A Novel Moving Objects Detection Model Based on Images Registration within Sliding Time Windows

Shaomang Huang, Jun Ma, and Qian Zhao

Dept. Computer Science and Technology, Shandong University
No.1500, Shunhua Road, Jinan, Shandong Province, China
yuletianxia@hotmail.com, majun@sdu.edu.cn, zhaoqian7616@126.com

Abstract. Moving objects detection is a fundamental step for automated video analysis, robot visual system and many other vision applications. There are limitations in the existing algorithms, such as assuming a static camera, a smooth motion and rigid motion of target objects, etc. In this paper, we present a novel model named IRTSW-model; a moving objects detection model which can work effectively no matter the camera is moving or static. In the approach, images registration is used to eliminate the relative movements between the background and the camera; unsupervised codebook model is constructed to model the background; and then the moving objects are detected accurately. Experiments on the segtrack database demonstrate the effectiveness of our model.

Keywords: Moving objects detection, images registration, unsupervised codebook model, time sliding window.

1 Introduction

The moving objects detection approach intends to separate all the moving objects from background. It is important for many vision applications such as surveillance, traffic monitoring, augmented reality, vehicle navigation, robot visual system, etc.[1][2]. Its algorithms are divided into two kinds: ones working in a static scene and ones working in a dynamic scene, depending on the camera is static or moving (the camera position changes or the camera rotates or both). [3].

Moving objects detection methods in static scenes include background subtraction method [4], inter-frame difference method, as well as some background modeling methods such as codebook model[5][6] and Gaussian mixture model, and so on. Although they have their own advantages in different situations, they have some common drawbacks, such as training procedures are required and the limitation of they can only be used in a static scene. Moving objects detection methods in dynamic scenes include algorithms based on object detectors, optical flow algorithm [7], block-matching algorithm[8] and several motion segmentation algorithms[9][10]. In the algorithms based on object detectors, classifiers should be trained and built either by off-line learning on separate databases [10] or by on-line learning initialized with a manually labeled frame at the start of a video [11]. The optical flow method and motion segmentation algorithms separate moving targets from background using the motion information. Although they avoid training phases, however, they assume rigid motion [9] or smooth motion [10] in respective regions, which is not generally true in practice.

In this paper, we present a novel moving objects detection model named IRTSW-model (a moving objects detection model based on Images Registration within Sliding Time Window). In our approach, we find a way to build a background model for the scene (both dynamic and static). The moving objects are detected based on the background model. It is robust, effective and has a wide application.

The main contributions can be summarized as: First, we present a sliding time window method for processing prolonged videos effectively. Second, we present the background modeling method for dynamic scenes based on images registration. Third, we present an unsupervised codebook model which classifies pixels depending on the statistical characteristics of the observation value sequences.

2 IRTSW Model Description

A video is defined as: $V_{\Delta t} = [I_0, I_2, \dots, I_{N-1}]$. Where, Δt is the time interval between adjacent frames. N is the number of frames in the video. $I_i (0 \le i < \mathrm{N})$ is the i-th frame in $V_{\Delta t}$. Usually, the camera gets more than 24 frames per second. So, Δt is very small, and the adjacent frames, I_i and I_{i+1}, have most of their contents overlapped. During Δt , the motion of camera could be approximately regarded as self-rotation, or small-scale panning, in which conditions homography works well. We extend this from two to several frames adjacent, and get a short time made up with several Δt. During the short time, the motion of camera is still simple and homography works well within error tolerance. We call the short time as a time window, and the part of video inside it is a video clip.

The detection result of each frame I_i is denoted as $Mark_{<obj,i>}$, which is a 2D binary matrix having the same length and width with I_i. $Mark_{<obj,i>}(x, \mathrm{y})$ is the classification result of $I_i(x, \mathrm{y})$. If $I_i(x, \mathrm{y})$ is a pixel on background, the value is 0; otherwise, the value is 1.

