REAL-TIME MOTION DETECTION BASED ON DISCRETE COSINE TRANSFORM

Tae-Hyun Oh, Joon-Young Lee, In So Kweon

Robotics and Computer Vision Lab, KAIST, Korea

ABSTRACT

We present a motion detection algorithm by a change detection filter matrix derived from Discrete Cosine Transform. Recently, a Fourier reconstruction scheme shows good results for motion detection. However, its computational cost is a major drawback. We revisit the problem and achieve two orders of magnitude faster than the previous algorithm with better performance. The proposed algorithm runs at about 800 frames per second for VGA resolution images on a consumer hardware by using only integer matrix multiplication and the symmetric property of the change detection filter matrix. In addition, our algorithm is fundamentally robust to sudden illumination changes because it works based on edge information. We verify our algorithm with challenging datasets that contain strong and sudden illumination changes.

Index Terms— Motion detection, discrete cosine transform, video surveillance, change detection

1. INTRODUCTION

Motion detection is a fundamental issue for video surveillance and widely used as a preprocessing step for many computer vision and image processing applications, such as event detection, object tracking, behavior recognition, and so on.

For a surveillance system that uses a static camera, background subtraction is a conventional approach to detect moving objects. Background subtraction algorithms compare the difference between an input image and a reference background model. To determine proper threshold values, these methods should learn statistic parameters of environment variations using a Gaussian mixture model [1] [2] [3], kernel density estimation [4], and so on. However, once abrupt changes appear in a scene, algorithms could easily fail to extract moving object regions.

Recently, Tsai and Chiu [5] show that a Fourier reconstruction algorithm can extract exact boundaries of moving objects from a static camera. The algorithm uses frequency analysis on a 3D x-y-t spatial-temporal space compiled by stacking consecutive frames of 2D spatial images. This approach extracts moving object regions by removing background patterns using the Fourier transforms. If there is no moving object over a limited number of consecutive frames, a vertical line pattern exists on a 2D x-t plane which is a slice

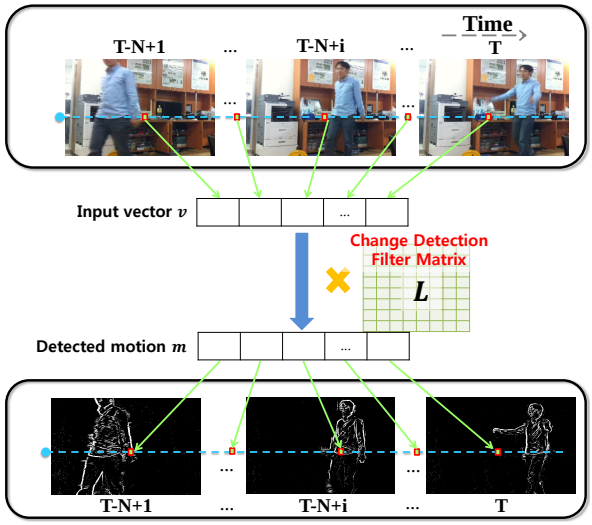


Fig. 1. Overview of our algorithm. We construct an input vector from N recent consecutive frames on the same pixel position. Then, we determine motion boundaries by multiplying a change detection filter L to the input vector as described in Sec. 2

of the 3D x-y-t space. Conversely, if moving objects appear in the x-t slice, the vertical line pattern structure is partially broken. With assuming pixel dependency, the pattern can be efficiently removed by 2D frequency analysis on the x-t plane. It is robust to image noise and adapt well to both gradual and sudden changes. However, the algorithm has high computational complexity, because it requires 2D Fourier transform. Our work is motivated by Tsai and Chiu's Fourier reconstruction(FR) approach [5]. We present a motion detection algorithm using Discrete Cosine Transform (DCT). Our algorithm works two orders of magnitude faster than the FR method [5] with better performance.

