REAL-TIME DETECTION OF MOVING OBJECTS IN A ROTATING AND ZOOMING CAMERA

Yingbo Li, Won-ho Cho, and Ki-sang Hong

Department of Electronic & Electrical
Pohang University of Science & Technology
Pohang, Korea
{lantuzi, ellescho, hongks}@postech.ac.kr

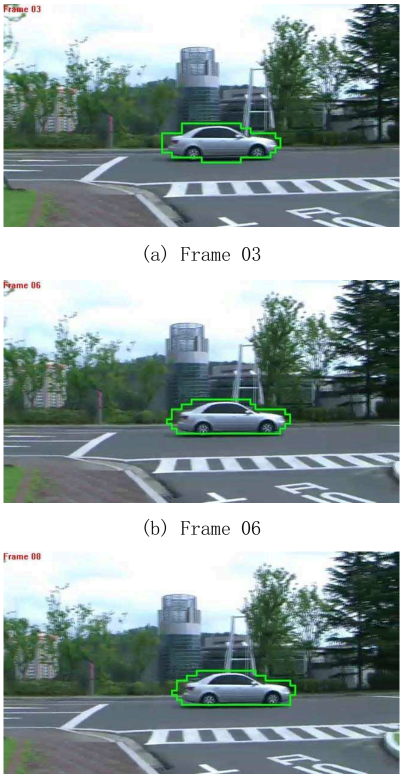
ABSTRACT

In this paper, we present a real-time method to detect moving objects in a rotating and zooming camera. It is useful for camera surveillance of fixed but rotating camera, camera on moving car, and so on. We first compensate the global motion, and then exploit the displaced frame difference (DFD) to find the block-wise boundary. For robust detection, we propose a kind of image to combine the detections from consecutive frames. We use the block-wise detection to achieve the real-time speed, except the pixel-wise DFD. In addition, a fast block-matching algorithm is proposed to obtain local motions and then global affine motion. In the experimental results, we demonstrate that our proposed algorithm can handle the real-time detection of common object, small object, multiple objects, the objects in low-contrast environment, and the object in zooming camera.

Keywords: block-matching, moving object detection, Displaced Frame Difference, confidence map, MRMCS, ARPS

1. INTRODUCTION

Detection of moving objects is a basic problem for object tracking, video surveillance, and so on. If we ignore the depth of background, it is relatively easy to capture the moving objects in a rotating and zooming camera. For example, we want to detect the car in Fig. 1. In previous papers, there are two kinds of detection algorithms: (1) The first kind reconstructs the background using image mosaic [1], trajectories of feature points [4] or a spatial distribution of Gaussian (SDG) model [2], and then subtracts the background from the frames to extract the moving objects. This kind of algorithms need enough frames to prepare the stable background model, so they cannot obtain the satisfied results in few frames. (2) The other kind exploits the optical flow and compensates the background motion between two frames [3]. In [3], it uses DFD to find moving objects after the compensation. But it is pixel-wise detection, so it is too slow to achieve real-time detection. While in our algorithm we use both advantages of fast block-wise detection and precise pixel-wise DFD, therefore the real-time detection is realized and the minimum number of frames for detection preparation is just three.



(a) Frame 03

(b) Frame 06

(c) Frame 08

Fig. 1. A moving car in moving camera

Our real-time algorithm for a rotating and zooming camera is practical for surveillance camera, because we can reduce the number of necessary cameras if a single moving camera can search larger region. And it can be used in road patrol for a moving police car too. We do not need the police car is stationary and it can carry out the real-time tasks when moving.

The proposed algorithm is illustrated in Fig. 2. In Section 2 we describe the overall algorithm of detection, based on local motions, global motion and DFD. If needed, we create confidence map to combine the detections of previous frames. In Section 3, we explain the novel BMA, which is much faster than real-time computation. In Section 4, we show by experimental results that our algorithm can detect different objects in diverse environments. Section 5 is the conclusion of this paper.

