ISSN 1746-7659, England, UK
Journal of Information and Computing Science
Vol. 5, No. 2, 2010, *pp*. 089-098

A Fast Vision-Based Air Object Detection Algorithm

Huan Wang[+], Mingwu Ren, Jingyu Yang

School of computer science, Nanjing university of science & technology, Nanjing, PR. China. 210094

(*Received July 2, 2008, accepted October 20, 2008*)

Abstract. Most of air object detection techniques are based on laser or radar sensor. This paper addresses a novel technique for air object detection which is based on computer vision, namely it detects object with sequence images acquired by visible or thermal infrared cameras. It is based on single frame detection other than classical background subtraction, which means that it can tolerate motions or jitters of a camera and sudden changes of illumination. Moreover, it can be used both in visible-light and infrared images. Extensive experiments demonstrate the effectiveness and efficiency of the proposed algorithm. It has been successfully used in movable surveillance system.

Keywords: Air object detection, Chain-code, OTSU Segmentation, MRFs

1. Introduction

As a special aspect, air object detection is an important research portion in object detection domain. It can be used in many fields such as battlefield surveillance, missile intercepting, and so on. Currently, most of air object detection techniques are based on laser or radar sensor. They detect objects actively by launching laser or electromagnetic waves. Unfortunately, these transmitted signal are often spoofed by adversaries' air objects aiming at battlefield surveillance, therefore, this technique often leads to the exposure of our military targets, while a passive sensor, cameras for example, can be more competent in the case, they launch no signals, and can be freely moved when put in movable vehicles, they can also obtain plentiful air object information. In this paper, we address a novel air object detection technique which is based on computer vision, namely, it detects object with sequence images acquired by cameras. In computer vision, although a lot of studies about object detection (such as face detection, pedestrian detection, vehicle detection, and more generally, moving object detection, just to name a few) have been conducted in recent years, there are only a few literatures in terms of air object detection, its studies often be included in moving object detection, so regular methods about air object detection are similar to moving object detection approaches. Generally, there are three types of these methods: optical flow method, temporal difference method and background subtraction method. Optical flow [1],[2] methods aim at computing an approximation of the 2D motion field by means of spatio-temporal changes of image intensity. They can be used in the presence of camera motion, but most optical flow methods are computationally complex, and it cannot be applied to full-frame video streams in real-time without specialized hardware. In contrast, temporal difference [3-4] method takes into account differences between two consecutive frames, this approach is very adaptive to dynamic environment, but it is strictly dependent on the velocity of moving object in the scene and it is subject to the foreground aperture problem. Compared with the above two methods, background subtraction [5],[6] method is more practical and effective. It makes use of the image sequence itself to generate and maintain a background image, and considers differences between current image and the background image. The straightforward background includes average background, median background. In [5], the mixture of Gaussians approach is used to generate background, where each pixel is modeled as mixture of several Gaussians. What's promising of this method is it can effectively detect objects in the scene background which is not completely static, such as the background includes repetitive motions like swaying vegetation, rippling water, flickering monitors, camera jitter, etc. However this algorithm has to allocate a huge computer memory, and it takes a long time to produce a valid background. LBP based object detection algorithm [6] uses local binary pattern to describe the background texture, each pixel is modeled by multiple LBP histograms. However, LBP detection algorithm is very time-consuming, this is mainly due to the need of many LBP histogram

[+] Corresponding author. *E-mail address*: wanghuan_ywzq@tom.com.

operations (construction, update and comparison) for each pixel in each frame. Moreover, it performs poorly in weak texture backgrounds, especially the sky background which will be discussed in this paper.

For an air object detection system, cameras are often put in a movable vehicle, where camera motions and jitters always occur, so temporal difference method or background subtraction method often fails in this case. In addition, it has been stated that sky scene is a weak texture background, so it is difficult for some feature corresponding-based algorithms [7],[8] to detect objects reliably. Besides, a fast processing speed is absolutely necessary since air objects often have a high motion speed. In this paper, we present a fast air object detection algorithm. It is based on single frame detection other than classical background subtraction approach, which means that it can tolerate motions or jitters of the camera and sudden illumination changes. Moreover, it can be used both in visible-light and infrared image. Extensive experiments demonstrate the effectiveness and efficiency of the proposed algorithm. It has been successfully used in movable surveillance system.

This paper is organized as follows, in section 2, the proposed algorithm is described in detail; Experiment results and performance analysis of the algorithm are given in section 3; conclusions are drawn in the last section.