2. MOTION DETECTION BY DISCRETE COSINE TRANSFORM

In an image sequence, we consider a sequence of intensities at a pixel position along time axis as a vector $\mathbf{v}$. Then, elements of $\mathbf{v}$ in background regions have similar values along time. If $\mathbf{v}$ is in motion boundaries, abrupt intensity changes appear in $\mathbf{v}$. The abrupt changes in $\mathbf{v}$ can be easily extracted

by high-pass filtering in frequency domain. Frequency domain is robust to noise and intensity variations, and useful to remove static values on individual pixels by high-pass filtering [5].

We choose DCT as a frequency analysis tool because of two reasons. First, DCT is efficient to filter static values due to an energy compaction property. Second, DCT results in only real values, while Fourier transform has both real and imaginary components. DCT can be easily constructed to a matrix form so that we derive a change detection filter from DCT in the following.

Derivation of a change detection filter. Our method is based on temporal consistency of intensities on the same pixel position in recent N frames over time. $N \times N$ DCT matrix D_N is denoted by

$$\{D_N\}_{i,j} = \sqrt{\frac{2}{N}} \cos\left[\frac{\pi}{N}\left(i-\frac{1}{2}\right)\left(j-\frac{1}{2}\right)\right], \quad (1)$$

where i and j are a row and a column index of D_N matrix. To detect moving object boundaries, we use edge images as the input. Therefore, an observation vector $\mathbf{v}$ contains intensities of edge images on one pixel position over the recent N frames.

Abrupt changes in $\mathbf{v}$ can be detected by frequency analysis. The observation vector $\mathbf{v}$ is transformed to frequency domain by DCT and then high-pass filtering is applied to the transformed vector. Finally, we apply inverse DCT to convert the filtered vector into the result vector $\mathbf{m}$ in time domain. All these operations, DCT, high-pass filtering and inverse DCT can be represented into matrix forms. Specifically, we define a $N \times N$ diagonal matrix $E_{N,\Delta w}$ to represent high-pass filtering into matrix form as

$$E_{N,\Delta\, w} = diag(\underbrace{0,\cdots,0}_{\Delta w},\underbrace{1,\cdots,1}_{N-\Delta w}), \quad (2)$$

where Δw is a filter width parameter. Theoretically, the result of our algorithm would suffer from ringing artifact, since Eq. (2) is an ideal high-pass filter. In practice, the artifact is not observed in our experimental results. Magnitudes of low frequencies in the transformed vector are relatively small because most of texture-less regions have zero values in edge images. Thus, the artifact due to the filtering are negligible.

Combining all the operations in matrix forms, we can solve the motion detection problem as matrix multiplications by

$$\mathbf{m} = {D_N}^T \cdot E_{N,\Delta w} \cdot D_N \cdot \mathbf{v} = L \cdot \mathbf{v}. \quad (3)$$

We call the matrix L in Eq. (3) as a change detection filter, which can be pre-calculated. Consequently, we construct an observation vector $\mathbf{v}$ for each pixel in the input, then we can get a motion detection result $\mathbf{m}$ for each pixel by multiplying L and $\mathbf{v}$.

In an implementation, we only need one component in a vector $\mathbf{m}$ for fast computation. Namely we use one component in the middle that is less affected by Gibbs effect of DCT. Also, our method processes each pixel independently, therefore it is very suitable for parallel processing.

Fixed point implementation. To implement our algorithm, we need floating point operations because the change detection filter L has floating values due to DCT basis. To reduce computational complexity, we can convert floating point operations into fixed point operations by scaling. Converting procedures from floating values to fixed values consist of up-scaling, quantization, change detection filtering and down-scaling.

