A Novel Approach for Robust Surveillance Video Content Abstraction

LiMin Wang[1], Yirui Wu[1], Zhiyuan Tian[1], Zailiang Sun[1], and Tong Lu[1,2,*]

[1] State Key Lab. for Novel Software Technology, Nanjing University, China
[2] Jiangyin Institute of Information Technology of Nanjing University, China
{07wanglimin@gmail.com;wuyirui1989@163.com;
tianzhiyuan126@126.com;
shunzailiang@gmail.com;lutong@nju.edu.cn}

Abstract. Efficient video content analysis is an unsolved problem, especially for real-life surveillance videos due to their low resolution and illustration variations. A novel framework to efficiently and robustly convert a surveillance video clip into one abstraction image containing the integrated contour of interested objects is proposed. It has the following novelties: 1) an improved w-SIFT algorithm for Y-axis frames offset calculation, 2) a trapezoid-based compensation algorithm for X-axis perspective distortion correction, and 3) an incremental video content integration approach. Experimental results show that our method is robust for real-life low resolution videos and efficient for real-time analysis.

Keywords: surveillance video; video abstraction; content; w-SIFT.

1 Introduction

Video has been widely used in many fields like digital entertainment, e-business, military and security monitoring in recent years. How to effectively extract video contents is becoming one of the hotspots in real-life applications. Existing content analysis algorithms include key frame extraction [1-2], video summarization [3-5], object recognition [6] or scene interpretation [7-8]. However, automatic extraction of video contents is still one of the unsolved problems due to the following three reasons: 1) video content analysis is easily affected by illumination, perspective distortions or even shot quality; 2) video content is relatively complicate since even one frame may contain a number of objects, making precise contents analysis very difficult; and 3) real-time video content analysis is difficult due to the low efficiency of existing algorithms. Therefore, how to effectively and efficiently extract important contents from a large amount of binary video data streams has become one of the bottlenecks in video-based applications.

In this paper, we present a novel approach for robust surveillance video content abstraction. Our method consists of the following three steps: 1) extract moving regions from sampled frames of a surveillance video; 2) calculate frame offsets by an improved SIFT-based algorithm within the extracted contours; and 3) eliminate the perspective

G. Qiu et al. (Eds.): PCM 2010, Part II, LNCS 6298, pp. 660–671, 2010.

distortions by gradient compensations. Finally one single abstraction image containing the complete contours of the moving objects can be obtained after merging and fusion operations, well revealing the interested contents of the surveillance video.

Our method has the following advantages. First, our method provides an integrated abstract image in a more direct approach than other algorithms like key-frame extraction. Second, our method is robust for real-life low resolution videos (e.g., with illumination variations or perspective distortions). Moreover, our method is efficient and can provide nearly real-time video content extraction results according to our experimental results.

The rest of the paper is organized as follows. Section 2 introduces the background of our research and related work, and then Section 3 gives our approach. Experimental results and discussions are presented in Section 4. Finally, Section 5 concludes the paper with discussions of future work.

2 Related Research

Typical video content analysis processes are shown in Fig. 1. Though the detailed algorithms in each process may be distinct from each other, they share a similar idea of selecting the lowest number of frames to describe video contents. However, it is difficult to accurately define such frames and different methods may obtain varied results.