Processes of IRSTW-model are described as follows:

1) The video $V_{\Delta t}$ is divided into lots of clips, $\{S_0, \dots, S_{n-1}\}$. Each clip S_t corresponds to a time window and contains certain number of frames: $S_t = [I_0, \dots, I_m] (0 \le t < \mathrm{n}; \mathrm{m}, \mathrm{n} > 0)$. The adjacent clips, S_i and S_{i+1}, are partially overlapped, which means there exist some frames shared by S_i and S_{i+1}. The algorithm processes the clips one after another. When one clip is finished and the next is started, it is called the time window is sliding. When the time window slides, homography matrix between the previous coordinate space and the new one is calculated, and then the background model and arguments will be projected into the new coordinate space according to the homography.
2) Within each time window, we select the coordinate space of the first frame in the clip, $\mathcal{R}_{I_0}$, as the projection coordinate space. Then, homography matrix, $H_{<t_0,i>}$, between frame I_i and $\mathcal{R}_{I_0}$ is calculated. Using $H_{<t_0,i>}$, I_i can be mapped into $\mathcal{R}_{I_0}$. The corresponding projection image is denoted as I_i^*.

3) In $\mathcal{R}_{I_0}$, the average of all the projection images is calculated as the fusion image, denoted as J. At the same time, a background model is built using unsupervised codebook model algorithm which will be described later, according to the observation value sequences at each coordinate location through all projection images.
4) In $\mathcal{R}_{I_0}$, the pixels are classified in each projection image I_i^*. Then, using homography matrix $H_{<t_0,i>}$, we can get the classification result in each frame I_i. Thus, the moving objects in each frame are detected.

The procedure of images registration within the time window, the algorithm of unsupervised codebook model and the acquisition of moving objects detection results will be detailed described below.

2.1 Images Registration within Time Window

Given a video $V_{\Delta t}$ and the time window length $T(1 \le T \le \mathrm{N})$, the video clip which starts with I_{t_0} as the first frame can be denoted as $S_{<t_0,T>} = [I_{t_0}, I_{t_0+1}, \dots, I_{t_0+T-1}]$. $I_{t_0+i}(0 \le i \le \mathrm{T}-1)$ is the $(i+1)$-th frame inside $S_{<t_0,T>}$. As we said before, a point in the real world is projected into different locations of different frames in $S_{<t_0,T>}$ due to the camera motion. It is called that different frames, e.g. I_{t_0+i} and I_{t_0+j}, are in different coordinate spaces. The goal of images registration is to project images into a same coordinate space, so that they are free to compute with each other. In this paper, the projection coordinate space, which is denoted as $\mathcal{R}_{I_{t0}}$, is always selected to be the coordinate space of the first frame in the clip. In $\mathcal{R}_{I_{t0}}$, a fusion image J is calculated, because we need it to locate each pixel position. J is the average of projection images of all the processed frames within the clip. Obviously, J is an image in $\mathcal{R}_{I_{t0}}$, so the homography between I_{t_0+i} and J is also the one between I_{t_0+i} and $\mathcal{R}_{I_{t0}}$.

Images registration is an iterative procedure. One frame is processed per iteration. We uses a SIFT+RANSAC method similar to Matthew Brown [18] to calculate the homography matrix between J and I_{t_0+i}. For more details about getting a homography using a SIFT+RANSAC method, you may refer to the paper [18].

As shown in Alg.1, Ж, J and $\underset{(x,y)\in\mathrm{R}_J}{\mathrm{V}_{<x,y>}}$ are outputted. Ж is an array saving all the homography matrix which can project each frame in the clip into $\mathcal{R}_{I_{t0}}$. Once a homography $H_{<t_0,i>}$ is calculated, it is saved in Ж, as shown at line 6 of Alg.1. J is the fusion image which is the average of all the projection images. In each iteration, J is updated by : $J = \frac{I_{t_0}+I^*_{t_0+1}+\cdots+I^*_{t_0+i}}{i+1} = \frac{J*i+I^*_{t_0+i}}{i+1}$. We use $(x,y) \in \mathrm{R}_J$ to represent every pixel location in J. And $\underset{(x,y)\in\mathrm{R}_J}{\mathrm{V}_{<x,y>}}$ means the sequences of observation values at each pixel location of J. $\mathrm{V}_{<x,y>}$ is the sequence of observations at the coordinate (x,y), and is the array of pixel values at (x,y) location in each projection image.

