Detection of Moving Objects with Non-Stationary Cameras in 5.8ms: Bringing Motion Detection to your Mobile Device

Kwang Moo Yi, Kimin Yun, Soo Wan Kim, Hyung Jin Chang, Hawook Jeong and Jin Young Choi
Seoul National University, ASRI
Daehak-dong, Gwanak-gu, Seoul, Korea
{kmyi, ykmwww, soowankim, changhj, hwjeong, jychoi}@snu.ac.kr

Abstract

*Detecting moving objects on mobile cameras in real-time is a challenging problem due to the computational limits and the motions of the camera. In this paper, we propose a method for moving object detection on non-stationary cameras running within **5.8** milliseconds (ms) on a PC, and real-time on mobile devices. To achieve real time capability with satisfying performance, the proposed method models the background through dual-mode single Gaussian model (SGM) with age and compensates the motion of the camera by mixing neighboring models. Modeling through dual-mode SGM prevents the background model from being contaminated by foreground pixels, while still allowing the model to be able to adapt to changes of the background. Mixing neighboring models reduces the errors arising from motion compensation and their influences are further reduced by keeping the age of the model. Also, to decrease computation load, the proposed method applies one dual-mode SGM to multiple pixels without performance degradation. Experimental results show the computational lightness and the real-time capability of our method on a smart phone with robust detection performances.*

1. Introduction

Detection of moving objects in a scene is without doubt an important topic in the field of computer vision. It is one of the basic steps in many vision-based systems. For example, applications such as human computer interface (HCI), robot visions, and intelligent surveillance systems [6] require detection of moving objects. Various methods have been proposed and have proven to be successful for detection of moving objects in case of stationary cameras [15, 2, 9, 14], but in case of mobile or pan-tilt-zoom (PTZ) cameras these methods do not work well due to many unaccounted factors that arise when using of movable camers.

Many methods have been proposed for moving object detection with a non-stationary camera, but the applicability of them are still doubtful. The most critical reason restricting the applicability of these methods is the amount of computation they require to work. In [4], the authors noted that it took 30 to 60 seconds per frame with their unoptimized MATLAB implementation, and also noted that [13] takes over 2.6 seconds. The method proposed in [10] also requires much computation and cannot run in realtime since they use dense optical flows and nonparametric belief propagation (BP) to optimize a Markov random field (MRF). Therefore, even though these algorithms may provide promising results off-line, in realtime, they are unusable unless a machine with great computation power is provided. The methods presented in [8, 7] work in realtime, but they are still not enough when considering that other visual inference task are usually performed after detection, or when considering platforms with less computation power. Smart phones or embedded platforms, such as robots or head mount displays, would be examples of platforms with relatively low computational power which could benefit much from fast motion detection.

Figure 1. Example of the proposed method running on a mobile device (foreground pixels highlighted in red). Our implementation runs approximately 20 frames per second, comfortably in realtime.

To reduce the computation load of methods targeted for non-stationary cameras, it is important that the model design itself also considers the computation load required for

978-0-7695-4990-3/13 $26.00 © 2013 IEEE
DOI 10.1109/CVPRW.2013.9

applying the model, such as the computation load arising from motion compensation. For example, the method proposed by Barnich and Droogenbroeck [1] is one of the well-known fast background subtraction algorithms showing robust performances. However, when applied to non-stationary cameras, the motion compensation procedure for the algorithm requires computation load proportional to the number of samples used for a pixel. This could slow down the method in significant amounts (usually requiring more computation than the detection algorithm itself), unless there is some sort of hardware support.

Besides the computation load, when modeling the scene, it is also important that the model considers not only the errors and noises that arise in stationary cameras, but also the errors that arise when compensating for the motion of the camera. This is a critical reason that we cannot just simply apply background subtraction algorithms for stationary cameras with simple motion compensation techniques. Stationary camera background modeling algorithms usually focus on building a precise model for each pixel. But for non-stationary case, we cannot guarantee that the model used to evaluate a pixel is actually relavant to that pixel. Even the slightest inaccuracy in motion compensation could end up in making the algorithm use wrong models for some pixels. To account for such motion compensation errors, in [12, 8, 11], small nearby neighborhoods are considered. However, considering neighborhoods increases the necessary computation, slowing down the whole algorithm.

In this paper, we propose a method for detecting moving objects on a non-stationary camera in realtime. Furthermore, our method is not aimed to work in realtime for PC environments only, but for mobile devices as well. Our background model is designed in a way that minimizes computational requirements and shows robust detection performances. The novel dual-mode SGM with age in the proposed model prevents our relatively simple model from being harmed by the foreground. The motion compensation is performed in a way specifically tuned to our model. The compensation is done in a way so that not much warping computation is required, and the model tries to learn compensation errors within the model itself. Experimental results show that our method requires average of **5.8** milliseconds to run for a 320×240 image sequence on a desktop PC, with acceptable detection performance compared to other state-of-the-art methods. Also, our implemetation of the proposed method on a smart phone is able to run in real-time.

