Sensor-Assisted Video Encoding for Mobile Devices in Real-World Environments

Xiaoming Chen, *Member, IEEE,* Zhendong Zhao, Ahmad Rahmati, *Student Member, IEEE,* Ye Wang, *Member, IEEE,* and Lin Zhong, *Member, IEEE*

***Abstract*—In this paper, we present a comprehensive study on sensor-assisted video encoding (SaVE) schemes for video capturing on mobile devices in real-world environments. Our purpose is to reduce the computational complexity of video encoding by leveraging sensors that are increasingly available on mobile devices, e.g., accelerometers and digital compasses. Motion estimation is a key component of video encoding. In this paper, SaVE calculates the rotational movement of a camera (on mobile devices) and then infers the global motion in the camera imager. SaVE subsequently employs the estimated global motion as predictors to simplify motion estimation algorithms for state-of-the-art H.264/AVC video coding. We have constructed a prototype of SaVE and evaluated its performance with a pair of accelerometers, a digital compass, and their combination. Our experimental results show that SaVE can significantly reduce the computations of motion estimation while achieving equal or better video quality. Our results also show that SaVE has a strong noise-resistant capability. Therefore, it can be practically employed in real-world environments.**

***Index Terms*—Accelerometer, digital compass, H.264/AVC, motion estimation, MPEG, sensor, video encoding.**

I. Introduction

VIDEO CAMERAS have already become a standard component of Smartphones and other handheld devices. Amateur video clips captured by such cameras have populated social network portals, e.g., YouTube, and enabled amateur journalism, e.g., iReport. Yet, video capturing on mobile devices is compute-intensive and therefore power-hungry. A key compute-intensive module in video encoding is *motion estimation*. Motion estimation seeks to identify blocks from neighboring video frames that match each other and subsequently eliminate the redundancy. In modern video coding standards such as H.264/AVC, motion estimation may examine a video frame for block matching from multiple reference frames and using multiple block sizes [1]. Not surprisingly, the power and computational cost of motion estimation in H.264/AVC is posing a significant challenge to video capturing on mobile devices. Our solution toward addressing this challenge is sensor-assisted video encoding (SaVE). SaVE leverages low-power sensors to estimate camera movement and subsequently apply the estimated camera movement to significantly simplify motion estimation.

SaVE is motivated by the following observations. First, the motion of an object in a video frame can be decomposed to *global motion* introduced by the camera movement and *local motion* introduced by the movement of the object itself. In many video sequences, particularly in amateur-captured video clips, global motion due to camera movement, particularly rotation, is very common. Second, modern mobile devices have embraced ultra-low-power and low-cost sensors including digital compasses and accelerometers, e.g., HTC G1 [2] and Nokia 5140 mobile phones [3]. These sensors can provide accurate information regarding the camera movement.

In this paper, we conduct a comprehensive study of SaVE in real-world environments and present solutions that significantly improve motion estimation in such environments. Fig. 1 shows the system structure of SaVE (an H.264/AVC encoder incorporated with the proposed encoding scheme). SaVE employs three combinations of sensors (as shown in Table I) to estimate camera rotation. Using these estimations, SaVE infers the global motion in the subsequent frames and then utilizes the estimated global motion as "SaVE predictors" in motion estimation.

We have built a prototype of SaVE using custom and commercial sensors attached to a commercial camcorder. Extensive experiments on the prototype have been conducted with different sensor combinations in various conditions, considering external interference in particular. Our experimental results show that, even with the simple prototype, SaVE can reduce the complexity of the unsymmetrical-cross multi-hexagon-grid search (UMHS) [4], [5] and enhanced predictive zonal search (EPZS) [1], [6] motion estimation algorithms by up to 27% while achieving even better video quality. To the best of our knowledge, SaVE is the first publicly reported attempt in using

Manuscript received October 26, 2009; revised May 18, 2010; accepted November 12, 2010. Date of publication February 10, 2011; date of current version March 23, 2011. The work by the NUS team was supported by the Singaporean MOE, under Grant R-252-000-236-112. The work by the Rice University team was supported in part by the NSF awards, under CNS/CSR-EHS 0720825 and IIS/HCC 0713249, and an equipment donation from Texas Instruments. This paper was recommended by Associate Editor E. Steinbach.

X. Chen was with the School of Computing, National University of Singapore, 117576, Singapore, while conducting this research. He is currently with Digital Media Lab, Umeå University, SE-901 87 Umeå, Sweden, and also with the Autonomous Systems Lab (Brisbane), Commonwealth Scientific and Industrial Research Organization, Qld. 4069, Australia (e-mail: xiaoming.chen29@gmail.com).

Z. Zhao and Y. Wang are with the School of Computing, National University of Singapore, 117576, Singapore (e-mail: zhendongzhao@comp.nus.edu.sg; yewang@comp.nus.edu.sg).

A. Rahmati and L. Zhong are with the Department of Electrical and Computer Engineering, Rice University, Houston, TX 77005 USA (e-mail: rahmati@rice.edu; lzhong@rice.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2011.2114210

TABLE I
THREE SENSOR COMBINATIONS ARE USED IN THIS PAPER

Sensor Combination / Descriptions	SC1	SC2	SC3
Components	Two accelerometers	One compass plus one accelerometer	One compass plus two accelerometers
Vertical rotational estimation	Use single accelerometer	Use single accelerometer	Use single accelerometer
Horizontal rotational estimation	Use two accelerometers	Use one compass	Use one compass or two accelerometers

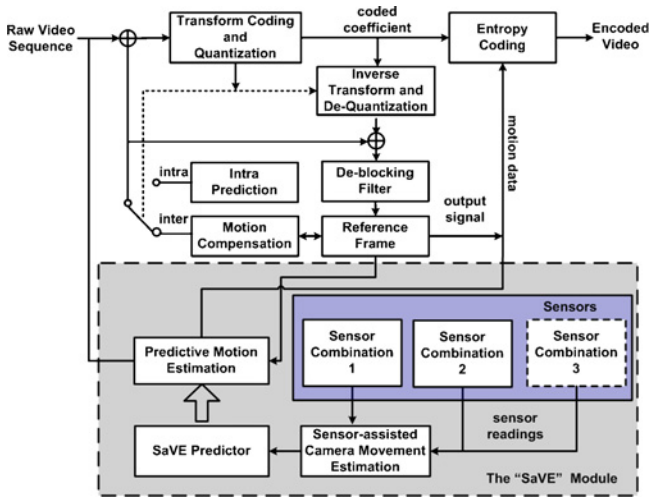


Fig. 1. System structure of SaVE (an H.264/AVC encoder incorporated with the proposed encoding scheme).

sensors to improve video encoding. It embodies a new research direction in multimedia processing that explicitly employs physical information obtained from sensors.

Our previous papers [7], [8] presented earlier results on employing sensors to improve motion estimation for MPEG2 and H.264/AVC. In particular, [8] employed a noise-sensitive digital compass but had not evaluated SaVE's resilience to the external noise. This paper presents a complete evaluation of SaVE and, in particular, the following new contributions over our previous paper.

1) We discovered that the *selective application* of SaVE predictors provides a strong noise-resistant capability, confirmed by extensive evaluations. This enables SaVE to be practically employed in real-world environments.
2) We demonstrated the effectiveness of intelligently switching between SaVE-based methods for camera movement estimation, i.e., switching between the digital compass and the accelerometers. This method allows more utilization of the detected global motion even if the digital compass is interfered.
3) We conducted intensive evaluations on the different methods of utilizing SaVE predictors (i.e., predictor insertion strategies) and justified their effectiveness.

The rest of this paper is structured as follows. In Section II, we outline the background information and related work in video coding. In Section III, we describe our method for camera movement estimation. The technical details of utilizing the estimated global motion with SaVE are explained in Section IV. In Section V, we describe the constructed prototype of SaVE. The experimental results based on the prototype implementation are shown in Section VI. We discuss the limitations and future work of SaVE in Section VII and conclude in Section VIII.

II. BACKGROUND AND RELATED WORK

In this section, we provide background information and discuss related work.

A. Motion Estimation

1) *Traditional Motion Estimation Algorithms:* The simplest motion estimation algorithm, namely, naïve Full Search, will exhaust all candidate blocks for identifying a matched block. While the Full Search algorithm provides highest block matching accuracy, it is extremely slow. Extensive research has been conducted in exploring fast motion estimation algorithms. Some algorithms attempt to reduce the number of searching positions in block matching, e.g., three-step search [9], new three-step search [10], four-step search [11], diamond search [12], [13], cross-diamond search [14], kite cross-diamond search [15], and others. Some others seek to reduce the number of pixels used in the matching distortion calculations, e.g., partial distortion search (PDS) [16], alternative sub-sampling search algorithm [17], normalized PDS [18], adjustable PDS [19], dynamic search window adjustment [20], and others. There are also some hybrid methods combining multiple techniques, e.g., motion vector field adaptive search technique (MVFAST) [21], predictive MVFAST [22], UMHS [4], [5], and EPZS [1], [6]. In particular, UMHS and EPZS can reduce the computational cost of Full Search by 90% while maintaining a fairly good rate distortion performance [1], [4]. In this paper, we have used the implementation of UMHS and EPZS in JM version 14.2 H.264/AVC reference software as the "host" of our SaVE scheme and also as baselines for evaluation.

2) *Predictive Motion Estimation:* An emerging feature of state-of-the-art video encoding algorithms, such as UMHS and EPZS, is predictive motion estimation. Instead of exhausting all candidate blocks within a search range, predictive motion estimation initially attempts a few promising predictors, and applies simplified search patterns around the predictor with a early-termination criterion [4], [6]. For example, the predictors used in UMHS and EPZS include median predictor, corresponding block predictor, UPLayer predictor, and others

[4], [6]. Among these predictors, the median predictor is believed to be more reliable [6]. A median predictor refers to the median motion vector of the top, left, and top-right (or top-left) neighbor blocks of the current block, which is frequently used as the *initial* search predictor and also for motion vector prediction encoding [6]. However, the predictors used in UMHS and EPZS are only estimated according to the spatial and temporal correlations between adjacent blocks and successive frames. These predictors have no knowledge about the real camera movement, thus the global motion is caused.

3) *Global Motion Estimation and Compensation:* Some video encoding algorithms have also considered the global motion introduced by the camera movement with the employment of global motion estimation and compensation (GMEC). GMEC usually describes the global motion in parametric motion models. It is to be noted that GMEC would require extra computations, e.g., the GMEC employed in MPEG4 [23]. There is a considerable amount of low-complexity GMEC methods proposed, e.g., [24] and [25] only use a small subset of pixels in motion estimation and calculating motion model parameters; [26] allows fast estimation of model parameters from coarsely sampled motion vector field. Moreover, there are some other methods employed GMEC for predictive motion estimation, e.g., [27]–[29]. However, H.264/AVC has not adopted GMEC in the standard due to its suboptimal rate-distortion performance [30] and increased complexity [30] (while [31] proposed a much simpler method that offers better overall rate-distortion performance). SaVE takes advantage of the concept of traditional GMEC but obtain the real global motion introduced by camera in a new way, i.e., by using reliable sensors. This renders SaVE being able to achieve even better results than original H.264/AVC, reported in Section VI later.