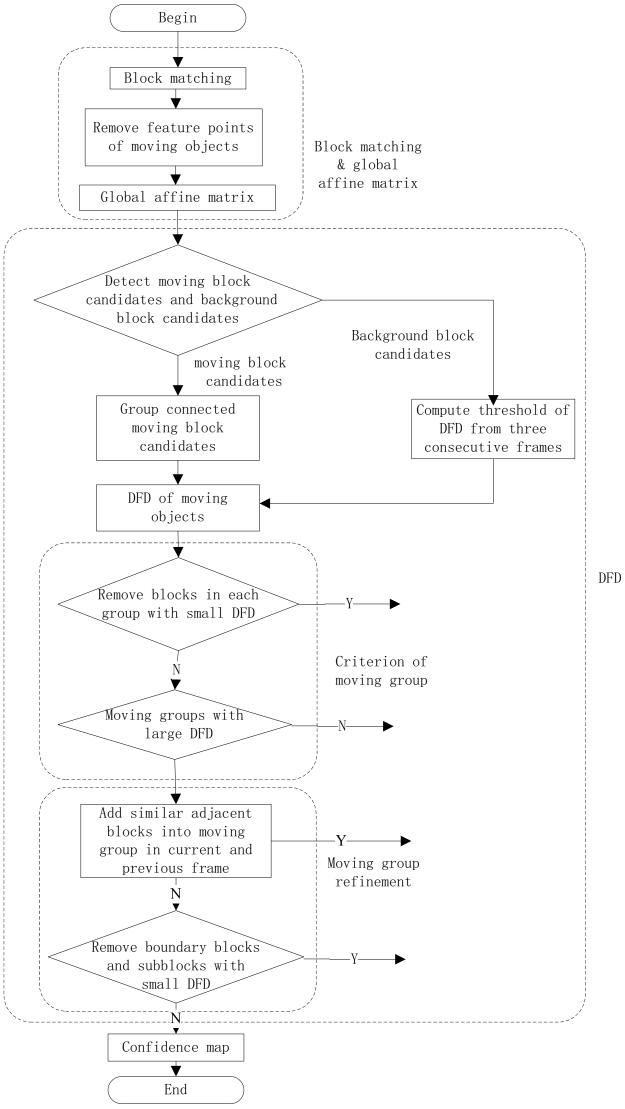

Begin
Block matching
Remove feature points of moving objects
Global affine matrix
Block matching & global affine matrix
Detect moving block candidates and background block candidates
Background block candidates
moving block candidates
Group connected moving block candidates
Compute threshold of DFD from three consecutive frames
DFD of moving objects
DFD
Remove blocks in each group with small DFD
Y
N
Criterion of moving group
Moving groups with large DFD
N
Add similar adjacent blocks into moving group in current and previous frame
Y
Moving group refinement
N
Remove boundary blocks and subblocks with small DFD
Y
N
Confidence map
End

Fig. 2. Detection flowchart

2. MOVING OBJECTS DETECTION ALGORITHM

Assume that we already have the local motion vectors (MVs) from a novel BMA, explained in Section 3. For a rotating and zooming camera, the frame motion can be described by affine transformation, defined as:

$$X_2 = AX_1 \tag{1}$$

where X_1 and X_2 are the coordinates of corresponding points belonging to one local MV between two frames, and A is affine matrix. For each local MV, the relation like Eq. 1 exists. We divide the whole frame into gridding, and the center of each block in the gridding is called block point. Then from local MVs of block points, we can compute the affine matrix A using Least Square algorithm.

2.1. Background Affine Motion

For the video with moving objects, some block points are on the moving objects. All the local MVs of block points used to compute affine matrix of background should be from the background. So it is necessary to decide a block point belongs to background or moving object.

We assume that most area are background, thus the global affine matrix, G, from all the block points should be similar with the background motion. And we can compute a global MV, $MV_G(x,y)$, for each block point from G. Squared distance difference (SDD) between $MV_G(x,y)$ and local MV $MV_l(x,y)$ from BMA is:

$$SDD = (MV_G(x) - MV_l(x))^2 + (MV_G(y) - MV_l(y))^2 \tag{2}$$

In the histogram of SDD, we consider 50% smaller differences and their corresponding blocks belong to background, while the others belong to moving objects. Here we call a block as moving block candidate (MBC) or background block candidate (BBC). We can compute the background affine matrix G_b from local MVs of BBCs by Least Square algorithm.