For up/down-scaling the change detection matrix efficiently without overflow, we only consider bit shift operator. With bit shift operator, a proper up-scaling level satisfy the following inequality:

$$2^l \left(i_{max}|L|_{max}N\right) < 2^{B-1}, \quad (4)$$

where l is a bit shift level for scaling, i_{max} is the maximum intensity level (e.g., 255), $|L|_{max}$ is the maximum absolute value in the matrix L, N is the size of the $\mathbf{v}$, and B is the number of system data bus bits (e.g., 32 or 64 bits). To determine the maximum integer value of l from Eq. (4), we derive

$$l = floor(B - 1 - \log_2(i_{max}N|L|_{max})). \quad (5)$$

Therefore, an up-scaled change detection filter $\tilde{L}$ is given by

$$\tilde{L} = round\left(2^l L\right), \quad (6)$$

where $round$ is an element-wise rounding operator. In Eq. (3), we substitute L with $\tilde{L}$, then we can get a result using only fixed point operations. To maintain the scale level of $\mathbf{v}$, down-scaling is required to $\tilde{L} \cdot \mathbf{v}$. Therefore, the fixed point version of Eq. (3) is denoted by

$$\mathbf{m}_f \simeq 2^{-l}\left(\tilde{L} \cdot \mathbf{v}\right) = 2^{-l}\left(round\left(2^l L\right) \cdot \mathbf{v}\right), \quad (7)$$

where $\mathbf{m}_f$ is the motion detection result with fixed point values. In Eq. (7), 2^l and 2^{-l} are implemented by left and right bit shift operators. Also, $\tilde{L}$ can be pre-calculated.

3. EXPERIMENTS

In this section, we perform experiments to evaluate our algorithm. For our method, we set the number of temporal image frames N to 5 and the bit shift level l to 23. To extract edge images, we use a 3x3 Sobel filter. Tsai and Chiu [5] show that setting the first and the second frequency components to zero is suitable for removing static values from background and for reducing small variations of magnitude in the vector $\mathbf{m}$. Therefore we also use the filter width $\Delta w = 2$ to extract pixels in moving object boundaries.

For evaluation, we compare our algorithm to simple temporal difference [6], one of the state-of-the-arts background subtraction algorithm [3] and the Fourier reconstruction [5].

Temporal difference is implemented by the following equation:

$$\frac{|I_t(\mathbf{p}) - I_{t-1}(\mathbf{p}) - \mu_d|}{\sigma_d}, \tag{8}$$

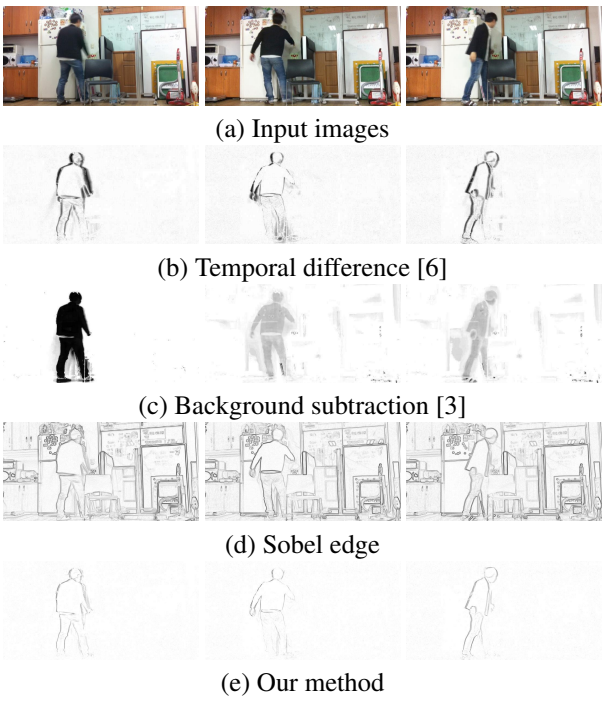
where $I_t(\mathbf{p})$ is an intensity at a pixel position $\mathbf{p}$ in an input sequence at time t, μ_d and σ_d are mean and standard deviation of the difference between two images, $I_t - I_{t-1}$. Pilet *et al.*'s method [3] is robust to sudden illumination changes due to robustness of Normalized Cross Correlation (NCC) for illumination changes. Our implementation of the FR method [5] use FFT algorithm in FFTW C++ library[1]. We compare the algorithms with two challenging datasets. Both datasets are videos of 720×480 and 640×480 (VGA) resolution, and have strong and sudden illumination changes to demonstrate robustness to illumination changes of each algorithm. One of the datasets was captured with an auto-exposure setting and the other one was captured with a fixed-exposure setting.