B. Low-Power Video Coding

According to [32], the methods for reducing the video encoding complexity for H.264/AVC can be classified into three categories: 1) reducing the number of search points in motion estimation, e.g., the fast motion estimation and predictive motion estimation we mentioned in Sections II-A1 and II-A2; 2) reducing the number of candidate encoding modes for a given macroblock, e.g., [33]–[35]; and 3) applying efficient mode selection by improving the optimization cost function, e.g., [32] itself proposed a simplified Lagrange multipliers cost function for reducing the number of operations in mode selection. A particular benefit of the third category, as claimed by [32], is that any reduction achieved by this category can also be applied on the top of the other two categories. Our proposed SaVE scheme falls into the first category so it can be combined with other low-power video encoding schemes, e.g., [32], to achieve further complexity reductions.

C. Sensors

1) *Related Work on Sensors-Assisted Applications:* Some sensor-based work has been done in the areas of vehicle movement detection and computer vision. For example, the European patent application EP1921867 presented the idea of using vehicle movement information detected by sensors to assist video compression [36]. The authors of [37] proposed a method for estimating autonomous vehicle movement by using cameras together with inertial sensors. However, [36] and [37] focused on detecting vehicle movement and vehicle-mounted cameras. Moreover, [38] proposed a method for object shape and camera motion recovery by using a gyro sensor, focusing on applications in the area of computer vision. To the best of our knowledge, no research has been done on SaVE for handheld devices except our work.

2) *Sensors in SaVE:* SaVE employs low-power and low-cost sensors to estimate camera rotation. Assuming negligible linear acceleration of the camera, a single tri-axis accelerometer can provide the vertical angle of the camera with respect to ground. Two accelerometers placed apart can measure rotational acceleration, both horizontally and vertically. However, accelerometers are unable to provide the absolute angle of the device. In contrast, a tri-axis digital compass can directly measure both horizontal and vertical angles. In this paper, SaVE uses readings from a single accelerometer for the vertical angle. For the horizontal angle, we have implemented SaVE to use either absolute angle readings from single digital compass or a pair of accelerometers. It is important to note that, compared to accelerometers, digital compasses are easily subject to external influence from nearby magnets and ferromagnetic objects, and radio interference (e.g., from mobile phones). Therefore, care must be taken when employing the digital compass in SaVE. This problem is addressed in Section IV.

3) *Power Consumptions of Sensors:* The power consumption of digital compasses and accelerometers is very small in comparison to the power required for video encoding by processor. For example, the commercial digital compass board with an embedded tri-axis accelerometer used in our prototype consumes 66 mW [39]. According to our measurements, our custom sensor with a pair of accelerometers consumes 15 mW (without Bluetooth). Furthermore, much of the power is consumed by components that will be obsolete when the sensors are properly integrated into the camera or mobile device. For example, the Honeywell HMC6042/1041Z tri-axis compass solution consumes a total of 23 mW [40], [41], and each of the KXM52 tri-axis accelerometers used in our custom Orbit Platform sensor board [42] consumes less than 5 mW [43]. Such power overhead is negligible compared to that of a H.264/AVC encoder, which is typically over a Watt [44].

D. Summary

In this section, we have discussed related concepts in video coding and sensor technology that can be summarized as follows. The traditional GMEC methods are suboptimal in their rate-distortion performance and they would also increase the encoding/decoding complexity. The existing predictive motion estimation methods have no knowledge about the real camera movement, thus the global motion is caused. The increasing availability of sensors in mobile devices brings us an emerging opportunity to reduce video encoding complexity in a new way—SaVE. SaVE naturally combines the concept of GMEC and predictive motion estimation; it detects the real camera movement by reliable sensors, infers global motion

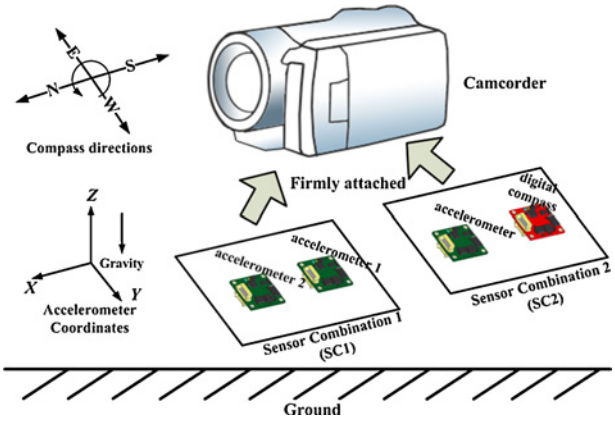


Fig. 2. SaVE prototype with two physical sensor boards.

in the camera imager, and subsequently employs the inferred global motion as predictors to simplify motion estimation. We next introduce SaVE's global motion estimation procedure in Section III and describe how SaVE predictors are applied in Section IV.

III. Proposed Global Motion Estimation

The global motion of a camera is contributed by two kinds of motion: 1) *intended* camera motion, e.g., panning, and 2) *unwanted* camera motion, e.g., hand jiggles. While [45] reported a method that integrates a digital stabilizer for dealing with unwanted camera motion, we present in this paper a new approach to handle the intended camera motion. The intended global motion of a camera is caused by *linear movement* and *rotational movement*. Linear movement, i.e., translational movement, is introduced by the camera location change, while rotational movement is introduced by tilting or panning the camera. It is worth noting that there are currently no sensor technologies for reliable estimating linear movement. Therefore, SaVE addresses rotational movement only. In this paper, the global motion to be detected by sensors is described by a global movement vector (*GMV*), which specifies both vertical and horizontal movements of objects between two successive frames due to camera rotation.

A. Rotational Change Estimation

In this paper, we have used two physical sensor boards, defined as Sensor Combination 1 (*SC1*) and Sensor Combination 2 (*SC2*), as shown in Fig. 2. Both *SC1* and *SC2* are attached to the camcorder used in our prototype. We have also employed an additional "logical" sensor combination, defined as Sensor Combination 3 (*SC3*), which uses the sensor readings of the digital compass of *SC2* and the two accelerometers of *SC1* for estimating the camera motion. All of the three sensor combinations use single accelerometers to estimate the vertical camera rotation, but use different sensors to estimate the horizontal rotation as described in Table I.

1) *Vertical Rotation Estimation:* By measuring the static gravity acceleration, a single accelerometer is sufficient to obtain the vertical angle, the camera or mobile device is tilted at with respect to the earth [7]. For example, let a_x, a_y, and a_z denote the earth's gravity on acceleration measurement in three axes, a_x will increase and a_z will decrease when the camera rolls down from the illustrated position in Fig. 2. As a result, SaVE calculates the vertical angle V_n at the video frame F_n as follows:

$$V_n = \tan^{-1}\left\{\frac{a_x}{\sqrt{a_y^2 + a_z^2}}\right\}. \tag{1}$$

The vertical rotational change $\Delta\theta_v$ for two successive video frames F_n and F_{n-1} is then calculated as follows:

$$\Delta\theta_v(n) = V_n - V_{n-1}. \tag{2}$$

The above approach is adopted in all of *SC1*, *SC2*, and *SC3*.

2) *Horizontal Rotation Estimation:* SaVE can calculate the horizontal rotation by using two accelerometers (*SC1*), a digital compass (*SC2*), or a digital compass plus two accelerometers (*SC3*), as shown in Table I.

a) *Using Two Accelerometers (SC1):* A single accelerometer cannot provide the horizontal angle, but two accelerometers placed apart from each other, as for *SC1*, can measure the angular acceleration of the device. According to [7], we can calculate the angular acceleration ($\dot{\omega}$) of the device in the horizontal direction using measurements from the two accelerometers as follows:

$$\dot{\omega} = \frac{S_y - S_{1y}}{d} \tag{3}$$

where S_{0y} and S_{1y} are the acceleration measurements in the y direction of the two sensors, respectively. Let $\Delta\theta_h$ denote the horizontal rotational change, assuming the time between two subsequent frames is t, for frame n we have

$$\begin{aligned}
\Delta_{\theta_h} &= \omega \cdot t \\
\Delta_{\theta_h} &= \frac{\Delta_{\theta_h(n)} - \Delta_{\theta_h(n-1)}}{t} = \dot{\omega} \cdot t \\
\Delta_{\theta_h(n)} - \Delta\theta_h(n-1) &= \frac{S_{0y} - S_{1y}}{d} \cdot t^2 \\
\Delta_{\theta_h(n)} - \Delta_{\theta_h(n-1)} &= k \cdot (S_{0y} - S_{1y}).
\end{aligned}$$

So we have

$$\Delta_{\theta_h} = \Delta_{\theta_h(n)} = \Delta_{\theta_h}(n-1) + k \cdot (S_{0y} - S_{1y}). \tag{4}$$

Therefore, by using the two accelerometers, we can calculate the horizontal movement of each video frame according to [7].

b) *Using Digital Compass (SC2):* With the digital compass used in *SC2*, SaVE can directly obtain the horizontal angle of the camera with respect to the magnetic north. Therefore, the horizontal rotational movement $\Delta\theta_h$ between frame F_n and F_{n-1} can also be easily obtained as follows:

$$\Delta\theta_h(n) = H_n - H_{n-1} \tag{5}$$

where H_n and H_{n-1} are the horizontal angles at frame F_n and F_{n-1} from the compass readings.

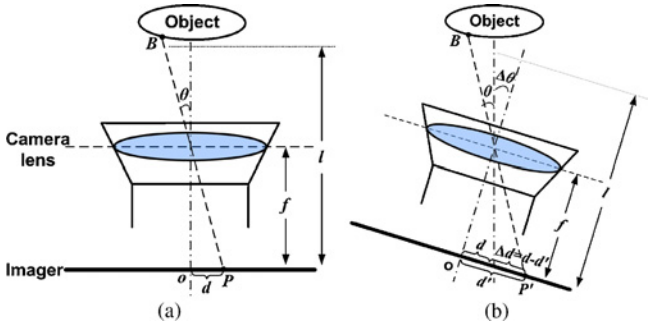


Fig. 3. From camera movement to projection movement in the video frame. (a) Initial camera position. (b) Camera rotates by $\Delta\theta$.

c) *Using Digital Compass and Accelerometers (SC3):* *SC3* uses either a digital compass or two accelerometers to estimate the horizontal rotation, depending on whether the digital compass readings are affected by external interference, i.e., *SC3* use the compass (of *SC2*) as the default sensor to estimate camera rotation, but will switch to using accelerometers (of *SC1*) when the compass readings are affected by external noise. The details of this switching procedure will be given in Section IV-C.

