Adaptive-Motion-Detector-Based Skip-Mode Predecision in Motion Estimation for Video Surveillance

Jia Su and Takeshi Ikenaga

Graduate school of Information, Production, System, Waseda University
2-7 Hibitino, Wakamatsu-ku, Kitakyushu-shi, Fukuoka 808-0135, Japan
E-mail: selene@suou.waseda.jp, ikenaga@waseda.jp

Abstract  H.264/AVC often leads to high computational complexity, which induces high power dissipation when a tremendous number of cameras is installed to build a large-scale surveillance system. Moreover, there is much background video compression redundancy in the surveillance scene. To solve this problem from both computer vision and video compression points of view, we propose a multi-objective optimized motion block detector for skip-mode predetermination in motion estimation H.264/AVC. The proposed algorithm contains two stages: multi-objective-optimization-based motion block detector and an application in the skip-mode decision of the fast motion estimation algorithm for video surveillance scenes. In the prestige, a motion detector with an enhanced difference filter and weighted erosion filter was designed, in which multi-objective optimization was introduced for automatically determining the weights for each filter. For the uninterested static macroblock, a low-bit-rate skip mode can be directly chosen as the best mode while performing motion estimation. Compared with the statistical and knowledge-based object detector (SAKBOT), the proposed motion detector can achieve a time saving of more than 40% on average, while attaining a high accuracy for most indoor and outdoor surveillance test sequences. Compared with the UMHexagonS fast block matching method of JM11.0, it can achieve a time saving of 23%-41% in motion estimation (ME) and higher detection accuracy for uncompressed surveillance videos.

Keywords: multi-objective optimization, motion detection, video surveillance

1. Introduction

Video surveillance has been a popular security tool for years. Banks, retail stores, and countless other end users depend on the protection provided by video surveillance. During the last few years, it has been proved that algorithms for wireless video surveillance must be as power conscious as possible [1]-[3]. The most popular method for motion detection is based on the frame difference; basically the absolute difference between a reference frame and the frame under test is computed. A lot of variants of this method were proposed. In [4], Toyama et al. proposed a statistically adaptive update scheme for the reference frame. Jin et al demonstrated the effectiveness of the dynamically chosen threshold [5]. These algorithms alone are weak against light changes. Recently, Matsushita et al. [6] proposed a preprocessing filter that is able to enhance algorithm robustness. However, the complexity of these algorithms can be decreased much more.

An important issue in computer vision is the determination of weights for multiple objective function optimizations. This problem arises naturally in many reconstruction problems, where one wishes to reconstruct a function belonging to a constrained class of signals based upon noisy observed data. There is usually a trade-off between reconstructing a function that is true to the data, and constructing one that is true to the constraints. Instances of this tradeoff can be found in constructing shape from shading [7], optical flow [8], surface interpolation [9], edge detection [10], visible surface reconstruction [11], and brightness-based stereo matching [12]. The recently popularized regularization method for solving ill-posed problems [13], [14] always requires the tradeoff of conflicting requirements. The basic framework defines a cost or error functional that reflects the “badness” of a proposed solution to the reconstruction problem. Mathematical techniques, such as the calculation of variation is used to find the best solution to the reconstruction problem.
The contribution of each constraint to the cost functional is weighted, and the weights may be adjusted to achieve the desired trade-off. In some cases, a priori knowledge may be used to determine the "best" set of weights. Typically, one must know some property of the observed data, such as the signal-to-noise ratio, before proceeding. However, it is not practical. Thus, a method for determining weights that does not depend on prior knowledge is proposed. Motion estimation (ME) is one of the most important parts of a video encoder. It is used to reduce the temporal redundancy in video sequences by reducing the amount of data of the encoded video. However, it is also one of the most computationally intensive blocks. In the H.264/AVC coding standard, it represents up to 80% of the computations of the whole set. Many fast ME algorithms are proposed to reduce the encoding time of the ME process, of some which (UMHS [15]-[18] and simplified UMHS [19] and EPZS [20]) are very successful: more than 90% of ME time is removed from full search (FS) while still maintaining very good rate-distortion (R-D) performance. However, even these fast algorithms still require a significant amount of computations and little contribution to bit-rate reduction. Often, they do not consider the feature of the video surveillance system. As a consequence, fast ME algorithms can be combined with motion detection algorithms that make use of search range decision techniques. The aim of these techniques is to separate the motion blocks from the inputs, and mainly focus on the part of interest to achieve computation saving for the video surveillance system. The motion-detection-based fast motion estimation algorithm is proposed in this paper; it is particularly suited to be used in the H.264/AVC standard for video surveillance systems. This algorithm speeds up the ME procedure compared with other algorithms, while still maintaining its coding quality performance of both the obtained bit-rate and the subjective performance.

