JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 30, 1755-1771 (2014)

Classifying Useful Motion Vectors for Efficient Frame Rate Up Conversion of MC-DCT Encoded Video Streams*

JONGHO NANG, SANGCHUL KIM AND HYUK-JUN LEE
Department of Computer Science and Engineering
Sogang University
Seoul, 121-742 Korea
E-mail: jhnang@sogang.ac.kr; smaslayer1@nate.com; hyukjun1@sogang.ac.kr

Frame Rate Up Conversion (FRUC) is a technique that makes natural video playback possible by increasing the frame rates of video streams at the decoder side. Previous FRUC methods estimated the motions of the all blocks between nearby decoded frames to generate an interpolated frame, and hence, required a lot of computational resources. This paper proposes a new frame interpolation method that reuses the motion vectors (MV's) that are generated by MC-DTC based encoder for primarily reducing the bitrate of video stream, but still useful for FRUC. In order to classify the MV to a useful one for FRUC, the visual similarities of two blocks that are located at the same position at the reference and current frames, the visual similarities of two blocks that are connected by the MV, and the visual similarity with the surroundings blocks, are measured. If these similarities exceed some thresholds, the MV will be directly reused for frame interpolation without additional block motion estimation in the proposed method. Additionally, if this decision is ambiguous, the pre-trained SVM (Support Vector Machine) is used to decide its usefulness. Experimental results using MC-DCT encoded videos with various motion energies show that the recall and precision of the proposed classifying method are about 0.72~0.99 and 0.83~0.99, respectively. Since the motion estimations for the blocks that have the MV's which are classified as useful ones could be skipped in the frame interpolation process, the total time for FRUC can be reduced by 50-99% without a drastic video image quality loss.

Keywords: frame rate up conversion (FRUC), MC-DCT motion vector, classify useful motion vector, detect useful motion vector processing (DUMP), transcoding

1. INTRODUCTION

Recent increases in the sales of digital appliances with embedded cameras (*e.g.*, mobile phones, smartphones, tablets), coupled with improvements to the wireless internet environment, have led to a boom in applications for using videos in a wireless environment. Since these wireless connections usually offer a lower bandwidth than wired services, videos are offered at low quality or low frame rates. In particular, in the case of low frame rates (*e.g.*, less than 15 frames/second) the receiver of the video will experience a stuttering effect, which causes a motion judder/blur effect on LCD and LED screens due to the low frequency, which in turn causes problems such as eye fatigue [1]. Hence, to solve this problem with low frame rates, the FRUC (Frame Rate Up Conversion) method, which inserts a virtual frame, is applied [2]. The MC (Motion Compensa-

Received February 13, 2013; revised June 6, 2013; accepted July 19, 2013.
Communicated by Jen-Hui Chuang.
* This work was supported by the IT R&D program of MSIP/KEIT. [10044615, Development of Open-Platform /Social Media Production and Delivery System for Fused Creation, Editing, and Playing of Broadcasting Media Contents on Cloud Environments].

tion)-FRUC method [3-5], a recently proposed such method, involves estimating the motion vectors (MVs) of the all blocks between nearby decoded frames, then using the virtual MV to interpolate the inserted frame. It is being widely used because it is resistant to noise. However, since the estimating the MVs of all blocks require a lot of computational resources, it could not be applied to the wireless systems such as mobile phones.

This paper proposes a new frame interpolation method that partially reuses the MV embedded in MC-DCT encoded video streams (*e.g.* H.264) for FRUC. Since the MV in MC-DCT encoded video streams primarily aims to reduce the bitrate of the video stream, there are MV's that may not be adequate for FRUC although they attempt to point the block with the smallest difference. This paper proposes a classifying method that determines the usefulness of MV's for FRUC by analyzing the visual similarities of some blocks in the current and reference frames. The first classifying rule uses the temporal redundancy between these two frames. If the visual difference of two blocks at the same position in these frames is less than a certain threshold, the corresponding block's MV is set to (0,0) and is classified as a useful one. The second rule uses the relationship between two blocks linked by a MV. If the MV of the block in question is accurate, the block of the current frame and the block pointed by the MV in the reference should be similar. If the visual difference of these two blocks is less than a certain threshold, then the MV is classified as a useful MV. The third rule uses the spatial redundancy of frame by comparing the visual similarity between a target block and its neighboring blocks. If the difference is less than a certain threshold, these blocks may belong to the same object, which results in the MV of the neighboring block being used as the MV for the target block as well. During these considerations, if the MV of the neighboring block cannot be trusted, usefulness of MVs can be determined using a SVM classifier that is trained with pre-classified useful MV's extracted from various video streams. If the MV for a block is classified as useless for FRUC, a conventional motion estimation is adopted to find the MV of that block suitable for FRUC. Experimental results with various video streams show that the recall and precision of the proposed classifying algorithm are about 0.72~0.99 and 0.83~0.99, respectively. Since the ratio of useful MVs depends on the motion energy of the video stream (the less motion energy in video stream, the more useful MVs for FRUC), the speedup of FRUC with the proposed interpolation method would be different according to the motion energy of video. Our experimental results show that the total FRUC time could be reduced by 50-99% without a noticeable degradation of visual quality of the interpolated video frame.

This paper is structured into a total of five sections. Section 2 introduces previous researches on FRUC, while section 3 deals with the useful MV classifying method for FRUC in H.264 encoded video stream. Section 4 carries out experiments on proposed classifying method, then compares and discusses the performance and its usefulness. Section 5 is the conclusion and details concerning future works.

2. PREVIOUS WORKS ON FRUC

FRUC is a technique that inserts a virtual frame between adjacent two decoded frames, making videos to play more naturally. Early research on FRUC used an interpolation method using the average values of different part of two frames, but this method had problems, such as severe video distortion or blurring [2]. Much more recently, MC-

FRUC (Motion Compensated Frame Rate Up Conversion) methods were actively researched. It first estimates the motion of pixels/blocks/objects between adjacent two frames (Motion Estimation: ME) and uses such estimation to interpolate the virtual frame (Motion-Compensated Interpolation: MCI). Among these methods, the per-block ME method (or Block Matching Algorithm: BMA) [6]) is widely researched because it is robust to noisy and requires less computations than per-pixel or per-object ME methods. Current research on BMA is looking at eliminating inaccurate MVs that occur during ME, and finding methods to minimize the blocking artifact in the interpolated frame that is a distortion that appears usually in compressed video stream as abnormally large pixel blocks.