Fig. 2 and Fig. 3 show comparison results with both datasets. We put Sobel edge images to compare with our motion boundary images. In both figures, temporal difference algorithm [6] fails to extract accurate boundaries of motion parts because it is weak for local illumination changes and it extracts not only motion boundaries but also union regions of differences observed in both images. Especially, the results of temporal difference [6] in Fig. 3 show poor performances due to low contrast. Background subtraction [3] works reasonably for most input sequences, however it fails to detect moving object regions when large illumination changes happen in short duration.

On the other hand, we can observe both the FR [5] and our method detect motion boundaries well, even there are large illumination changes in short duration. It means both algorithms based on frequency analysis are more robust to sudden illumination changes than the others. Basically, the FR and the proposed method come from similar analyses. Therefore both algorithms show similar results in both Fig. 2 and Fig. 3. A detailed comparison between the FR and our method is presented in Fig. 4. Our method has clearer boundaries because the process of the FR algorithm affects neighbor pixels each other due to the pixel-dependency assumption of 2D frequency analysis. It causes noisier boundaries in the FR than our motion boundaries that each pixel is processed independently.

We compare computation time of both the FR [5] and our methods in Table 1. Algorithms are tested on Intel i7 3.0 GHz processor. With the general floating point implementation in Eq (3), our method is around 25 times faster than the FR method because we do not perform 2D FFT operation and we can utilize the pre-calculation of the change detection filter L. In case of the fixed point implementation in Eq (7), it takes about 1ms for processing one VGA image and we achieve more than 100 times faster processing time comparing to the FR method.

[1] http://www.fftw.org/



(a) Input images

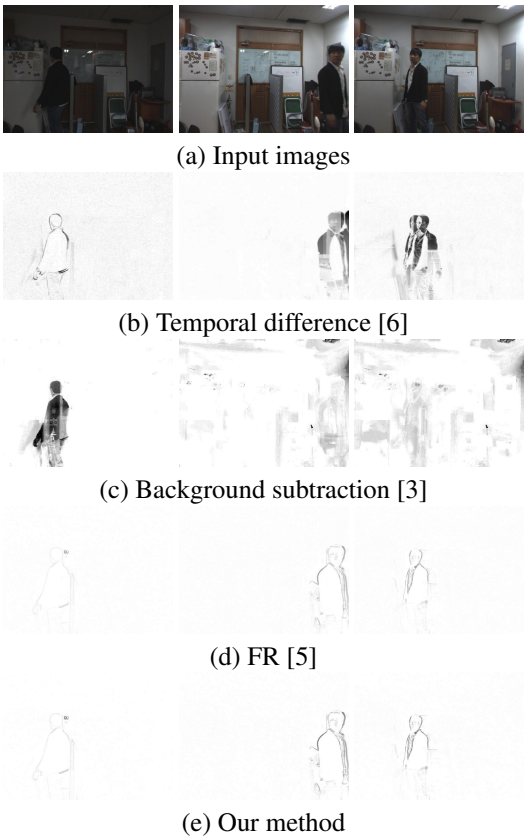
(b) Temporal difference [6]

(c) Background subtraction [3]

(d) Sobel edge

(e) Our method

Fig. 2. Experiment results for strong and sudden illumination change with an auto-exposure setting.



(a) Input images

(b) Temporal difference [6]

(c) Background subtraction [3]

(d) FR [5]

(e) Our method

Fig. 3. Experiment results for strong and sudden illumination change with a fixed-exposure setting.

(a) Input image (b) FR [5] (c) Ours

Fig. 4. A detailed comparison between FR [5] and our method.

Algorithm	Elapsed time (ms)
FR [5]	131.58
Ours 1 (floating point operations)	5.44
Ours 2 (fixed point operations)	1.14

Table 1. Processing time for VGA (640×480) images. *Ours 1* is the result that use floating point operations in Eq. (3). *Ours 2* is the result of the fixed point implementation of *Ours 1* in Eq. (7).