B. GMV Estimation

In this section, we describe how we convert the estimated camera rotation (from *SC1*, *SC2*, or *SC3*) to the pixel movements in the camera's imager.

1) *From Rotational Change to Global Movement:* When a camera rotates, the projection of an object in the view to the camera image sensor also moves, as illustrated in Fig. 3. The projection movement can be described by the *GMV*. To calculate the *GMV*, we must understand the camera characteristics and build an optical model. In Fig. 3(a) and (b), O denotes the optical center of the camera image sensor, f denotes the focal length, and l denotes the distance between the object to the camera lens; B is a point in the object. In Fig. 3(a), the projection P of point B on the image sensor is located at a distance of d from O; θ is the angle between the line BP and the perpendicular bisector of the camera lens. The situation when the camera is turned by $\Delta\theta$ is shown in Fig. 3(b), where the new projection P' is located at d' from O. The movement for projections of point B can be calculated as $\Delta d = d - d'$. From the optical model, we can easily calculate d and d' with

$$\begin{cases} d = f \cdot \tan\theta \\ d' = f \cdot \tan(\theta + \Delta\theta). \end{cases} \tag{6}$$

Hence, the projection movement Δd can be calculated as follows:

$$\Delta d = d - d' = f \cdot \{\tan(\theta + \Delta\theta) - \tan\theta\}. \tag{7}$$

As $\Delta\theta$ is usually very small between two successive frames of a video clip, we have

$$\tan(\theta + \Delta\theta) - \tan\theta \approx \Delta\theta \cdot \frac{d(\tan\theta)}{d\theta} = \Delta\theta \cdot \sec^2(\theta). \tag{8}$$

Thus, we can obtain Δd as follows:

$$\Delta d \approx f \cdot \Delta\theta \cdot \sec^2(\theta). \tag{9}$$

Typically, θ ranges between zero and half of the field of view of the lens. Hence, for all camera lenses except for extreme wide-angle and fisheye ones, θ is reasonably small and Δd can be further reduced to

$$\Delta d \approx f \cdot \Delta\theta \cdot \sec^2(\theta) \approx f \cdot \Delta\theta. \tag{10}$$

From the above formulas, we have that f and $\Delta\theta$ are adequate to calculate the movement. We can then convert the movement in pixels (denoted by *f'*) by dividing the calculated distance by the pixel pitch of the image sensor. In this paper, f is obtained by using available calibration tools based on MATLAB from [46] to [48]. It is to be noted that the calibration is not part of SaVE motion estimation, and will not be required by real applications.

2) *GMV Per Reference Frame:* Having the horizontal and vertical rotation $\Delta\theta_{\mathrm{h}}$ and $\Delta\theta_{\mathrm{v}}$, we can calculate the *GMV* for two successive frames F_n and F_{n-1} as follows:

$$GMV_n(\Delta d_h, \Delta d_v) = (f' \cdot \Delta\theta_h, f' \cdot \Delta\theta_v) \tag{11}$$

where Δd_h and Δd_v are the movement of the projection along the horizontal and vertical directions, respectively. Multiple reference-frame motion vector prediction is an important feature of H.264/AVC. For a video frame F_n, a single *GMV* calculated for merely its previous reference frame F_{n-1} is inadequate to obtain accurate predictors in other reference frames. To address this problem, SaVE dynamically calculates the reference-dependant *GMV*s. For example, when using F_{n-k} as the reference frame, the vector GMV_n^k for the frame F_n can be calculated as follows:

$$GMV_n^k(\Delta d_h, \Delta d_v) = \sum_{i=n-k}^{n} GMV_i. \tag{12}$$

IV. SaVE Motion Estimation

A. SaVE Predictor

To improve motion estimation, we can use the calculated *GMV* $(\Delta d_h, \Delta d_v)$ in UMHS and EPZS as a *SaVE predictor* (SPx, SPy), where $SPx = x + \Delta d_h$ and $SPy = y + \Delta d_v$, x and y are the horizontal and vertical coordinates of the current block to be encoded. Fig. 4 demonstrates the benefits of using *GMV*-based SaVE predictor; Fig. 4(a) shows, for a given block (gray), the original predictor has no knowledge of global motion and, therefore, it may require a fairly large search window to identify the best matching block (black). In contrast, the SaVE predictor based on *GMV* will only need a much smaller search window, as shown in Fig. 4(b). This will be confirmed in our experimentation, reported in Section VI later.

It is important to note that the SaVE predictor obtained from sensors may not be absolutely accurate in all cases and there are exceptions, e.g., when encoding a region with extensive local motion, or when the compass is interfered by external noise (as mentioned in Section II-C2). In these cases, the original predictors in UMHS or EPZS may achieve

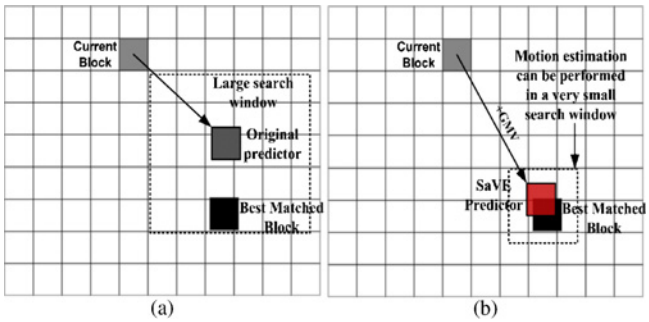


Fig. 4. SaVE predictor reduces the search window of motion estimation. (a) Original predictor. (b) SaVE predictor.

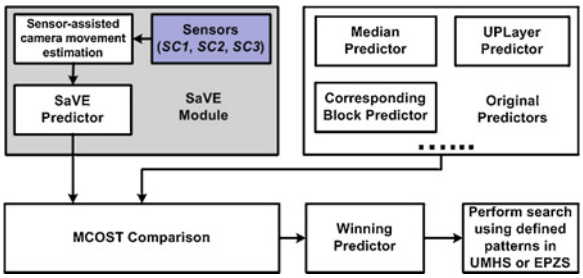


Fig. 5. MCOST-based application.

better results than SaVE. Therefore, we selectively apply SaVE predictors as follows.

B. Selective Application of SaVE Predictors

1) *MCOST-Based Application:* The most straightforward selective application is to use SaVE predictors only when they lead to reduced motion encoding cost (MCOST) (block distortion plus the cost of encoding motion vectors) than the original predictors defined in UMHS and EPZS, as shown in Fig. 5.

2) *Location-Based Application:* We have also found that where in a frame to apply a SaVE predictor is important. We next examine two basic strategies to decide where in a frame to apply a SaVE predictor. The first, called *arbitrary strategy*, is adopted in our preliminary work for MPEG2 reported in [7]. The arbitrary strategy employs the SaVE predictor as the only predictor for all blocks in a video frame, so its drawback is that it excessively emphasizes on the detected global motion while ignoring the local motion and the spatial correlation of adjacent blocks.

We next devise *selective strategies* that consider both the global and local motion. As mentioned in Section II-A2, since the median predictor (usually used as the initial predictor) in UMHS and EPZS highly relies on the *top* and *left* neighbors of the current block, it can "spread" the current motion vector tendency to the remaining blocks in the *lower* and *right* part of the video frame. We have examined a number of selective strategies, i.e., attempting the insertion with different number of blocks and in different locations of the video frame. For example, Fig. 6 shows that we have inserted SaVE predictors in the blocks located at the top two boundaries and left two boundaries of the frame. When a SaVE predictor is inserted into a block, the SaVE predictor is attempted for motion

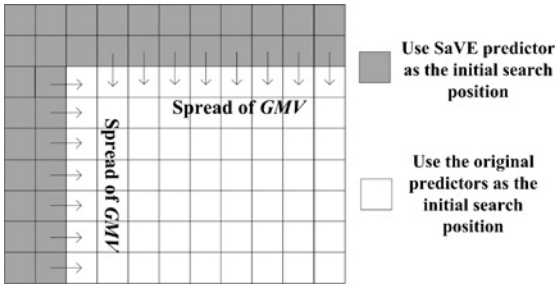


Fig. 6. Selective strategy—two-row/column strategy.

estimation first before attempting those original predictors in UMHS or EPZS. Consequently, selective strategies can spread the estimated global motion from sensors to the entire frame by using a mixture of the SaVE predictor and the original predictors. Intensive evaluations of the insertion strategies are presented and analyzed in Section VI-B.

We have also discovered that the selective applications (both MCOST-based and location-based) of SaVE predictors provide particular benefits under strong external interference, i.e., a SaVE predictor will be only attempted into a few locations in a frame, and it will be only adopted if it produces a smaller MCOST than the original predictors. Our experimental results showed that the selective application of SaVE predictors provide good noise-resistant capability of SaVE, confirmed by our experiments presented in Section VI-D1.

C. Switching Between Accelerometers and Compass (SC3)

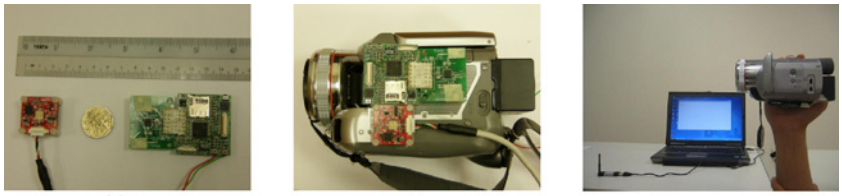
In addition to the selective application of SaVE predictors, we also propose another optional method that is able to utilize the detected camera rotation even if the compass is interfered, based on *SC3* as described in Section III-A2(c).

Basically, *SC3* will use the compass of *SC2* as the default sensor to calculate the horizontal rotation, but will switch to using the two accelerometers of *SC1* when the compass is interfered. For example, when an abnormal reading is detected from the compass (indicating the compass is interfered), SaVE will discard the compass data and use the accelerometer data as substitutions. In this paper, we have adopted a simple yet effective detection method for abnormal compass readings, which utilizes the coherence between the *GMV*s calculated by the compass and by the accelerometers at the same frame, and the coherence between the *GMV*s of the compass at two successive frames. This means that we consider a compass reading is abnormal if the current *GMV* calculated by the compass is significantly different from the corresponding *GMV* calculated by accelerometers, as well as the neighbor *GMV* calculated by the compass. We experimentally determined that the threshold is 60 for detecting the abnormal *GMV* of compass. The evaluation and analysis of this *SC3*-based method is given in Sections VI-D2 and VI-D3.

V. Prototype Implementation

A. Hardware Implementation

To evaluate SaVE, we implemented a prototype using a consumer-grade camcorder and two physical sensor boards



(a) (b) (c)

Fig. 7. Prototype implementation of SaVE. (a) Commercial board with a digital compass and a tri-axis accelerometer (left) and the in-house built board with dual tri-axis accelerometers (right) used in SaVE. (b) Camcorder and the digital compass and dual accelerometers bundled for video capturing. (c) Prototype in working.