The standard FRUC method using BMA, which uses decoded frames as a reference, is a method that estimates the MV by tracking the movement of a block from the previous frame f_b to the current frame f_c. This method is further divided into two approaches. The first one, called Unidirectional Motion Estimation (ME) [7], is to create the MV of the to-be-generated frame from f_b using f_c as a reference frame, as shown in Fig. 1. In this approach, the visual difference between a block in the previous frame and the blocks within the search area of current frame is calculated and the smallest one is searched using Eqs. (1) and (2), respectively. Eq. (1) is the sum of absolute difference (SAD) value between the pixel brightness of the candidate block in f_b and the block in f_c. Eq. (2) represents the smallest vector among the SAD values and is designated to be the final motion vector $MV_{x,y}$. The vector (dx, dy) represents MV candidate, $S_{x,y}$ represents a set of MV candidates, and M represents the size of the search area.

$$SAD(dx, dy) = \sum_{x \in S_x} \sum_{y \in S_y} | f_c(x - dx, y - dy) - f_b(x, y) | \tag{1}$$

$$MV_{x,y} = \arg \min_{(dx,dy) \in S_{x,y}} \{SAD(dx, dy)\} \tag{2}$$

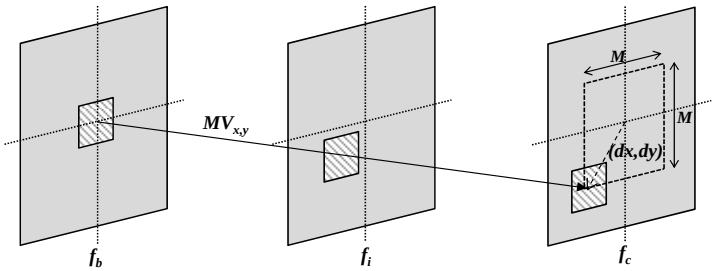

Fig. 1. Uni-directional motion estimation.

In unidirectional ME, interpolation is performed by using f_b and f_c to generate f_i. However, if the MV is incorrectly estimated, the interpolated block cannot be generated properly. Additionally, it may lead to a hole effect where parts of the image are not generated, or an overlapped effect where blocks overlap with each other, covering up the final image [10]. B.T. Choi [9] proposed a method to use bi-directional ME to resolve the hole effect. D. Vincent's method [11], when estimating the MV for unidirectional FRUC, considers the spatial and temporal connections between multiple frames. Their another method [12], rather than use bilateral vectors, employs two separate unidirectional vectors – one that points from the past frame to the current frame, and another

vector that points from the current frame to the past frame – to carry out MCI.

The second FRUC method using BMA is the bilateral ME [8] that finds two blocks at the symmetric position within the search area in the previous and current frames with minimum differences, as shown in Fig. 2. It is a method made to completely avoid the hole and overlap effects due to incorrectly estimated MVs when estimating with unidirectional ME, because the MV is estimated using the block's location at the interpolated frame. The MV is estimated by assuming the occurrence of an instantaneous symmetry in the movements of f_b and f_b central to the interpolated frame f_i. Eq. (3) is used to express the visual difference between two blocks during MV estimation, and through Eq. (4), the MV with the lowest difference is selected.

$$SBAD(dx,dy) = \sum_{x \in S_x} \sum_{y \in S_y} | f_b(x - dx, y - dy) - f_c(x + dx, y + dy) | \tag{3}$$

$$MV_{x,y} = \arg \min_{(dx,dy) \in S_{x,y}} \{SBAD(dx,dy)\} \tag{4}$$

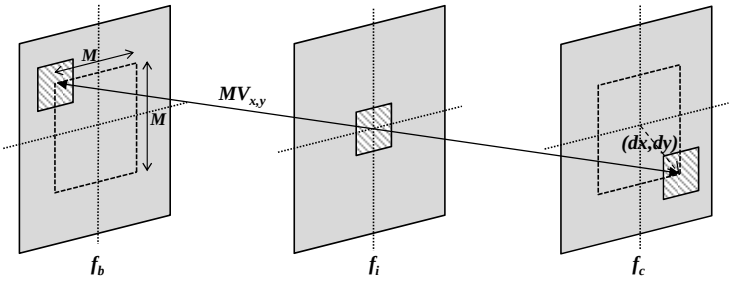

Fig. 2. Bilateral motion estimation.

Although this method does not produce the hole or overlapping effects, it may cause a block artifact problem because of the incorrectly estimated MV's [13]. To solve this new problem, OBMC (Overlapped Block Motion Compensation) [14, 15], a method that uses the area around the blocks, and a weighted matrix, which smooth out the brightness change, was proposed, However, this smoothing process led to motion blur occurring, which significantly lowered the image quality [16]. To resolve this problem, Suk-ju Kang proposed a method [17] which reduced the judder effects by generating another block in the overlapping area between the blocks in the interpolated frames. Additionally, Bo-Won Jeon *et al.* also proposed a method [18] to reduce motion blurring in bilateral-FRUC. However, these ME methods, either unidirectional ME or bilateral ME, are very hard to apply to mobile devices with limited computing power, because they are required to estimate the motion of all blocks in the previous or interpolated frames, which is a very time consuming process. An efficient algorithm to reduce the time for ME by reusing the MVs embedded in MC-DCT encoded video stream will be presented in the following section.