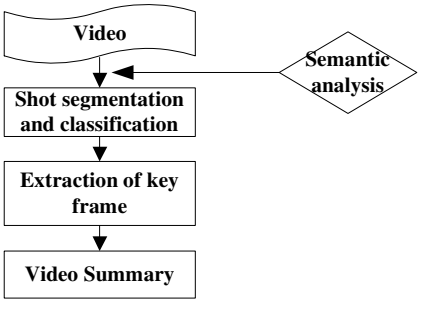


Fig. 1. Typical video content analysis processes

Our research focuses on directly converting a clip of surveillance video into one abstraction image containing the integrated interested contents. Take a real-life buried vehicle chassis surveillance video as an example, our target is to extract an abstraction image containing the complete chassis contour. Existing stitching algorithms for panoramic images generation are somewhat similar to our target. For example, Szeliski [9] uses Levenberg-Marquardt iteration for nonlinear minimization to align related images, while Brown and Lowe [10] merge images by calculating SIFT feature points. However, these algorithms have the following hypotheses: 1) there should be enough matched feature points between two related images to use the least square method, and 2) the matched feature points should be relatively accurate and there can't be three collinear feature points to solve the following equation:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} m_o & m_1 & m_2 \\ m_3 & m_4 & m_5 \\ m_6 & m_7 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \tag{1}$$

where (x, y) and (x_1, y_1) are from different images to be matched, and m_0 to m_7 are 8 unknown parameters needing 4 matched feature points to solve. Unfortunately, in low resolution surveillance videos with varied illuminations or perspective distortions, such two hypotheses can't be well met.

3 Our Approach

We propose a novel approach for robust surveillance video content abstraction as follows: *moving objects detection*, *Y-axis frame offset calculation*, *X-axis perspective distortion correction*, and *video content integration*. The framework of our algorithm is shown in Fig. 2.

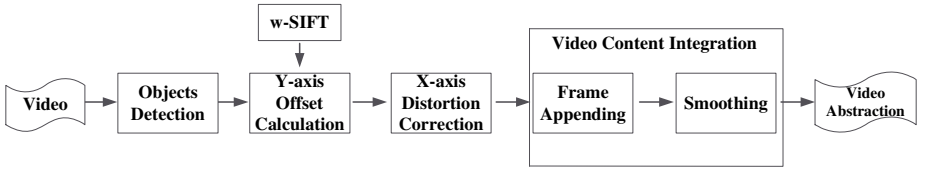


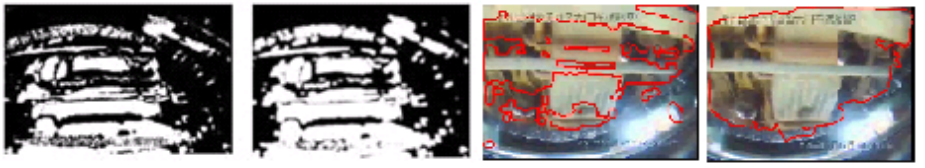
Fig. 2. The framework of our algorithm

3.1 Moving Objects Detection

In vehicle surveillance videos, we consider the moving vehicle chassis as our interested content. We use the differential frames arithmetic method [11] to detect the moving objects on a vehicle chassis. Considering real-time response, we take every 3 frames to detect our interested contents. Fig. 3 shows an example of the detected results from a real-life automotive chassis surveillance video, where Fig. 3(a) is the binarization result of the difference between two neighboring frames, while Fig. 3(b) gives the merged regions after using Gaussian pyramid decompositions to expand the holes in Fig. 3(a). Then we represent the irregular moving regions as follows:

(1) Sample along with each region contour and fit the sampled points by linear Hough transformations. As a result, we obtain a group of polygonal contours (see Fig. 3(c));
(2) Traverse the polygonal contours and discard each tiny polygon p_i by

$$Area(p_i < \lambda_0(x, y)) \tag{2}$$



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

Fig. 3. Moving objects detection and representation. (a) Binarization of the difference result from a vehicle chassis surveillance video, (b) inflation result from (a), (c) corresponding polygons, and (d) the result chassis fitted by snake curves.

where *Area* is the area of p_i, λ_0 is a threshold ratio to reject tiny regions, while x and y denote the width and height of a video frame, respectively;

(3) Fit the rest polygons by [12] and obtain the contour point set $snake(f_i)$.

Fig. 3(d) gives the fitted chassis, where the static background regions can be easily removed. Then we define *Ic* as the maximum inner rectangle (see Fig. 4) shared by all the detected snake contours from each frame. The succeeding video abstraction process in our framework is actually the integration result of $Ic(f_i)$ from each frame.