2. Proposed Method

The proposed method consists of three major parts; preprocessing to reduce noise, background modeling through the novel dual-mode SGM with age, and the specifically tuned motion compensation for the background movements by mixing models. Figure 2 is an illustration of the framework. Pre-processing on the image is performed with simple spatial gaussian filtering and median filtering on the image. To reduce the amount of computation required, the same model is used for multiple pixels (grids). To cope with the errors arising from this configuration, dual-mode SGM with age is proposed. The proposed model prevents the background model being contaminated by foreground and noise, while still robustly learning the background. The motion compensation is performed in a simple manner, with traditional KLT [16]. However, rather than *moving* the SGM model to its correct positions, similar to [10], we *mix* the background models from the previous frame to construct a model for the present frame. Finally, we obtain the detection results using the trained model.

2.1. SGM model with Age

One of the main reasons for background subtraction methods with statistical models failing for non-stationary cameras is that they usually have a fixed learning rate. Having a fixed learning rate means that for a pixel, the first observation of the pixel is being considered as the mean of an infinitely learned model. This does not cause critical problems for stationary camera since for many cases the pixel value of a certain pixel does not change much for background pixels. However, for non-stationary cameras, motion compensation errors are apt to exist no matter how accurate the compensation is, and we cannot assume that the first observation of a pixel would be similar to the mean value we would actually get by acquiring further observations. Therefore, we need a varying learning rate. Although the authors of [5] thought it to be a problem with initialization and fast adaptation, the notion of constructing a model with expected sufficient statistics works nicely for non-stationary cameras as well. In [8], the authors also use the *age* of a pixel to define a variable learning rate, which is actually the same as using expected sufficient statistics.

To model the scene, we use SGMs. With the notion of sufficient statistics, to use only the observed data to form a model, as in [8] we keep the age of a SGM as well as its mean and variance. Also, to reduce the computation load, we divide the input image into equal grids of size $N \times N$ and keep one SGM for each grid. If we denote the group of pixels in grid i at time t as $\mathbf{G}_i^{(t)}$, the number of pixels in $\mathbf{G}_i^{(t)}$ as $\left|\mathbf{G}_i^{(t)}\right|$, and the observed pixel intensity of a pixel j at time t as $I_j^{(t)}$, then the mean $\mu_i^{(t)}$, the variance $\sigma_i^{(t)}$, and the age $\alpha_i^{(t)}$ of the SGM model applied to $\mathbf{G}_i^{(t)}$ is updated as

$$\mu_i^{(t)} = \frac{\tilde{\alpha}_i^{(t-1)}}{\tilde{\alpha}_i^{(t-1)}+1}\tilde{\mu}_i^{(t-1)} + \frac{1}{\tilde{\alpha}_i^{(t-1)}+1}M_i^{(t)} \qquad (1)$$

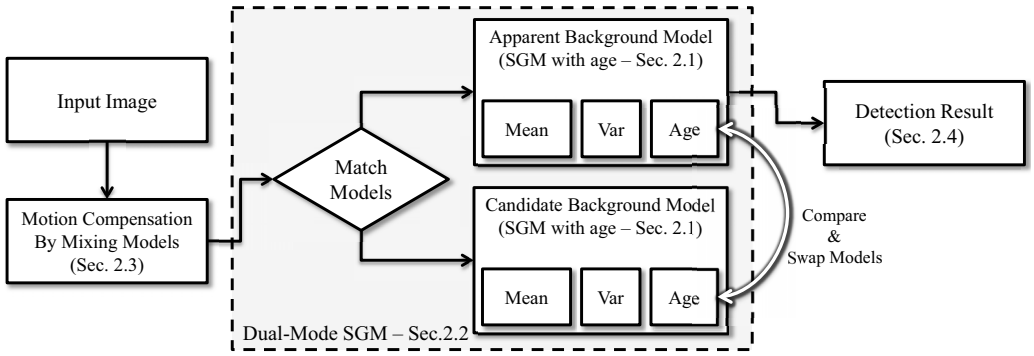


Figure 2. Framework of the proposed method

$$\sigma_i^{(t)} = \frac{\tilde{\alpha}_i^{(t-1)}}{\tilde{\alpha}_i^{(t-1)}+1}\tilde{\sigma}_i^{(t-1)} + \frac{1}{\tilde{\alpha}_i^{(t-1)}+1}V_i^{(t)} \quad (2)$$

$$\alpha_i^{(t)} = \tilde{\alpha}_i^{(t-1)} + 1 \quad (3)$$

where $M_i^{(t)}$ and $V_i^{(t)}$ are defined as

$$M_i^{(t)} = \frac{1}{|\mathbf{G}_i|}\sum_{j\in\mathbf{G}_i} I_j^{(t)} \quad (4)$$

$$V_i^{(t)} = \max_{j\in\mathbf{G}_i}\left(\mu_i^{(t)} - I_j^{(t)}\right)^2 \quad (5)$$