3. METHOD FOR CLASSIFYING USEFUL H.264 MV FOR FRUC

The process of generating the interpolated frame in the previous FRUC methods

required estimation of the motion of all blocks in the current or interpolated frames in an uncompressed form, which requires a large amount of computations. To resolve this problem, one could directly reuse the MVs embedded in MC-DCT compressed video stream to generate the interpolated frame because they are designed to point the most visually similar blocks in the reference frame. However, since the objectives of MV generated for video encoding and MV for FRUC are different from each other, the MV embedded in video stream could not directly be used for FRUC. The reason is that the basic principle of MC-DCT based video encoders such as H.264 is to find the most visually similar block (or macroblock) in the reference frame to the block in current frame, then to DCT-encode the visual differences between these two blocks. Since the primary goal of this encoding algorithm is to reduce the bitrate of the video stream, the blocks in the reference frame would be declared as most similar by the MV although it belongs to another meaningful moving object. It may help to reduce the overall bitrate of the encoded stream, but this MV could not be used for FRUC. FRUC using this MV could lead to a visually poor interpolated frame, because two semantically unrelated blocks are used to generate the block in the interpolation frame. Additionally, when the motion of the block is similar to the motion of adjacent blocks, MC-DCT encoder may generate the MV, called an approximated MV, in order to save the bits for the MV itself. This MV may point to the block that is not the most similar to the target block. It means that the MV's embedded in the encoded video stream should be first checked and modified to be reused for FRUC. Let us present the basic rules to detect these useful MV's among the MV's embedded in MC-DCT compressed video stream.

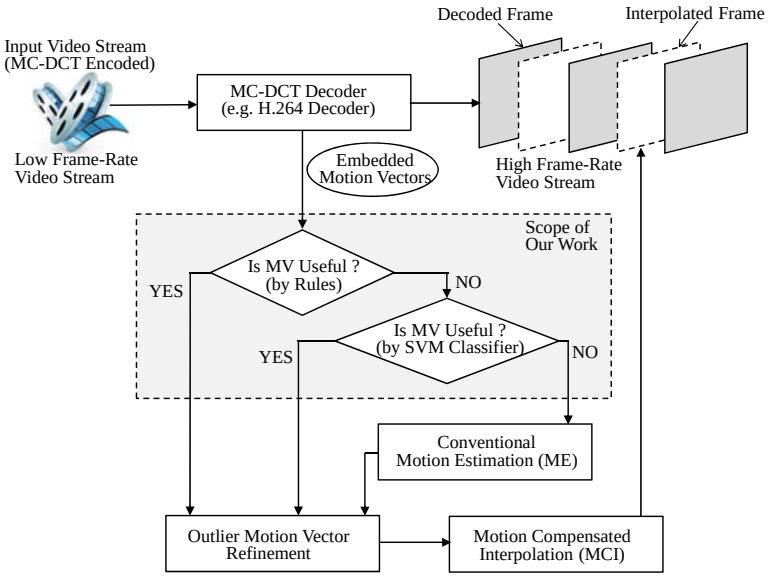

Fig. 3. Overall FRUC process with proposed useful MV classifying method.

Fig. 3 shows an overall structure of proposed FRUC algorithm proposed in this paper, in which the MV's embedded in the H.264 video stream (a MC-DCT encoded video

stream) are first classified using the proposed rules. If the MV is classified as useless, a conventional ME algorithm is applied to find the MV for FRUC. Otherwise, it is directly reused for FRUC without re-estimation. A MCI process is applied with these MVs after outlier MV's are removed, which are the same as in the previous FRUC methods based on Unidirectional ME.

The basic classifying rules in more detail are presented. Basically, among the MVs embedded in MC-DCT encoded stream, those useful in FRUC could be classified by comparing the visual similarities with 8 surrounding blocks in the current frame, the block at the same position in the reference frame, and the block pointed at by the MV, as shown in Fig. 4. Each of the three rules is presented.

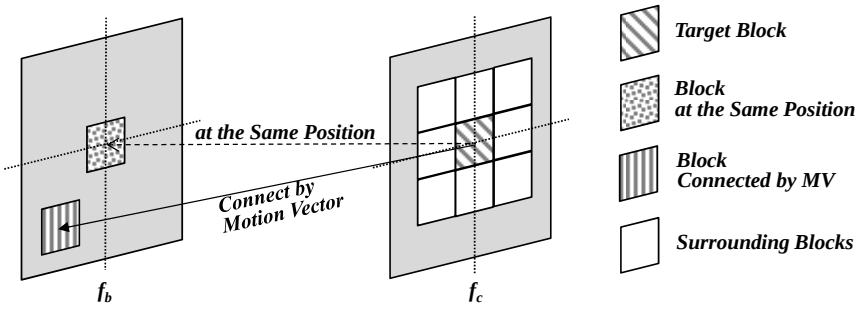

Fig. 4. 10 Blocks in current and reference frames for testing visual similarities.

Rule 1: Rule of using visual similarity of two blocks in the same position: If two blocks at the same position in the current and reference frames have a SAD value lower than a certain threshold ($Thr_{temporal}$), then regardless of the generated MV from MC-DCT encoder, the two blocks can be used to generate the interpolated frame block. This rule is very useful to detect the useless MV that is set to (0, 0) but two blocks at the same position are visually very different. The reason that this kind of MV is generated by MC-DCT encoder is to reduce the computations for the ME for this block. When the visual difference of two blocks at the same position in both the current and reference frames is larger than a certain threshold, the encoder gives up the ME process and sets the MV as (0, 0) because it guesses that similar blocks could not be found although the ME process is performed. This analysis led us to use the MV whose value is (0, 0) only when these two blocks are visually similar. This rule's condition and editing method of the MV are represented in Eq. (5), in which $Block(x, f_n)$ and $BMV(x, f_n)$ are the block and its MV at position x in frame f_n, respectively.

$$Rule\ 1: \begin{cases} Cond: SAD(Block(x, f_n), Block(x, f_{n-1})) \le Thr_{temporal} \\ Action: MV(x, f_n) = (0,0) \text{ and } MV(x, f_n) \text{ is } Useful \end{cases} \tag{5}$$

Rule 2: Rule of using visual similarity of two blocks connected by a MV: If the current frame's block and the reference frame's block connected by the MV have a SAD value lower than the certain threshold ($Thr_{temporal}$), the MV can be determined to be useful for FRUC. These two blocks then can be used to interpolate the interpolated frame. This rule is to detect the approximated MV that is generated by the encoder to save the

bits for the MV itself when the block's MV is similar to the neighboring blocks' MV. This MV may point to the block in the reference frame that is not similar enough. Since this MV is useless for FRUC, the MV is classified as a useful one only when the blocks connected by the MV are visually similar enough. This rule's condition is represented in Eq. (6).