(Figs. 2 and 7) as mentioned in Section III-A. One sensor board (*SC1*) was custom-designed and carries two tri-axis accelerometers. The other board (*SC2*), the OceanServer Technology OS5000 [39], is a commercial tri-axis digital compass with an embedded tri-axis accelerometer. It is to be noted that the *SC3* described in Sections III-A2(c) and IV-C is not a physical sensor board but a logical combination of *SC1* and *SC2*. Our custom sensor (*SC1*) outputs raw accelerometer readings and we perform the necessary calculations offline. The commercial sensor (*SC2*) computes and reports the absolute horizontal and vertical angles using its tri-axis compass and tri-axis accelerometer, respectively. We firmly attach both sensor boards to the camcorder as shown in Fig. 7.

B. Data Collection and Synchronization

While it is straightforward to synchronize video and sensor readings on an integrated hardware implementation, e.g., commercial mobile devices, our hardware prototype is limited in that the video and its corresponding sensor data are collected separately; video were captured directly by the camcorder and the sensor data were captured by the digital compass or accelerometers but stored by a laptop. Thus, we have to *manually* synchronize them. The synchronization between accelerometer data and video clips has been introduced in [7]. We have used a similar approach to synchronize the compass data and the video clips.

C. Overhead

SaVE is implemented in approximately 260 lines of C code in addition to the H.264/AVC JM software. Such simplicity makes it easy to be incorporated into practical encoding systems.

VI. Experimental Evaluation

A. Experimental Setup and Design

To evaluate SaVE, experiments were carried out using the standard H.264/AVC reference software (JM version 14.2), which implements up-to-date UMHS and EPZS algorithms. The test clips were encoded at a frame rate of 25 f/s and a bitrate of 1.5 Mb/s, using Baseline profile, variable block sizes, and five reference frames on a personal computer with 2.66 GHz Intel Core 2 CPU. Rate-distortion optimization was turned on. Intra period was set to 10. Peak signal-to-noise ratio (PSNR) was used as an objective measurement of the encoded video quality.

TABLE II
Descriptions of Clips in ''Normal Cases''

Camera (Global Motion) \ Object (Local Motion)	Still	Moving
Keep almost still	Clip01	Clip02
Medium vertical movement	Clip03	Clip04
Faster vertical movement	Clip05	Clip06
Medium horizontal movement	Clip07	Clip08
Faster horizontal movement	Clip09	Clip10
Irregular movement	Clip11	Clip12
Camera	Extensive Local Motion	
Irregular movement	Clip13	
Irregular movement	Clip14	

TABLE III
Descriptions of Clips in ''Difficult Cases''

Camera (Global Motion) \ External Interference	Mobile Signal	Moving Metal object	Microwave
Keep almost still	Clip15	Clip17	Clip19
Irregular movement	Clip16	Clip18	Clip20

We have systematically captured 20 test clips (namely, Clip01–Clip20) with a resolution of 720×576. They fall into two categories. The first category, namely, "normal cases," contains 14 clips (Clip01–Clip14) with different combinations of global (camera) and local (object) motion. The second category, namely, "difficult cases," includes six clips (Clip15–Clip20) captured under external interference. The descriptions of these clips are shown in Tables II and III (the detailed descriptions are given in the remaining sections later).

Using the techniques proposed in Section IV, we have implemented and tested three SaVE-enhanced methods with the three sensor combinations presented in Table I. The first method, denoted as SaVE/*SC1*, uses *SC1* to calculate and utilize the SaVE predictor according to Sections IV-A and IV-B. The second method, denoted as SaVE/*SC2*, uses the techniques described in Sections IV-A and IV-B as well but it employs *SC2* to calculate the SaVE predictor. The third method, denoted as SaVE/*SC3*, will use *SC2* as default but will switch to *SC1* for horizontal rotation estimation in certain conditions, as described in Section IV-C. When the above SaVE-enhanced methods are applied to UMHS or EPZS, they are denoted as UMHS+*SC1*, UMHS+*SC2*, EPZS+*SC3*, and so on.

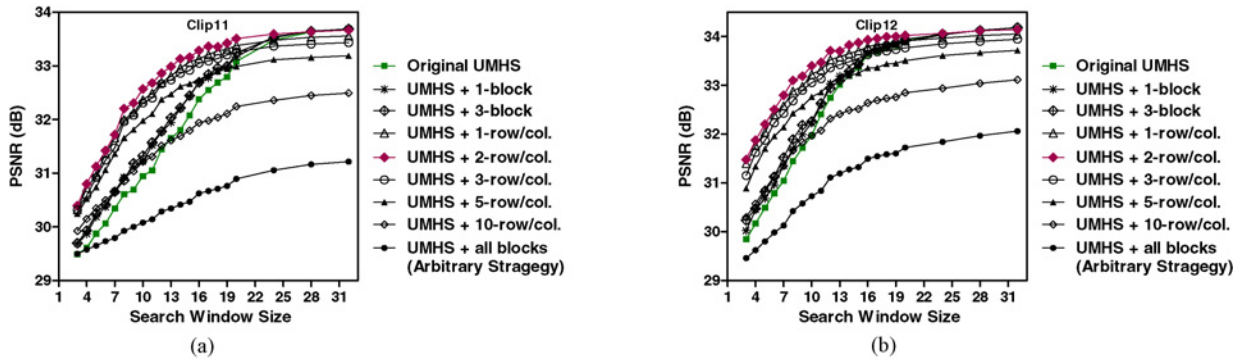


Fig. 8. Evaluations on insertion strategies. (a) Clip11. (b) Clip12.

Our experiments are divided into three parts: 1) we first examine the insertion strategies we presented in Section IV-B2; 2) we next evaluate SaVE's performance in "normal cases," where there are different kinds and levels of global motion introduced; and 3) we finally test SaVE in "difficult cases," where significant external interference is introduced.

B. Insertion Strategies

We have tested a number of insertion strategies according to Section IV-B2. This evaluation is based on UMHS using SaVE/*SC2*. The tested insertion strategies are described in Table IV. Fig. 8 presents their PSNR performance (vertical axis) for Clip11 and Clip12 based on UMHS. The search window size (SWS) used in the encoding ranges from ± 3 to ± 32 (horizontal axis).

We have observed that the SaVE predictor is very effective with the help of the median predictor as described in Section IV-B2, e.g., it will achieve a noticeable amount of PSNR gains over UMHS even if we insert it into only one block ("1-block" strategy). However, we observed decreased PSNRs compared to UMHS when there are too many SaVE predictors inserted, e.g., "5-row/col." strategy, "10-row/col." strategy, and arbitrary strategy. This is because that these strategies are increasingly eliminating the existing local motion and spatial correlations of adjacent blocks while introducing overmuch global motion. We have observed that the "2-row/col." strategy achieves the best tradeoff in this regard as shown in Fig. 8. We believe the "2-row/col." strategy particularly improves UMHS due to the following two reasons. On one hand, it benefits from a reasonable use of SaVE predictors that reflect the global motion estimated from sensors. On the other hand, it respects the local motion and spatial correlation of adjacent blocks by using the original UMHS predictors. We have observed similar results of the "2-row/col." strategy for EPZS and for other test clips. Therefore, this strategy is adopted in the remaining of the experiments.

C. Normal Cases: Clip01 to Clip14

In the evaluation for normal cases, we test the improvement of SaVE with UMHS and EPZS over original algorithms. The global motion in Clip01–Clip14, as described in Table II, was introduced by the camera movement while the local motion was introduced by walking pedestrians (a snapshot of the captured videos is shown in Fig. 11).

TABLE IV
SAVE PREDICTOR INSERTION STRATEGIES EXAMINED BASED ON UMHS

Insertion Strategies	SaVE/SC2 Predictor Inserting Position
1-block	Top-left block
3-block	Three top-left blocks
1-row/col.	Top one boundary blocks and left one boundary blocks
2-row/col.	Top two boundary blocks and left two boundary blocks (as shown in Fig. 6)
3-row/col.	Top three boundary blocks and left three boundary blocks
5-row/col.	Top five boundary blocks and left five boundary blocks
10-row/col.	Top ten boundary blocks and left ten boundary blocks
Arbitrary strategy	All blocks in the video frame

Figs. 9 and 10 show the PSNR improvement achieved by SaVE in comparison to the original UMHS and EPZS. For Clip06, Clip07, and Clip11, we show the results for SWS ranging from ± 3 to ± 32. For other clips, we only show SWS = ± 3 to ± 20, as the current prototype of SaVE will not provide gains over this range. We also present the results regarding the computation reduction realized by SaVE in Table V (for clips with vertical movement only) and Fig. 12 (for clips that contain horizontal movement). We measure the computation load of encoding with the motion estimation time.

1) *Still Camera: Clip01 and Clip02:* Clip01 and Clip02 were captured with the camera held still. As expected, the SaVE-enhanced algorithms cannot achieve higher PSNR over the original UMHS and EPZS (Fig. 9), since there is almost no camera movement. Nevertheless, SaVE will not deteriorate the performance.

2) *Vertical Movement: Clip03 to Clip06:* For Clip03–Clip06, as all of SaVE/*SC1*, SaVE/*SC2*, and SaVE/*SC3* use a single accelerometer to calculate the vertical rotation, we only present in Fig. 9 the results obtained by using SaVE/*SC2*. The results show that UMHS+*SC2* and EPZS+*SC2* achieve higher PSNR (up to 2.59 dB) than original UMHS and EPZS.