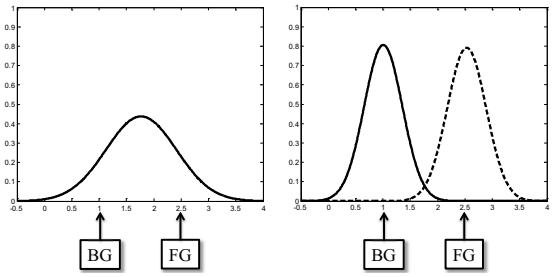
and $\tilde{\mu}_i^{(t-1)}$, $\tilde{\sigma}_i^{(t-1)}$, and $\tilde{\alpha}_i^{(t-1)}$ denote the SGM model of time $t-1$ compensated for use in time t which will be discussed in detail at Section 2.3. Note that (5) is used instead of

$$V_i^{(t)} = \left(\mu_i^{(t)} - M_i^{(t)}\right)^2. \quad (6)$$

Since the SGM model is for grid $\mathbf{G}_i$, (6) would seem logical. However, in our case, because the model is applied to multiple pixels, some pixels within the grid may be considered to be outliers with (6). Therefore, we learn (5) instead to prevent such false foregrounds. The advantage of the proposed SGM model for grids with age is that it allows the motion compensation errors to be learned in the model properly with a variable learning rate based on the age of a model. Also, having the number of SGMs less than the number of pixels reduces computation load.

2.2. Dual-Mode SGM

Using a SGM to model the scene usually works well in simple cases, however, when fast learning rates are used, the background model suffers from getting contaminated with the data coming from foreground pixels. In our method, since we have a variable learning rate, having a fast learning rate is a common case. For example at initialization, all pixels start with age of one, meaning that the learning rate of these pixels at next frame would be 0.5. As illustrated in Figure 3 (a), the fast learning rate causes the background model to describe some portion of the foreground as well.



(a) SGM (b) Dual-Mode SGM

Figure 3. Illustration of the effects of using dual-mode SGM. Learning results of (a) SGM and (b) dual-mode SGM with the same data. The same 1-D data generated with gaussian noise centered at 1 and 2.5 with standard deviation 0.5 is given. In (b), the solid line denotes the apparent background model whereas the dotted line denotes the candidate background model.

This can be seen easily in the case of large objects passing through the scene. A naive solution to this problem would be to update with only the pixels determined as the background. However, in this case, a single misclassification of pixel would have an everlasting effect on the model since false foregrounds would never be learned.

To overcome this defect, we use another SGM which acts like a candidate background model. The candidate background model remains ineffective until its age becomes older than the apparent background model, when, at that time, the two models are swapped. This dual-mode SGM is different from Gaussian mixture models (GMM) [15] with two modals, considering the fact that using a bi-modal GMM would still have the foreground data contaminating the background whereas our method does not. If we denote the mean, variance, and age of the candidate background model and the apparent background model at time t for grid i as $\mu_{C,i}^{(t)}$, $\sigma_{C,i}^{(t)}$, and $\alpha_{C,i}^{(t)}$, and $\mu_{A,i}^{(t)}$, $\sigma_{A,i}^{(t)}$, and $\alpha_{A,i}^{(t)}$, respectively, then, if the squared difference between the observed mean $M_i^{(t)}$ and $\mu_{A,i}^{(t)}$ is less than a threshold with respect to $\sigma_{A,i}^{(t)}$, *i.e.*

$$\left(M_i^{(t)} - \mu_{A,i}^{(t)}\right)^2 < \theta_s \sigma_{A,i}^{(t)}, \quad (7)$$

we update $\mu_{A,i}^{(t)}$, $\sigma_{A,i}^{(t)}$, and $\alpha_{A,i}^{(t)}$ according to (1), (2), and (3), where θ_s is a threshold parameter. Also if the above condition does not hold and if the observed mean matches the candidate background model,

$$\left(M_i^{(t)} - \mu_{C,i}^{(t)}\right)^2 < \theta_s \sigma_{C,i}^{(t)}, \tag{8}$$

then we update $\mu_{C,i}^{(t)}$, $\sigma_{C,i}^{(t)}$, and $\alpha_{C,i}^{(t)}$ according to (1), (2), and (3). If none of the conditions hold, we initialize the candidate background model with the current observation. When updating according to this process, only one of the two models is updated and the other remains untouched.