$$Rule\ 2: \begin{cases} Cond: SAD(Block(x, f_n), Block(x - MV(x, f_n), f_{n-1})) \le Thr_{temporal} \\ Action: MV(x, f_n) = (0,0) \text{ and } MV(x, f_n) \text{ is } Useful \end{cases} \tag{6}$$

Rule 3: Rule of using visual similarity of blocks within the same object: For any frame, the blocks belonging to the same object should have MVs in the same direction. If the SAD between the block of interest in the current frame and its adjacent neighboring blocks is below a certain threshold ($Thr_{temporal}$), or in other words, if there are adjacent blocks that are extremely visually similar, then the adjacent blocks and current blocks are regarded as belonging to the same object and set the MV of current block to the MV of the most similar neighboring block. Of course, in this case the new motion vector is classified as useful for FRUC. Eq. (7) shows this classifying and modification rule, where $N(x)$ represents a set of 8 neighboring blocks as shown in Fig. 4.

$$Rule\ 3: \begin{cases} Cond: SAD(Block(x, f_n), Block(x', f_n)) \le Thr_{spatial} \\ \text{where } x' = \arg\min_{x' \in N(x)} SAD(Block(x, f_n), Block(x', f_n)) \\ Action: MV(x, f_n) = MV(x', f_n) \text{ and } MV(x, f_n) \text{ is } Useful \end{cases} \tag{7}$$

SVM Classifier: Classifying useful MVs through SVM learning: For the MVs whose usefulness could not be determined with above deterministic three rules, a learning mechanism using the visual characteristics of the 10 blocks shown in Fig. 4 could be used to classify useful MVs. In the proposed classifying method, a SVM (Support Vector Machine) is used to finally determine the usefulness of MV's. It is trained with a lot of relationships between visual characteristics of 10 blocks when it is useful for FRUC, and the training set is extracted from various video streams with different motion energies. Note that, this SVM classifier can be solely used for classifying the usefulness of all MVs since it is a superset of above three classifying rules. However, since there are a lot of MV's in MC-DCT encoded video stream and classifying MV with SVM requires some non-neglectable computations, classifying all MVs with SVM would require a lot of computational resources. Hence this classifier works only with remaining MV's that could not be classified using the previous three deterministic rules. Fig. 5 shows how to classify useful MVs in FRUC, and Fig. 6 shows how to detect outlier MVs in FRUC in detail. In these algorithm ms $Pos(f_n)$ represents a set of positions of all blocks in f_n, and $N(x)$ represents a set of positions of 8 neighboring blocks of $Block(x, f_n)$.

Algorithm 1: DUMP (Detect Useful Motion vector Processing) Algorithm
for each $x \in Pos(f_n)$ do /* for each block in frame */
  /* Check the visual similarity of two blocks in the same position */
  if ($SAD(Block(x, f_n), Block(x, f_{n-1})) \le Thr_{temporal}$))

```
    then {MV(x, f_n) = (0, 0); MV(x, f_n) is Useful}
    else /* Check the visual similarity of two blocks connected by motion vector */
      if (SAD(Block(x, f_n), Block(x − MV(x, f_n), f_n-1)) ≤ Thr_temporal)
        then MV(x, f_n) is Useful
        else /* Check the visual similarity between neighboring blocks */
          x′ = arg min_{x′∈N(x)} SAD(Block(x, f_n), Block(x′, f_n));
          if (SAD(Block(x, f_n), Block(x′, f_n)) ≤ Thr_spatial)
            then {MV(x, f_n) = MV(x′, f_n); MV(x′, f_n) is Useful}
            else /* Check with SVM Classifier */
              if (SVM_classification(MV(x, f_n)))
                then MV(x, f_n) is useful
                else MV(x, f_n) is not useful
              end if
          end if
      end if
  end if
  /* Outlier Detection */
  if (Outlier Detection(x, f_n)) then MV(x, f_n) is not useful
end for
```

Fig. 5. Proposed DUMP (Detect Useful Motion vector Processing) algorithm.

Algorithm 2: Outlier Detection Algorithm

function OutlierDetection(x, f_n)

$o(x) = \phi$;

For each $x' \in N(x)$ **do** /* for each neighboring block */

  if ($SAD(Block(x, f_n), Block(x', f_n)) \le Thr_{spatial}$)

    /* $Block(x', f_n)$ is in same object with $Block(x, f_n)$ */

    $o(x) \leftarrow o(x) \cup \{x'\}$;

  end if

end for

$MV_{mean} = (\sum_{i=0}^{|o(x)|} MV_{i \in o(x)}(i, f_n)) / |o(x)|$;

if ($MV(x, f_n) - MV_{mean} > Thr_{outlier}$) /* $Thr_{outlier}$ is a threshold for detecting outlier MV*/

  then return *true*;

  else return *false*;

end if

Fig. 6. Outlier MVs detection algorithm.

4. EXPERIMENTAL RESULTS

In this section, the algorithms presented in Section 3 are experimented and analyzed. This experiment used the FFMPEG Library [19], the currently most used MC-DCT based encoder/decoder, to encode/decode the video stream, and LibSVM [20], a SDK for developing classifier based on SVM, to train and classify the useful MV.

4.1 Experimented Video Streams and Evaluation Criteria

To investigate the possibilities of applying the proposed FRUC algorithm to different video stream content, a variety of videos, such as high-motion videos, low-motion videos, 3D game playing videos, PC desktop application (word processing, Web surfing) screen shot videos, and video conferencing videos were experimented with and analyzed. The tested videos used for experiments with their motion energies are presented in Table 1, in which the motion energy is computed by averaging the difference in frames in every frame of the video.