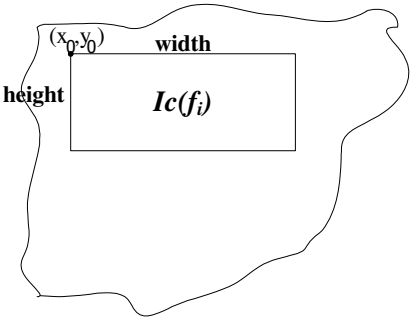


Fig. 4. *Ic* is defined as a rectangular region shared by all the video frames, and $Ic(f_i)$ is a current rectangular region of the frame f_i

To calculate *Ic*, we first calculate $Ic(f_i)$ from each frame f_i as follows:

$$Ic(f_i) \begin{cases} x_0 = (1-a)\min(x_i) + a\max(x_i) \\ y_0 = (1-c)\min(y_i) + c\max(y_i) \\ height = (1-c-d)(\max(y_i) - \min(y_i)) \\ width = (1-a-b)(\max(x_i) - \min(x_i)) \\ (x_i, y_i) \in snake(f_i) \end{cases} \tag{3}$$

where a, b, c and d are the coefficients to adjust the size of $Ic(f_i)$ (We set a=0.05, b=0.35, c=0.05, and d=0.50). Then we use the following function to calculate *Ic*:

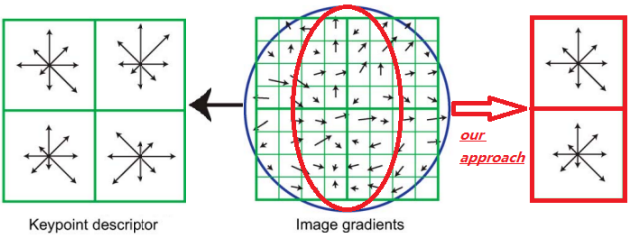


Fig. 5. An improved w-SIFT algorithm to match more feature points

$$Ic \begin{cases} x_{avg} = \sum_{i=0}^{N} w_i x_i \\ y_{avg} = \sum_{0}^{N} w_i y_i \\ height_{avg} = \sum_{0}^{N} w_i height_i \\ width_{avg} = \sum_{0}^{N} w_i width_i \qquad where \sum_{0}^{N} w_i = 1 \end{cases} \tag{4}$$

where w is a weight factor to refine the size of Ic.

3.2 Y-axis Offset Calculation

Next, we need further match the detected regions for each two neighboring frames to calculate their offsets. SIFT [13] is proved to be a robust and stable descriptor in image matching or content analysis [14-15]; however, there exist the following difficulties in dealing with our surveillance videos. First, too few matched key points can be extracted using SIFT due to the low resolution and illustration variations in real-life vehicle surveillance videos, making neighboring frames matching even impossible. Irregular vehicle chassis and movements make this problem much worse. Second, 128-dimension SIFT features make video content extraction inefficient, especially for real-time video content analysis.

To solve the problems, we hypothesize that a surveillance camera has a fixed viewpoint and a vehicle moves along straightly in a short time interval. Obeying the hypothesis, we propose a weighted SIFT (w-SIFT) descriptor to match more key points from neighboring frames. w-SIFT has two improvements against [13]: in *descriptor generation* phase, w-SIFT computes the 2D Gaussian function using

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{\frac{-((\xi_x x)^2 + (\xi_y y)^2)}{2\sigma^2}} \tag{5}$$

where ξ_x and ξ_y are two coefficients representing the weights along X-axis and Y-axis, respectively. Fig. 5 gives an example of $\xi_x = 2$ and $\xi_y = 1$, producing a reduced

64-dimension feature vector for fast SIFT matching. As a result, in *key points matching* phase, we can fast choose the nearest neighboring key point pair (x_1, y_1) and (x_2, y_2) by minimizing

$$d_{final}(x_1, y_1),(x_2, y_2) = d_{Euclidean}((x_1, y_1),(x_2, y_2)) + d_{extra}((x_1, y_1),(x_2, y_2)) \quad (6)$$