Afther updating, the two background models for grid i are swapped if the age of the candidate exceeds the apparent meaning,

$$\alpha_{C,i}^{(t)} > \alpha_{A,i}^{(t)}. \tag{9}$$

The candidate background model is initialized after swapping. Finally, we only use the apparent background model, which is now an uncontaminated background model, when determining foreground pixels in Section 2.4. Through the dual-mode SGM, we can prevent the background model from being corrupted by the foreground data. As in Figure 3 (b), the foreground data is learned by the candidate background model rather than the apparent background model. Also, we do not have to worry about false foregrounds never being learned into the model since if the age of the candidate background model becomes larger than the apparent background model, the models will be swapped and correct background model will be used.

2.3. Motion Compensation by Mixing Models

For image sequences obtained from a non-stationary camera, the model learned until time $t-1$ cannot be used directly for detection in time t. To use the model, motion compensation is required. However, since we use a single model for all the pixels inside a grid (*i.e.* a single model for all j such that $j \in \mathbf{G}_i^{(t)}$), simple warping on the background model based on interpolation strategies would cause too much error. Thus, instead of simply warping the background model, we construct the compensated background model at time t by merging the statistics of the model at time $t-1$. For obtaining the background motion, we divide the input image at time t into 32×24 grids, and perform KLT [16] on every corner of the grid with the image from time $t-1$. With these point tracking results, we perform RANSAC [3] to obtain a homography matrix $\mathbf{H}_{t:t-1}$ which warps all pixels in time t to pixels in time $t-1$ through a perspective transform. We consider this to be the movement of the background. For further explanation, we will denote the position of pixel j as $\mathbf{x}_j$, the position of the center for $\mathbf{G}_i^{(t)}$ as $\bar{\mathbf{x}}_i^{(t)}$, and the perspective transform of $\mathbf{x}$ according to $\mathbf{H}_{t:t-1}$ as $f_{PT}\left(\mathbf{x}, \mathbf{H}_{t:t-1}\right)$.

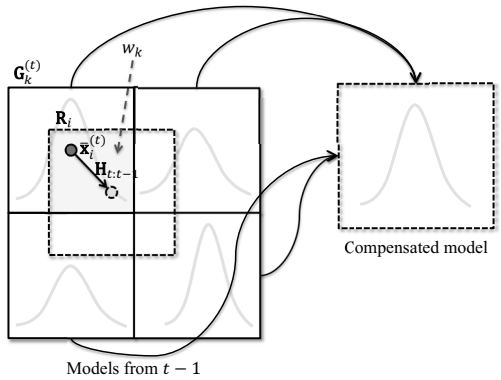


Figure 4. Illustration of the proposed motion compensation by mixing models. Models of overlapping regions are mixed together to a single model.

As in Figure 4, for each grid i, we consider $\bar{\mathbf{x}}_i^{(t)}$ to have moved from $f_{PT}\left(\bar{\mathbf{x}}_i^{(t)}, \mathbf{H}_{t:t-1}\right)$. Then, assuming that there was no significant scale change, if we think of an $N \times N$ square region $\mathbf{R}_i$ having $f_{PT}\left(\bar{\mathbf{x}}_i^{(t)}, \mathbf{H}_{t:t-1}\right)$ as the center, the model at time $t-1$ for $\mathbf{R}_i$ would be the motion compensated background model for pixels $\mathbf{G}_i^{(t)}$. $\mathbf{R}_i$ overlaps with multiple grids and we mix the SGMs at time $t-1$ of these overlapping grids to obtain $\tilde{\mu}_i^{(t-1)}$, $\tilde{\sigma}_i^{(t-1)}$, and $\tilde{\alpha}_i^{(t-1)}$ of the motion compensated background model. The mixture weights are assigned so that they are proportional to the overlapping area. If we denote the group of grids overlapping with $\mathbf{R}_i$ as $\mathbf{O}_i^{(t)}$, and the mixture weights as w_k where $k \in \mathbf{O}_i^{(t)}$, then we obtain the compensated background model simply by mixing the Gaussian models

$$\tilde{\mu}_i^{(t-1)} = \sum_{k \in \mathbf{O}_i^{(t)}} w_k \mu_k^{(t-1)} \tag{10}$$

$$\tilde{\sigma}_i^{(t-1)} = \sum_{k \in \mathbf{O}_i^{(t)}} w_k \left[\sigma_k^{(t-1)} + \left\{\mu_k^{(t-1)}\right\}^2 - \left\{\tilde{\mu}_i^{(t-1)}\right\}^2\right] \tag{11}$$

$$\tilde{\alpha}_i^{(t-1)} = \sum_{k \in \mathbf{O}_i^{(t)}} w_k \alpha_k^{(t-1)}, \tag{12}$$