In our experiments, the precision and recall factors, two evaluation methods often used in the fields of computer learning and pattern recognition, were used to evaluate the performance of the proposed classifying method. An ideal set of MVs (or ground truth) for FRUC could be obtained by the method shown in Fig. 7. As exemplified by Fig. 7, for a given block in the current frame, all possible interpolated blocks are first generated by using all possible MV's from the target block to the blocks in the search range of the reference frame. Among these MVs, the one that can generate an interpolated block with the lowest SAD to the block at the same position in the skipped frame is an ideal MV for FRUC. It is an ideal one because it can generate the most similar interpolated block with the blocks in the current and reference frames. If the MV embedded in the video stream is the same as this ideal MV, then it should be classified as the useful MV with our rules. Otherwise, it should be classified as useless. Using this ground truth of classification, the recall and precision of our rules are tested. The training set for the SVM classifier is col-

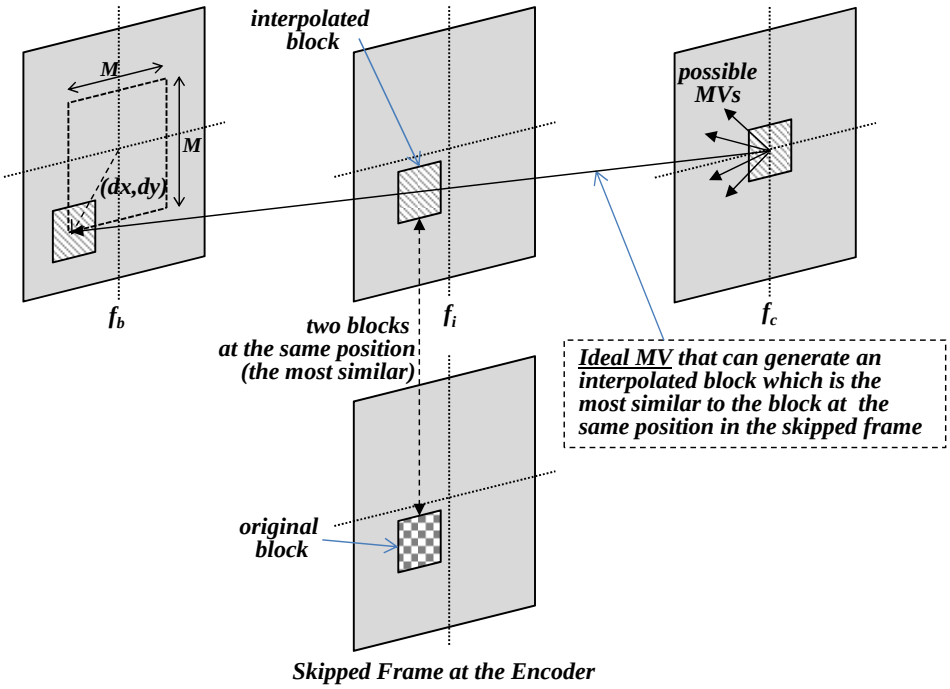

Fig. 7. Example of ideal MVs.

lected from various video streams encoded with FFMPEG. For each video frame, 10 degrees of features (as explained in Fig. 4) were extracted, and a total of approximately 200,000 MVs with ground truths are used to train the SVM. Non-linear Radial Basis Functions (RBF) kernel and sigmoid kernel are used to train the SVM.

4.2 Experiments and Analyses

4.2.1 Experiments on performance of classifying algorithm

To evaluate the DUMP rule presented in Fig. 5, the precision and recall of the DUMP rule were experimented with the video streams presented in Table 1. Table 2 shows the precisions and recalls of our classifying algorithm at each threshold value: Thr8 ($Thr_{temporal} = Thr_{spatial} = 512$), Thr2 ($Thr_{temporal} = Thr_{spatial} = 128$), Thr0 ($Thr_{temporal} = Thr_{spatial} = 0$). As shown in these experiments, as the threshold value becomes smaller, the MVs are evaluated much more rigorously, which results in a decrease of the recall factor. In our experiments, although the recall is decreased by lowering the threshold value, the precision factor is not changed so much. The reason is that the MVs that are classified as useless by the three rules are re-tested by our SVM classifier so that the precision of the proposed algorithm is governed mainly by the SVM classifier. Also note that the recall/precision factors of the proposed algorithm are relatively high when there is less motion energy in the video stream because, in this case, the abnormal MVs (or useless MVs) are not generated by MC-DCT encoders and they are easily classified by the three static rules.

Table 1. List of experimented video streams with their motion energy[+].

Video Files	Motion Energy
Run1 (High Motion Energy)	11.081
3D Game (High Motion Energy)	8.159
Run2 (Low Motion Energy)	2.075
VideoPhone3 (Very Low Motion Energy)	1.025
Desktop (Extremely Low Motion Energy)	0.112

[+] The term, “high motion energy” and “low motion energy”, are used to emphasize the relative amount of motion energy of tested video compared to others.

Table 2. Precision and recalls of DUMP rule for each threshold.

	Recall (%)			Precision (%)		
	Thr8	Thr2	Thr0	Thr8	Thr2	Thr0
Desktop	99.8	99.3	99.0	99.8	99.8	99.8
VideoPhone3	89.9	96.6	81.0	98.4	98.3	97.9
3DGame	99.1	92.4	84.1	76.2	75.5	74.9
Run2	99.5	90.2	72.7	97.6	97.3	96.9
Run1	91.0	83.7	79.3	85.8	84.9	83.9

To test the validity of the three deterministic rules and a statistical SVM classifier, the ratios of useful MVs that are classified by each rule and SVM classifier are experimented. The results are summarized in Fig. 8. For videos with a low motion energy (*e.g.*

desktop application videos), the second rule caused most of the MVs to be classified as useful, whereas in videos with a high motion energy (*e.g.* 3D game videos) or with low thresholds, most of the cases were determined by the SVM classifier. The test results imply that in videos with higher motion energy than the average (*e.g.*, *3DGame*(*Thr0*) or *Run1*(*Thr0*)), it becomes much more difficult to find blocks that are similar to the reference frame at encoder. Thus, the SAD between two blocks pointed at by the MVs and the difference in MVs among adjacent blocks become large. In this case, the three deterministic rules could not be used to classify the usefulness of MVs, and the SVM classifier is primarily used for classification. Also note that the intra-coded blocks without MV are classified as useless in the proposed algorithm.