2.2. Displaced Frame Difference (DFD)

1. Group Connected MBC Every group of connected moving block candidates is considered as a candidate of moving object, called moving group candidate (MGC).
2. DFD Points In each MGC, compute DFD for three consecutive frames $I(x,y,t_1)$, $I(x,y,t_2)$ and $I(x,y,t_3)$ $(t_1 < t_2 < t_3)$.

$$M(x,y,t_2) = \begin{cases} 0, & if \;\; (|I_c(x,y,t_1) - I(x,y,t_2)|) < Th \cdot \hat{\sigma_1} \\ & and \; (|I_c(x,y,t_2) - I(x,y,t_3)|) < Th \cdot \hat{\sigma_2} \\ 1, & otherwise. \end{cases} \tag{3}$$

where,

$$\begin{aligned} \hat{\sigma_1}^2 &= \frac{\{[I_c(x,y,t_1) - I(x,y,t_2)] \cdot B_{(t_1,t_2)}(x,y)\}^2}{\{\sum_{\forall(x,y)} B_{(t_1,t_2)}(x,y)\} - 1} \\ \hat{\sigma_2}^2 &= \frac{\{[I_c(x,y,t_2) - I(x,y,t_3)] \cdot B_{(t_2,t_3)}(x,y)\}^2}{\{\sum_{\forall(x,y)} B_{(t_2,t_3)}(x,y)\} - 1} \end{aligned} \tag{4}$$

where B of one point in Eq. 4 is 1, if this point is in BBC; otherwise, $B = 0$. Th in Eq. 3 is variable depending on the video illumination, and I_c represents the compensated frame by G_b. M is called DFD value. If $M = 1$, the corresponding point belongs to moving object, and $M = 0$ means that it is a point of background. The examples of MBCs (blue blocks), BBCs (white blocks) and their DFD image are shown in Fig. 3.
3. Criterion of Moving Group First, for the blocks in a MPC, if DFD value in a block is less than 18% of the pixel number, this block is removed. Second, if DFD value in the residual blocks of MPC is larger than the threshold, 10% of pixel number, this MPC is moving group, and it is the position of a moving object.
4. Moving Group Refinement First, supplement the lost blocks. The intensity histograms of the blocks of the same

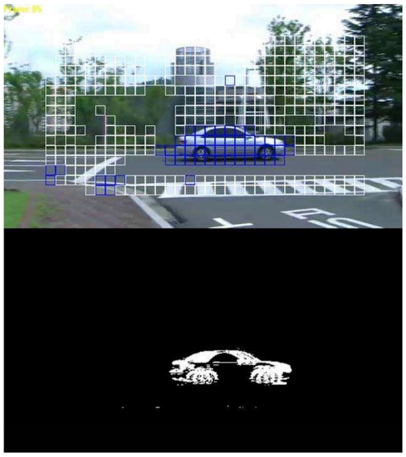


Fig. 3. (a)MBC, BBC and (b)DFD image

object are similar. So we check the similarity between adjacent blocks around moving group in current and previous frame, and the blocks of moving group by similarity function (Bhattacharyya coefficient).

$$f = f[\vec{q}, \vec{p}] = \sum_{u=1}^{m} \sqrt{p_u q_u} \tag{5}$$

where p, q are the intensity histograms of two blocks.

$$\begin{array}{ll} \vec{q} = \{q_u\}_{u=1\ldots m} & \sum_{u=1}^{m} q_u = 1 \\ \vec{p} = \{p_u\}_{u=1\ldots m} & \sum_{u=1}^{m} p_u = 1 \end{array}$$

If f is large, these two blocks belong to one object and the adjacent block is added into this moving group.

Second, refine the boundary of moving group. The blocks on the boundary are divided into four subblocks. The blocks and subblocks with small DFD are removed from moving group to achieve a more precise boundary.

2.3. Confidence Map

If it is impossible to obtain the full boundary of moving objects in each frame, we create an image, same size with frame and called confidence map (CM), to construct the boundary from the results of previous frames. And a value, confidence value (CV), is given to each pixel of CM. The initial CV is 0. If one pixel is on moving object in the first frame, CV of the same position is plus 1. Then the block with nonzero CV is on existing moving object (EMO) in CM. Frame by frame we check the similarity between all the detected objects of current frame and EMO. If the similarity is satisfied, the CVs of the same positions with detected objects are plus 1.

For the similarity, the detected object and EMO should be near in frame. And intensity histogram of two objects should be similar by Eq. 5. Because the blocks of a moving object move in the same direction and speed, MVs of two are similar. Then we can prove two objects are near and from the same moving object. Especially if MVs is different, CV is minus 1. At last, the blocks with large CVs are from moving objects.

