ICIC Express Letters
Volume 16, Number 11, November 2022

ICIC International ©2022 ISSN 1881-803X
pp. 1235–1240

A MOVING OBJECT DETECTION METHOD BASED ON DISCRETE FOURIER TRANSFORM

Shanqian Sun and Kohei Inoue
Department of Communication Design Science
Kyushu University
4-9-1 Shiobaru, Minamiku, Fukuoka 815-8540, Japan
sun.shanqian@kyudai.jp; k-inoue@design.kyushu-u.ac.jp

Received April 2022; accepted June 2022

Abstract. *Moving object detection in video sequences is a challenging task and contributes to missions like pedestrian detection. In this paper, we propose a discrete Fourier transform (DFT) based moving object detection method and utilize it to extract pedestrians from a publicly available dataset. Considering the real-time requirement in practical application, we compare DFT with fast Fourier transform (FFT), experimentally. Data analysis shows that the FFT based method is a low time consumption moving object detection method with good performance.*
Keywords: Moving object detection, Frame sequence, Time complexity, DFT, FFT

1. **Introduction.** In the domain of computer vision, moving object detection in video sequence is a challenging task and plays a fundamental role in pedestrian detection, pedestrian track, person re-identification, etc. The target of moving object detection is to divide each video frame into foreground (moving object) and background (static object).

Generally, it is assumed that the video is captured by stationary camera and the light is stable without flashing. Moreover, the video frame rate and resolution should be high enough [1]. The traditional moving object detection methods can be divided into frame difference method, optical flow method and background modeling method [2].

Frame difference method calculates pixel-value difference between every two or three consecutive frames and the pixel would be recognized as foreground if beyond a threshold. The idea is simple and it is easy to implement. However, this method would have poor performance if the object moves slowly since it takes previous frame as a reference and may be a hole in the detected object [3, 4].

Optical flow method calculates optical flow field, clustering and tracking moving object according to the optical flow distribution characteristic of the image. However, it requires large quantity of calculations [5].

Background modeling method estimates a model of the background according to local pattern of texture, photometric feature and a mixture of Gaussian distribution and classifies the pixels into background and foreground according to the similarity to the modeled background. This method requires few frames without moving object to initialize the background model, which cannot be guaranteed in real-life scenarios [6].

DFT based moving object detection method in this paper is enlightened by frame difference method. However, our method deals with the hole problem. Moreover, we use its advanced method FFT to improve the computation speed. The rest of this paper is organized as follows. Section 2 and Section 3 introduce the theories of DFT and FFT and apply them to the task, respectively. Section 4 gives a numerical example and compares the efficiency according to the time complexity. Section 5 concludes this paper.

DOI: 10.24507/icicel.16.11.1235

2. DFT Based Algorithm.

In mathematics, the DFT is a method which can transform time domain signal into frequency domain signal [7], which is defined as

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i\frac{2\pi}{N}kn} \tag{1}$$

where $\{x_n\} := x_0, x_1, \ldots, x_{N-1}$ and $\{X_k\} := X_0, X_1, \ldots, X_{N-1}$ indicate time and frequency domain signals separately. n and k are serial numbers, and N is the length of input frame sequence. And the inverse DFT (IDFT) is defined as

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{i\frac{2\pi}{N}kn} \tag{2}$$

We can get the frame sequence only with foreground by processing in frequency domain. Here are the specific steps. Firstly, we separate the RGB frame sequence into 3 gray-scale frame sequences and transform them into frequency domain. Then, we set X_0 to $0 + 0i$, in which i is the imaginary unit, to remove the background. Next, we inversely transform the $\{X_k\}$ back to $\{x_n\}$ and divide the pixels into foreground and background according to the experimental threshold. Finally, we reunite the R, G and B channels images with OR operation, as it shows in Algorithm 1.

Algorithm 1 DFT based algorithm

```
1: Prepare gray-scale frame sequences
2: Transform every frame sequence $\{x_n\}$ to $\{X_k\}$ by DFT
3: Set $X_0$ to $0 + 0i$
4: Transform $\{X_k\}$ back to $\{x_n\}$
5: Segment foreground and background by the experimental threshold
6: Process with OR operation
```

We use the UCSD (the University of California San Diego) pedestrian dataset [8] to test our algorithm. This dataset contains videos of pedestrians on UCSD walkways, taken from a stationary camera, and is 740×480 at 10 fps. Figure 1 shows the results of steps 1 to 4. 1(a) shows a frame of the original frame sequence, 1(b) shows its G channel image, 1(c) to 1(i) show the result with different input's length N. It should be noted that there is an $N = 256$ image between 1(h) and 1(i) but ignored due to small difference.