$d_{Euclidean}$ is Euclidean distance between the two points, while d_{extra} is used to reject the false matching feature pairs and keep the correct ones. d_{extra} is defined as follows:

$$d_{extra} = \begin{cases} \infty, & y_2 - y_1 > 0 \; or \; |x_2 - x_1| > \xi_{\max} \\ 0, & otherwise \end{cases} \quad (7)$$

where ξ_{max} is a maximum offset threshold along X-axis. As a result, w-SIFT obtains a group of matched key point pairs for two neighboring frames f_i and f_{i+1} as follows:

$$\left\{\left\langle A_{(i+1)k}(x_{(i+1)k}, y_{(i+1)k}), A_{ik}(x_{ik}, y_{ik})\right\rangle\right\}(i = 1,2,..., N-1)(k = 1,2,..., n(f_{i+1}, f_i)) \quad (8)$$

where $n(f_{i+1}, f_i)$ is the number of the matched key point pairs.

Next, we estimate the Y-axis offset $\Delta\tilde{y}_i(f_{i+1}, f_i)$ by matching $Ic(f_{i+1})$ and $Ic(f_i)$ as follows. Let $\Delta x_{ik} = x_{(i+1)k} - x_{ik}, \Delta y_{ik} = y_{(i+1)k} - y_{ik}$, we obtain the following set S_i:

$$S_i = \left\{\left\langle \Delta x_{i1}, \Delta y_{i1}\right\rangle, \left\langle \Delta x_{i2}, \Delta y_{i2}\right\rangle,..., \left\langle \Delta x_{ik}, \Delta y_{ik}\right\rangle,...\right\} \quad (9)$$

Then $\Delta\tilde{y}_i(f_{i+1}, f_i)$ can be calculated by

$$\Delta\tilde{y}_i(f_{i+1}, f_i) = \sum_{j=1}^{n(f_{i+1}, f_i)} \xi_j \Delta y_j \quad (10)$$

where ξ_j is the weight of Δy_j and can be calculated by

$$\xi_j = \frac{K_j}{\sum K_j} \quad (11)$$

K_j is a distance operator defined by

$$K_j = \frac{1}{n(f_{i+1}, f_i) - 1} \sum_{k=1}^{n(f_{i+1}, f_i)} (\Delta y_j - \Delta y_k)^2 \quad (12)$$

Fig. 6 shows an example of Y-axis offset estimation, where $A_i(x_{Ai}, y_{Ai})$ and $B_i(x_{Bi}, y_{Bi})$ are example matched feature points from two neighboring frames f_A and f_B, respectively.

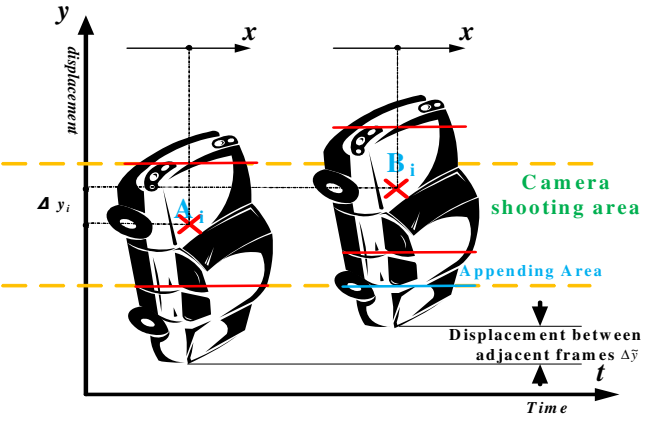


Fig. 6. Estimation of the Y-axis offset $\Delta\bar{y}(f_A, f_B)$ between f_A and f_B. $A_i(x_{Ai}, y_{Ai})$ and $B_i(x_{Bi}, y_{Bi})$ are two example matched feature points from two neighboring frames f_A and f_B

3.3 Correction of X-axis Perspective Distortions

In Y-axis offset calculation, we have a hypothesis of $|\Delta x| \approx 0$ to fasten the frame matching process. Actually, there still exists X-axis perspective distortions since camera lens may be not exactly vertical to the vehicle chassis plane (see Fig. 7(a)). Therefore, our next task is to correct the X-axis perspective distortions before video content integration.