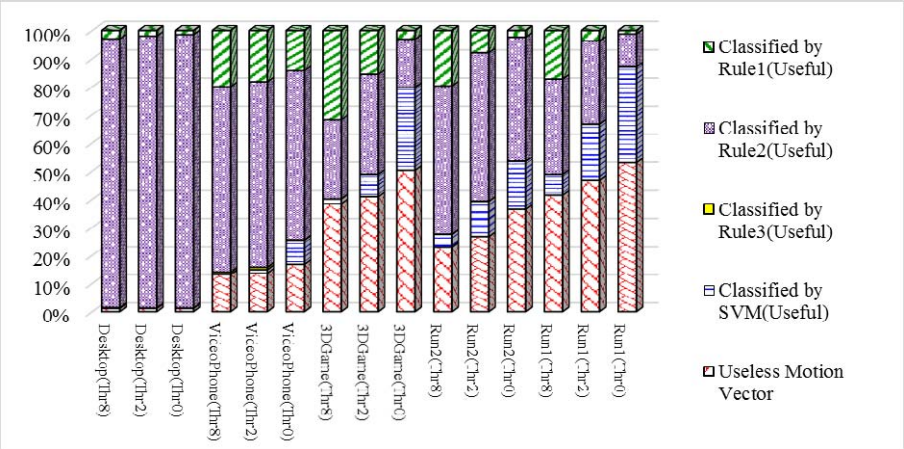

Fig. 8. Percentages of useless and useful MVs classified with each rule and SVM.

4.2.2 Time complexity and experimental FRUC time

As explained in the section 2, the previous FRUC methods estimate the MVs for all blocks in a current or interpolated frame, whereas the proposed FRUC method estimates the MVs that are classified as useless ones. The time complexities of estimating the MVs in the previous FRUC and proposed FRUC can be presented as Eqs. (8) and (9), respectively. In these equations, M represents the area of the search region, l is the block's width and height, P represents the number of blocks in each frame, O(SVM) is the temporal complexity of using SVM, K represents the number of times SVM is used for MV classification, and α represents the ratio of MVs that are classified as useless or intra-coded;

$$\text{Searching MVs in Previous FRUC: } O(M^2 * l^2 * P) \quad (8)$$

$$\text{Searching MVs in Our FRUC: } O(l^2 * 10 * P * SVM * K * + (M^2 * l^2) * (P * \alpha)) \quad (9)$$

The expected speedup of proposed method could be formalized as follows:

$$SpeedUp = \frac{O(M^2 * l^2 * P)}{O(l^2 * 10 * P * SVM * K + (M^2 * l^2) * (P * a)} \quad (10)$$

useless MVs (α) and the number of MVs (K) that are classified with *SVM*. If $K \approx 0$, the expected speedup would be $O(M^2 * l^2 * P)/O(l^2 * 10 * P * SVK * K * + (M^2 * l^2) * (P * \alpha) \approx 1/\alpha$. Furthermore if $K \approx 0$ and $\alpha \approx 0$, the expected speedup would be $O(M^2 * l^2 * P)/O(l^2 * 10 * P) \approx M^2/10$.

Experimental MV searching time for FRUC are presented in Figs. 9 and 10, in which the test video streams with high motion energy (*Run1*) and low motion energy (*VideoPhone3*) are experimented, respectively. Note that in Figs. 9 and 10, "Full SVM" is the time needed to classify all MVs with the SVM classifier. From these experiments, we can find out that, as the threshold value is smaller, the number of times the SVM classifier is used increases, which correspondingly negatively affects the processing speed. Comparing Figs. 9 and 10, one can see that when there is higher motion energy in the video stream, the number of useful MVs decreases. This conversely increases the number of times SVM is called, which also causes negative effects to processing speed. The reason behind these results is that in videos with high motion energy, objects and backgrounds that disappear and reappear, or faces and body parts that rotate, are common, which forces MC-DCT encoder produce MVs that are abnormal and useless for FRUC. Our method can estimate the MVs for FRUC about 2.2 ($0.703/0.316 \approx 2.2$) times faster than the conventional method in the case of the *Run1* video stream as shown in Fig. 9, and about 700 times faster ($0.703/0.001 \approx 700$) in the case of *VideoPhone3*, as shown in Fig. 10, when thresholds are *Thr*8.

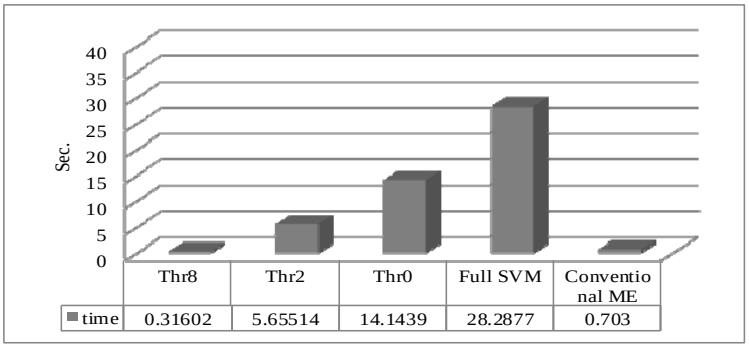

Fig. 9. MV searching time for each DUMP threshold value and comparison to the conventional ME method ("*Run1*" video stream).

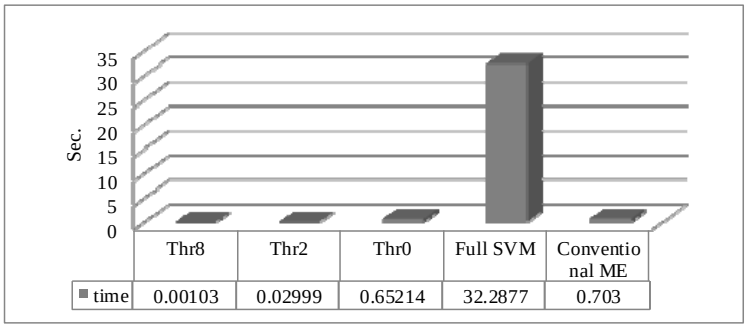

Fig. 10. MV searching time for each DUMP threshold value and comparison to the conventional ME method ("*VideoPhone3*" video stream).