Fig.1 shows the homography matrixes and the fusion image of a series of images. In the fusion image, the background is the merger of backgrounds in all the input images, while the foregrounds (i.e. the moving objects) disappear gradually.

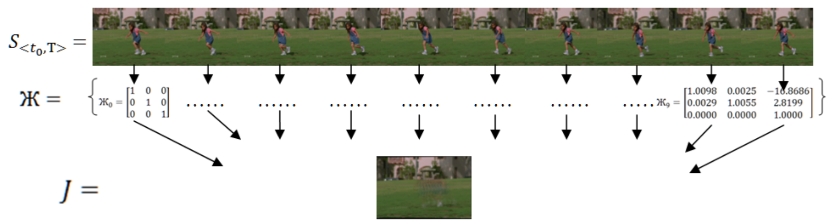

$S_{<t_0,T>} =$
$Ж =$
$Ж_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
$Ж_9 = \begin{bmatrix} 1.0098 & 0.0025 & -16.8686 \\ 0.0029 & 1.0055 & 2.8199 \\ 0.0000 & 0.0000 & 1.0000 \end{bmatrix}$
$J =$

Fig. 1. Homography matrixes and fusion image of image series

Alg.1. Images Registration within the Time Window

```
1.  Input: $S_{<t_0,T>} = [I_{t_0}, I_{t_0+1}, \ldots, I_{t_0+T-1}]$
2.  Initialize: $J \leftarrow I_{t_0}$, $i \leftarrow 1$, $Ж[0] \leftarrow \begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix}$, $\underset{(x,y)\in R_J}{V_{<x,y>}}[0] \leftarrow I_{t_0}(x,y)$, $T$.
3.  for $i = 0$ to $T-1$ do
4.      Extract SIFT features for $J$ and $I_{t_0+\mathrm{i}}$
5.      Compute homographic matrix $H_{<t_0,i>}$ between $J$ and $I_{t_0+\mathrm{i}}$ using
    RANSAC
6.      $Ж[\mathrm{i}] \leftarrow H_{<t_0,i>}$
7.      $I^*_{t_0+i} \leftarrow H_{<t_0,i>} * I_{t_0+i}$
8.      for each $(x,y)$ in $I^*_{t_0+i}$
9.          $\underset{(x,y)\in R_J}{V_{<x,y>}}[\mathrm{i}] \leftarrow I^*_{t_0+i}(\mathrm{x},\mathrm{y})$
10.     next
11.     $J \leftarrow \frac{J*i+I^*_{t_0+i}}{i+1}$
12.     $i \leftarrow i+1$
13. next
14. Output: $Ж$, $\underset{(x,y)\in R_J}{V_{<x,y>}}$, $J$
```

2.2 Unsupervised Codebook Model

The unsupervised codebook model takes the outputs of Algorithm 1 as inputs. It produces background codewords at a pixel location from the statistical characteristics of the observation value sequence there. In $\mathcal{R}_{I_{t_0}}$, the observation value sequence at location (x,y) is $V_{<x,y>} = [v_{<x,y,0>}, \ldots, v_{<x,y,T-1>}]$. $v_{<\mathrm{x,y},i>}$ means the pixel value at (x,y) on $I^*_{t_0+i}$: $v_{<\mathrm{x,y},i>} = I^*_{t_0+i}(\mathrm{x},\mathrm{y}) = \langle r_{<x,\mathrm{y},i>}, g_{<\mathrm{x,y},i>}, b_{<\mathrm{x,y},i>} \rangle$. Here $r_{<x,\mathrm{y},i>}$, $g_{<\mathrm{x,y},i>}$ and $b_{<\mathrm{x,y},i>}$ represent the RGB values of $v_{<\mathrm{x,y},i>}$ respectively. The intensity (brightness) of $v_{<\mathrm{x,y},i>}$ can be calculated by:

$int(v_{<x,y,i>}) = \sqrt{r^2_{<x,y,i>} + g^2_{<x,y,i>} + b^2_{<x,y,i>}}$. For each observation value sequence $V_{<x,y>}$, we create a model containing several codewords: $C_{<x,y>} = \{c_{<x,y,0>}, \dots, c_{<x,y,L-1>}\}$. Where, each $c_{<x,y,i>}(0 \le i < L_{<x,y>})$ represents a codeword. Different from classic codebook model, $c_{<x,y,i>}$ not only could be a background codeword, but also could be a foreground codeword. $c_{<x,y,i>}$ is a foreground codeword when it has not been identified as a background codeword. A codeword is a multi-tuple: $c_{<x,y,i>} =< \check{I}, \hat{I}, \check{d}, \hat{d}, num, \Delta t, l, \tau, s, p, q, f >$. The meaning of each parameter is described below:

$\check{I}, \hat{I}$: the min and max brightness, respectively, that the codeword accepted.

$$\check{I} = \min_{i\in[1,N]}[int(v_{<x,y,i>})], \quad \hat{I} = \max_{i\in[1,N]}[int(v_{<x,y,i>})].$$

$\check{d}, \hat{d}$: the min and max brightness of the decision boundary of the codeword.

If the intensity of an observation value is between $\check{d}$ and $\hat{d}$ of a codeword, it is called they are matched [6]. $\check{d} = \alpha\check{I}$, $\hat{d} = \min(\beta\hat{I}, \check{I}/\alpha)$.

num : the cycle number of a cyclical background codeword.

Δt: the appearance interval time of the codeword.

l: the duration time of the codeword.

τ: the time tolerance parameter.

s: the time when the codeword is created.

p, q: the last start and end time, respectively, that the codeword has occurred.

f: the codeword type identifier. If the value is 0 or 2, indicating a background codeword; if the value is 1, indicating a foreground codeword.

At a certain location in $\mathcal{R}_{I_{t0}}$, the foreground parts appear suddenly, and the duration is short. Thus, the corresponding codewords appear suddenly, and they rarely appear before and after. Their durations are short too. The background parts may be static or move regularly and cyclically.

Two types of background codeword patterns are defined. (a). The static background codeword. The observation value sequence matched with it has the same or similar pixel values for a long time, continuously. Parameters are like : $\Delta t = 0, l = 1, \tau = 2$. (b). The cyclical moving background codeword. It could be used to express background like shaking branches and fluttering flags etc. Parameters are like: $\Delta t = \Delta t_1 + \Delta t_2 > 0, l = \Delta t_1, 0 < \tau \le \Delta t * \gamma$. Where, $\gamma \in (0,1)$.

To decide whether a codeword should be a background code word or not, it should be monitored for a certain period of time, λ. If a codeword $c_{<x,y,i>}$ meets one of the backgrounds codeword patterns for more than λ time, it is converted to be a background codeword; otherwise, it is cleared up (line 18 to 23 in Alg.2). For improving the accuracy, we create a special background codeword using the fusion image J. It is the similar thought with the mean background subtraction method. The special background codeword is identified as: $f = 2$. It is updated by recreating per iteration (line 5 in Alg.2). Whenever an observation value $v_{<x,y,t>}$ is matched with a codeword $c_{<x,y,i>}$, the parameters of $c_{<x,y,i>}$ are judged and then updated according to several cases defined as follows:

M_1: if $\Delta t = 0, f = 1, t - q > \tau$
Do updates: $\Delta t = t - s, l = q - p, p = t, num = num + 1, q = t$
M_2: if $\Delta t > 0, f = 1, t - q \le \tau$
Do updates: $l = \max(l, t - p), q = t$
M_3: if $\Delta t > 0, f = 1, t - q > \tau$
Do updates: $\Delta t = \frac{\Delta t * num + (t - q)}{(num + 1)}$, $\tau = \max(|\Delta t + q - t|, |q - p - l|, \tau)$, $num = num + 1, q = t$
M_4: if otherwise
Do updates : $q = t$
In $[\mathbf{M_1}, \dots, \mathbf{M_4}]$, the other parameters except $\Delta t, l, p$, q and τ should be updated according to their definition.