where the mixing weights are

$$w_k \propto Area\left\{\mathbf{R}_i \cap \mathbf{G}_k^{(t)}\right\} \tag{13}$$

and

$$\sum_k w_k = 1. \tag{14}$$

Note that in (11) $\left\{\mu_k^{(t-1)}\right\}^2 - \left\{\tilde{\mu}_i^{(t-1)}\right\}^2$ exists, since when we merge multiple Gaussian models into one, it is the expectation for the square of the observation that is merged

with weight w_k and not the variance itself. During the mixing process, as in Figure 4, models where nearby regions differ a lot (*e.g.* edges) will have excessively large variances after compensation (as in (11)). Normally, a SGM would not have too large variances, and having such large variance would mean that the model has not learned much of the target object. Therefore, after compensation, if the variance is over a threshold θ_v, *i.e.* $\tilde{\sigma}_i^{(t-1)} > \theta_v$, we reduce the age of the model as

$$\tilde{\alpha}_i^{(t-1)} \leftarrow \tilde{\alpha}_i^{(t-1)} \exp\left\{-\lambda\left(\tilde{\sigma}_i^{(t-1)} - \theta_v\right)\right\}, \qquad (15)$$

where λ is a decaying parameter. Through this decaying of age, we prevent the model from having a model with too large variance. This especially helps removing false foregrounds near edges.

2.4. Detection of Foreground Pixels

After obtaining the background model for time t as in Section 2.1, 2.2, and 2.3, we select pixels that have distances larger than threshold from the mean as foreground pixels. This is not an exact solution to the problem theoretically, since to be exact, we should find pixels with lower probability of being the background with respect to the learned background model. However, finding such pixels require much computation, due to the square-root and natural logarithms operations in the exact equation [15]. We found empirically that the results are useable even with simple thresholding with respect to the variance, without complicated computation. Mathematically, for each pixel j in group i, we classify the pixel as a foreground pixel if

$$\left(I_j^{(t)} - \mu_{A,i}^{(t)}\right)^2 > \theta_d \sigma_{A,i}^{(t)}, \qquad (16)$$

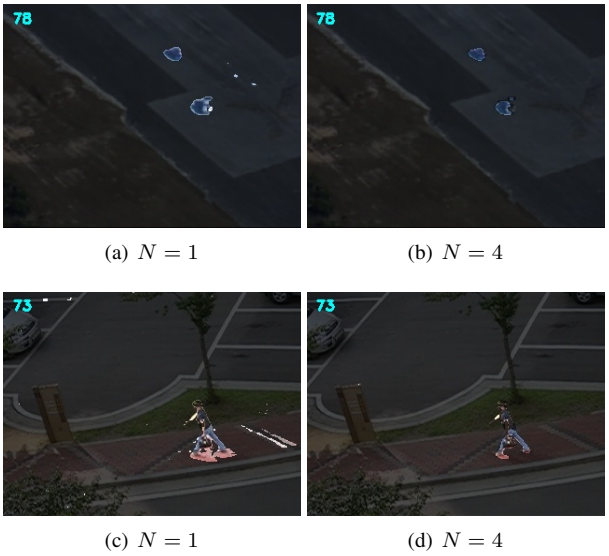
where θ_d is a threshold parameter. Note that we only use the apparent background model in determining the foreground. Through this way, we can avoid false backgrounds arising from contamination of the background model.

3. Experiments

For the experiments, the proposed method was implemented using C++ with the KLT from OpenCV[1] library. For the parameters, the grid size $N = 4$, the threshold for matching $\theta_s = 2$, the decaying parameter for age $\lambda = 0.001$, the threshold for decaying age $\theta_v = 50 \times 50$, and the threshold for determining detection $\theta_d = 4$. For the initialization of the variance, we simply set variance to be a moderate value (*e.g.* 20×20). The age was truncated at 30 to keep a minimum learning rate. The method was experimented with eight image sequences, where six is identical to the sequences used in [8], one is our own, and the last one is downloaded from YouTube[2].

[1] http://opencv.org/downloads.html

[2] http://www.youtube.com/watch?v=0K4XKTx7T4g



(a) $N = 1$ (b) $N = 4$

(c) $N = 1$ (d) $N = 4$

Figure 6. Detection results with different grid size N for the same sequence.

3.1. Qualitative Comparisons

To evaluate the quality of our method against the state-of-the-art, we compared our method against two other methods by Barnich and Droogenbroeck [1], and by Kim *et al.* [8]. For [1] we implemented the method with the pseudo code provided in their paper and applied simple warping of the background based on the $\mathbf{H}_{t:t-1}$ in Section 2.3. This compensation strategy is also noted in the authors websites[3]. For [8] we used the implementation provided by the authors. Some critical frames are shown on Figure 5. Results for all sequences are provided in the supplementary video.