We can find that when we calculate with 4 images, the outline of pedestrian is fuzzy but the background is clean, as 1(c) shows. As the images increase, the outline becomes clear but the background becomes a little dirty. And as the images continually increase, the outline becomes sharp and the background becomes clean, as 1(i) shows. Moreover, no visible difference remains when N exceeds 128.

Then, we use experiential threshold to divide pixels in the $N = 512$ images into foreground and background, and reunite by OR operation at the end. Figure 2 shows the result. 2(a), 2(b) and 2(c) are the R, G and B channels images without background. 2(d), 2(e) and 2(f) are the after thresholds' images from the images above. 2(g) is the after OR operation images from 2(d), 2(e) and 2(f), and 2(h) is the original image.

From the result, we can find that 1) the foreground is not marked completely, like the backpack of the right man; 2) some pixels are mistaken into foreground, for example, on the top right of the image, some leaves are distinguished into foreground due to the shaking, which requires further effort to deal with.

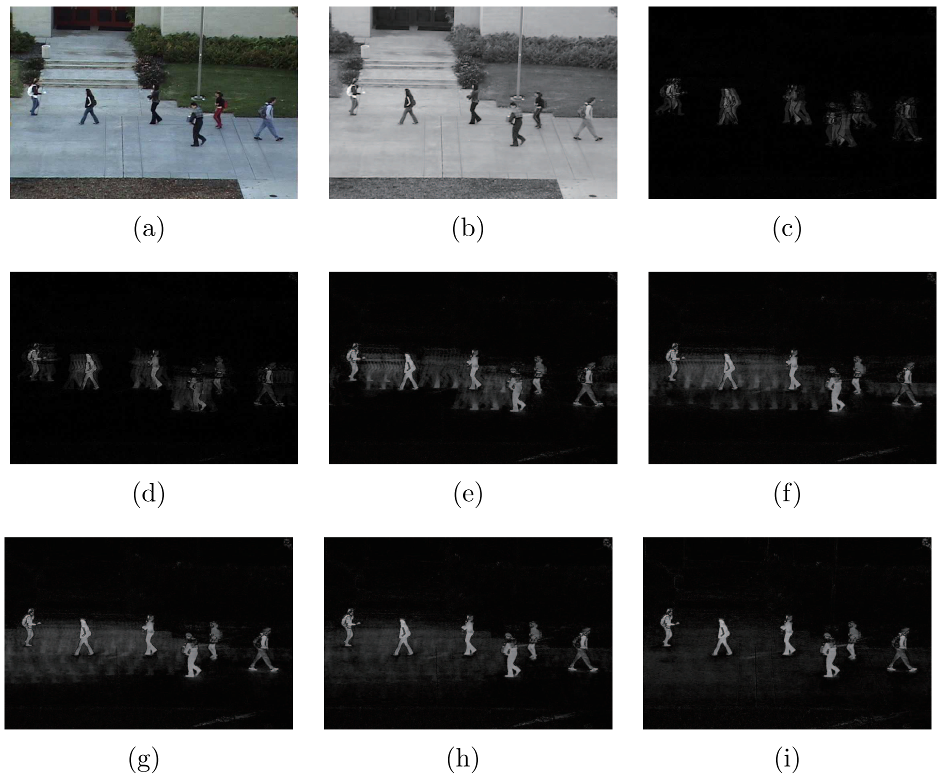

FIGURE 1. Foreground extraction images with different length N: (a) Original image; (b) G channel image; (c) $N = 4$; (d) $N = 8$; (e) $N = 16$; (f) $N = 32$; (g) $N = 64$; (h) $N = 128$; (i) $N = 512$