Table V shows, for Clip03–Clip06, the speedup the UMHS+*SC2* and EPZS+*SC2* achieved over the originals while obtaining equal or even *higher* PSNRs. We have selected a very small SWS (= ± 3) and a relatively large SWS (= ± 11) for comparison. It is apparent that UMHS+*SC2* with SWS =

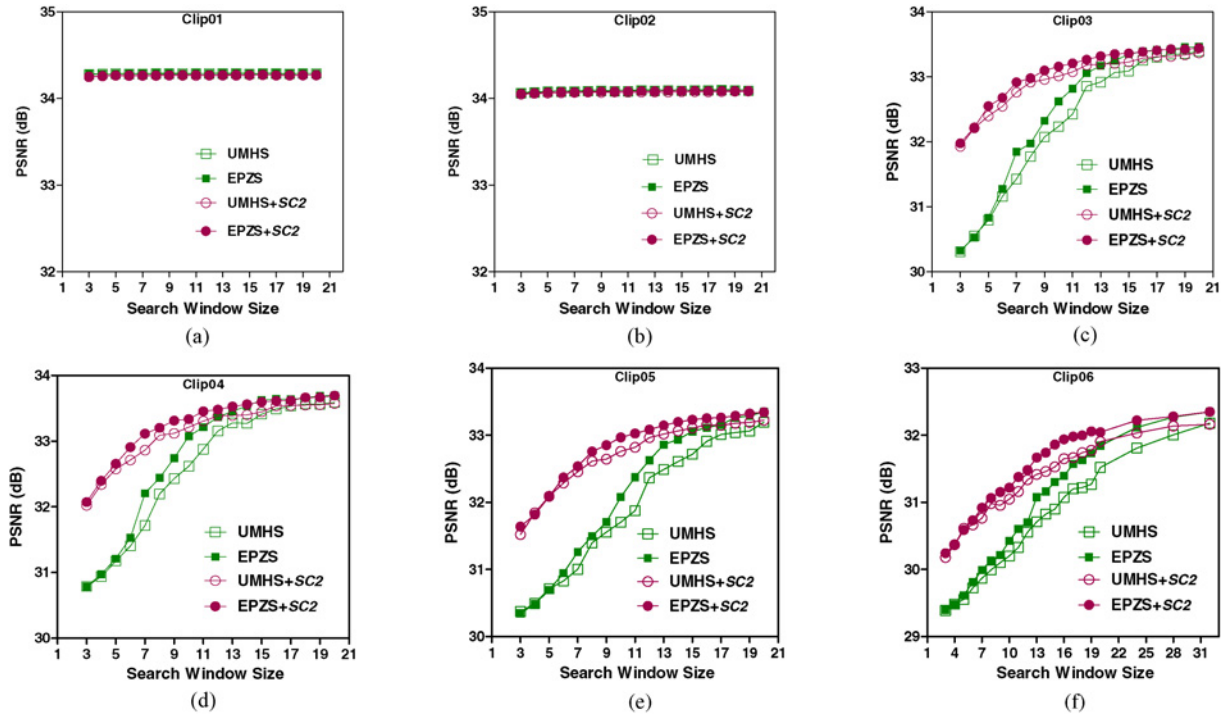


Fig. 9. PSNR comparisons for UMHS, EPZS, and SaVE-enhanced UMHS and EPZS (clips with vertical movement). Note that because SaVE/*SC1*, SaVE/*SC2*, and SaVE/*SC3* are identical in vertical motion estimation, we only show the results of SaVE/*SC2*. (a) Clip01. (b) Clip02. (c) Clip03. (d) Clip04. (e) Clip05. (f) Clip06.

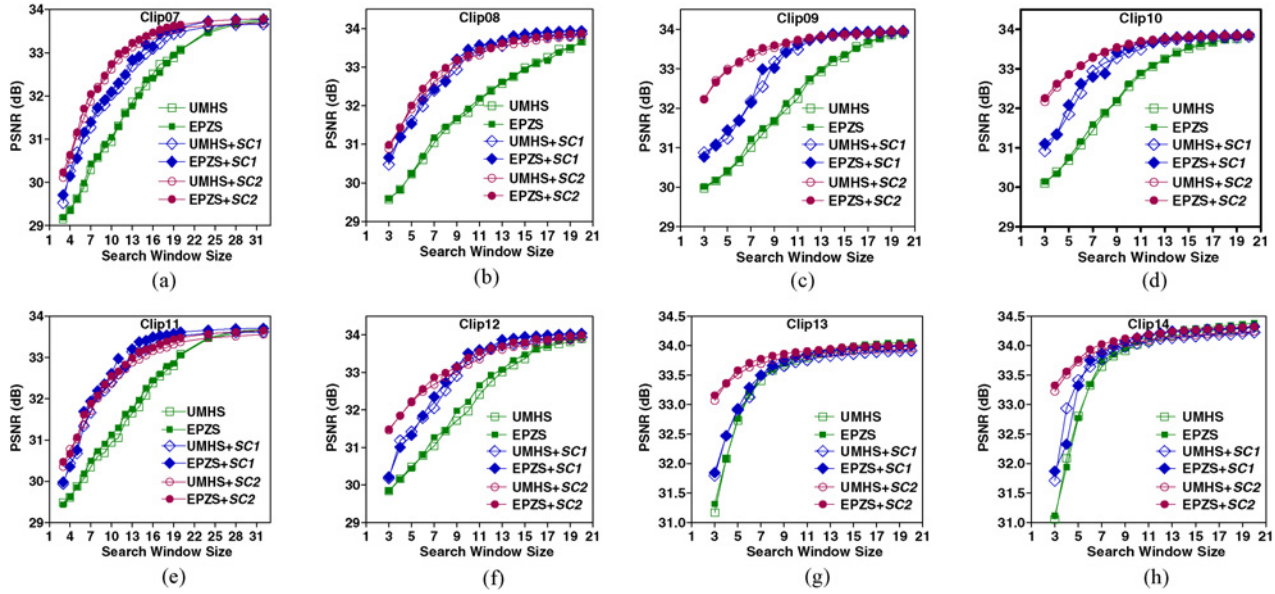


Fig. 10. PSNR comparisons for UMHS, EPZS, and SaVE-enhanced UMHS and EPZS (clips that contain horizontal movement). (a) Clip07. (b) Clip08. (c) Clip09. (d) Clip10. (e) Clip11. (f) Clip12. (g) Clip13. (h) Clip14.

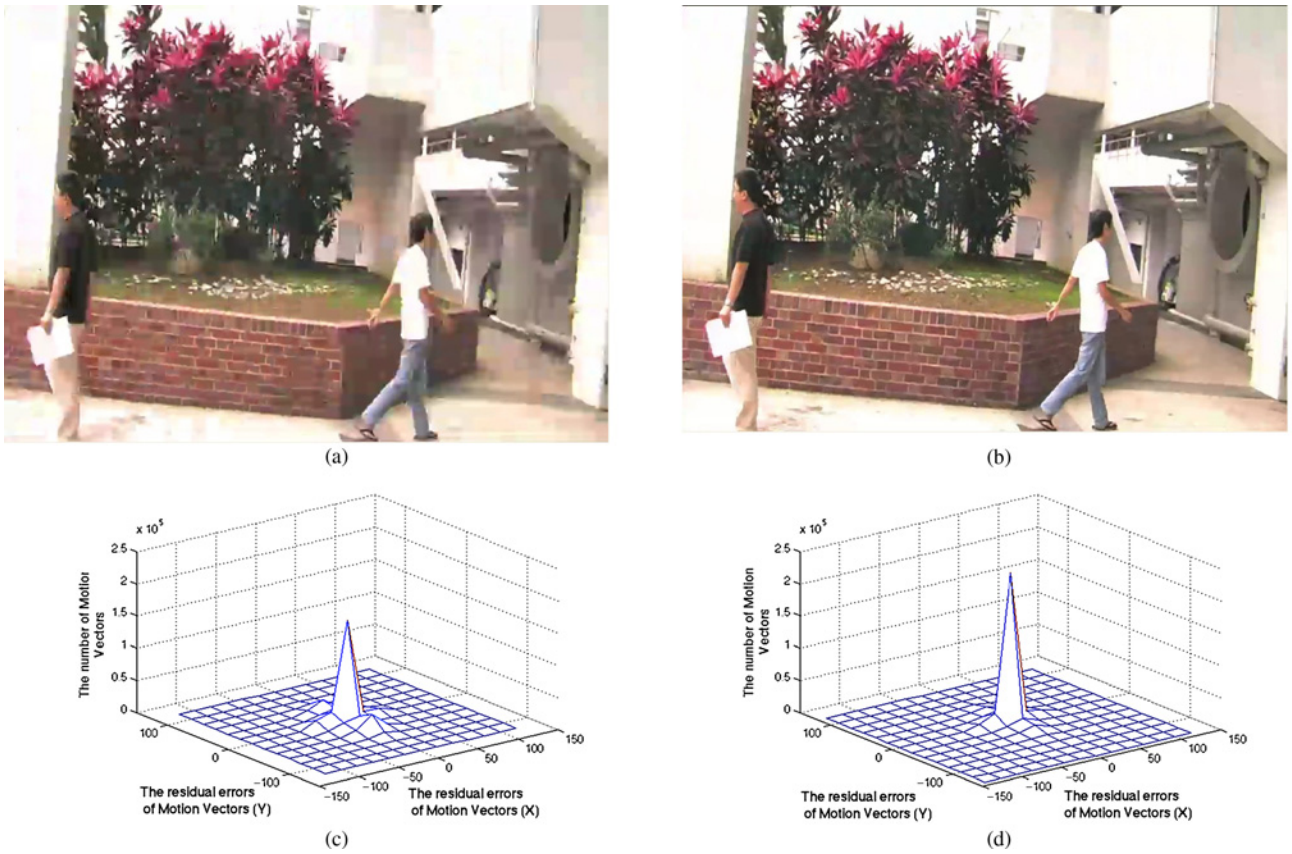


Fig. 11. Results of Clip11. (a) Sample picture decoded by EPZS with SWS = ±11 (27.01 dB). (b) Sample picture decoded by EPZS+*SC2* with SWS = ±11 (31.42 dB). (c) Motion vector residues produced by EPZS compared to a 64 × 64 full search. (d) Motion vector residues produced by EPZS+*SC2* compared to a 64 × 64 full search.

TABLE V
PSNR GAINS AND SPEEDUP ACHIEVED BY SAVE-ENHANCED UMHS AND EPZS WHEN SWS = ±3 AND SWS=±11 FOR CLIPS WITH VERTICAL MOVEMENT

SaVE-Enhanced UMHS				
Clip	SWS = ±3		SWS = ±11	
	PSNR Gains (dB)	Speedup (%)	PSNR Gains (dB)	Speedup (%)
03	+0.15	22.87	+0.01	7.10
04	+0.31	15.77	+0.03	6.86
05	+0.12	23.59	+0.10	8.72
06	+0.06	27.00	+0.11	13.80
SaVE-enhanced EPZS				
Clip	SWS = ±3		SWS = ±11	
	PSNR Gains (dB)	Speedup (%)	PSNR Gains (dB)	Speedup (%)
03	0.01	12.07	+0.04	3.08
04	+0.54	7.65	0	3.50
05	+0.14	10.03	+0.09	5.71
06	+0.07	11.44	+0.03	4.58

±3 (or SWS = ±11) can usually obtain a higher PSNR than original UMHS with SWS = ±7 to ±9 (or SWS = ±13 to ±19). This results in up to 27% time saving over the original algorithms (since we can apply a very small SWS with SaVE to achieve good video quality).

3) *Horizontal Movement: Clip07 to Clip10:* For clips of "normal cases" that contain horizontal movements, we show the results of SaVE/*SC1* and SaVE/*SC2* only, as according to our results, SaVE/*SC3* will obtain the same results with SaVE/*SC2* since there is no external interference introduced. As shown in Fig. 10, both SaVE/*SC1* and SaVE/*SC2* can obtain significant PSNR gains over the originals. It is apparent that SaVE/*SC2* outperforms SaVE/*SC1* since the digital compass is more accurate in estimating the horizontal movement.