2. Proposed algorithm

Procedure of our proposed air objects detection algorithms are shown in Fig 1. The detection algorithm is based on single frame detection. For each frame, a background is extracted by a filter operator with a large size filter window from the frame. In comparison to other classical object detection algorithms, such as background subtraction or temporal difference, it can tolerate the moving of the camera, and adapt illumination changes automatically.

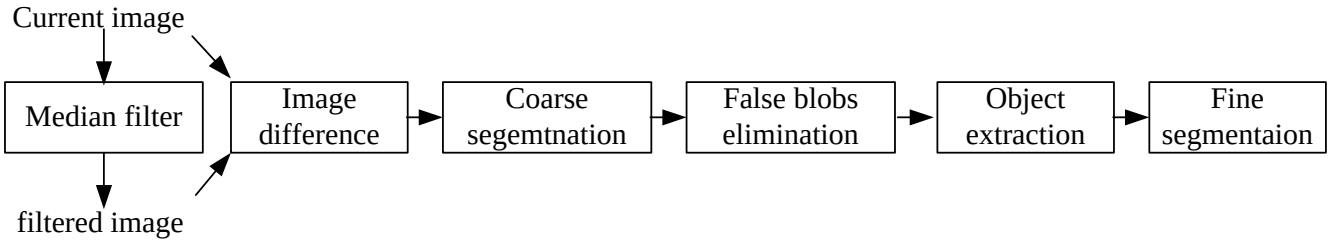

Fig.1 procedure of proposed method

2.1. Background Extraction

Considering sky background takes up most of an image and air objects submerges in the sky background, so we use median filter to current image with a fixed filter window to obtain a background image without any object, the background image will be utilized for detection in the following steps. Note that the window size should be larger than the maximum size of all objects appear in the image. In general, it is enough to use a 41×41window size, all the experiments in this paper adopt this size. Fig. 2(b) shows a background image extracted from Fig. 2(a) with Median filter. Median filter is good both at removing image noises and preserving image details such as edges. However, it is very time-consuming when a large window size is used. In order to accelerate the processing speed of median filter algorithm, a slip window technique-based median filter algorithm [9] is used to improve its efficiency, it utilizes histogram statistic and the overlapping relation of adjacent filter windows to reduce the time cost in sorting. For each row in each frame, a histogram of the left most image region within the filter window is constructed, and median value of all pixels gray value within the window image is obtained from the histogram, then the window is shifted horizontally pixel by pixel, the histogram of the new window centered at a new pixel is obtained iteratively by subtracting the left most column of the last filter window and add the right most column of current filter window from the last histogram, the new median value is obtained automatically, The algorithm is summarized as follows:

Table 1: The fast median filter algorithm

1. Set $th = \frac{w \cdot h}{2}$, w, h represent the width and height of filter window respectively.

2. Move the window to a new image row, construct the histogram H of the image in the window, then find the median value med from the histogram, record the number of pixels Lt_med which is less or equal to med .

3. For each pixel p whose value is p_g in the most-left column of current window, do $H(p_g) = H(p_g) - 1$

4 Move the window toward right by a pixel, for each pixel p whose value is p_g in the most-right column of current window, do

$$H(p_g) = H(p_g) + 1$$

$$\text{If } p_g < med \text{ , do } Lt\_med = Lt\_med + 1$$

5. If $Lt\_med > th$, go to step 6.

Else Repeat:

$$Lt\_med = Lt\_med + H(med)$$

$$med = med + 1$$

Until $Lt\_med > th$

6. Repeat:

$$med = med - 1$$

$$Lt\_med = Lt\_med - H(med)$$

Until $Lt\_med \le th$

7. If the right most column of current window is less than the image width, go to Step3.

8. If the bottom most row of the window is less than the image height, go to Step2.

This algorithm is independent of the windows size, which means that its time cost doesn't rely on the size of filter window.

2.2. Image Difference

After the background image has been extracted, it is subtracted from the original image for obtaining a difference image. Assuming current frame is $f(x, y)$, its resulting image after filtering (or background image) is $\tilde{f}(x, y)$, their difference image is defined by their absolute difference (see Fig. 2(c)):

$$d(x, y) = \left| f(x, y) - \tilde{f}(x, y) \right| \tag{1}$$

This procedure is more similar to the classical background subtraction method, but in our method, the background image is extracted just from the frame being processed, other than a serial of history frames, if the camera is moving, classical background subtraction method cannot obtain a valid background image, while our single frame background subtraction method is competent in this case.