We model this problem as follows. Considering vehicle moves along straightly in a short time interval, different objects on a chassis may be approximately described by a set of parallel lines (see Fig. 7(b)). Similarly, considering the characteristics of a camera lens, we can use Fig. 7(c) to describe an imaging plane. As a result, the projections from a vehicle chassis to the imaging plane should be a group of divergent trapezoid shapes. Therefore, X-axis perspective distortion correction is actually a reverse trapezoid-based compensation process.

According to the above analysis, we use trapezoid-based reverse transformations for X-axis perspective distortion correction. Suppose (x, y) is a point from a captured video frame, while (x', y') is its corresponding corrected point, then (x, y) and (x', y') obey the following linear transformation model:

$$\begin{bmatrix} x' \\ y' \\ \omega' \end{bmatrix} = \begin{bmatrix} S_u & 0 & 0 \\ 0 & S_v & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ \omega \end{bmatrix} \tag{13}$$

where (ω, ω') are Homogeneous coordinates and S_u, S_v are zoom ratios in X-axis and Y-axis, respectively. Obviously (x', y') can be calculated from (x, y) as follows:

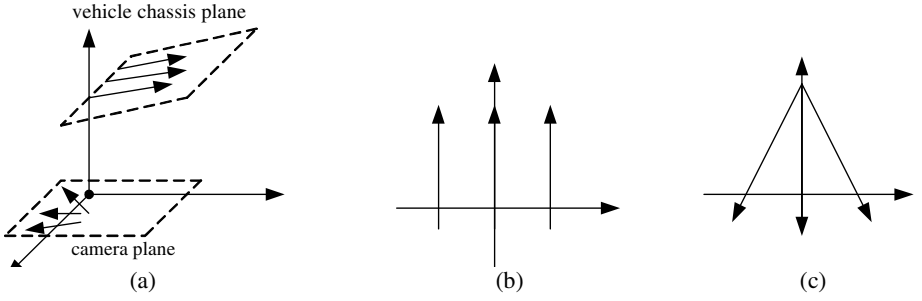


Fig. 7. X-axis perspective distortion model. (a) Perspective distortion is caused by the angle between the camera and vehicle chassis planes, (b) the vehicle chassis plane, and (c) the camera imaging plane

$$\begin{cases} x = x'/S_u \\ y = y'/S_v \end{cases} \tag{14}$$

Considering the results may not be integers, we use the following bilinear interpolation method to estimate:

$$\begin{aligned} F(Ic(f_n))(x', y') = d_y[d_x Ic(f_n)(\lceil x \rceil - 1, \lceil y \rceil - 1) + (1 - d_x) Ic(f_n)(\lceil x \rceil, \lceil y \rceil - 1)] \\ + (1 - d_y)[d_x Ic(f_n)(\lceil x \rceil - 1, \lceil y \rceil) + (1 - d_x) Ic(f_n)(\lceil x \rceil, \lceil y \rceil)] \end{aligned} \tag{15}$$

where $\lceil x \rceil$ denotes the *ceiling*(x) function and $d_x = \lceil x \rceil - x$, $F(Ic(f_n))$ is the result of distortion correction. Since we only consider distortions along X-axis in this step, then (15) can be simplified as follows:

$$F(Ic(f_n))(x', y') = d_x Ic(f_n)(\lceil x \rceil - 1, \lceil y \rceil) + (1 - d_x) Ic(f_n)(\lceil x \rceil, \lceil y \rceil) \tag{16}$$

Considering our compensation is a trapezoid-based process, the X zooming ratio should become greater from the top to the bottom in the final abstraction image. Therefore we can calculate the X zooming ratio as a linear function of coordinate Y:

$$S_u = S_o + Ky \tag{17}$$

S_0 and K can be initially estimated by the prior parameters such as vehicle speed range or height range. Fig. 8 illustrates this compensation process.