Alg.2. Unsupervised codebook model

```
1.  Input: $\underset{(x,y)\in R_J}{V_{<x,y>}}, J, \lambda$
2.  Initialize: $\underset{(x,y)\in R_J, i\in[0,T-1]}{Dec_{<x,y,i>}} \leftarrow 0$, $\underset{(x,y)\in R_J}{L_{<x,y>}} \leftarrow 0$, $t \leftarrow 0$, $\underset{(x,y)\in R_J}{C_{<x,y>}} \leftarrow \{\}$
3.  for each $(x, y) \in R_J$ do
4.    for $t \leftarrow 0$ to $T - 1$ do
5.        $c_{<x,y,0>} \leftarrow < int(J(x,y)), int(J(x,y)), \check{d}, \hat{d}, 0,0,1,2, t, t, t, 2 >$
6.        if $v_{<x,y,t>}$ does not match any codeword within $C_{<x,y>}$ then
7.          $L_{<x,y>} \leftarrow L_{<x,y>} + 1$
8.          $c_{<x,y,L>} \leftarrow < int(v_{<x,y,t>}), int(v_{<x,y,t>}), \check{d}, \hat{d}, 0,0,1,2, t, t, t, 1 >$
9.        elseif $v_{<x,y,t>}$ matches with codeword $c_{<x,y,i>} \in C_{<x,y>}$ then
10.          Update parameters of $c_{<x,y,i>}$ according to the cases $[M_1, \dots, M_4]$
11.          if $f = 1$ then
12.            $Dec_{<x,y,t>} = 1$
13.          elseif $f = 0$ or $f = 2$ then
14.            $Dec_{<x,y,t>} = 0$
15.          endif
16.        endif
17.        for each $c_{<x,y,i>} \in C_{<x,y>}$
18.          if parameters of $c_{<x,y,i>}$ have : $t - q > \lambda$ then
19.            clear $c_{<x,y,i>}$
20.            $L_{<x,y>} \leftarrow L_{<x,y>} - 1$
21.          elseif $(\Delta t = 0$ or $\tau \le \Delta t * \gamma)$ and $q - s > \lambda$ and $f = 1$ then
22.            set parameter of $c_{<x,y,i>}$ : $f = 0$
23.          endif
24.        next
25.    next
26. next
27. Output: $\underset{(x,y)\in R_J, i\in[0,T-1]}{Dec_{<x,y,i>}}$
```

As shown in Alg.2 from line 11 to line 15, the moving objects detection results in $\mathcal{R}_{I_{t0}}$ are obtained at the same time when the codebook models are built and updated. If $v_{<x,y,t>}$ is matched with $c_{<x,y,i>}$ whose parameter $\mathrm{f} = 0$ or $\mathrm{f} = 2$, then the pixel at (x, y) position of $I^*_{t_0+t}$ is regarded as a background pixel; if $v_{<x,y,t>}$ is matched with a foreground codeword, then the corresponding pixel is regarded as a foreground pixel. $\mathrm{Dec}_{<x,y,\mathrm{t}>}$ is the classification result of the pixel at location (x, y) on $\mathrm{I}^*_{t_0+\mathrm{t}}$. $\underset{(x,y)\in \mathrm{R}_J, \mathrm{i}\in[0,\mathrm{T}-1]}{\mathrm{Dec}_{<x,y,\mathrm{i}>}}$ means the classification results of all the pixels in all projection images in the time window.