3. MRMCS-ARPS BMA

In this section, we explain a fast Block-Matching Algorithm, named as MRMCS-ARPS, which improves the speed of the whole detection algorithm a lot. This BMA is based on Multi-Resolution Motion Search algorithm (MRMCS)[5], and Adaptive Rood Pattern Search algorithm (ARPS)[7][6].

We illustrate the proposed BMA in Fig. 4. MRMCS exploits the structure of 3 levels. The images of upper levels are constructed via 2:1 subsampling from the lower levels. MRMCS uses FSBMA to find two points with two minimum sum of absolute difference (SAD) in the coarsest level. And around these two points, it finds one result in middle level by modified FSBMA and the standard FSBMA is exploited in the finest level to find the final point. As we know, FSBMA wastes the computation time. And we find that a worse candidate can be accepted in the coarsest and middle level, because the result is refined by lower levels. So we use faster ARPS instead of FSBMA in the coarsest level. ARPS initially searches 5 points of rood pattern with the arm sizes which are the same with local MV of adjacent left block, illustrated in Fig. 5, for one time to predicate the approximate position, and refines the result by repeating search of rood pattern with variable arm, which is smaller and smaller from the size of initial search to the size of 1. In middle level, we reduce every point as searched candidate for FSBMA to one of every two points, because it achieves very similar result in our experiments. Standard FSBMA is still used in the finest level .

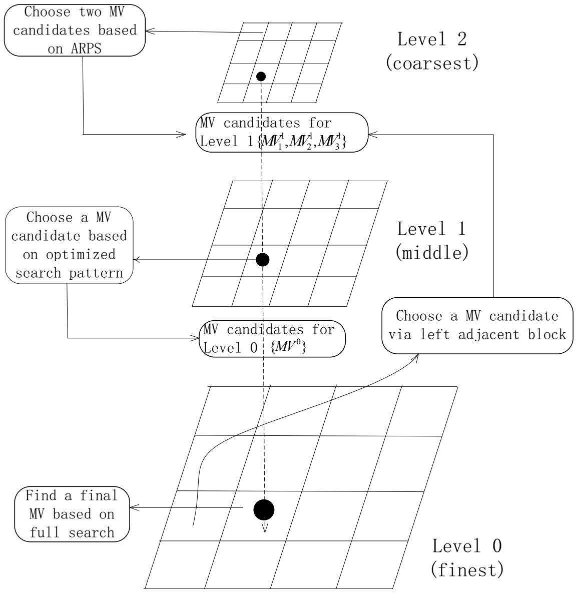

Choose two MV candidates based on ARPS
Level 2 (coarsest)
MV candidates for Level 1 $\{MV_1^1, MV_2^1, MV_3^1\}$
Choose a MV candidate based on optimized search pattern
Level 1 (middle)
MV candidates for Level 0 $\{MV^0\}$
Choose a MV candidate via left adjacent block
Find a final MV based on full search
Level 0 (finest)

Fig. 4. MRMCS-ARPS Flowchart

MRMCS-APRS is much faster than the real-time application, and has the similar quality with FSBMA, which is shown in Table 1 and 2. In Table 1 and 2, we show the speed and PSNR compare of FSBMA, MRMCS and MRMCS-ARPS. The proposed algorithm achieves similar PSNR with

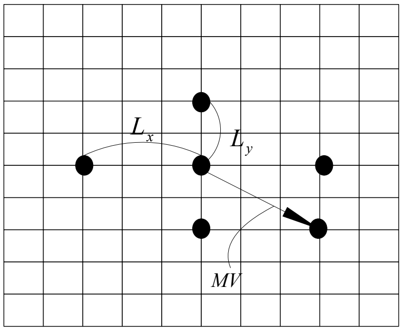


Fig. 5. ARPS Pattern

the other two methods, but it is 4 times the speed of FSBMA, and 1.75 times the speed of MRMCS. The experiments of standard MPEG test video are executed on Pentium4 2.6G. Video size is 352*288, and block size is 32. The main reason of fast speed is that we reduce the searched points. By MRMCS, we reduce about 75% points in each level, because of 2:1 subsampling. Furthermore, we converge to the result point much faster by prediction of ARPS.