The rest of the paper is organized as follows. The next section overviews the proposed adaptive-motion-detection-based skip-mode prediction in motion estimation for video surveillance. Section 3 deals with an adaptive-motion-block-detector, which consists of the proposed enhanced difference filter, weighted erosion filter, and linear square optimized threshold filter. Section 4 illustrates skip-mode detection for motion estimation algorithms. Section 5 presents the experimental results, and the last section concludes the paper.

2. Overview of Proposed Algorithms

The proposed idea combines motion block detection with fast motion estimation. The following section presents a detailed description of motion block detection. Then, how the proposed motion block detection acts on the motion estimation part will be illustrated. There exist previous works that are designed using condition-based skip-mode selection, flown by other mode selection algorithms for fast mode selection [21]-[24]. However, our proposal is mainly an adaptive motion detector for directing the mode selection, which means that by using the mathematical optimum, our proposed motion detector does not need a priori knowledge to achieve an adaptive balanced result in a real-time video surveillance system. Figure 1 displays the whole algorithm system reported in this paper. The input sequence is the raw data to be compressed, while the motion detector is aimed at detecting the motion macroblocks (MBs). In order not to add additional computation, the simplest motion-detection algorithm was chosen to detect the interesting point. For environment modeling, the threshold filter was used to ignore insignificant colors and the difference filter replaced the motion segmentation.
module; the erosion filter is particularly useful for binary image processing, where it removes pixels that are not surrounded by specified number of neighbors. It gives the ability to remove noisy pixels (stand-alone pixels) or shrink objects. Finally, the proposed motion block detector was set up below. After that, motion blocks are transmitted to the fast motion estimation algorithm which integrated to JM software, while the static blocks were set to skip mode. Video analysis algorithms can sustain a significant degradation in the input video quality without a decrease in their accuracy. This finding leads to the problem that prior knowledge also has a tradeoff between bit rate and accuracy. Our approach to the solution of this problem is to focus on video features that affect the performance of the algorithm. It can yield satisfactory results without prior knowledge. After a multi-objective optimal is obtained within several seconds of calculation, the detection result can reach a stable balance, which can be reflected in the fitness score graphs. By studying how a video adapts, where in video bit-rate is reduced and the video features are degraded, the rate accuracy tradeoff can be estimated.

3. Proposed Adaptive Motion Block Detector

In H.264, the unit of processing is MBs. In order to link the motion-detection algorithm to the motion-estimation part, the motion-detection algorithm is used here to detect motion blocks, which can be recognized as a foreground or interested block. The motion detection embedded in the SAKBOT [25] system is based on background subtraction and models the background using statistics and knowledge-based assumptions. In fact, the background model is computed frame by frame using a statistical function (temporal median) and taking into account the knowledge acquired about the scene in previous frames. In practice, the background model is updated differently if the considered pixel belongs to a previously detected moving visual object MVO; in this case, the background model is kept unchanged because the current value does not describe the background. Moreover, if an object is detected as “stopped” (i.e., the tracking system detects that it was moving but is now stationary) for more than a “time-out” number of frames, its pixels are directly inserted into the background, without using the statistics.

Figure 2 displays the block diagram of motion-block detection. It should be mentioned that, different from other motion detection algorithms, the input video sequence is the raw data to be compressed. Video analysis algorithms can sustain a significant degradation in the input video quality without a decrease in their accuracy. This finding leads to the problem of determining the tradeoff between video bit-rate and the accuracy of a given algorithm. Our approach to the solution of the problem is to focus on video features that affect the performance of the algorithm. By studying how a video adaptation, which reduces the video bit-rate, degrades video features, the rate accuracy tradeoff can be estimated [26]-[28]. Moreover, by multi-objective optimization (MOO) and linear square (LS) least optimization, 2 conditions for switching among different combinations of filters have been derived. The details on how to decide the conditions are described in Sec. 3.1 and Sec. 3.2 respectively. Finally, the proposed motion-block detector was set up below.