In Alg.2, given a video containing N frames, about $N*m*n*L$ times codewords are created and updated. Where, m and n are the max length and width of the fusion images of all the clips respectively. L is the max of the numbers of codewords appeared, generally $L < 10$. So, the time complexity of Alg.2 is about $O(Nmn)$.

2.3 Moving Objects Detection Results

In Alg.2, we have already gotten $\underset{(x,y)\in R_J, i\in[0,T-1]}{Dec_{<x,y,i>}}$, the detection results at each location in each projection image. $\underset{(x,y)\in R_J}{Dec_{<x,y,t>}}$ is the detection result of $I^*_{t_0+t}$ which can be abbreviated as Dec_t. Dec_t is in $\mathcal{R}_{I_{t0}}$, using the reverse of homography $Ж[t]$ we can transfer Dec_t to the original coordinate space to get the detection result of I_{t_0+t}: $Mask_{<obj,t>} = Ж[t]^{-1} * Dec_t$, as shown in Fig.2. In $Mask_{<obj,t>}$, there is some noise. Some operations such as deleting the detected areas and the dilation and erosion operations of the image are used to eliminate the noise.

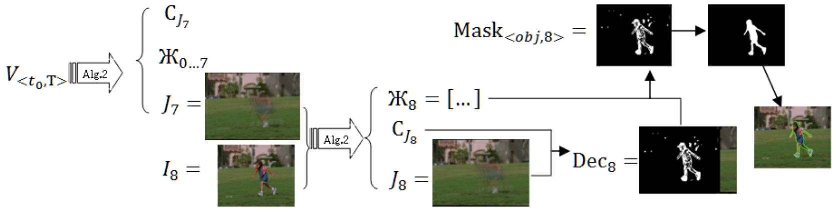


Fig. 2. Moving objects detection in a frame

3 Experiments

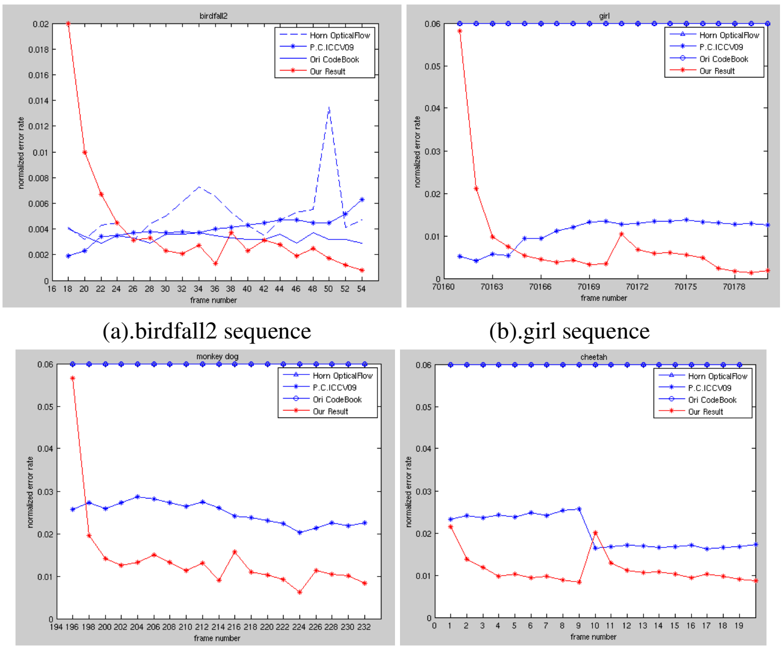
Four videos in SegTrack Database (2011) [19] are selected as our experimental data. Background parts and foreground parts (moving objects) in these videos are without any ambiguity. In these videos, foreground parts are the running girl in the girl sequence, the falling bird in the birdfall2 sequence, the running monkey as well as the moving dog in the monkeydog sequence, and the cheetah as well as the antelope in the cheetah sequence.