4) *Irregular Movement: Clip11 and Clip12:* Again, Fig. 10 shows that SaVE can achieve considerable PSNR gains over the original algorithms. When medium SWSs are used, the PSNR gains are typically from 0.4 dB to 1.6 dB.

5) *Extensive Local Motion: Clip13 and Clip14:* Clip13 and Clip14 were captured in a busy crossroad with *significant local motion* introduced by fast moving vehicles and slow moving pedestrians at various distances to the camera. As shown in the figure, SaVE/*SC2* can still outperform the original algorithms, although not as much as Clip03–Clip12. The improvement is less in SaVE/*SC1* because it partially relies on the movement of its previous frame [7]. The reduction in improvement is expected because SaVE only provides extra information about the global motion.

For Clip07–Clip14, we found that in some cases the SaVE with SWS = ±3 (or SWS = ±11) can achieve higher PSNR than the original algorithms with SWS = ±10 (or SWS = ±20). In Fig. 12, we present and compare the speedups achieved by SaVE/*SC*1 and SaVE/*SC2* while achieving 0.01–0.42 dB *higher* PSNR over the original algorithms. The figure shows that the speedups are up to 24.78%. It also shows that using the

digital compass is more efficient than the two accelerometers overall. For Clip13 and Clip14, SaVE/*SC1* will not achieve speedup, however, we found that it can improve the PSNR by up to 0.75 dB while consuming the same computation.

Fig. 11(a) and (b) shows a sample frame (frame 76) of Clip11 decoded by EPZS (27.01 dB) and EPZS+*SC2* (31.42 dB) with same SWS (= ±11). Due to the camera movement, the picture decoded by EPZS is blurred (as EPZS has no knowledge about the global motion). It is obvious that the picture decoded by EPZS+*SC2* has much better quality than EPZS. Fig. 11(c) and (d) shows, for the same clip, the distributions of motion vector residuals produced by EPZS and EPZS+*SC2* (SWS = ±11) compared to a 64 × 64 Full Search (SWS = ±32). Assuming that the 64 × 64 Full Search has produced very accurate motion vectors (Section II-A1), the figures show that the resulted motion vectors of EPZS+*SC2* are closer to the Full Search compared to original EPZS, and therefore EPZS+*SC2* can yield higher PSNRs than EPZS.

D. Difficult Cases: Clip15 to Clip20

In this evaluation, we first test the noise-resistant capability of SaVE/*SC1* and SaVe/*SC2* based on EPZS in Section VI-D1. In Section VI-D2, we further evaluate SaVE/*SC3* and compare it with SaVE/*SC2* since it is based on SaVE/*SC2*. In this evaluation, we intended to see SaVE's performance in realistic environments, in particular under external interference, e.g., a phone call comes when the user is capturing a clip with his or her mobile phone. Specifically, Clip15–Clip20 were taken with various sources of external interference introduced. These clips are described in Table III. We placed a mobile phone (that was receiving calling signals) closely to the compass while capturing Clip15 and Clip16. When capturing Clip17 and Clip18, interference was introduced by waving a metal object around the compass. We captured Clip19 and Clip20 next to a microwave oven to test the SaVE/*SC2* with microwave interference.

1) *SaVE/SC1 and SaVE/SC2 with Selective Use of SaVE Predictors:* As an example, Fig. 13 shows the horizontal *GMV*s calculated by accelerometers of SaVE/*SC1* and the digital compass of SaVE/*SC2* for Clip16 and Clip19. We can see that the *GMV*s calculated by the compass are significantly affected by the external interference. This is not surprising, since the compass is a magnetic sensor designed to measure the earth's weak magnetic field. Therefore, the compass of *SC2* is more easily affected by nearby ferromagnetic objects and electromagnetic interference. On the contrary, the accelerometers of *SC1* are relatively immune to this interference.

Fig. 14 compares the average PSNR obtained by original EPZS and by the selective application of SaVE predictors (SWS = ±11), as described in Section IV-B. To show the benefits of the selective application, we have also shown in the figure the results of SaVE without the selective application as a reference, i.e., SaVE predictor is inserted into all blocks (arbitrary strategy) and without comparisons to the original predictors in EPZS, denoted by EPZS+*SC2**. We discovered that the performance of EPZS+*SC2** is below EPZS+*SC1* and EPZS+*SC2*, and even the original EPZS. On the contrary, EPZS+*SC2* (with the selective application of SaVE predictors)

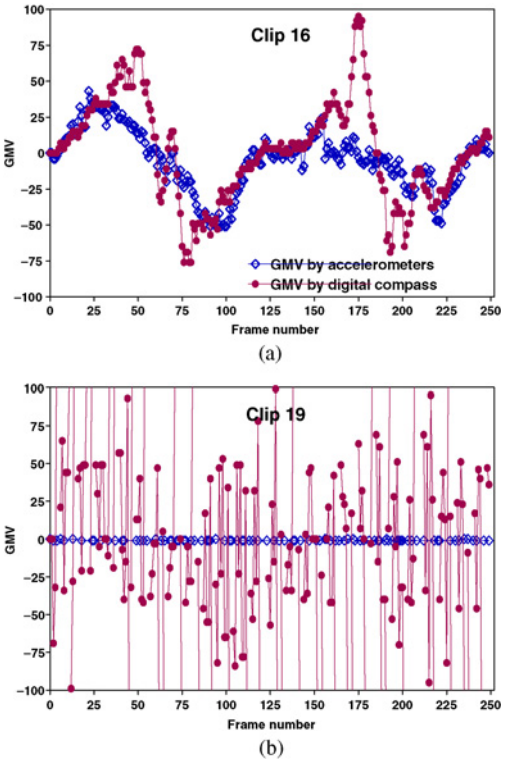


Fig. 13. Horizontal GMVs calculated by the two accelerometers and the digital compass under external interference. (a) Clip16. (b) Clip19.

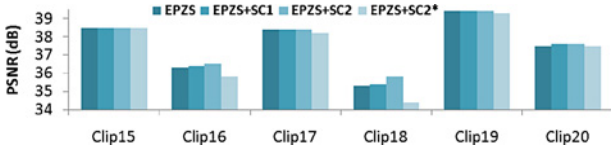


Fig. 14. PSNR comparisons for EPZS and SaVE-enhanced EPZS, for Clip15–Clip20 (clips that were taken under external interference).

still can outperform the original EPZS, especially when there is global motion introduced (Clip16, Clip18, and Clip20). For clips without global motion (Clip15, Clip17, and Clip19), EPZS+*SC2* can still obtain the same average PSNR as the original EPZS. As explained in Section IV-B, the resulted SaVE predictor is only attempted in selected positions of a video frame and it will only be adopted when it is "better" than the original predictors (i.e., when it produces smaller MCOST). This explains why the performance of SaVE is always above the original algorithms, even under significant interference.

As an example, Fig. 15 shows the PSNRs of two segments of Clip16 (frames 40–100, and frames 160–230). The compass readings of *SC2* for these two segments are particularly affected by the interference (refer to Fig. 13). It shows that EPZS+*SC2* may obtain lower PSNRs than EPZS+*SC1* in some cases, e.g., from frames 80 to 100 (marked with *Portion A* in Fig. 15), and from frames 210 to 230 (marked with *Portion B* in Fig. 15) since the accuracy of the SaVE/*SC2*'s predictor is somewhat affected by the interference. But at least EPZS+*SC2* will obtain the similar PSNR as the original EPZS. Besides, EPZS+*SC2* can still provide benefits for other portions without, or with less interference. As a result, EPZS+*SC2* can still

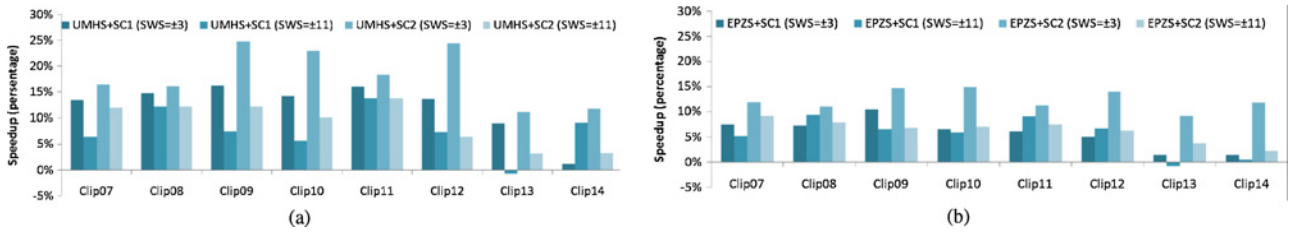


Fig. 12. Speedup comparisons for SaVE/*SC1* and SaVE/*SC2* for Clip07–Clip14 (clips that contain horizontal movement). (a) Based on UMHS. (b) Based on EPZS.

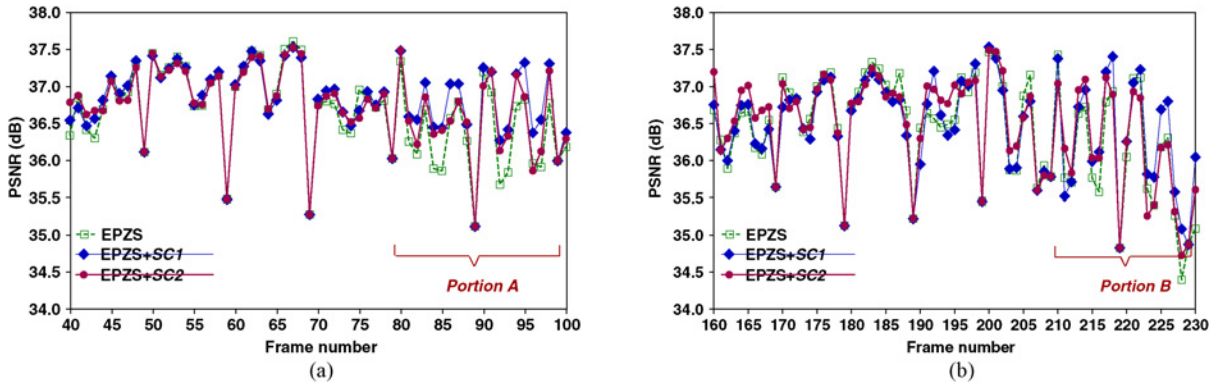


Fig. 15. PSNR comparisons for SaVE/*SC1* and SaVE/*SC2* for affected portions in Clip16. (a) Frames 40–100. (b) Frames 160–230.

slightly outperform EPZS+*SC1* overall, as shown in Fig. 14.

2) *SaVE/SC3 (Logical Sensor Combination):* In this section, we have tested SaVE/*SC3* described in Section IV-C. Due to space limitations, here we only present the results of Clip16 as an example (while we observed similar results for other test clips).