3.4 Video Contents Integration

Finally we integrate video contents from each frame as follows. We first divide each $F(Ic(f_i))$ into two regions: a fusion region and a stretching region. The former is used to fuse with the result abstraction image *Ir* and the latter is used to directly append to the end of *Ir*. Fig. 9 shows the details.

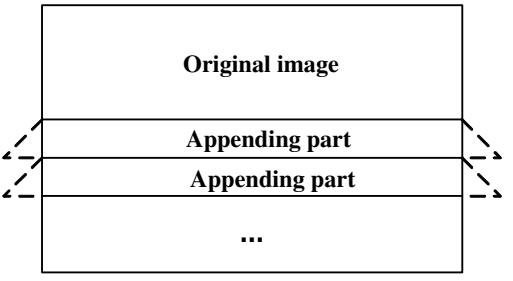


Fig. 8. Trapezoid-based compensation for X-axis perspective distortion

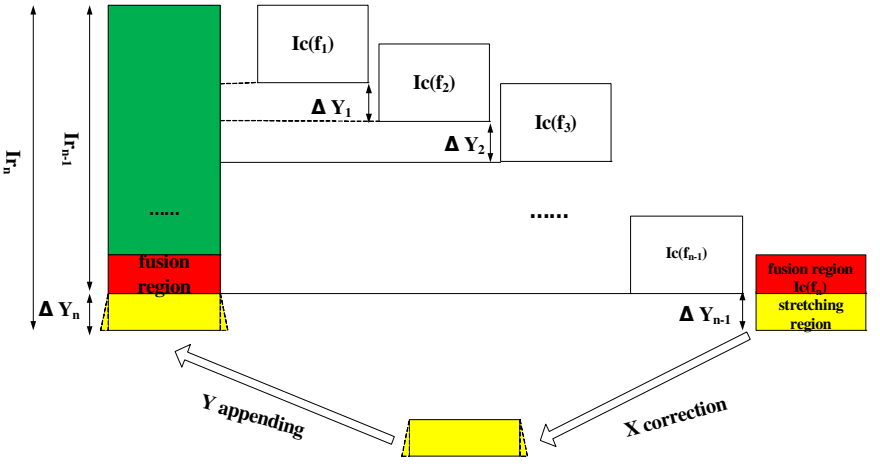


Fig. 9. A new $F(Ic(f_n))$ is divided into a fusion region and another stretching region (right). The fusion region is used to fuse with the result abstraction result image Ir_{n-1} (left) while the stretching region is directly appended at the end of Ir_{n-1}.

In Fig. 9, we integrate all the frames by a recurrence equation of

$$Ir_n(x,y) = \begin{cases} Ir_{n-1}(x,y), (x,y) \in Ir_{n-1} - Ic(f_n) \\ (1-\omega(y))Ir_{n-1}(x,y) + \omega(y)Ic(f_n)(x,y), (x,y) \in Ir_n \cap Ic(f_n) \\ F(Ic(f_n))(x,y), (x,y) \in Ic(f_n) - Ir_{n-1} \end{cases} \quad (18)$$

where Ir_n is the result of n-th integration and note that $Ir_1 = Ic(f_1)$, $F(Ic(f_n))$ is defined in (16) and $\omega(y)$ is the weight function. In our experiment, we set it as a linear function of coordinate Y:

$$\omega(y) = y / length \quad (19)$$

where *length* denotes the length of fusion region.

4 Experiments and Discussions

We test our algorithm on real-life low resolution surveillance videos on VC++ 6.0 platform with 2.40 GHz CPU. The video abstraction result is shown in Fig. 10, where Fig. 10(a) shows a group of example frames from a vehicle chassis surveillance video, while Fig. 10(b) illustrates their corresponding *Ic* images. Fig. 10(c) is part of the final video abstraction after content integration and fusion from Fig. 10(a).