First of all, the regions where these two frames are differ slightly are detected. For this purpose, difference and threshold filters can be utilized. To remove random noisy pixels, the erosion filter was chosen. The difference filter takes two images (reference
and current images) of the same size and pixel format and produces one image, where each pixel equals the absolute difference between corresponding pixels of the provided images. The filter accepts 8 and 16 bpp (bits per pixel) gray-scale images for processing. The threshold filter performs image binarization using a specified threshold value. All pixels with intensities equal to or higher than the threshold value are converted to white pixels. All other pixels with intensities below the threshold value are converted to black pixels. In order to simplify the model, only gray level images were considered to obtain the motion blocks.

The use of at threshold filter is a technique that helps us to “delete” unwanted pixels from an image and concentrate only on the ones we want. For each pixel in an image, if the pixel value is above a certain threshold, it is converted to 255 (white); otherwise, it is converted to 0 (black). The blob counter is a very useful feature and can be applied in many different applications. It can count objects in a binary image and extract them.

The idea comes from “Connected components labeling”, a filter that colors each separate object with a different color. By utilizing the blob counter, we can obtain the number of objects, their position and the dimension on a binary image.

The proposed idea utilizes both linear and nonlinear weight constraint in different motion detector parts. First, an optimized divided difference filter (DDF) and weighted erosion filter are described in the next subsection; after that, the proposed linear square optimized threshold filter is described in detail.

3.1 Enhanced difference filter and weighted erosion filter

The comparison of the current frame with the first one would reveals the moving object independent of its motion speed, if there are no objects in the initial frame. The motion can be detected on the place where the moving object was. Therefore, the initial frame can be renewed sometimes, but it still will not give satisfactory results in cases where the first frame contains only static background. Thus, an inverse situation occurs, for example, a picture is put on the wall in the room. The motion detected from the initial frame should be renewed. Then the following condition is derived.

Consider the general discrete-time nonlinear system [29]

\[
s_{k+1} = f(s_k, u_k) \\
t_k = h(s_k, v_k)
\]

where \( s_k \in \mathbb{R}^n \) the \( n \times 1 \) state vector, \( t_k \in \mathbb{R}^m \) is the measurement vector, \( u_k \in \mathbb{R}^q \) the \( q \times 1 \) state noise process vector, and \( v_k \in \mathbb{R}^r \) the \( r \times 1 \) measurement noise vector. It is assumed that the noise vectors are uncorrelated white Gaussian noise processed with expected means and covariance.

\[
E\{w_k\} = \bar{w} \\
E\{(w_k - \bar{w})(w_j - \bar{w})\} = Q_k
\]

In this paper, because the measurement equation is linear, the nonlinear system in the divided difference filter [30] has been transformed to a linear system, so the proposed enhanced difference filter procedure is:

1) Initialize filter at \( k = 0 \), as \( \bar{x}_{0/0} = \bar{x}_0 \) and

\[
P_{0/0} = P_0 \\
P_{k/k} = \tilde{S}_x S_x^T, \quad P_{k+1/k} = \tilde{S}_x S_x^T \\
Q = S_u S_u^T
\]

The interval step size is set at \( h = 3 \), (optimal for Gaussian distribution based on the square-root of the kurtosis). The state dimension is set as \( n_x = \text{length}(x) \).