Our model is compared with three other approaches, which are the classic codebook model [20], the optical flow method [21] and the adaptive fragments-based tracking proposed by Prakash Chockalingam in ICCV 2009 [8]. In this paper, we represent these algorithms as Ori_codebook, Horn_OpticalFlow and P.C.ICCV09 for short, respectively.

To provide quantitative evaluation of our model, we use an evaluation method the same as the method in the paper [8], which is NER (normalized error rate). NER can be computed by counting the number of misclassification pixels and normalizing it by the image size [8].

Our model is implemented in matlab R2011b on Linux CentOS operating system. Ori_codebook model is implemented in VS 2008 C++ using the opencv [22] library functions. Modh Kharbat's source codes [23] are used to get the Horn_OpticalFlow method's detection results. The results of P.C.ICCV09 method are obtained by a video published on the internet [24].

In our experiments, the length of time window is set by $T = 10$; the step length for each window sliding is set by $\Delta T = \frac{1}{3}T$; the parameters in the unsupervised codebook are set by $\lambda = 8$，$\alpha = 0.8$，$\beta = 1.2$.



(a).birdfall2 sequence (b).girl sequence

(c).monkey-dog sequence (d).cheetah sequence

Fig. 3. NER of each approach on four sequences

Fig.3 shows the NER of each algorithm on four sequences. On all the four sequences, our model shows better NER than the other three approaches. The comparison with the Ori_codebook illustrates the effectiveness of images registration phase in our model, and the possibility of transplanting static scene detection algorithms into dynamic scenes. The camera taken the birdfall2 sequence is static, while the ones taken the other three are moving. The NER of the Horn_OpticalFlow method is below 0.01 on the birdfall2 but above 0.06 on the other sequences, which illustrates the limitations of the Horn_OpticalFlow when it is used to handle a scene taken by a quickly moving camera or containing a non-rigid shape deformation object. The data in Fig.3 also illustrates that our model gets good detection results both in a dynamic scene and in a static scene.

4 Conclusion and Future Work

The moving objects detection model proposed in this paper eliminates the relative movements between the background parts of different frames, then builds a background model using a unsupervised codebook model, and detects the moving objects in each frame precisely based on the background model. The comparasion with the classic codebook model illustrates the effectiveness of the images registration procedure on eliminating the motion of the camera, and the comparasions with the optical flow method and P.C.ICCV09 method illustrate the robustness and higher accuracy of our model.

But there are some defects in our model. The model contains some procedures with a high time complexity, such as extracting the SIFT features and computing the homography matrixes using a RANSAC method. It is difficult to be used in a real-time application temporarily. And exploring the speed up techniques, such as finding some faster image feature extraction methods and some faster way to get the homography matrixes, will be our future study.

Acknowledgement. This work was supported by Natural Science Foundation of China (61272240,60970047, 61103151), the Doctoral Fund of Ministry of Education of China (20110131110028) and the Natural Science foundation of Shandong province (ZR2012FM037).

References

1. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Computing Surveys 38(4), 1–45 (2006)
2. Moeslund, T., Hilton, A., Kruger, V.: A survey of advances in vision-based human motion capture and analysis. Comput. Vis. Image Und. 104(2-3), 90–126 (2006)
3. Zhang, J., Mao, X., Chen, T.: Survey of moving object tracking algorithm. Application Research of Computers 26(12) (December 2009)
4. Piccardi, M.: Background subtraction techniques: a review. In: 2004 IEEE International Conference on Systems, Man and Cybernetics, vol. 4, pp. 10–13, 3099–3104 (October 2004), doi:10.1109/ICSMC.2004.1400815
5. Huan, R., Wang, Z., Tang, X., Chen, Q.: Moving target detection under complex background based on code book. In: 2011 IEEE International Conference on Computer Science and Automation Engineering (CSAE), June 10-12, vol. 1, pp. 203–207 (2011), doi:10.1109/CSAE.2011.5953204