As shown in the third column of Figure 5, results of [1] with simple motion compensation have many false foregrounds. Most error arises near the edges, showing that we cannot simply use a method designed for stationary cameras in case of non-stationary cameras even with motion compensation. Our method (second column) generally outperforms or is comparable to the method proposed by Kim *et al.* [8] (last column), but as shown in the second row, in cases where the parts of the foreground is similar to the background, false backgrounds do occur. Still, the performance of our method is acceptable even in such cases. Note that in the third row, a pedestrian walking by is detected even though the size of the pedestrian is very small.

3.2. Runtime and Effects of Grid Size

To demonstrate the computational efficiency of our method, we measured the computation time required for each major steps compared to other methods. We have also

[3] http://www2.ulg.ac.be/telecom/research/vibe/

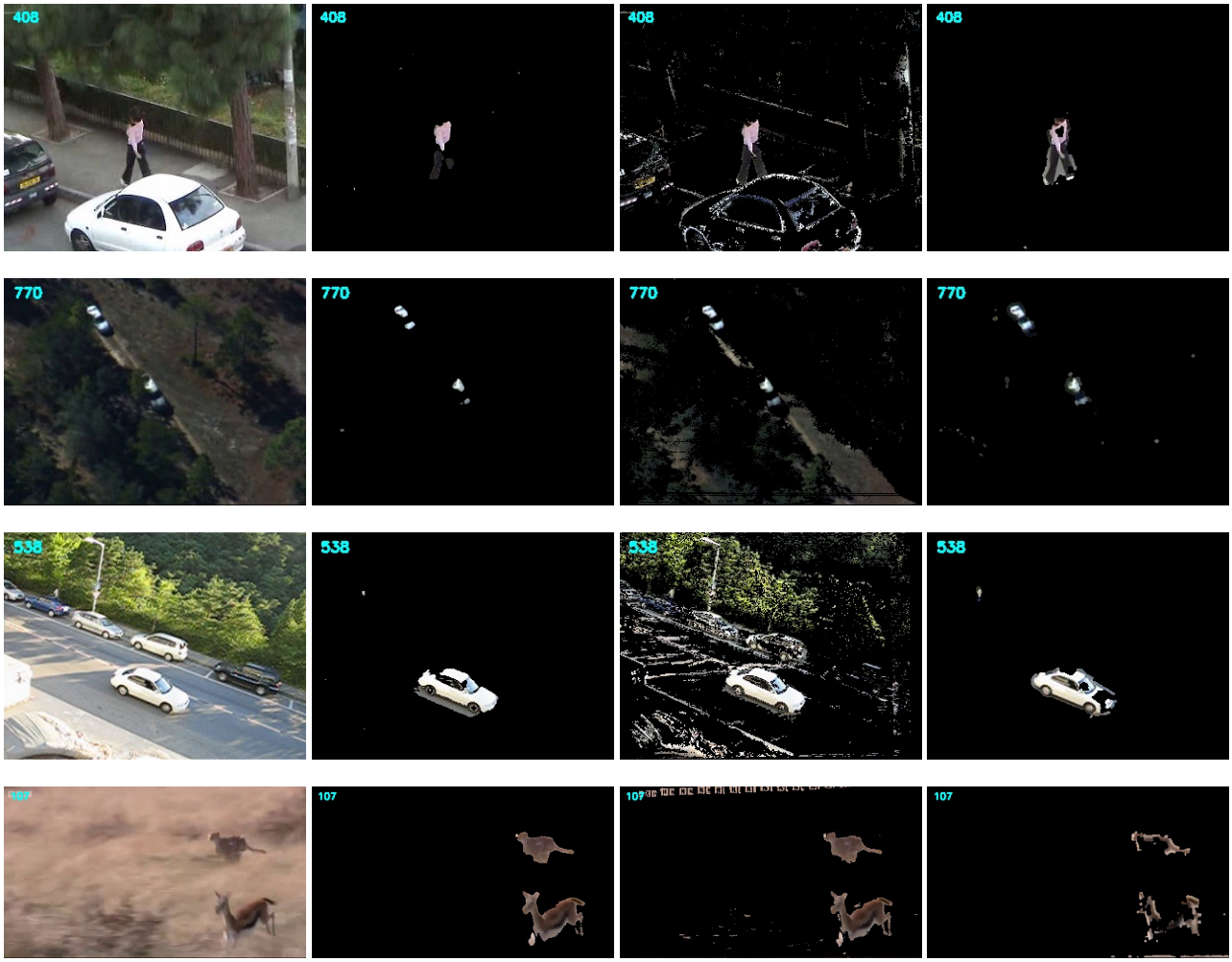
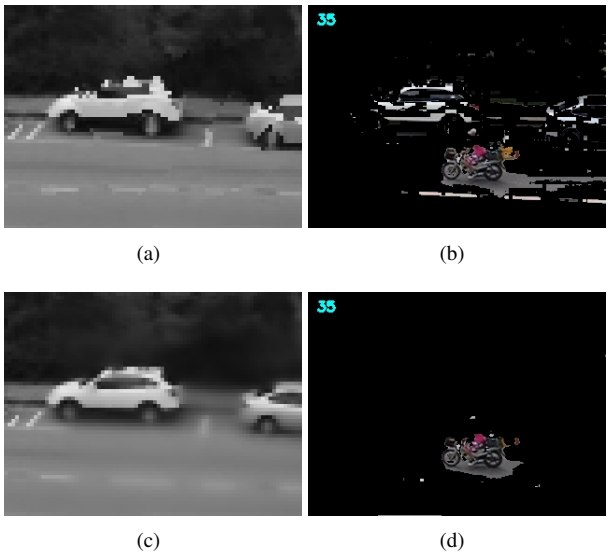