![Fig. 3 Block diagram of DDF](image)

2) The state prediction at time instant \( k + 1 \) and the corresponding covariance matrix is

\[
\hat{x}_{k+1} = h^2 - n_k - n_p \psi(\hat{f}_k) \\
+ \frac{1}{2h^2} \sum_{p=1}^{n_p} \psi(\hat{x}_k + h\tilde{S}_x, \psi(\hat{x}_k - h\tilde{S}_x)) \]

3) The measurement prediction is

\[
\hat{z}_{k+1/k} = H\hat{x}_{k+1/k}
\]

4) Update

\[
K_{k+1} = P_{k+1/k} H^T (H P_{k+1/k} H^T + R_k)^{-1}
\]

5) The estimation at time instant \( k + 1 \) given all the measurement up to time instant \( k + 1 \) is

\[
\hat{x}_{k+1/k+1} = \hat{x}_{k+1/k} + K_{k+1}(z_{k+1} - \hat{z}_{k+1/k})
\]

\[
P_{k+1/k+1} = (I - K_{k+1}H)P_{k+1/k+1}
\]

6) The square root factorization of estimation error covariance is

\[
\tilde{S}_e(k + 1/k + 1) = \{\text{chol}(P_{k+1/k+1})\}^T
\]

7) Increase \( k \) to \( k + 1 \) and repeat from step 2).

The main changes are in step 2, step 3 and step 5,
compared with DDF. The erosion filter is particularly useful for binary image processing, where it removes pixels that are not surrounded by specified amounts of neighbors. It gives the ability to remove noisy pixels (stand-alone pixels) or shrink objects [31]. Previous work [26]-[28] on using the linear square method to get a more flexible solution has been carried out. However, considering the accuracy of the motion detection parts, we choose a more strict condition in the difference and erosion filter designs. Moreover, we need to control the bit rates. We have not modified the calculation procedure of the erosion filter. The originality of the proposed pyramid erosion filter is to use the symmetrical feature of one macroblock to decide the weights between each direction, just as Fig. 5 shows. In Fig. 4, DF means the conventional difference filter. DDF represents the proposed divided difference filter. Figure 5 displays the proposed adaptive erosion filter for a 16×16 macroblock, which contains two parts. The left paragraph shows the pixels of a macroblock. For example, the linear distance between the blackened pixels is 1. The right figure shows the different weights for the different distances between the selected pixels with the center of each macroblock. The linear, inverse, and the inverse squared distances represent x, y and z in the noise function, respectively. In order to collect every central noise, the weight should be decreased adaptively on the basis of the different location of each pixel. In Fig. 6, EF means the conventional erosion filter, WEF represents the weighted erosion filter. Figures 4 and 6 show the progress of the best values, worst values and the mean values of the population evolution found by using the DDF and WEF over 30 runs for 7 outdoor and indoor sequences that show the same situation as the evaluation section. Each picture exhibits the two algorithms in the same number of frames and resolution. Four different algorithms are tested. DDF and WEF perform significantly better than DF and EF on the best values of every generation for different video sequences. With the increase in the number of frames, DDF and WEF also are better than DF and EF on the progress of the best and worst values. The lines of average values in each figure are calculated for 30 runs. The best values of each run in every generation are collected. The average values of DDF and WEF are also larger than DF and EF in 1000 generations. Most the experiments are carried out under the Linux system with an Intel core 2 Duo CPU@3.16GHz.

3.2 Linear square optimized threshold filter

Figure 7 takes the 300th frame of the Paris CIF sequence as an example. It shows the experimental results for the threshold set from 1 to 10 and the motion blocks set to white. The threshold is alert to the difference between the current frame and reference frame. The smaller the threshold is set, the more blocks are recognized as motion blocks. In order to find the mathematical relationship between the threshold and the motion-detection blocks, the sta-
Fig. 7 Example of threshold setting and motion-block detection

Statistical analysis is shown in Fig. 8. Here, only CIF sequences for each sequence size have been shown for simplicity. Through 10 QCIF sequences, 9 CIF sequences and 7 sequences of the 720p test, which covers most video application with little movement, large movement, few detailed features, many detail feature, near distance, far distance and the others, the rules of the motion-block ratio and threshold have been calculated. For Eqs. (9) to (13) listed below, in order to attain sufficient accuracy, the coefficient of each linear polynomial has been set to retain three significant digits has been set. After linear, quadratic, cubic, 4th and 5th-degree polynomial fittings, the 4th-degree model was confirmed to be the best fit with fewest errors and residue at a minimum level. In the bottom part of a Fig. 8, the vertical axis means the residue after linear square fitting, the horizontal axis shows the threshold number. The residual norms are also displayed. It can be discovered that the 4th degree can obtain an absolute bias within 0.002. Although the 5th degree has a smaller residual, the differences between different sequences fill the gap. Moreover, the complexity is higher than the 4th degree. Therefore, the 4th degree is sufficient. From this analysis, the threshold can be adaptively chosen for different video types. It can be conclude that in a video with little background movement, the threshold should be set to a larger value for motion block detection. The motion-detection scalability versus threshold statistical analysis was to maintain a high accuracy with a residue smaller than 0.002. After the blobs are calculated by the blob filter, the linear square (LS) condition is set. The LS condition is if $y > x$ in Eq. (11), the weighted erosion filter is chosen; else the erosion filter is chosen.