Table 1. Speed (frames/second) Performances

Video	FSBMA	MRMCS	MRMCS-ARPS
Coastguard	35	80	137
Mobile	35	81	138
Foreman	35	80	141

Table 2. Average PSNR(dB) Performance

Video	FSBMA	MRMCS	MRMCS-ARPS
Coastguard	27.24	27.21	26.65
Mobile	23.74	23.71	23.69
Foreman	31.78	30.99	30.30

4. EXPERIMENTAL RESULTS

Block size for detection is 16 pixels, and all the experiments are executed on PC: $Intel\ Core2\ E6400$, and $2G$ memory. The speed is $30 \sim 35$ frames/s for $352 * 288$ video. The threshold Th in Eq. 3 is 6.0 for common illumination, and 3.0 for dark environment. The threshold for intensity similarity of blocks is 0.7. Fig. 1 is the detection of common object in good illumination. From Fig. 6 to 9, we show other situations, and confidence map is used in Fig. 6 and 7. In Fig. 6, we want to show the ability of our method to detect small object, which is just larger than the size of block. We can see that the boundary precision is good, though moving object is small. While, the precision and detection ability depend on the block size. In Fig. 7, it is the case of dark illumination. So it is hard to capture the object at the beginning of the video frames. In Fig. 7(a), only one part of object is detected. But more parts of object are detected frame by frame, just like Fig. 7(b) and (c). In Fig. 8 we demonstrate the proposed method could detect multiple objects, not only single object. In Fig. 9 the video is captured in the case of small rotation, but large zooming of camera. This example illustrates the capability of resolving this special case, and the moving car is perfectly detected.

5. CONCLUSION

In this paper, we have proposed a real-time detection algorithm for video captured by a rotating and zooming camera. The proposed algorithm could begin to robustly detect in few frames, and the minimum frame number is three frames. Furthermore, the detection speed is fast enough for real-time application. Together with the good speed, our algorithm can achieve a block-wise boundary based on pixel-wise DFD, and the detection precision is half of the block size. So we perfectly combine the block-wise and pixel-wise detection, and achieve a fast and robust detection algorithm. In addition, we suggest a novel BMA to quickly obtain the local motions, which accelerates the whole algorithm.

6. REFERENCES

[1] Alessandro Bevilacqua and Pietro Azzari. “A Fast and Reliable Image Mosaicing Technique with Application to Wide Area Motion Detection.” International Conference on Image Analysis and Recognition Vol. LNCS 4633, pp. 501-512, 2007.

[2] Ying Ren, Chin-Seng Chua, and Yeong-Khing Ho. “Motion Detection with Nonstationary Background.” Machine Vision and Applications Vol. 13, pp. 332-343, 2003.

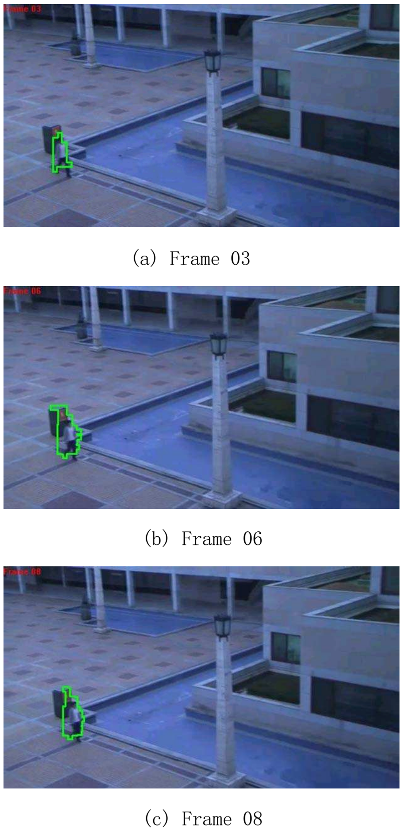
[3] Ninad Thakoor and Jean Gao. “Automatic Video Object Detection And Recognition With Camera In Motion.” IEEE International Conference on Image Processing 2005.

[4] Yasuyuki SUGAYA and Kenichi KANATANI. “Extracting Moving Objects from a Moving Camera Video Sequence,” Memoirs of the Faculty of Engineering, Okayama University Vol. 39, pp. 56-62, January 2005.

[5] Jae Hun Lee, Kyoung Won Lim, Byung Cheol Song, and Jong Beom Ra. “A Fast Multi-Resolution Block Matching Algorithm and its LSI Architecture for Low Bit-Rate Video Coding.” IEEE Transactions on Circuits and Systems for Video Technology Vol. 11, No. 12, pp. 542–545, December 2001.

