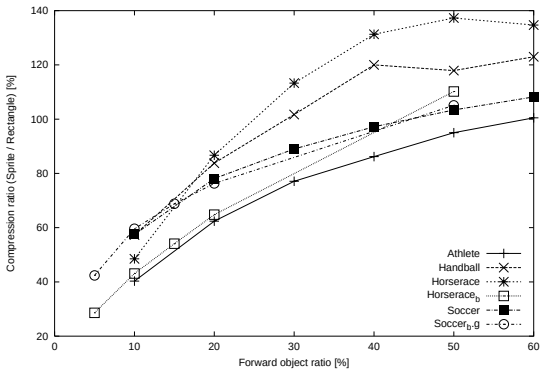

Fig. 5. Foreground object ratio and coding efficiency with sprite mode coding

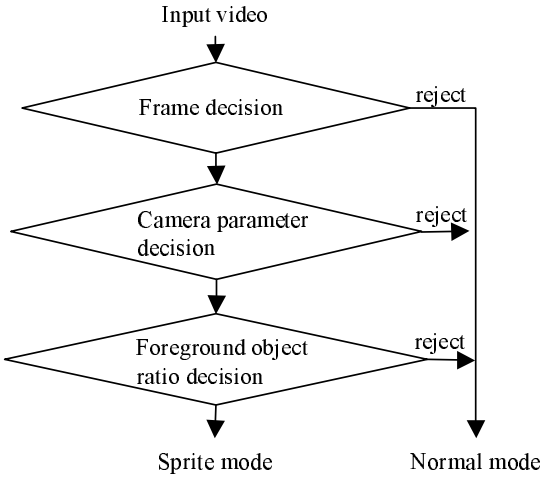

Fig. 6. A flow chart of automatic mode switching algorithm

isting video coding method. If the foreground object ratio is large (above 30%), it has rather low coding efficiency.

4. IMPLEMENTATION

We developed an automatic mode switching algorithm and used it to realize a prototype multi mode coding system. This algorithm has three decision modules. These offer frame decision, camera parameter decision, and foreground object ratio decision. All modules must decide that the input shot is suitable for sprite mode coding; rejection by any module means that the shot is normal mode coded. The thresholds used in the modules can be determined empirically. Fig.6 shows a flow chart of the proposed algorithm. A shot consists of

Image	Mode*1	Sprite mode ratio*2 (%)	Bitrate (kbps)
Paris	normal	-	160
	multi	100	30
Baseball	normal	-	244
	multi	7	227
Horserace	normal	-	477
	multi	81	295
Soccer	normal	-	163
	multi	87	129

*1 normal:fully normal mode, multi:multi mode

*2 the average percentage of sprite mode frames in full video

Table 1. Coding results(1)-Comparison of bitrate (Target conditions: SIF, QP=12, 15[fps])

frames in the period from T1 to T2($T1 \leq t \leq T2$).

i) Frame decision
$T2 - T1 \geq TH_{frame}$

ii) Camera parameter decision
$pan(t) \geq TH_{pan} \| \; tilt(t) \geq TH_{tilt} \; \| \; zoom(t) \geq TH_{zoom} \| \; rotate(t) \geq TH_{rotate}$

iii) Foreground object ratio decision
$fg\_ratio(t) \leq TH_{fgratio}$

5. EXPERIMENTAL RESULTS

We used the prototype system to code the four video sequences shown in Table.1 (SIF, 30[fps], 30[sec]). Two coding methods were compared: MPEG-4 multi mode coding and full shot normal mode coding (MPEG-4 simple profile rectangle coding) from the view point of bit rate (coding efficiency) needed to achieve the same objective image quality. Coding conditions were: QP = 12, frame rate = 15 [fps]. Our results shows that the proposed multi mode coding method offers higher coding efficiency (up to five times higher) than the normal coding scheme for video sequences containing static scenes or those with camera operation.

Fig.7 shows some other coding results. Note that frame rate is written in the graph to compare multi mode to fully normal mode. Cod-

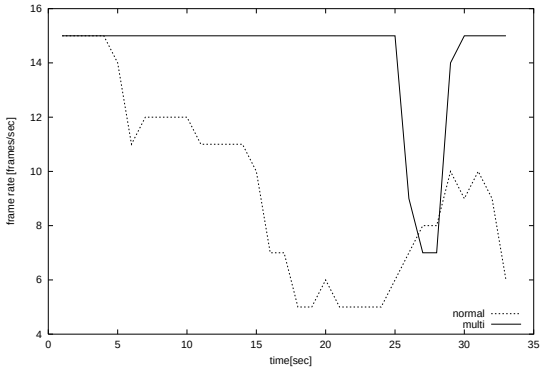

Fig. 7. Coding results(2)-Comparison of frame rate (Image : Horserace, Target conditions : 128[kbps], SIF, 15[fps])

ing conditions were: bit rate = 128 [kbps], SIF, frame rate = 15[fps]. Fully normal mode often must drop below the target framerate, but multi mode can satisfy the desired frame rate continuously. The one slip occurred when one shot was rejected by the foreground object decision module and so was coded in normal mode.

6. CONCLUSION

We focused on the sprite coding supported by MPEG-4 Version 1 Main Profile. In this paper, we analyzed the characteristics of sprite coding and proposed an automatic mode switching algorithm for multi mode coding. Several experiments on MPEG-4 multi mode coding were conducted. Coding results shows constantly higher frame rate or higher coding efficiency than existing coding schemes.

There are several remaining issues in realizing a truly effective multi mode coding system. Examples include multi mode rate control and faster processing. We are now investigating more useful systems that can handle any video.

Acknowledgement

The authors gratefully acknowledge Ms. Noriko YONEHARA for her help in the simulations.

7. REFERENCES

[1] ISO/IEC JTC 1/SC 29/WG 11: “Information technology – Coding of audio-visual objects – Part 2: Visual,” 14496-2, Dec. 1999.

[2] K. Jinzenji, S. Okada, H. Watanabe, N. Kobayashi, “Automatic Two-layer Video Object Plane Generation Scheme And Its Application To MPEG-4 Video Coding,” IEEE ISCAS2000, pp.606-609, May 2000.

[3] K. Jinzenji, H. Watanabe, N. Kobayashi, “Global Motion Estimation For Static Sprite Production And Its Application To Video Coding,” IEEE ISPACS’98, pp.328-332, November 1998.