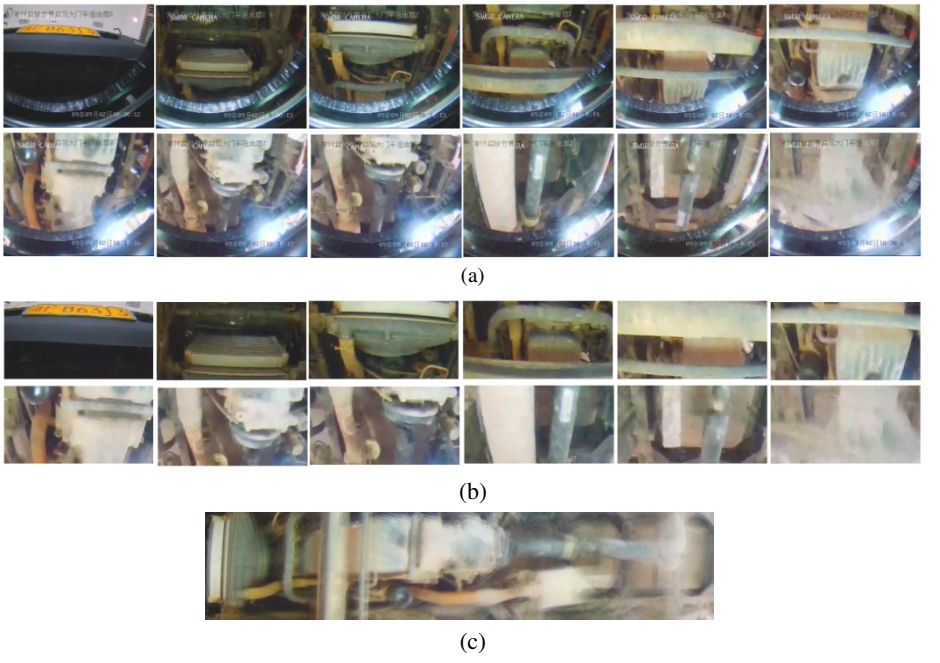


Fig. 10. Video abstraction result. (a) Example frames from a real-life vehicle chassis surveillance video, (b) their corresponding common images (I_c), and (c) part of the final video abstraction image after content integration and fusion

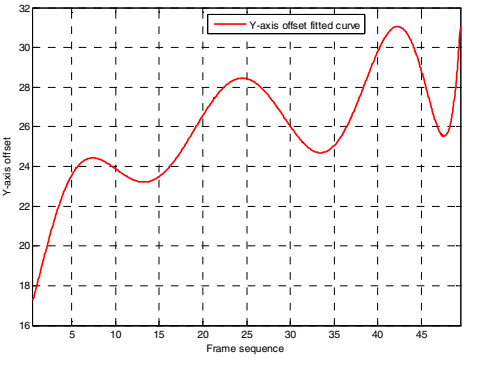


Fig. 11. The fitted Y-axis offset $\Delta\tilde{y}(f_A, f_B)$ curve from video frames

Note that the quality of the last frame in Fig. 10(b) is too poor to obtain any SIFT features. We use a fitted $\Delta\tilde{y}(f_A, f_B)$ curve (see Fig. 11) to estimate its Y-axis offset.

Our next experiment is to compare the performances between w-SIFT and [13] using our low resolution surveillance videos. Fig. 12 gives the experimental results of matched SIFT features between neighboring video frames. It can be seen that nearly 340% matched SIFT features can be improved with our method.