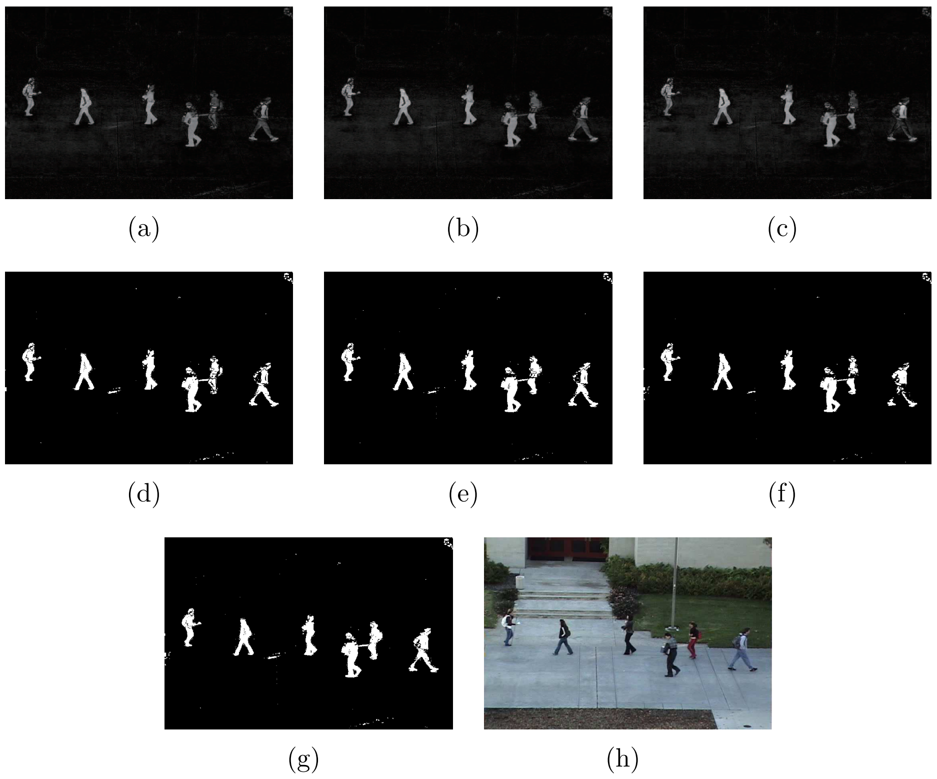

FIGURE 2. Process with threshold and OR operation: (a) R channel image; (b) G channel image; (c) B channel image; (d) R channel image binarization; (e) G channel image binarization; (f) B channel image binarization; (g) after OR operation image; (h) original image

3. **FFT Based Algorithm.** Although the DFT works well, it can still be improved. We replace W_N with $e^{-i\frac{2\pi}{N}}$ to simplify the expression, and Equation (3) shows the regulations of W_N, in which m, $\frac{N}{m}$ and r are integers.

$$\begin{aligned} W_N^{nk} &= W_{\frac{N}{m}}^{\frac{nk}{m}} \\ W_N^{nk} &= W_N^{(n+rN)k} = W_N^{(k+rN)n} \end{aligned} \tag{3}$$

Equation (1) can be rewritten as the first line in Equation (4), then, using $2r$ to replace n and applying the first law in Equation (3), we can get the third line equation, in which e_r is the even sequence and o_r is the odd sequence, and it is the sum of even sequence DFT and odd sequence DFT as the final line shows.

$$\begin{aligned} X_k &= \sum_{n=0\ (n \text{ is even})}^{N-1} x_n W_N^{nk} + \sum_{n=0\ (n \text{ is odd})}^{N-1} x_n W_N^{nk} \\ &= \sum_{r=0}^{\frac{N}{2}-1} x_{2r} W_N^{2rk} + \sum_{r=0}^{\frac{N}{2}-1} x_{2r+1} W_N^{(2r+1)k} \\ &= \sum_{r=0}^{\frac{N}{2}-1} e_r W_{\frac{N}{2}}^{rk} + W_N^k \sum_{r=0}^{\frac{N}{2}-1} o_r W_{\frac{N}{2}}^{rk} \\ &= DFT(e_r) + W_N^k DFT(o_r) \end{aligned} \tag{4}$$

However, the equation above only contains half of the result, which means X_k, $0 \le k \le \frac{N}{2} - 1$. Equation (5) shows the result of $X_{\frac{N}{2}+k}$, $0 \le k \le \frac{N}{2} - 1$ by applying the second law in Equation (3).

$$\begin{aligned} X_{\frac{N}{2}+k} &= \sum_{r=0}^{\frac{N}{2}-1} e_r W_{\frac{N}{2}}^{r\left(\frac{N}{2}+k\right)} + W_N^{\frac{N}{2}+k} \sum_{r=0}^{\frac{N}{2}-1} o_r W_{\frac{N}{2}}^{r\left(\frac{N}{2}+k\right)} \\ &= DFT(e_r) - W_N^k DFT(o_r) \end{aligned} \tag{5}$$

Equation (6) sums the result [9]. We can reduce repetition to cut down the runtime through this method. We calculate every N sequences 10000 times. The algorithm is run on AMD Ryzen 7 4800H CPU 2.90 GHz with 16 cores, and the result shows in Table 1, where N stands for the length of frame sequence. We can find that the time growth rate of FFT is much lower than the DFT's and the FFT is faster than the DFT when N is huge.