From Fig. 15, we have observed that in many cases SaVE/*SC2* can obtain at least similar PSNR to original EPZS, even under very strong interference, e.g., from frames 175 to 200 (refer to Fig. 13). This is because that SaVE/*SC2* is able to discard significantly interfered compass readings and use the original predictors in EPZS as substitutions in such cases. In some other cases, the slightly interfered compass readings can still provide partial knowledge of global motion, e.g., the PSNRs of EPZS+*SC2* is slightly higher than the original EPZS in *Portion A* and in *Portion B* as shown in Fig. 15. However, this improvement of EPZS+*SC2* in *Portion A* and *Portion B* is smaller than that of EPZS+*SC1* (since the accelerometer is not interfered thus providing more knowledge of global motion).

Fig. 16 shows the results of SaVE/*SC3*, which is expected being able to utilize more detected global motion, and compares its performance to SaVE/*SC2* (since it is based on SaVE/*SC2*). According to our results, the overall improvement of SaVE/*SC3* over SaVE/*SC2* is not much (around 0.03 dB). For example, Fig. 16(a) shows that the PSNR of EPZS+*SC3* is similar to EPZS+*SC2* for most frames. The small amount of overall improvement is expected because SaVE/*SC3* and SaVE/*SC2* process most video frames in the same way. However, for some other frames, SaVE/*SC3* is able to provide noticeable gains, especially in portions where EPZS+*SC2* is somewhat interfered by insignificant noise, e.g., results of *Portion A* displayed in Fig. 16(b) and results of *Portion B* displayed in Fig. 16(c). We have also observed similar results for other tested clips. Therefore, we believe SaVE/*SC3* can be served as an attractive option to SaVE/*SC1* and SaVE/*SC2* when sufficient sensors are available.

3) *Summary:* According to the results shown in the above two sections, our analysis and conclusion can be summarized as follows.

a) Due to the hardware features of *SC1*, SaVE/*SC1* is not sensitive to external interference. In some cases, SaVE/*SC1* is able to obtain better results than SaVE/*SC2* for certain frames.
b) *SC2* is easily subject to external interference. However, SaVE/*SC2* is able to use original predictors in UMHS or EPZS when *SC2* is significantly interfered. Due to the better estimation for horizontal camera rotation, SaVE/*SC2* still can slightly outperform SaVE/*SC1* overall.
c) In some cases, SaVE/*SC3* is able to utilize more global motion detected by sensors than SaVE/*SC2* by switching between SaVE/*SC1* and SaVE*SC2*. Such being the case, SaVE/*SC3* can provide gains in some frames over SaVE/*SC2*, though the overall improvement is not much.
d) We believe all of the three SaVE-based methods, i.e., SaVE/*SC1*, SaVE/*SC2*, and SaVE/*SC3*, can be employed in real-world environments due to their robust noise-resistant capabilities. In particular, SaVE/*SC2* and SaVE/*SC3* can obtain better overall results than SaVE/*SC1*, but they would require integration of two types of sensors, i.e., both accelerometer and digital compass. In most cases, we believe SaVE/*SC2* is good enough to deal with the external interference and also it requires fewer sensors than SaVE/*SC3*.

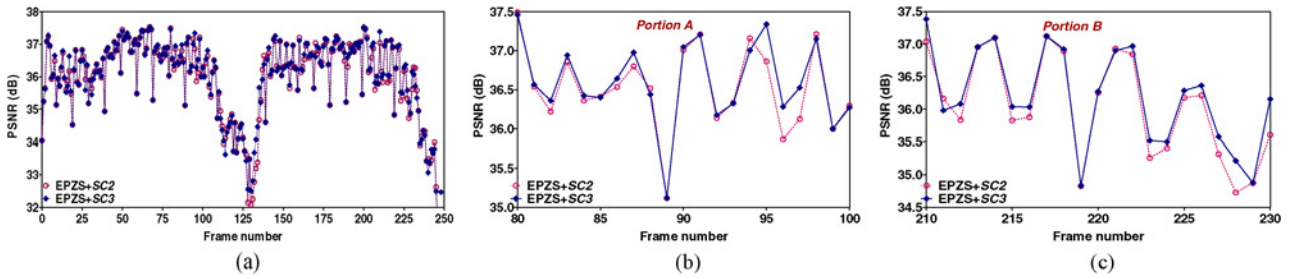


Fig. 16. PSNR comparisons for SaVE/*SC2* and SaVE/SC3 for Clip16. (a) Full range. (b) *Portion A* (frames 80–100). (c) *Portion B* (frames 210–230).

VII. Limitations and Discussions

A. Limitations and Future Work

While we show that SaVE can reduce video encoding complexity, we acknowledge that our paper is limited in the following aspects.

First, our paper is limited in the manually synchronization of video frames and sensor data (Section V-B). This will of course introduce considerable amount of errors and noise and therefore deteriorates SaVE's performance. However, an industrial implementation of SaVE will readily solve this problem with access to built-in digital sensors.

Second, SaVE cannot handle the linear movement (including zooming) adequately at this moment (Section III). Thus, more sophisticated algorithms or more sensors, e.g., gyroscope, may be required to accurately separate linear motion from rotation.

Third, we have only used a translational model for global motion estimation. In our future work, we will attempt to build an affine or perspective model to estimate the global motion. However, we expect these advanced models would require extra computations. The tradeoff between the model complexity and power consumptions will be studied in our future work as well.

B. Discussions

1) *Sensors in Mobile Devices:* As is apparent from Section VI, the digital compass-based SaVE implementation (SaVE/*SC2*) performs better than the accelerometer-only implementation (SaVE/*SC1*), in particular when horizontal camera rotation dominates. However, accelerometers are cheaper and lower-power than digital compasses because of advancement in micro-electro-mechanical systems (MEMS) technologies. Apart from digital compasses and accelerometers, a number of phones, e.g., iPhone4, are becoming equipped with low-power MEMS gyroscopes, which could also be well utilized in video encoding on mobile devices by employing a similar scheme to SaVE.

2) *Applications of SaVE:* In the consumer electronics industry, sensors are increasingly integrated into mobile devices for multiple purposes. For video phones that already have the sensors, e.g., HTC G1, SaVE would not impose any additional hardware overhead or power overhead (since the sensors are consuming power even if without SaVE). SaVE simply utilizes the information already available from preexisting sensors for another purpose—video encoding. As another example, many cameras utilize accelerometers and/or a compass to calculate camera angle changes for image stabilization. Given this context, we can consider SaVE as a "free lunch" for video capturing on sensor-rich mobile devices.

VIII. Summary and Conclusion

We reported a novel video encoding scheme, SaVE, that utilizes sensors to improve H.264/AVC motion estimation on mobile devices. We built a prototype of SaVE and conducted extensive evaluations on it with different sensor combinations and various predictor insertion strategies. We showed that SaVE is able to significantly reduce computational costs required by video capturing on mobile devices, which can be readily achieved by using a voltage/frequency scalable processor to save energy and prolong the battery life. We also demonstrated that SaVE has a strong noise-resistant capability when employed with particular predictor insertion strategies. Therefore, SaVE can be practically employed in real-world environments even if there exists external interference.

SaVE improves global motion estimation, i.e., motion due to a moving video camera. The handheld video capturing devices, intended for user-created, amateur video capturing, *inherently* experience significant camera motion. Due to the explosive growth in video-enabled small devices such as video phones, the creation of video by amateur users is increasingly popular. The phenomenal success of portals such as YouTube has demonstrated such a social trend. It was one of the motivations of our research and also attests the practical significance and broader impact of our work.

Our paper has shown that the system-level computational complexity thus energy consumption can be effectively reduced by leveraging the synergy between different modalities, acceleration, angular rotation, and vision in our case. We believe that there is plenty of *redundant computation* in a multi-component multimedia device such as a video phone, which could be removed by exploiting a similar methodology presented in this paper.

Acknowledgment

The authors would like to thank G. Hong for helpful discussions and those students at Sound and Music Computing Laboratory, National University of Singapore, Singapore, who helped in capturing the test video clips. Moreover, we are very grateful for the anonymous reviewers and Associate Editor Prof. E. Steinbach for their help in improving the final version of this paper.

REFERENCES

[1] A. M. Tourapis, *Fast ME in the JM Reference Software*, document JVT-P026, Joint Video Team, Poznan, Poland, 2005, pp. 24–29.

[2] *HTC G1 Mobile Phone*, HTC Corporation, Taoyuan, Taiwan [Online]. Available: http://www.htc.com/www/product/g1/overview.html

[3] *Nokia 5140 Mobile Phone*, Nokia, Finland [Online]. Available: http://www.forum.nokia.com/devices/5140

[4] Z. Chen, J. Xu, Y. He, and J. Zheng, "Fast integer-pel and fractional-pel motion estimation for H.264/AVC," *J. Vis. Commun. Image Represent.*, vol. 17, no. 2, pp. 264–290, 2006.

[5] Z. Chen, P. Zhou, and Y. He, *Fast Motion Estimation for JVT*, document JVT-G016, Joint Video Team, Pattaya, Thailand, 2003, pp. 7–14.

[6] A. M. Tourapis, "Enhanced predictive zonal search for single and multiple frame motion estimation," *Proc. SPIE Conf. Vis. Commun. Image Process.*, vol. 4671, pp. 1069–1079, Jan. 2002.

[7] G. Hong, A. Rahmati, Y. Wang, and L. Zhong, "SenseCoding: Accelerometer-assisted motion estimation for efficient video encoding," in *Proc. ACM Multimedia*, 2008, pp. 749–752.

[8] X. Chen, Z. Zhao, A. Rahmati, Y. Wang, and L. Zhong, "SaVE: Sensor-assisted motion estimation for efficient H.264/AVC encoding," in *Proc. ACM Multimedia Conf.*, 2009, pp. 381–390.

[9] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, "Motion compensated interframe coding for video conferencing," in *Proc. IEEE Nat. Telecommun. Conf.*, Nov.–Dec. 1981, pp. G5.3.1–5.3.5.

[10] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 4, no. 4, pp. 438–442, Aug. 1994.

[11] L. M. Po and W. C. Ma, "A novel four-step search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, pp. 313–317, Jun. 1996.

[12] S. Zhu and K. K. Ma, "A new diamond search algorithm for fast block matching motion estimation," in *Proc. ICICS*, 1997, pp. 292–296.

[13] Z. Shan and M. Kai-Kuang, "A new diamond search algorithm for fast block-matching motion estimation," *IEEE Trans. Image Process.*, vol. 9, no. 2, pp. 287–290, Feb. 2000.

[14] C. H. Cheung and L. M. Po, "A novel cross-diamond search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 12, pp. 1168–1177, Dec. 2002.

[15] C. W. Lam, L. M. Po, and C. H. Cheung, "A novel kite-cross-diamond search algorithm for fast block matching motion estimation," in *Proc. ISCAS*, 2004, pp. 729–732.