$$\begin{align*}
y &= -0.626x + 1 \\
y &= -0.0035x^2 - 0.037x + 1 \\
y &= -0.00048x^3 + 0.0015x^2 - 0.053x + 1 \\
y &= -0.0013x^4 + 0.018x^3 - 0.086x^2 + 0.011x + 0.93 \\
y &= -0.00026x^5 + 0.0032x^4 - 0.012x^3 + 0.0027x^2 - 0.01x + 0.99
\end{align*}$$

Fig. 8 Linear-square-optimized threshold with motion-blocks ratio
Table 1 Different filter combinations (ms)

<table>
<thead>
<tr>
<th>TH</th>
<th>DDF + EF</th>
<th>DF + WEF</th>
<th>DDF + WEF</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3856</td>
<td>40664</td>
<td>14384</td>
</tr>
<tr>
<td>5</td>
<td>794</td>
<td>22816</td>
<td>16976</td>
</tr>
<tr>
<td>10</td>
<td>1108</td>
<td>28464</td>
<td>17128</td>
</tr>
<tr>
<td>15</td>
<td>1396</td>
<td>33800</td>
<td>19940</td>
</tr>
<tr>
<td>20</td>
<td>1156</td>
<td>33552</td>
<td>22464</td>
</tr>
</tbody>
</table>

4. Skip-Mode Detection for Motion Estimation Algorithms

In H.264, variable block-size motion estimation (ME) is adopted, in which eight inter prediction modes (SKIP, Inter 16×16, Inter 16×8, Inter 8×16, Inter 8×8, Inter 8×4, Inter 4×8, and Inter 4×4) are defined. Therefore, an exponential growth of complexity in inter mode prediction has been analyzed [23]. Among them, the skip mode provides good coding performance and requires little complexity, and it usually occurs when the contents of video sequences are at a standstill or with very slow motion, such as the static background. Therefore, if those parts can be detected, a large amount of computation and bit rate can be saved by directly setting the skip mode. Temporal prediction techniques are usually based on the information concerning the Motion Vectors (MVs) from previous frames. Most common approaches, which use the MVs corresponding to the homologous or surrounding MBs as temporal predictors, require a considerable amount of storage space. In contrast, the proposed approach for temporal prediction, which reduces memory, is based on the following two assumptions: (a) temporal prediction improves the required prediction significantly less than spatial prediction and its importance is only justified by the high complementarities that it provides when combined with spatial prediction; (b) motion block is belonged to this homologous which has already moved and consequently, the stored information might only be useful if the same motion pattern involves the surrounding MBs. Hence, it can be shown that temporal prediction may be significantly improved if the mean value of more neighboring MBs is also considered.

Moreover, to make temporal prediction usable by hardware motion estimators, the required memory should be reduced. On the basis of this assumption, we propose the following new area based temporal prediction approach: i) the image is first divided into equal-sized square areas, for instance, as shown in Fig. 9; ii) for all these areas, the mean MVs are calculated and used as temporal predictors. Additionally, if the MB being processed is close to the edge of one of such areas, the mean MVs of the adjacent areas may also be taken into account. By considering assumption (b), it was also decided to perform the temporal prediction using only the maximum considered block size for the MBs (16×16 pixels). This granulation of the ME blocks presents an inherent trade-off between prediction precision and ME memory requirements.