2.3. Coarse Segmentation

The following stage is to segment the difference image by a threshold T , which is used to classify all pixels in $d(x, y)$ into object pixels (labeled with 1) and background pixels (labeled with 0):

$$M(x, y) = \begin{cases} 1 & d(x, y) > T \\ 0 & else \end{cases} \tag{2}$$

This threshold T is more important since lower threshold will produce more false positive, and objects are easy to be lost with higher threshold, and it is known that a fixed threshold can't adapt the variance the image illumination, so here we select an adaptive threshold by modeling all the pixels in the difference image as normal distribution $N(\mu, \sigma^2)$, where μ and σ are mean and deviation of the difference image, and

set the threshold as:

$$T = \mu + \alpha\sigma \tag{3}$$

where α is a tunable coefficient, low false alarm rate will be obtained with higher value, but the rate of being lost will increase. In our experiment, $\alpha = 2.5$ is a suitable value.

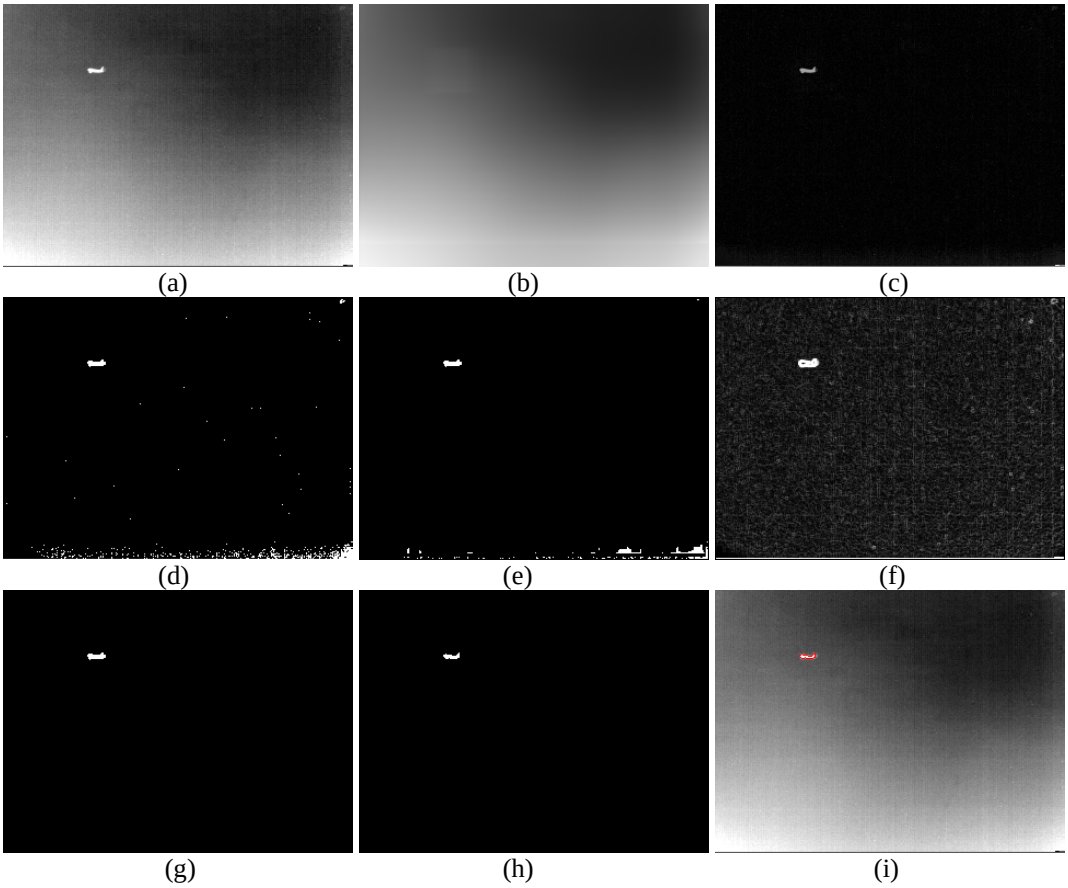

Fig.2 (a) original image; (b) background image obtained with median filter; (c) difference image of (a) and (c); (d)coarse segmentation image; (e) filter result of (d) with morphologic filter and isolate points removing;(f) gradient image of (a) with Sobel operation, (g) the result of removing falseblobs from (e); (h) fine segmentation result; (i) final detection result by marking object's contour with chain-code contour following.

2.4. False Blobs Elimination

A morphologic close operation is then applied in the coarse segmentation image (Fig.2(d)) to fill the holes inside each blob and connect adjacent regions to a integral blob, and isolated noise points are also removed here, the result is shown in Fig.2(e). Although most of noise blobs are removed, there still exist some false positive blobs. If we neglect them, the false alarm rate will be high and the computation complex of following steps will increase, so we have to remove them as early as possible. It is known that there often exist high gradient values in the boundary of objects, but low gradient value in the background. So we first obtain a gradient image by using Sobel operation (Fig.2(f)), then calculate the average gradient value $\overline{g}$ of the boundary pixels of each candidate blob. Set a threshold T_g ($T_g = 30$ in our experiments), if $\overline{g} < T_g$, we consider the blob is a false positive blob and remove it from coarse segmentation image, the result after removing false positive blobs is shown in Fig.2(g).