Figure 5. Comparison results against other methods. Input frames (first column), proposed method (second column), method of Barnich and Droogenbroeck [1] with simple motion compensation (third column), and method by Kim *et al*. [8] (fourth column).

	Pre-Processing	Motion Comp.	Modeling	Post-Processing	Total
Ours with $N = 4$ and OpenMP	0.82ms	2.00ms	0.88ms	-	3.71ms
Ours with $N = 4$	0.84ms	2.65ms	2.31ms	-	5.81ms
Ours with $N = 1$	0.84ms	18.72ms	6.40ms	-	25.96ms
Barnich & Droogenbroeck [1]	-	7.20ms	4.15ms	-	11.35ms
[1] with our motion compensation	-	39.22ms	3.64ms	-	42.86ms
Kim *et al*. [8]	1.15ms	5.90ms	2.32ms	7.85ms	17.23ms

Table 1. Average computation time for each method.

measured our method with parallel processing used (implemented with OpenMP), and in case of $N = 1$, which means that all pixels have their own dual-mode SGM. All experiments were performed on an Intel Core i5-3570 3.4GHz PC with 320×240 image sequences. Table 1 is the average runtime required for each algorithm. As shown in Table 1, our method outperforms other methods with respect to computation load. Even our method with $N = 1$ runs in average of **25.96**ms assuring real-time performance. The computation time of the method by Barnich and Droogenbroeck [1] is comparable to our method (11.35ms) but shows relatively poor performance considering the quality of detection results (as previously discussed in detail in Section 3.1.) Also, the influence of the computation load arising from motion compensation increases when a more precise motion compensation method, such as our compensation method, is used rather than simple warping. The method by Kim *et al*. [8] also require computation time suitable for real-



(a) (b)

(c) (d)

Figure 8. Example results when simple nearest neighbor warping is performed ((a) and (b)) and when the proposed motion compensation by mixing models is performed ((c) and (d)). (a) and (c) are the means of the apparent background model, and (b) and (d) are the detection results.

time performance on a PC, but is still computationally expensive to run on a machine with relatively less computation power. Furthermore, a simple parallel implementation with OpenMP reduces the required computation even more, making our method run in **3.71**ms on average. As shown in Figure 6, the detection performance of our method does not degrade with respect to N. Rather, false foregrounds arising from motion compensation errors near edges are reduced.

3.3. Effects of Dual-Mode SGM

As noted in Section 2.2, using one SGM model causes the background model to be contaminated by the foreground. This causes problems when the scene contains objects that are relatively large or moving slowly, since they will contaminate the background significantly and affect the final detection result. An example of this is shown in Figure 7. In Figure 7 (a), it can be seen that traces of moving objects are left in the background model when using one SGM model, whereas in (b) and (c), with dual-mode SGM, the traces are learned in the candidate background model (c) and the apparent background model (b) is preserved and clear. As in (d) and (e), this degradation of the background model decreases the performance of the detection algorithm.

3.4. Effects of Motion Compensation

Since we use the same dual-mode SGM for multiple pixels inside a grid, the motion compensation method proposed in Section 2.3 plays a critical role for the performance of the whole algorithm. If we simply apply a nearest neighbor warping to *move* the pixels, even the SGM model with age is not able to cope with such compensation errors. Figure 8 is an example showing the effectiveness of the proposed compensation scheme. In the case of simple nearest neighbor warping, as shown in (a), the background model becomes distorted due to quantization effects. This results in bad detection performance as shown in (b). However, with the proposed motion compensation scheme, the background motion is well compensated as in (c). Also, as in (d), detection performance is unharmed.