$$\begin{aligned} X_k &= DFT(e_r) + W_N^k DFT(o_r) \\ X_{\frac{N}{2}+k} &= DFT(e_r) - W_N^k DFT(o_r) \end{aligned} \tag{6}$$

TABLE 1. Operation time (s) with different length sequences and algorithms

	$N=4$	$N=8$	$N=16$	$N=32$	$N=64$	$N=128$	$N=256$	$N=512$
DFT	0.09	0.11	0.21	0.60	2.09	7.95	36.69	146.31
FFT	0.14	0.35	0.76	1.58	3.23	6.56	13.22	26.77

4. **Numerical Example.** Here goes an example of specific computational process of DFT and FFT with 4 points sequence.

We take a pixel-value sequence as $\{x_0 = 89, x_1 = 40, x_2 = 91, x_3 = 90\}$ from one pixel of 4 continuous R channel images to DFT. So, in Equation (1), $N = 4$, and $W_4 = e^{-i\frac{2\pi}{4}} = \cos\left(\frac{\pi}{2}\right) - i\sin\left(\frac{\pi}{2}\right) = -i$, then X_k $(k = 0, 1, 2, 3)$ can be expressed as

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \end{bmatrix} = \begin{bmatrix} W_4^0 & W_4^0 & W_4^0 & W_4^0 \\ W_4^0 & W_4^1 & W_4^2 & W_4^3 \\ W_4^0 & W_4^2 & W_4^4 & W_4^6 \\ W_4^0 & W_4^3 & W_4^6 & W_4^9 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix} \begin{bmatrix} 89 \\ 40 \\ 91 \\ 90 \end{bmatrix} = \begin{bmatrix} 310 \\ -2+50i \\ 50 \\ -2-50i \end{bmatrix} \tag{7}$$

From the equation above, we can easily find that it has $N \times N$ times complex multiplications and $N \times (N-1)$ times complex additions. One complex multiplication has 4 times real-number multiplications and 2 times real-number additions, and one complex addition has 2 times real-number additions. So, it takes $4N^2$ times real-number multiplications and $4N^2 - 2N$ times real-number additions. So, the time complexity is $O\left(N^2\right)$.

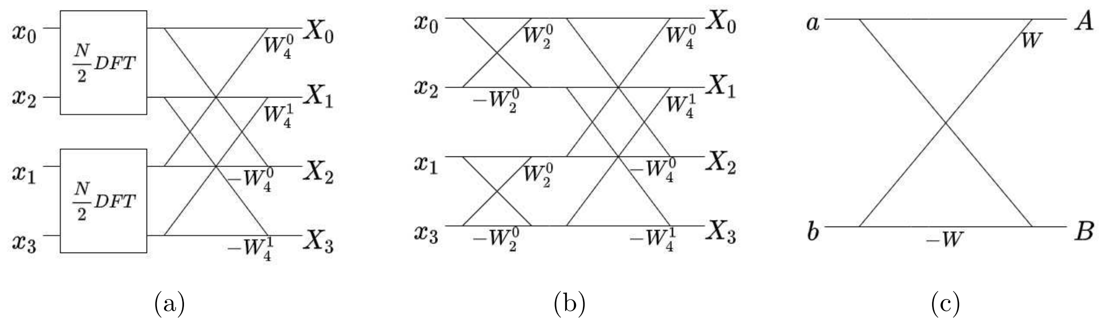

FIGURE 3. Process of 4 points FFT: (a) First time decomposition; (b) second time decomposition; (c) butterfly computation unit

Figure 3 shows the process of 4 points FFT. As the length of $N = 4 > 2$, $\{x_0 = 89, x_1 = 40, x_2 = 91, x_3 = 90\}$ is divided into even sequence $e_r = \{x_0 = 89, x_2 = 91\}$ and odd sequence $o_r = \{x_1 = 40, x_3 = 90\}$. The output can be expressed as Equation (8), having the same form as Equation (6). Then the length of e_r and o_r is 2 and cannot be separated further, so after their DFT, as Equation (9) shows, the process is finished.