[6] Yao Nie and Kai-Kuang Ma. “Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation” IEEE Transactions on Image Processing Vol. 11, No. 12, December 2002.

[7] Byung-Gyu KIM, Seon-Tae KIM, Seok-Kyu SONG, and Pyeong-Soo MAH. “Novel Block Motion Estimation Based on Adaptive Search Patterns.” IEICE TRANS. INF. & SYST. Vol. E89-D, No. 4, April 2006.

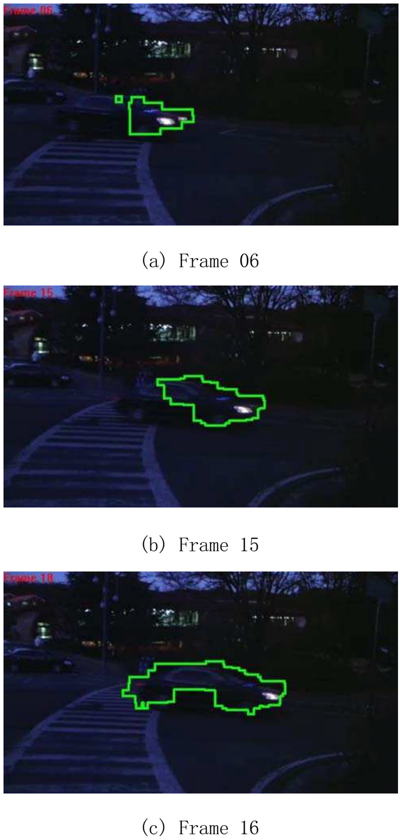


(a) Frame 03

(b) Frame 06

(c) Frame 08

Fig. 6. Detection of Small object

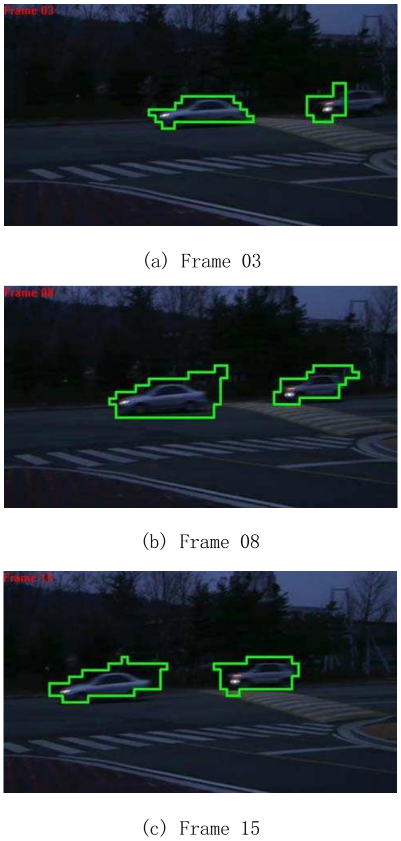


(a) Frame 06

(b) Frame 15

(c) Frame 16

Fig. 7. Detection in dark environment

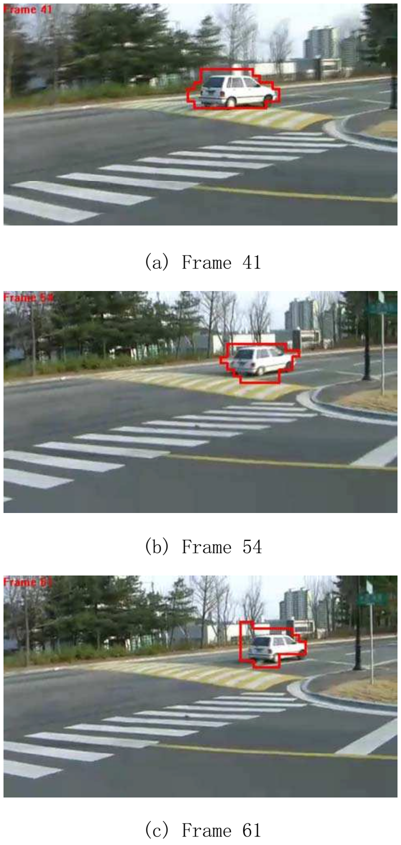


(a) Frame 03

(b) Frame 08

(c) Frame 15

Fig. 8. Detection of two objects



(a) Frame 41

(b) Frame 54

(c) Frame 61

Fig. 9. Detection in obviously zooming camera