Fig. 9 Detected motion blocks

5. Evaluation Result

5.1 MOO-based motion detector evaluation

Table 2 Motion-block detector time comparison (ms)

<table>
<thead>
<tr>
<th>Sequences</th>
<th>SAKBOT</th>
<th>Proposed</th>
<th>TS</th>
</tr>
</thead>
<tbody>
<tr>
<td>videoA0</td>
<td>34359</td>
<td>19634</td>
<td>42.86%</td>
</tr>
<tr>
<td>videoA1</td>
<td>24756</td>
<td>13528</td>
<td>45.35%</td>
</tr>
<tr>
<td>videoA2</td>
<td>25995</td>
<td>15566</td>
<td>40.12%</td>
</tr>
<tr>
<td>videoA3</td>
<td>30732</td>
<td>18078</td>
<td>41.18%</td>
</tr>
<tr>
<td>videoB1</td>
<td>64031</td>
<td>38114</td>
<td>40.48%</td>
</tr>
<tr>
<td>videoB2</td>
<td>66339</td>
<td>37133</td>
<td>44.00%</td>
</tr>
<tr>
<td>videoB3</td>
<td>63042</td>
<td>34237</td>
<td>45.69%</td>
</tr>
</tbody>
</table>

In order to evaluate the efficiency of the proposed motion-block detector, the time comparison and the accuracy have been listed in both Table 2 and Fig. 10. Evaluation based on Ground Truth (GT) offers a framework for objective comparison of the performance of alternate surveillance algorithms. In this kind of evaluation technique [32] the output of the algorithm is compared with the GT obtained manually by drawing bounding boxes around objects, marking up the pixel boundary of objects, or labeling objects of interest in the original video stream. The standard video surveillance sequences, including 4 outdoor videos (videoA0 to videoA3) and 3 indoor sequences (videoB1 to...
videoB3) sequences for high noise, low noise, little motion, and much motion from visor [33] with 30 frames per second (fps) and a resolution of 352x288, were tested to validate our proposed algorithm. In Fig. 10. AccuracyRate = TPObjects/GTObjects, where TP denotes True Positive accuracy rate means the motion-detection rate. Once the percentage of objects, P_objects, is greater than or equal to the defined value for overlap, the detected object belongs to TObjects. Here, TSAKBOT and TMD mean the motion-block detector times of SAKBOT and the proposed algorithm, respectively.

\[ T_{SM} = \frac{T_{SAKBOT} - T_{MD}}{T_{SAKBOT}} \times 100\% \quad (14) \]

Figure 11 shows the result of all cases of updated filters. Seven outdoor and indoor surveillance sequences have been tested. Here, two sequences are taken as an example. From the comparison between the SAKBOT and Proposed Algorithm, we found that the proposed algorithm can detect moving cars and humans accurately in both outdoor and indoor situations. Although for the indoor sequence, some misdetections part on the left side of the sequence exits. The reason is mainly the presence of the smoke detection module, which can also be respected by adding it into the proposed algorithm as future work.

5.2 Fast motion estimation evaluation

The fast algorithms were combined with UMHexagonS adopted in the H.264 reference software version JM11.0. Several high and low motion sequences were selected, including the CIF sequences Paris and hall, which can be recognized as indoor video surveillance test sequences. For all these sequences, the picture rate was 30f/s, and 300 frames were encoded. IPPP GOP and a single slice per picture were used. The search range was ±32 for the CIF sequence. The reference frame number was set as 5. High-complexity RD-optimization (RDO ON) and the CABAC entropy method were also used. For each sequence, quantization parameter values of 16, 24, 32, and 40 were tested, which covered high, moderate, and low bit rates. The ME time comparisons are shown in Table 3. HEX represents the UMHexagonS algorithm, which is integrated in to JM software. For ME time comparison, as UMHexagonS is the fastest algorithm in JM11.0, here, only is shown the comparison with the UMHexagonS algorithm. To quantify the coding quality of the algorithms, bit rate and subjective performance were utilized, and the original UMHexagonS algorithm with constant early termination thresholds was used as the standard. In the coding log file generated by JM11.9, the computation time required by intermode motion estimation is derived to analyze its performance. To evaluate the coding speedup of the proposed fast algorithms in motion estimation processing, motion estimation time saving is defined as