2.5. Object Extraction

Now, all the blobs are extracted by connecting region analysis since only objects regions are retained. In connecting region analysis, we often use region labeling method to extract each blob. However, region labeling method needs scanning one image several times in order to label each blob with a unique mark. In this paper, we explore chain-code based object extraction method [3-4] instead. Chain-code is a very promising object descriptor which is used to represent the object contours, contour following [10] and region filling [11] operations based on chain-code are very efficient, they need less memory space and less time

cost than region labeling method, and region area, region boundary rectangle and other features of a blob can also be obtained conveniently. In Fig.3, (a) shows a 8-direction chain-code, and (b) shows an object depicted by chain-code description, an illustrative experiment of time cost comparison between region labeling method and chain-code based contour following method is given in (c) and (d), we choose a database which includes 160 frames, these are binary images which are the segmentation result of several kinds of scenes, so both the image resolution and the region number in each frame are different. One of the images from the database is shown in Fig.3(c). We apply both methods to extract all objects in each frame, namely, a label image is obtained for region labeling and several chain-code series are obtained for the latter. In the whole experiment, the two methods both got accurate result. However, their time cost is significantly different, as shown in Fig.3(d), the time cost of region labeling method is varying according to the object number and the image resolution (see the curve between frame 100 to 120, it's time cost is over 12ms), but the latter is nearly constant. Its time cost is less than 1ms for all images. In our experiment, the time cost of each frame is the average time of 100 runs. From this comparison, it is definitely demonstrated that the latter method is more efficient than the former.

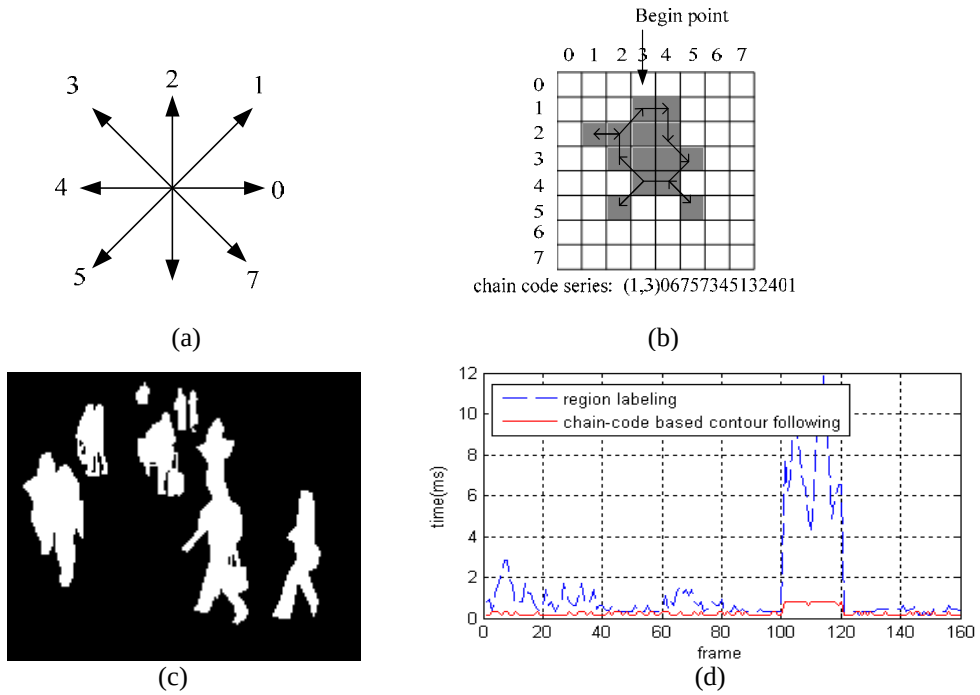

Fig.3 (a) chain-code direction. (b) an object depicted by chain-code. (c) an image from test database. (d) Average time cost curves of region labeling method and chain-code based contour following method.