4.2.3 Experiments on interpolated frame quality

After the MVs for all blocks are estimated as in the previous FRUC methods or classified/re-estimated in the proposed method, these motion vectors are used to generate the blocks in the interpolated frame. Fig. 11 shows an example of four frames that are interpolated with MVs that are classified/re-estimated with the proposed method (Fig. 11 (a)), with only MVs that are embedded in MC-DCT encoded video stream (Fig. 11 (c)), with MVs that are estimated with uni-directional ME (Fig. 11 (b)) and bilateral ME (Fig. 11 (d)). As shown in Fig. 11 (c), FRUC using MVs embedded in MC-DCT encoded frame directly could not guarantee a high quality interpolated frame, because they are primarily generated to reduce the overall bitrate. For the cases using Uni-directional ME (Fig. 12 (b)) or Bilateral ME (Fig. 12 (d)), the MVs are searched for all blocks, so a relatively acceptable frame quality can be achieved[1]. However, since the MVs for all blocks in the current or interpolated frames have to be found again for FRUC in these approaches, their processing require a lot of computation. To solve this problem, this paper proposed a method to reduce the MV searching time by selectively reusing the MV embedded in MC-DCT encoded video stream. As a result, the interpolated frame generated with the proposed method maintains a similar image quality compared to the Unidirectional ME, but can reduce the FRUC processing time remarkably.

(a) FRUC using our algorithm.

(b) FRUC using Uni-directional ME.

(c) FRUC using embedded H.264 MV.

(b) FRUC using bilateral ME.

Fig. 11. Example frames created by each MV estimation/classify method.

Fig. 12 presents an experiment on the amount of time taken to find the MV[2] and PSNR value for the interpolated frame generated by the Run1 video after having each

[1] To solve the hole and overlap effects in the uni-directional ME and proposed method, the previously proposed method [12].

[2] The time is the average motion estimation time per frame. Since an exhaustive motion estimation method is used in our experiments to find the MVs in previous FRUC and our FRUC algorithm, they are somewhat larger than expected. However, since the purpose of our experiments is to know the relative time for FRUC, these experiments are still valid to show the relative elapsed time for FRUC.

method applied. In this experiment, since the ratio of MVs that are useful for FRUC is about 60%, the MVs for a large number of blocks (around 40%) must be found again. For this reason, the amount of time required to find the MVs of the proposed method is the midpoint between the case for when H.264 MVs are directly used and when all MVs are searched again, as shown in Fig. 12. Taking the opposite situation, videos such as *VideoPhone3* with very low motion energy have a ratio of MVs useful for FRUC that is very large (approximately 98%). Hence, there is no need for additional time to find the MVs, as shown in Fig. 13. Basically, the blocking effects caused by the proposed method and the traditional approaches are exactly the same, because our algorithm is only to reduce the time for ME for blocks that have MVs generated by encoder but still useful for FRUC. That is the reason why the PSNR values of interpolated frames by our FRUC algorithm and Uni-directional FRUC algorithm are almost the same in Figs. 12 and 13. Based on such experiments, this paper concludes that by using the proposed MV classifying method, the interpolated frame could be very quickly generated at a similar quality as the conventional FRUC methods.

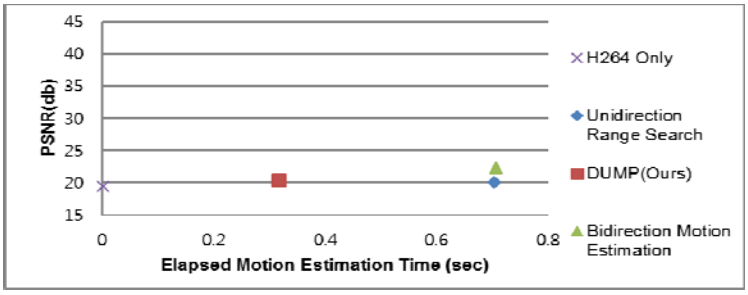

Fig. 12. PSNR and elapsed time for each MV estimation method (*Run*1 video, motion energy: 11.081).

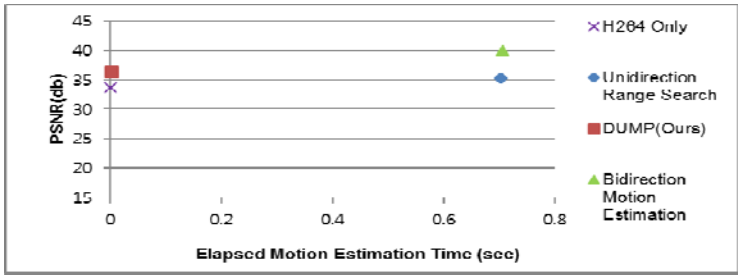

Fig. 13. PSNR and elapsed time for each MV estimation method (*VideoPhone*3 video, motion energy: 1.025).

5. CONCLUSION

Frame Rate Up Conversion (FRUC) is a technique that inserts an interpolated frame at the decoder side to improve the perceptual quality of video streams that are originally encoded in a low frame rate. Previously proposed FRUC methods generated the interpo-

lated frame by estimating the MVs for all blocks in the current frame or interpolated frame. However, since this motion estimation process for all blocks is a time consuming task, it is very difficult to be applied to mobile devices such as mobile phones. This paper proposes a method to reduce the time for motion estimation of blocks by reusing the MVs embedded in MC-DCT encoded video stream. Using three rules and a pre-trained SVM classifier, the useful MVs for FRUC can be classified with an accuracy of 0.72~0.99. Since the motion estimation for these MVs could be skipped in a FRUC process, the total FRUC time could be reduced dramatically without image quality degradation. According to experiments with videos with a varied range of motion energy, the FRUC using the method proposed in this paper resulted in a similar image quality as the conventional methods, while reducing the time taken to generate the interpolated frame by 50% to 99%. Especially for videos with low motion energy, the experiments demonstrated that a video with reasonable image quality could be generated without additional MV searches. The proposed FRUC method could be easily applied to the decoders on mobile phones in order to improve the visual quality of video streams encoded in a low frame rate.