6. Xu, C., Tian, Z., Li, R.: A Fast Motion Detection Method Based on Improved Codebook Model. Journal of Computer Research and Development 47(12), 2149–2156 (2010) ISSN: 1000-1239/CN 11-1777/TP
7. B.K.P. Horn, B.G. Schunck: Determining optical flow. Artificial Intelligence 17(1-3), 185–203 (1981) ISSN 0004-3702, 10.1016/0004-3702(81)90024-2
8. Chockalingam, P., Pradeep, N., Birchfield, S.: Adaptive fragments-based tracking of non-rigid objects using level sets. In: 2009 IEEE 12th International Conference on Computer Vision, September 29-October 2, pp. 1530–1537 (2009), doi:10.1109/ICCV.2009.5459276
9. Vidal, R., Ma, Y.: A Unified Algebraic Approach to 2-D and 3-D Motion Segmentation. In: 8th European Conference on Computer Vision, Prague, Czech Republic, May 11-14 (2004), ISBN 978-3-540-21984-2, doi:10.1007/978-3-540-24670-1_1
10. Papageorgiou, C.P., Oren, M., Poggio, T.: A general framework for object detection. In: Sixth International Conference on Computer Vision, January 4-7, pp. 555–562 (1998), doi:10.1109/ICCV.1998.710772
11. Babenko, B., Yang, M.-H., Belongie, S.: Robust Object Tracking with Online Multiple Instance Learning. IEEE Transactions on Pattern Analysis and Machine Intelligence 33(8), 1619–1632 (2011), doi:10.1109/TPAMI.2010.226
12. Hartley, R., Zisserman, A.: Multiple View Geometry in computer vision, pp. 32–33. Cambridge University Press (2003) ISBN 0-521-54051-8
13. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proceedings of the International Conference on Computer Vision, vol. 2, pp. 1150–1157 (1999), doi:10.1109/ICCV.1999.790410
14. Wang, G., Wu, Q.M.J., Zhang, W.: Kruppa equation based camera calibration from homography induced by remote plane. Pattern Recognition Letters 29(16), 2137–2144 (2008) ISSN 0167-8655, 10.1016/j.patrec.2008.07.012
15. Wang, L., Dai, X., Ju, H.: Homography-based visual measurement of wheel sinkage for a mobile robot'. In: 2010 IEEE International Conference on Systems Man and Cybernetics (SMC), October 10-13, pp. 3543–3548 (2010), doi:10.1109/ICSMC.2010.5642362
16. Oisel, L., Memin, E., Morin, L., Galpin, F.: One-dimensional dense disparity estimation for three-dimensional reconstruction. IEEE Transactions on Image Processing 12(9), 1107–1119 (2003), doi:10.1109/TIP.2003.815257
17. Fischler, M.A., Bolles, R.C.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. of the ACM 24(6), 381–395 (1981), doi:10.1145/358669.358692
18. Brown, M., Lowe, D.G.: Automatic Panoramic Image Stitching using Invariant Features. International Journal of Computer Vision (IJCV) 74(1), 59–73 (2007) ISSN 0920-5691, doi: 10.1007/s11263-006-0002-3
19. Tsai, D., Flagg, M., Nakazawa, A., Rehg, J.M.: Motion Coherent Tracking Using Multi-label MRF Optimization. International Journal of Computer Vision (IJCV) 100(2), 190–202 (2012) ISSN 0920-5691, doi: 10.1007/s11263-011-0512-5
20. Kim, K., Chalidabhongse, T.H., Harwood, D., Davis, L.: Real-time foreground–background segmentation using codebook model. Real-Time Imaging 11(3), 172–185 (2005), ISSN 1077-2014, 10.1016/j.rti.2004.12.004
21. Horn, B.K.P., Schunck, B.G.: Determining optical flow. Artificial Intelligence 17(1-3), 185–203 (1981) ISSN 0004-3702, 10.1016/0004-3702(81)90024-2
22. `http://opencv.org/`
23. `http://www.mathworks.com/matlabcentral/fileexchange/authors/43158`
24. `http://cpl.cc.gatech.edu/projects/SegTrack/bmvc10.avi`