2.6. Fine Segmentation

Although all object regions have been extracted now, each region only describes a coarse shape of the object since we select a global threshold in the above segmentation process. Consequently, a fine segmentation stage based on OTSU and Markov Random fields (MRFs) is applied to obtain an accurate silhouette for each object. It sufficiently considers the local spatial information of each object. The fine segmentation is applied to each target and local background independently, it is described as follows: firstly, we obtain the center, width and height of the boundary rectangle of each blob. Then we copy a corresponding rectangle region from the original image whose width and height are 3 times as the width and height of the boundary rectangle, then OTSU method [12] is applied to segment the copied region to get an initial result of fine segmentation of each object. The methodology of OTSU is maximizing the between-cluster variances, it doesn't employ the assumption of Gaussian distribution for image histograms, it just tries to partition the histogram into two uniform clutters and yield the largest distance between them, its main procedures is briefly described here:

For an L grey level image, its normalized histogram is denoted as

$$p_i = \frac{n_i}{N} \tag{4}$$

where n_i is the number of pixels in level i, $i \in \{1,2,\cdots L\}$, and $N = n_1 + n_2 + \cdots + n_L$. Then all the work is to find an optimal threshold k^* that maximize the following criterion:

$$\eta(k) = \sigma_B^2 / \sigma_T^2 \tag{5}$$

where

$$\sigma_B^2 = w_{k,0}(\mu_0 - \mu_T)^2 + w_{k,1}(\mu_1 - \mu_T)^2 \quad \sigma_T^2 = \sum_{i=1}^{L}(i - \mu_T)^2 p_i \tag{6}$$

$$w_{k,0} = \sum_{i=1}^{k} p_i \ , w_{k,1} = 1 - w_{k,0} \tag{7}$$

$$\mu_0 = \sum_{i=1}^{k} ip_i / w_0 \ , \mu_0 = \sum_{i=k+!}^{L} ip_i / w_1 \ \mu_T = \sum_{i=1}^{L} ip_i \tag{8}$$

We prefer OTSU method since it is a histogram-based segmentation method, so it is computational efficient, and what's more important is that OTSU is well-suited for the two cluster segmentation, while here each copied regions only includes one object and its local background, which is a standard two cluster segmentation problem. After OTSU segmentation, we obtain a more accurate object region than coarse segmentation result, but there still exists some disconnections. Then we use powerful effectiveness of Markov Random Fields (MRFs) based model [13] to refine a further accurate target silhouette for each object.

It is well known that the accurate target extraction problem can be posed as a binary labeling problem, namely, label each pixel i in each copied region with a binary flag $f_i \in \{0,1\}$, where $f_i = 0$ means i is a background pixel, $f_i = 1$ means i is a target pixel. Therefore, the following goal is to find a labeling set $F^* = \{f_i\}$ which can minimize the follow Gibbs energy:

$$E(F) = \sum_{i \in \Omega} E_1(f_i) + \lambda \sum_{i,j \in \Omega, i \leftrightarrow j} E_2(f_i, f_j) \tag{9}$$

Where $E_1(f_i)$ is region energy, encoding the cost when the label of pixel i as f_i, $E_2(f_i, f_j)$ is boundary energy, denoting the cost when the labels of adjacent nodes i and j are f_i and f_j respectively, and $i \leftrightarrow j$ denotes pixel i and j are adjacent nodes in eight neighbors, λ is a constant to balance the two energy items.

To model the likelihood of each pixel i belonging to target or background, a target intensity observation model and a local background intensity observation model are learned from Ω_B and Ω_T, respectively, where Ω_B corresponds to those pixels labeled as background in result of OTSU segmentation, and Ω_T corresponds to those pixels labeled as target. We assume that the image intensity obeys a Gaussian distribution. They are defined by:

$$p(g_i \mid f_i = 0) = \frac{1}{\sqrt{2\pi}\sigma_B^{\ 2}} \exp(-\frac{(g_i - \mu_B)^2}{2\sigma_B^2}) \tag{10}$$

$$p(g_i \mid f_i = 1) = \frac{1}{\sqrt{2\pi}\sigma_T^{\ 2}} \exp(-\frac{(g_i - \mu_T)^2}{2\sigma_T^2}) \tag{11}$$

where μ_B and σ_B are mean and standard deviation of the intensities of pixels in Ω_B, μ_T and σ_T are mean and standard deviation of the intensities of pixels in Ω_T .Then the two energy items are given as follows:

$$E_1(f_i) = -\log p(g_i \mid f_i) \tag{12}$$

$$E_2(f_i, f_j) = \begin{cases} 0 & f_i = f_j \\ \beta & f_i \neq f_j \end{cases} \tag{13}$$

where β is a positive constant.

In order to obtain optimal segmentation, we applied Iteration condition model (ICM) [14] to minimize the energy in (9), it can converge with several iterations.