[16] S. Eckart and C. Fogg, "ISO/IEC MPEG-2 software video codec," *Proc. SPIE Conf. Vis. Commun. Image Process.*, vol. 2419, pp. 100–118, Feb. 1995.

[17] B. Liu and A. Zaccarin, "New fast algorithms for the estimation of block motion vectors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, no. 2, pp. 148–157, Apr. 1993.

[18] C. K. Cheung and L. M. Po, "Normalized partial distortion search algorithm for block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 3, pp. 417–422, Apr. 2000.

[19] C. H. Cheung and L. M. Po, "Adjustable partial distortion search algorithm for fast block motion estimation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 1, pp. 100–110, Jan. 2003.

[20] L. W. Lee, J. F. Wang, J. Y. Lee, J.-D. Shie, "Dynamic search-window adjustment and interlaced search for block-matching algorithm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, no. 1, pp. 85–87, Feb. 1993.

[21] K. K. Ma and P. I. Hosur, *Performance Report of Motion Vector Field Adaptive Search Technique (MVFAST)*, document JTC1/SC29/WG11 MPEG99/m5851, ISO/IEC, Noordwijkerhout, The Netherlands, 2000.

[22] A. M. Tourapis, O. C. Au, and M. L. Liou, "Predictive motion vector field adaptive search technique (PMVFAST) enhancing block-based motion estimation," in *Proc. SPIE Conf. Vis. Commun. Image Process.*, vol. 4310. 2001, pp. 883–892.

[23] *MPEG-4 Video Verification Model*, document JTC1/SC29/WG11 N3908, ISO/IEC, Pisa, Italy, 2001.

[24] Y. Keller and A. Averbuch, "Fast gradient methods based on global motion estimation for video compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 4, pp. 300–309, Apr. 2003.

[25] H. Alzoubi and W. D. Pan, "Very fast global motion estimation using partial data," in *Proc. IEEE ICASSP*, Apr. 2007, pp. I-1189–I-1192.

[26] S. Yeping, S. Ming-Ting, and V. Hsu, "Global motion estimation from coarsely sampled motion vector field and the applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 2, pp. 232–242, Feb. 2005.

[27] K.-Y. Yoo and J.-K. Kim, "A new fast local motion estimation algorithm using global motion," *Signal Process.*, vol. 68, no. 2, pp. 219–224, 1998.

[28] H. Jozawa, K. Kamikura, A. Sagata, H. Kotera, and H. Watanabe, "Two-stage motion compensation using adaptive global MC and local affine MC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 1, pp. 75–85, Feb. 1997.

[29] D. Adolph and R. Buschmann, "1.15 Mbit/s coding of video signals including global motion compensation," *Signal Process.: Image Commun.*, vol. 3, nos. 2–3, pp. 259–274, 1991.

[30] A. M. Tourapis, F. Wu, and S. Li, "Direct macroblock coding for predictive (P) pictures in the H.264 standard," in *Proc. SPIE VCIP*, vol. 5308, no. 1. 2004, pp. 364–371.

[31] J. Lainema and M. Karczewicz, *Skip Mode Motion Compensation*, document JVT-C027, Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Fairfax, VA, May 6–10, 2002.

[32] M. Paul, M. R. Frater, and J. F. Arnold, "An efficient mode selection prior to the actual encoding for H.264/AVC encoder," *IEEE Trans. Multimedia*, vol. 11, no. 4, pp. 581–588, Jun. 2009.

[33] B. Jeon and J. Lee, *Fast Mode Decision for H.264*, document JVT-J033, Joint Video Team (JVT) of ISO/IEC MPEG and ITU-T VCEG, Waikoloa, HI, Dec. 8–12, 2003.

[34] W. Hanli, K. Sam, and K. Chi-Wah, "An efficient mode decision algorithm for H.264/AVC encoding optimization," *IEEE Trans. Multimedia*, vol. 9, no. 4, pp. 882–888, Jun. 2007.

[35] X. Jing and L. P. Chau, "Fast approach for H.264 inter mode decision," *Electron. Lett.*, vol. 40, no. 17, pp. 1050–1052, 2004.

[36] S. Martin and M. Ulrich, "Sensor assisted video compression," European Patent Application EP1921867, 2008.

[37] D. Strelow, and S. Singh, "Optimal motion estimation from visual and inertial measurements," in *Proc. IEEE WACV*, Dec. 2002, pp. 314–319.

[38] T. Mukai and N. Ohnishi, "The recovery of object shape and camera motion using a sensing system with a video camera and a gyro sensor," in *Proc. IEEE ICCV*, vol. 1. 1999, pp. 411–417.

[39] *OS5000-S and OS5000-US Digital Compass*, OceanServer Technology, Inc., Fall River, MA [Online]. Available: http://www.ocean-server.com/download/OS5000_Compass_Manual.pdf

[40] *HMC6042 2-Axis Magnetic Sensor Circuit Datasheet*, Honeywell International, Morristown, NJ, 2007.

[41] *HMC1041Z 1-Axis Magnetic Sensor Datasheet*, Honeywell International, Morristown, NJ, 2005.

[42] *Rice Orbit Platform*, Rice Efficient Computing Group, Houston, TX [Online]. Available: http://www.ruf.rice.edu/∼mobile/orbit

[43] *KXM52 Series Analog Output Accelerometers and Inclinometers*, Kionix, Inc., Ithaca, NY, 2005.

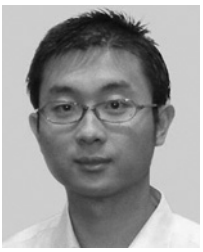
[44] *Texas Instruments DM6446 H.264 Media Processor* [Online]. Available: http://focus.ti.com.cn/cn/lit/an/spraad6a/spraad6a.pdf

[45] H. H. Chen, L. Chia-Kai, P. Yu-Chun, and C. Hung-An, "Integration of digital stabilizer with video codec for digital video cameras," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 7, pp. 801–813, Jul. 2007.

[46] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," in *Proc. IEEE Int. Conf. CVPR*, Jun. 1997, pp. 1106–1112.

[47] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *Proc. IEEE ICCV*, Sep. 1999, pp. 666–673.

[48] *Camera Calibration Toolbox for MATLAB* [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc



Xiaoming Chen (M'09) received the B.S. degree in computer science from the Royal Melbourne Institute of Technology, Melbourne, Australia, in 2003, and the M.Info.Tech. and Ph.D. degrees from the University of Sydney, Sydney, Australia, in 2005 and 2009, respectively.

From 2008 to 2010, he was a Research Fellow with the National University of Singapore, Singapore, and Nanyang Technological University, Singapore. He is currently jointly appointed as a Research Fellow with Digital Media Lab, Umeå University, Umeå, Sweden, and Autonomous Systems Lab, Commonwealth Scientific and Industrial Research Organization, Queensland, Australia. His current research interests include the areas of multimedia computing, image/video processing, low-power media processing for mobile devices, geometry video coding, processing and retrieval, and computer vision.

Dr. Chen was the recipient of the University of Sydney Postgraduate Award and the ETRI (Korea) Scholarship, both in 2005, and the China National Outstanding Overseas Postgraduate Award in 2008. He is a reviewer of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY and *ACM International Conference on Multimedia*.

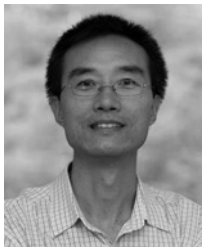
Zhendong Zhao received the B.E. and M.E. degrees from the Nanjing University of Posts and Telecommunications, Jiangsu, China, in 2005 and 2008, respectively, and the M.S. degree from the National University of Singapore, Singapore.

He is currently a Research Assistant with the School of Computing, National University of Singapore. His current research interests include multimedia information retrieval, large-scale machine learning algorithm, and data mining.

Ahmad Rahmati (S'09) received the B.S. degree in computer engineering from the Sharif University of Technology, Tehran, Iran, in 2004, and the M.S. degree in electrical and computer engineering from Rice University, Houston, TX, in 2008. He is currently pursuing the Ph.D. degree from Rice University.

He was a Research Intern with AT&T Labs-Research, Florham Park, NJ, Motorola Laboratories, Schaumburg, IL, and Deutsche Telekom Research and Development Laboratories, Los Altos, CA, in 2006, 2008, and 2010, respectively. His current research interests include the design and applications of efficient, context-aware mobile and pervasive systems, system power analysis and optimization, and human–computer interaction.

Mr. Rahmati received the ACM MobileHCI Best Paper Award in 2007.



Ye Wang (M'00) received the B.S. degree in telecommunications from the South China University of Technology, Guangzhou, China, in 1983, the M.S. degree in telecommunications from the Braunschweig University of Technology, Braunschweig, Germany, in 1993, and the Ph.D. degree in information technology from the Tampere University of Technology, Tampere, Finland, in 2002.

He is currently an Assistant Professor with the School of Computing, National University of Singapore (NUS), Singapore. Since April 2010, he has been a Faculty Member with the NUS Graduate School for Integrated Sciences and Engineering. Before joining NUS in 2002, he was a Research Engineer and a Senior Research Engineer with Nokia Research Center, Tampere, from 1994 to 2002. His research has evolved from error robust audio streaming and low power media processing for portable devices to sound and music analysis and retrieval. A significant portion of his research at NUS was exploring perception-aware low-power media processing methods and techniques for battery-powered mobile devices, addressing the fundamental tension between computational workload (and, hence, power consumption) and user perceived quality-of-service. His current research interests include large-scale music information retrieval with applications in healthcare and edutainment. His research has been funded by the Singaporean Ministry of Education, Ministry of Community Development, Youth and Sports, Nokia, and others.

Dr. Wang is the co-author of the paper that won the Best Student Paper Award at ACM MM2004. He has regularly served on the program committees of ACM multimedia conferences.

Lin Zhong (S'02–M'06) received the B.S. and M.S. degrees from Tsinghua University, Beijing, China, in 1998 and 2000, respectively, and the Ph.D. degree from Princeton University, Princeton, NJ, in September 2005.

He was with NEC Laboratories, Princeton, in 2003, and with Microsoft Research, Redmond, WA, in 2004 and 2005. He was with the Department of Electrical and Computer Engineering, Rice University, Houston, TX, in September 2005. His research interests include mobile and embedded system design, human–computer interaction, and nanoelectronics. His research has been funded by the National Science Foundation, Arlington, VA, Applied Research Center, Motorola Mobility, Schaumburg, IL, Nokia, Finland, Texas Instruments, Dallas, TX, and Microsoft Research.

Dr. Zhong received the AT&T Asian-Pacific Leadership Award in 2001 and the Harold W. Dodds Princeton University Honorific Fellowship in 2004–2005. He has co-authored a paper that was selected as one of the 30 most influential papers in the first ten years of the Design, Automation and Test in Europe Conferences. He and his students received the Best Paper Awards from the ACM MobileHCI in 2007 and the IEEE PerCom in 2009.