\[ TS = \frac{T_{ME_{HEX}} - (T_{MD} + T_{ME_{p}})}{T_{ME_{HEX}}} \times 100\% \quad (15) \]

where TMD + TMEp denotes the motion estimation time of proposed algorithms and TMEHEX is the time taken by the JM11.0 UMHexagonS method. Table 3 also clearly illustrates that by using our fast motion estimation algorithms, the bit rate was reduced by augmenting the Quantized Parameter(QP). It should be noted that for the same video content, the bit-rate performance is improved for most situation. The Bitrate saving is defined as the absolute difference compared with the referenced UMHexagonS algorithms. As the proposed algorithms utilized the computer vision algorithm, it was difficult to compare the conventional peak signal-to-noise ratio (PSNR) using JM software in most cases. In Fig. 12, the subjective analysis of Paris, which achieves average results, and Silent, which attains the greatest bit-rate saving, are displayed. It can be observed from the enlarged region, compared with the conventional proposal, that there is almost no background loss, while for the that of interest, for example, human faces, higher accuracy results can be achieved.

![Fig. 10 Accuracy rate compared with SAKBOT](image)

6. Conclusion

The addition of a robust engine to a motion-block detector, which combines an enhanced difference filter and weighted erosion filter, is proposed in this paper. The evaluation results prove that the proposed motion detector can achieve time saving of 40-46%, while reaching high accuracy for most indoor surveillance test sequences compared with the famous SAKBOT system. By implementing the first part of the encoder system into motion estimation, the proposed detector outperforms the original JM11.0 counterpart.
in both ME time saving and bit-rate saving. Theoretical analysis converts the divided difference filter into a linear function that can be easily embedded into this motion detection engine. Namely, with the renewed technique, more lightening noise and uncritical parts were neglected, while more motion estimation com-
<table>
<thead>
<tr>
<th>Sequences</th>
<th>QP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>HEX(ms)</td>
</tr>
<tr>
<td>Hall</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>40</td>
</tr>
<tr>
<td>Paris</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>40</td>
</tr>
<tr>
<td>Silent</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>40</td>
</tr>
<tr>
<td>News</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>40</td>
</tr>
</tbody>
</table>

Computation could be saved. Moreover, the approaches proposed in this paper can also be conveniently applied in other video standards. Experimental results show that when these methods are integrated with the UMHexagonS fast block matching method of JM11.6, 21-30% of bit rate and 23-41% motion estimation time can be saved with an acceptable coding quality loss.

References

Conventional Proposed
(a) Paris

Conventional Proposed
(b) Silent

Fig. 12 Subject performance of compressed video sequences


[33] The ViSOR repository, found at URL: http://www.openvisor.org

Jia Su received her B.E. degree from the Telecommunications Engineering School of Xidian University, China, in 2006 and M.E. degrees from Graduate School of Information, Production and Systems, Waseda University and School of Microelectronics in Xidian University in 2008 and 2009, respectively. She is currently working towards her PhD degree at Waseda University. Her research interests include video compression and computer vision.

Jia Su is a student member of IEEE.
Takeshi Ikenaga received his B.E. and M.E. degrees in electrical engineering and Ph.D degree in information & computer science from Waseda University, Tokyo, Japan, in 1988, 1990, and 2002, respectively. He joined LSI Laboratories, Nippon Telegraph and Telephone Corporation (NTT) in 1990, where he had been undertaking research on the design and test methodologies for high performance ASICs, a real-time MPEG2 encoder chip set, and a highly parallel LSI & system design for image-understanding processing. He is presently a Professor in the Graduate School of Information, Production and Systems and the School of Fundamental Science and Engineering, Waseda University. His current interests are the application SoCs for image and video processing, which covers video compression (e.g., H.264/AVC, H.264SVC, H.265/HEVC), video filter (e.g., super resolution, noise reduction), video recognition (e.g., feature point detection, object tracking) and video communication (e.g., UWB, LDPC, public key encryption). He also has interests in application-oriented many-core processor design. Dr. Ikenaga is a member of IEEE, IEICE, IPSJ and IIEEJ.

(Received July 1, 2011; revised October 25, 2011)