From Fig.2(h), we can see clearly that an accurate object shape is obtained duo to the fine segmentation step. Fig.2(i) gives the final detection result, which is described by marking the object contour with red color. It is also realized by using chain-code contour following method.

Our approach has a good compatibility in dealing with different images, namely visible or infrared image, this is due to the consistency of sky background, and our algorithm can effectively remove the noise point and false target in both images.

In terms of the time cost of the whole algorithm, its complexity mainly includes background extraction and fine segmentation, while the time cost of all other steps is less than 5ms for images small than resolution 768*576 thanks to the exploration of chain-code based methods. The time cost of background extraction depends on the resolution of images and the size of filter window, and for the fine segmentation, it depends on the size and number of existing objects.

3. Experiments

In this section, our approach will be tested by using several real image sequences include both visible and infrared images, some result images and quantitative descriptions about its performance will also be given.

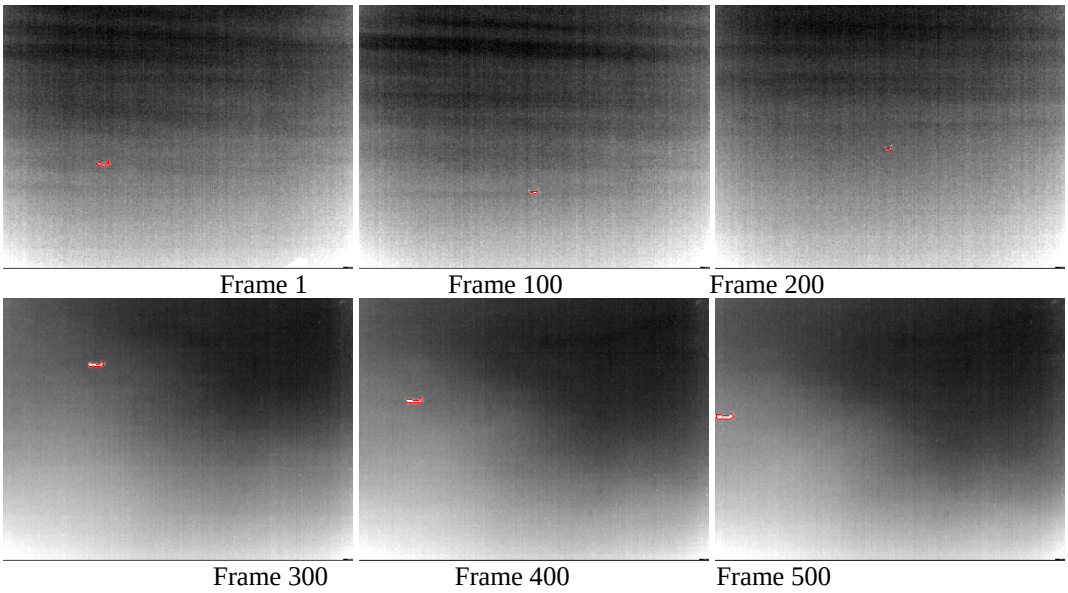

Fig.4 part of detection results of our method with the infrared sequence.

Fig.4 shows some results of our method in an infrared test sequence. The sequence is taken with a hand-hold Infrared camera, so the camera is non-static. A single plan object is included in this sequence. The image resolution is 320*240, although the images are characterized by low signal-to-noise ratios, poor object visibility and time varying object appearance, our detection approach can accurately detect the object, and with no false alarm. The object contour is also marked with red color.