$$\begin{aligned} \begin{bmatrix} X_0 \\ X_1 \end{bmatrix} &= DFT(e_r) + \begin{bmatrix} W_4^0 & 0 \\ 0 & W_4^1 \end{bmatrix} DFT(o_r) \\ \begin{bmatrix} X_2 \\ X_3 \end{bmatrix} &= DFT(e_r) - \begin{bmatrix} W_4^0 & 0 \\ 0 & W_4^1 \end{bmatrix} DFT(o_r) \end{aligned} \tag{8}$$

$$\begin{aligned} \begin{bmatrix} X_0 \\ X_1 \end{bmatrix} &= \begin{bmatrix} x_0 + W_2^0 x_2 \\ x_0 - W_2^0 x_2 \end{bmatrix} + \begin{bmatrix} W_4^0 & 0 \\ 0 & W_4^1 \end{bmatrix} \begin{bmatrix} x_1 + W_2^0 x_3 \\ x_1 - W_2^0 x_3 \end{bmatrix} = \begin{bmatrix} 310 \\ -2+50i \end{bmatrix} \\ \begin{bmatrix} X_2 \\ X_3 \end{bmatrix} &= \begin{bmatrix} x_0 + W_2^0 x_2 \\ x_0 - W_2^0 x_2 \end{bmatrix} - \begin{bmatrix} W_4^0 & 0 \\ 0 & W_4^1 \end{bmatrix} \begin{bmatrix} x_1 + W_2^0 x_3 \\ x_1 - W_2^0 x_3 \end{bmatrix} = \begin{bmatrix} 50 \\ -2-50i \end{bmatrix} \end{aligned} \tag{9}$$

From Figure 3, we can find that 3(a) and 3(b) consist of butterfly computation units in which $A = a + Wb$ and $B = a - Wb$ as 3(c) shows. An N-point FFT can be decomposed into $\log_2 N$ stages, then each stage has $\frac{N}{2}$ butterfly units, and each unit has one complex multiplication and two complex additions. All of them are $\frac{N}{2}\log_2 N$ complex multiplications and $N\log_2 N$ complex additions, so the time complexity is $O(N\log_2 N)$. When N is a huge number, the FFT costs less time than the DFT.

5. **Conclusions.**

This paper raises a DFT based moving object detection method. We utilize DFT rather than two or three frames to extract foreground and deal with the hole problem. Moreover, we replace the DFT with FFT to speed up the computational process and use an example of 4 points FFT to explain the reason by time complexity.

And many open questions have been raised that we want to answer in our further research. The first is about the threshold. We want to find a method to determine the threshold automatically, instead of the experimental value. And a post-processing method is needed to remove the noise in the foreground and complete the missing part.

Acknowledgment. This work was supported by JST SPRING, Grant Number JPM JSP2136 and JSPS KAKENHI Grant Number JP21K11964.

REFERENCES

[1] H. Zhu, X. Yan, H. Tang, Y. Chang, B. Li and X. Yuan, Moving object detection with deep CNNs, *IEEE Access*, vol.8, pp.29729-29741, 2020.
[2] S. D. Roy and M. K. Bhowmik, A comprehensive survey on computer vision based approaches for moving object detection, *2020 IEEE Region 10 Symposium (TENSYMP)*, pp.1531-1534, 2020.
[3] J. Hu, R. Liu, Z. Chen, D. Wang, Y. Zhang and B. Xie, Octave convolution-based vehicle detection using frame-difference as network input, *The Visual Computer*, pp.1-13, 2022.
[4] S. Cai, Q. Zhang, Q. Wang, Y. Lei and J. Yang, Multi-frame dimensionality-reduction difference method for extracting key frames of video, *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pp.1466-1470, 2020.
[5] L. Wang, Y. Guo, L. Liu, Z. Lin, X. Deng and W. An, Deep video super-resolution using HR optical flow estimation, *IEEE Trans. Image Processing*, vol.29, pp.4323-4336, 2020.
[6] A. Khalil, S. U. Rahman, F. Alam, I. Ahmad and I. Khalil, Fire detection using multi color space and background modeling, *Fire Technology*, vol.57, no.3, pp.1221-1239, 2021.
[7] X.-Q. Zhang and Z.-M. Lu, Discrete Fourier transform peak detection based robust audio watermarking against time scale modulation and pitch shifting, *International Journal of Innovative Computing, Information and Control*, vol.16, no.6, pp.1973-1985, 2020.
[8] A. B. Chan and N. Vasconcelos, Modeling, clustering, and segmenting video with mixtures of dynamic textures, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.30, no.5, pp.909-926, 2008.
[9] Y. Wang, Z. Jiang, Y. Li, J.-N. Hwang, G. Xing and H. Liu, RODNet: A real-time radar object detection network cross-supervised by camera-radar fused object 3D localization, *IEEE Journal of Selected Topics in Signal Processing*, vol.15, no.4, pp.954-967, 2021.