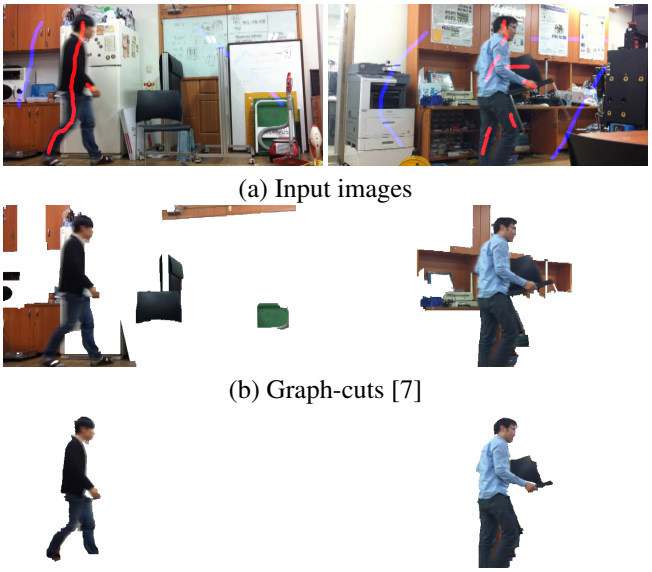
Applications. We introduce an application to moving object segmentation in Fig. 5. Graph-cuts [7] is widely used for the object segmentation. Graph-cuts formulates the segmentation problem into a Markov Random Field (MRF) graph optimization problem. In the MRF graph, a Gaussian mixture model of pixel colors is used as data term and edge is used as smoothness term. In our application, we use the result of our motion boundary detection for smoothness term. In Fig. 5 (a), red and blue scribbles represent initial seeds of foreground and background. The segmentation results are obviously improved when our motion detection is combined to the MRF model as smoothness term.

4. CONCLUSIONS

We have presented the motion detection algorithm for real-time video surveillance that runs at 1ms per one VGA resolution image on a consumer hardware without any parallel optimization. We presented frequency analysis for the motion boundary detection and derived the change detection filter from DCT that can be pre-calculated. Our key contribution is making the motion detection algorithm ultra high speed. Our algorithm is evaluated with challenging datasets and have shown robust results to sudden illumination changes. For future work, we will investigate an adaptive thresholding method to get clearer and robust results and will apply our results as a motion prior to improve performance of many image processing and computer vision applications.

5. ACKNOWLEDGEMENT

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MEST) (No.2012-0000986)



(a) Input images

(b) Graph-cuts [7]

(c) Graph-cuts with our motion detection

Fig. 5. Application of our motion detection to the object segmentation

6. REFERENCES

[1] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 246–252, 1999.

[2] P. KaewTraKulPong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," *European Workshop on Advanced Video Based Surveillance Systems*, vol. 25, 2001.

[3] J. Pilet, C. Strecha, and P. Fua, "Making background subtraction robust to sudden illumination changes," *European Conference on Computer Vision*, pp. 567–580, 2008.

[4] A. Mittal and N. Paragios, "Motion-based background subtraction using adaptive kernel density estimation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2004, vol. 2, pp. II–302 – II–309 Vol.2.

[5] Du-Ming Tsai and Wei-Yao Chiu, "Motion detection using fourier image reconstruction," *Pattern Recognition Letters*, vol. 29, no. 16, pp. 2145 – 2155, 2008.

[6] S. C. S. Cheung and C. Kamath, "Robust background subtraction with foreground validation for urban traffic video," *EURASIP Journal on Applied Signal Processing*, vol. 2005, no. 14, pp. 2330–2340, 2005.

[7] Y. Y. Boykov and G. Funka Lea, "Graph cuts and efficient N-D image segmentation," *International Journal of Computer Vision*, vol. 70, no. 2, pp. 109–131, Nov. 2006.