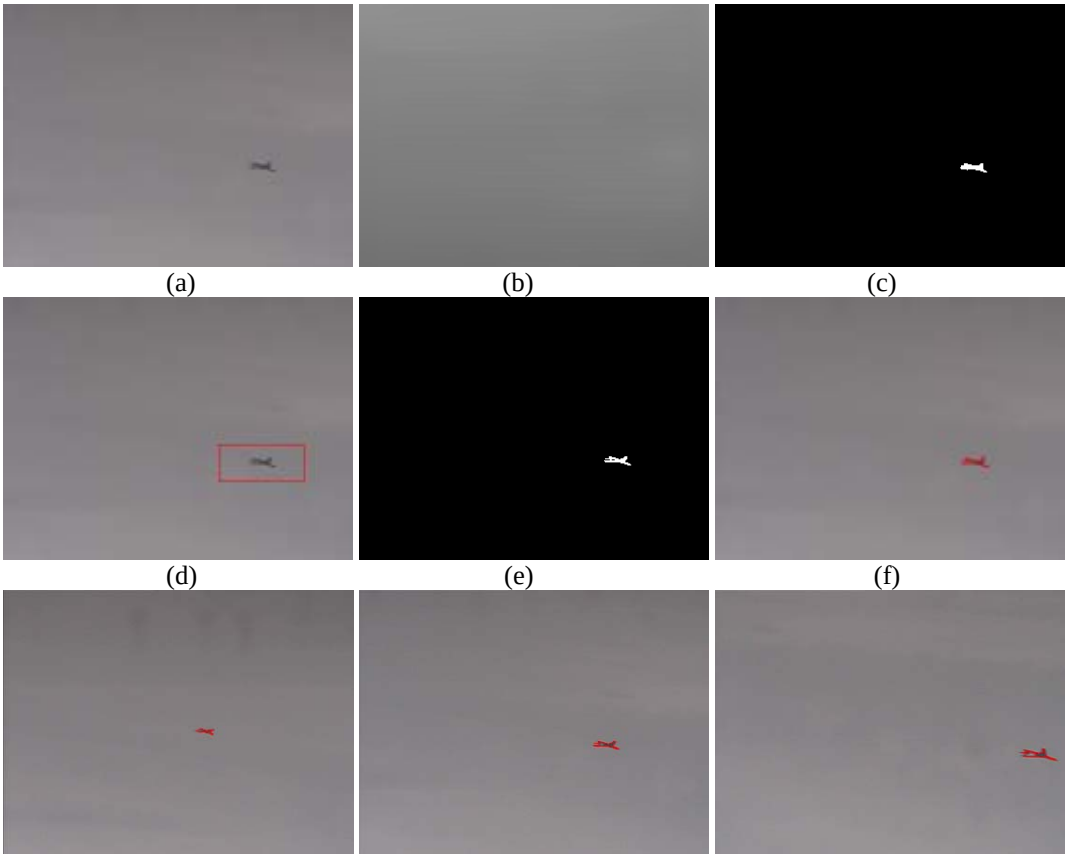

Fig.5 (a) original image; (b) background image obtained with median filter; (c) coarse segmentation image of difference between (a) and (b); (d) fine segmentation region; (e) fine segmentation result; (f) final detection result of (a) by marking object's contour with chain-code contour following. (g)~(i) other detection results of the sequence.

Fig.5 gives some results of our method in a visible-light image sequence which a dark plan object is included. Fig.5(a)~(f) show the procedure of our detection algorithm. Fig.5(g)~(i) are some detection results of the sequence.

Fig.6 gives another visible-light sequence which multiple objects are included. This sequence depicts a real scene of multiple cannonballs intercepting a missile. All objects' intensity are brighter than the background, our method also detects all objects accurately.

We also measure the speed of the proposed method, a standard PC with a 2.0 GHz AMD processor and 512 MB of memory is used in our experiments. The image resolution and average time cost for single frame is given in Table1, where the three contexts marked with "Single object in Infrared image", "Multiple objects in visible image", and "Single object in visible image" correspond to the sequences depicted in Fig.4, Fig.5 and Fig.6 respectively. We can see that the proposed algorithm is efficient. This makes the method well-suited to systems that require real-time processing.