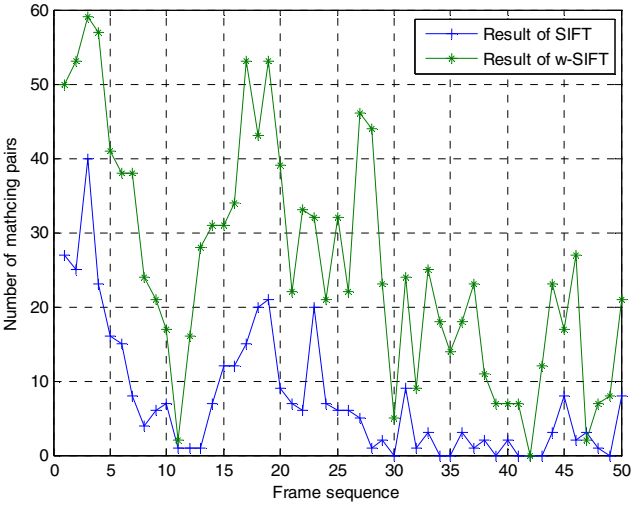


Fig.12. Performance comparison between w-SIFT and SIFT

5 Conclusions

This paper presents a novel approach to efficiently and robustly convert a surveillance video into one abstraction image containing the integrated contour of interested objects. Experimental results show that the method is robust for real-life low resolution videos and efficient for real-time analysis. Further research includes adapted threshold calculation and more accurate moving regions detection.

Acknowledgements. The work described in this paper was supported by the Natural Science Foundation of JiangSu under Grant No. BK2009082, the Natural Science Foundation of China under Grant No. 60723003 and 60721002, and the 973 Program of China under Grant No. 2010CB327903.

References

1. Gibson, N.D., Thomas, B.: Visual abstraction of wildlife footage using Gaussian mixture models. In: Proc. The 15th International Conference on Vision Interface, pp. 11–17 (2002)
2. Yu, X.D., Wang, L., Tian, Q., Xue, P.: Multi-level video representation with application to keyframe extraction. In: Proc. IEEE Multimedia Modeling (MMM), pp. 117–121 (2004)

3. Golman, D.B., Curless, B., Salesin, D., Seitz, S.M.: Schematic storyboarding for video visualization and editing. ACM Transactions on Graphics (ACM SIGGRAPH) 25(3), 862–871 (2006)
4. Damnjanovic, U.: Event detection and clustering for surveillance video summarization. In: Proc. IEEE 9th Workshop on Image Analysis for Multimedia Interactive Services, Klagenfurt, Austria, pp. 63–66 (2008)
5. Lu, S., King, I., Lyu, M.: Video summarization by video structure analysis and graph optimization. In: Proc. IEEE ICME 2004, Taipei, Taiwan, pp. 1959–1962 (2004)
6. Wang, G., Zhang, Y., Li, F.F.: Using dependent regions for object categorization in a generative framework. In: CVPR, pp. 1597–1604 (2006)
7. Kim, C.: An integrated scheme for object-based video abstraction. In: Proc. of ACM Multimedia 2001, pp. 303–309 (2001)
8. Liu, T.M., Zhang, H.J., Qi, F.H.: A novel video key frame extraction algorithm based on perceived motion energy model. IEEE Trans. on Circuits System for Video Technology 13(10), 1006–1013 (2003)
9. Richard, S.: Video mosaics for virtual environments. IEEE Computer Graphics and Applications 16(2), 22–30 (1996)
10. Matthew, B., David, L.: Automatic Panoramic Image Stitching using Invariant Features, pp. 50–73 (2007)
11. Rui, Y., Huang, T.S., Mehrotra, S.: Exploring video structure beyond the shots. In: International Conference on Multimedia Computing and System, pp. 237–240 (1998)
12. Kass, M., Witkin, A., Terzopolulos, D.: Snakes: Active Contour Models. International Journal of Computer Vision 4, 321–333 (1987)
13. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
14. Lyu, M.R., Song, J.: A Comprehensive Method for Multilingual Video Text Detection, Localization, and Extraction. IEEE Transactions on Circuits and Systems for Video Technology, 243–255 (2005)
15. Kim, W., Kim, C.: A New Approach for Overlay Text Detection and Extraction from Complex Video Scene. IEEE Transactions on Circuits and Systems for Video Technology, 401–411 (2005)