REFERENCES

1. K. Hsu, "Hardware architecture design of frame rate up-conversion for high definition videos with global motion estimation and compensation," in P*roceedings of IEEE Workshop on Signal Processing Systems*, 2011, pp. 90-95.
2. B. D. Choi, J. W. Han, C. S. Kim, and S. J. Ko, "Motion compensated frame interpolation using bilateral motion estimation and adaptive overlapped block motion compensation," *IEEE Transactions on Circuits and System for Video Technology*, Vol. 17, 2007, pp. 407-416.
3. S. H. Lee, Y.-C. Shin, S. Yang, H.-H. Moon, and R.-H. Park, "Adaptive motion-compensated interpolation for frame rate up-conversion," *IEEE Transactions on Consumer Electronics*, Vol. 48, 2002, pp. 444-450.
4. H. Blume, "Nonlinear vector error tolerant interpolation of intermediate video images by weighted medians," *Signal Processing-Image Communication*, Vol. 14, 1999, pp. 851-868.
5. O. A. Ojo and G. de Haan, "Robust motion compensated video up conversion," *IEEE Transactions Consumer Electronics*, Vol. 43, 1997, pp. 1045-1056.
6. H. S. Kang, *The Research for the Caption Processing in the Frame Rate Up-Conversion*, Master Thesis, Hanyang University, Seoul, Korea, 2011.
7. J. E. Santos Conde, A. Teuner, and B. J. Hosticka, "Hierarchical locally adaptive multi-grid motion estimation for surveillance applications," in *Proceedings of IEEE International Conference on Acoustics, Speech, Signal Process*, Vol. 6, 1999, pp. 3365-3368.
8. S. H. Lee, O. Kwon, and R. H. Park, "Weighted-adaptive motion-compensated frame rate up-conversion," *IEEE Transactions Consumer Electronics*, Vol. 49, 2003, pp. 485-492.
9. B. T. Choi, S. H. Lee, and S. J. Ko, "New frame rate up-conversion using bi-directional motion estimation," *IEEE Transactions Consumer Electronics*, Vol. 46, 2000,

pp. 603-609.
10. B. Choi, J. Han, and C. Kim, "Frame rate up-conversion using perspective transform," *IEEE Transactions on Consumer Electronics*, Vol. 52, 2006, pp. 975-982.
11. D. Vincent, A. Blanchfield, P. Klepko, and R. Commun, "Motion compensated frame rate up-conversion Part I: Fast multi-frame motion estimation," *IEEE Transactions on Broadcasting*, Vol. 56, 2010, pp. 133-141.
12. D. Vincent, A. Blanchfield, P. Klepko, and R. Commun, "Motion compensated frame rate up-conversion Part II: New algorithms for frame interpolation," *IEEE Transactions on Broadcasting*, Vol. 56, 2010, pp. 142-149.
13. T. Chen, "Adaptive temporal interpolation using bidirectional motion estimation and compensation," in *Proceedings of International Conference on Image Processing*, Vol. 2, 2002, pp. 313-316.
14. Y. Chen, "Parametric OBMC for pixel-adaptive temporal prediction on irregular motion sampling grids," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 22, 2012, pp. 113-127.
15. P. Ishwar and P. Moulin, "Switched control grid interpolation for motion compensated video coding," in *Proceedings of IEEE Conference on Image Processing*, Vol. 3, 1997, pp. 650-653.
16. G. Heising, D. Marpe, H. L. Cycon, and A. P. Petukhov, "Wavelet-based very low bit-rate video coding using image wrapping and overlapped block motion compensation," in *Proceedings of IEEE Visual Image Signal Processing*, Vol. 148, 2001, pp. 93-101.
17. S. J. Kang, D. G. Yoo, S. K. Lee, and Y. H. Kim, "Multiframe-based bilateral motion estimation with emphasis on stationary caption processing for frame rate up-conversion," *IEEE Transactions on Consumer Electronics*, Vol. 54, 2008, pp. 1830-1838.
18. B. Jeon, G. Lee, S. Lee, and R. Park, "Coarse-to-fine-frame interpolation for frame rate up conversion using pyramid structure," *IEEE Transactions on Consumer Electronics*, Vol. 49, 2003, pp. 499-508.
19. FFMpeg Team, *FFmpeg Version 0.11 Beta*, http://www.ffmpeg.org/, 2012.
20. Willow Garage Inc., *OpenCV: Version 2.3 Beta*, http://tech.groups.yahoo.com/group/opencv/, 2012.

JongHo Nang (浪鍾鎬) received his Ph.D. and M.S. degrees in Computer Science from Korea Advanced Institute of Science and Technology (KAIST), Daejon, Korea, in 1992 and 1988, respectively, and his B.S. degree in Computer Science from Sogang University, Seoul, Korea, in 1986. He has been a Professor of Computer Science and Engineering Department, Sogang University since 1993. His research interests include multimedia system, parallel processing, and internet technology.

Sangchul Kim (金相哲) received the B.S degree from Seokyung University, Seoul, Korea, in 2010, and the M.S. degree in Computer Science from Sogang University, Seoul, Korea, in 2012. He is currently working toward his Ph.D. degree in Computer Science at the same university. His research interests include multimedia system, contents based multimedia retrieval and object recognition.

Hyuk-Jun Lee (李赫濬) received the B.S. degree in Comuter Science and Engineering from the University of Southern California, Los Angeles, in 1993 and the M.S. and Ph.D. degrees in Electrical Engineering from Stanford University, Stanford, CA, in 1995 and 2001, respectively. From 2001 to 2011, he was a Senior Engineer at Cisco Systems, San Jose, CA, USA. He is currently an Assistant Professor with Computer Science and Engineering Department, Sogang University, Seoul, Korea. His research interests include computer architecture, embedded systems, parallel and distributed systems.