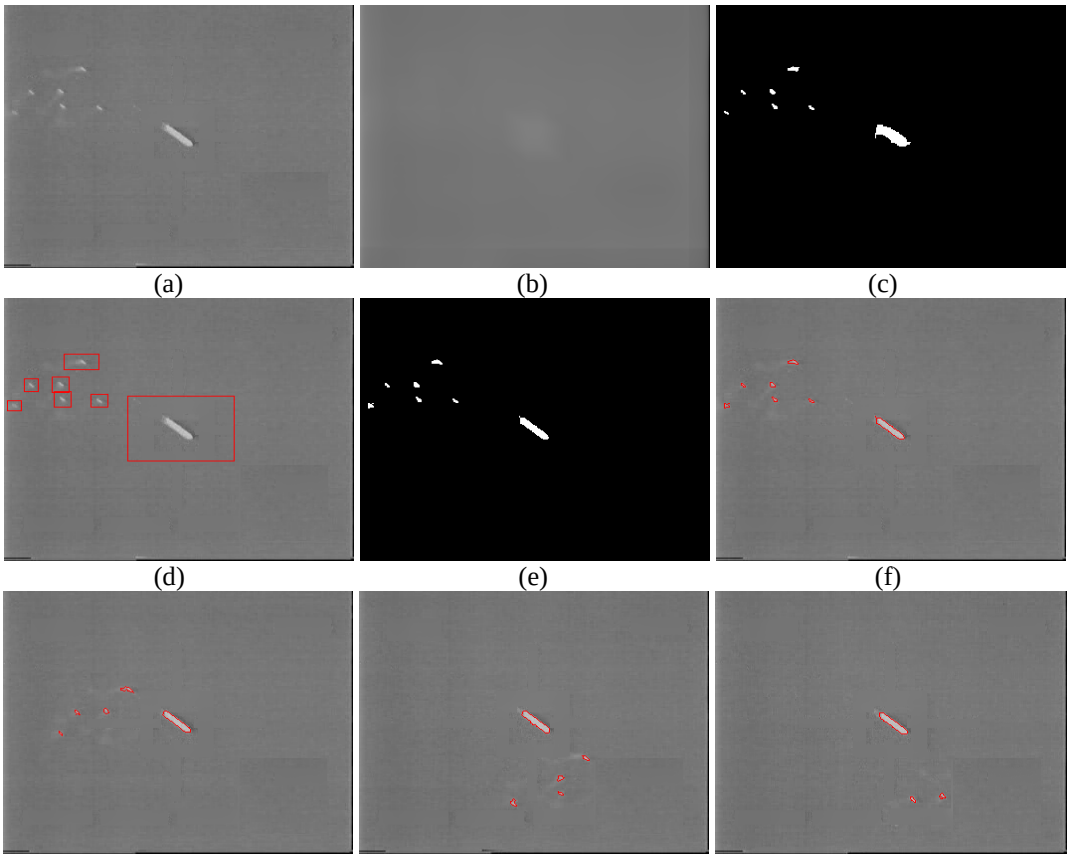

Fig.6 (a) original image; (b) background image obtained with median filter; (c) coarse segmentation image of difference between (a) and (b); (d) fine segmentation region; (e) fine segmentation result; (f) final detection result of (a) by marking object's contour with chain-code contour following. (g)~(i) other detection results of the sequence.

Table 2 Average processing time evaluated for each frame in each sequence.

Context	Quantity	Average time(ms)
Single object in Infrared image	320*240	38.9
Multiple objects in visible image	352*288	40.6
Single object in visible image	384*288	35.4

4. Conclusions

A novel vision-based approach for air object detection is presented in this paper, it is fast and insensitive to camera motions and sudden illumination changes. The method has been evaluated against several video sequences including both visible-light and infrared image, all experiments demonstrate the effectiveness and efficiency of the proposed algorithm. Currently, this method has been used in movable surveillance system successfully.

5. Reference

[1] A.Talukder, L.Matthies. Real-time detection of moving object from moving vehicles using dense stereo and optical flow. *IEEE Conference on Intelligent Robots and Systems.* 2004.

[2] L.Wixson. Detecting Sailient Motion by Accumulating Directionally Consistent flow. *IEEE Trans. Pattern Analysis and Machine Intelligence.* 2000, **22**(8): 774-780.

[3] R.T.Collins, A.J.Lipton, T.Kanade, et al.. *A system for video surveillance and monitoring.* Technical Report, Carnegie Mellon University 2000.

[4] R.Cucchiara, M. Piccardi. Vehicle detection under day and night illumination. *Proceedings of the Third*

International Symposia on intelligent industrial Automation and Soft Computing. 1999.

[5] C.Stauffer and W.E.L.Grimson. Adaptive background mixture models for real-time tracking. *Proc. IEEE Conf. Computer Vision and Pattern Recognition.* 1999, **2**: 246-252.

[6] M.Heikkila and M.Pietikainen. A texture-based method for modeling the background and detecting moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 2006, **28**: 657-662.

[7] D.Crandall, J.Luo. Robust color object detection using spatial-color joint propbbility function. *Proceedings of CVPR.* 2004, pp. 379-385.

[8] J.Weng, J,Ahuja, N,Huang. Matching two perspective views. *PAMI.* 1992, **14**(8): 806-825.

[9] M. Sonka, V. Hlavac, R. Boyle. *Image processing, Analysis, and Machine vision.* Thomson Learning and PT Prress. 1999, pp. 50-51.

[10] M.W.Ren, J.Y.Yang and H.Sun. Tracing boundary contours in a binary image. *Image and Vision Computing.* 2002, **30**(2): 125-131

[11] M.W.Ren, W.K.Yang and J.Y.Yang. A new and fast contour-filling algorithm. *Pattern Recognition.* 2005, **38**(12): 2564-2577

[12] N.Otsu. A threshold selection method from gray-level histograms. IEEE Trans. *Syst.Man Cybern*. 1979, **SMC-9**: 62-66.

[13] Huang S S, Fu L C, Hsiao P V. Region-Level Motion-Based Background Modeling and Subtraction Using MRFs. *IEEE Transaction on Image Processing*. 2007, **16**(5): 1446-1456.