3.5. Mobile Results

We have also implemented our method on a mobile device to further test its real-time capability. The implementation was done on an Quad-core 1.4 GHz Cortex-A9 android device with OpenCV for android[4] used to implement KLT. The implementation does not have any optimization techniques used and is basically the same code for PC tweaked so that it matches the android interface. Our method runs approximately 20 frames per second with the live capture resolution set to 160×120. Figure 9 shows actual still shots of our algorithm running on a mobile device. Although some small details are not detected due to low resolution, it is possible to see that our method performs well in real-time even on a mobile device.

4. Conclusions

A novel method for detecting moving objects in a scene with non-stationary cameras was proposed. The proposed method modeled the background through dual-mode SGM with age to cope with motion compensation errors and to prevent the background model from being contaminated by the foreground. A single dual-mode SGM was applied to multiple pixels to reduce the required computation load, without performance degradation. To reduce errors arising from motion compensation, models were mixed together in the compensation process. Experimental results showed that our method requires significantly less amount of computation, running within **5.8**ms, and yet with robust detection performances. Also, our method was implemented on a mobile device, confirming its real-time capability.

Acknowledgement

The research was sponsored by the SNU Brain Korea 21 Information Technology program.

References

[1] O. Barnich and M. Van Droogenbroeck. ViBe: A universal background subtraction algorithm for video sequences. *IEEE*

[4] http://opencv.org/platforms/android.html

(a) (b) (c) (d) (e)

Figure 7. Example result with SGM and dual-mode SGM. (a) mean of the background model for SGM, (b) mean of the apparent background model of the dual-mode SGM, (c) mean of the candidate background model of the dual-mode SGM, (d) SGM result, and (e) dual-mode SGM result. The mean of the background model of SGM (a) is contaminated by the foreground, whereas the mean of the apparent background model of dual-mode SGM (b) remains unharmed. This leads to different results as in (d) and (e).

Figure 9. Example detection results of our method on a mobile device (foreground pixels highlighted in red). Our implementation runs approximately 20 frames per second, assuring real-time performance.

Transactions on Image Processing, 20(6):1709–1724, June 2011. 2, 5, 6

[2] A. Elgammal, R. Duraiswami, D. Harwood, L. S. Davis, R. Duraiswami, and D. Harwood. Background and foreground modeling using nonparametric kernel density for visual surveillance. In *Proceedings of the IEEE*, pages 1151–1163, 2002. 1

[3] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981. 4

[4] G. Georgiadis, A. Ayvaci, and S. Soatto. Actionable saliency detection: Independent motion detection without independent motion estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. June 2012. 1

[5] P. KaewTrakulPong and R. Bowden. A real time adaptive visual surveillance system for tracking low-resolution colour targets in dynamically changing scenes. *Image and Vision Computing*, 21(10):913 – 929, 2003. 2

[6] I. Kim, H. Choi, K. Yi, J. Choi, and S. Kong. Intelligent visual surveillance? survey. *International Journal of Control, Automation and Systems*, 8(5):926–939, 2010. 1

[7] J. Kim, X. Wang, H. Wang, C. Zhu, and D. Kim. Fast moving object detection with non-stationary background. *Multimedia Tools and Applications*, pages 1–25, 2012. 1

[8] S. Kim, K. Yun, K. Yi, S. Kim, and J. Choi. Detection of moving objects with a moving camera using non-panoramic background model. *Machine Vision and Applications*, pages 1–14. 10.1007/s00138-012-0448-y. 1, 2, 5, 6

[9] T. Ko, S. Soatto, and D. Estrin. Warping background subtraction. In *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, pages 1331–1338. IEEE, 2010. 1

[10] S. Kwak, T. Lim, W. Nam, B. Han, and J. H. Han. Generalized background subtraction based on hybrid inference by belief propagation and bayesian filtering. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2174 –2181, nov. 2011. 1, 2

[11] A. Mittal and D. Huttenlocher. Scene modeling for wide area surveillance and image synthesis. In *In IEEE Conference on Computer Vision and Pattern Recognition*, pages 160–167, 2000. 2

[12] N. I. Rao, H. Di, and G. Xu. Panoramic background model under free moving camera. In *Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery - Volume 01*, FSKD '07, pages 639–643, Washington, DC, USA, 2007. IEEE Computer Society. 2

[13] Y. Sheikh, O. Javed, and T. Kanade. Background subtraction for freely moving cameras. In *ICCV*, pages 1219–1225. IEEE, 2009. 1

[14] Y. Sheikh and M. Shah. Bayesian modeling of dynamic scenes for object detection. *PAMI*, 27:1778–1792, 2005. 1

[15] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2, pages 2 vol. (xxiii+637+663), 1999. 1, 3, 5

[16] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical report, Carnegie Mellon University, Apr. 1991. 2, 4