The Research of Moving Object Tracking in Video Images based on ARM

Zicong LI[1,a], Fensu SHI[1,b],Qin LI[1],Xiao LIU[1]

[1]School of Computer Science and Engineering, Beifang University of Nationalities, Yinchuan, Ningxia, 750021, China

[a]email:395729604@qq.com,[b]email:shifensu@163.com

Keywords: ARM; Moving Object Tracking; Camshift; Three-frame differencing method

Abstract. With the rapid development of ARM processor, the computer vision systems begin to shift from the PC platform to the ARM platform. Moving object tracking is the key technology of the intelligent video surveillance, robot navigation, and other areas of precision-guided in the computer vision filed. This paper focuses on the principles of traditional Camshift algorithm and put forward an improved algorithm that combined with the three-frame differencing method and the Camshift algorithm. This paper also introduced how to implement the moving object tracking system on ARM platforms. The experiment on two videos indicates that compared to the traditional Camshift the proposed algorithm is more effective and feasible.

Introduction

The function of moving object tracking is to establish the pairing relationship between the continuous monitoring of moving objects in video scenes and the time-line, complete the marking of the object in space, get the position, trajectory, velocity, acceleration, etc. of the moving object in the scene at different times and according to the obtained information to analyze the object's behavior understanding and so on.[1] There's various of methods of the moving object tracking like tracking based on feature, tracking based on area, tracking based on model and tracking based on active contour. In order to adapt to different scenes, operation principles that use of feature information and parameters model of each method are different. This paper mainly describes the moving object tracking algorithm based on Camshift algorithm. It uses color feature of the moving object in the video image to find the object's position and size.

Traditional video surveillance system based on computer vision usually upload the video images captured by camera to the PC, and then use the PC to do the image processing, moving object tracking and follow-up processing tasks. In recent years, because of the rapid development of ARM processors, more and more computer vision systems are used by embedded devices. Besides, embedded devices that based on ARM processor because of its high performance and low power consumption, it is widely used in various fields. Therefore, this paper simply describes how to implement moving object tracking system on the ARM platforms.

Camshift algorithm

Camshift algorithm proposed by Bradski is a kind of moving object tracking algorithm that use the color histogram as the target mode and use the color information in the region to track the object. The algorithm is an evolution of MeanShift algorithm and it is short for Continuously Adaptive MeanShift. Its basic idea is to make all the frames of the video images do MeanShift operation, and make the results of the previous frame as the Search Window's initial value of the MeanShift algorithm for the next frame, and iterate the above processes.

MeanShift algorithm.

MeanShift is a kind of gradient ascent method that optimally find the maxima of the probability density. The moving object tracking technology based on MeanShift use the kernel function probability density to describe the characteristics of the target, and then use the MeanShift to search the target's location [4]. This method is stable, and able to adapt to the deformation and the continuously dimensional changes of the target. Besides, the algorithm's calculation is fast and its

anti-interference ability is strong so that the method can ensure the system's real-time performance and stability.

Given n sample points $x_i(i=1,...,n)$, $x_i \in R^d$, R^d is a d-dimensional space, MeanShift vector point x_i is defined as:

$$M_h(x) = \frac{1}{k}\sum_{x_i \epsilon S_h}(x_i - x) \quad (1)$$

Wherein, S_h is a high-latitude spherical region with its radius is h, the collection of point y satisfies with the equation (2):

$$S_h(x) = \{y: (y-x)^T(y-x) \leq h^2\} \quad (2)$$

k is the quantity of sample points x_i which fall into the area S_h, $k \leqslant n$

Assume that the sample point x_i is random sampling obtained by the probability density function $f(x)$. When the probability density is not zero, its gradient points to the maximum increment direction of the probability density. So in the area S_h, there should be more sample points fall on the gradient direction of the probability density. As can be seen from Fig. 1, the average offset $M_h(x)$ points to the sample area where the point distribution is the most intensive. That is the gradient direction of the probability density function.

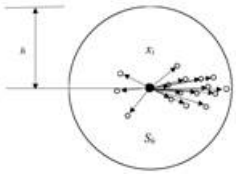


Fig. 1:MeanShift Schematic

As can be seen from equation (1), by the basic Meanshift, as long as the sample points fall into the area S_h, regardless of the distance between the x_i and the center point x, their contribution to the final calculation of $M_h(x)$ is the same. However, in the actual tracking process, when the tracking target is partial covered, since the pixel values of the outer layer is easily effected by the cover or background, so the pixels near the center of the target model is more reliable than the outer pixels. Therefore, Cheng introduced the concept of kernel function and weight coefficient. So the final form of Meanshift algorithm is:

$$M_h(x) = m_h(x) - x \quad (3)$$

Wherein, $G(x)$ is the kernel function, $\omega(i)$ is the weight, and

$$m_h(x) = \frac{\sum_{i=1}^{n} G(\frac{x_i - x}{h})\omega(x_i)x_i}{\sum_{i=1}^{n} G(\frac{x_i - x}{h})\omega(x_i)} \quad (4)$$

Specific steps of Meanshift algorithm is given an initial point, kernel function, threshold, Meanshift algorithm cyclically perform the following steps until conditions are satisfied:

1) Calculation the value of $M_h(x)$;
2) If $\| M_h(x) < x \|$, end of cycle, continue the else steps;
3) Assign the calculated value of $M_h(x)$ to x, proceed Step 1).

Three-frame differencing method.

The basic idea of the three-frame differencing method is to put the adjacent three frames as a group and every two adjacent frame of the group do differential operation, then the results of two differential do "and" operation. The algorithm can obtain the area of the target more accurately when the speed of the target or the frame rate is moderate. The algorithm is described as follows:

Assumed that the image sequence is represented as: $\{f_0(x,y), \dots, f_k(x,y), \dots, f_{n-1}(x,y)\}$, wherein $f_k(x,y)$ represents the k^{th} frame image of a video sequence. Select three successive frames of the video $f_{k-1}(x,y)$, $f_k(x,y)$, $f_{k+1}(x,y)$, calculate the difference of the two adjacent frames:

$$d_{(k-1,k)}(x,y) = |f_k(x,y) - f_{k-1}(x,y)| \quad (5)$$

$$d_{(k,k+1)}(x,y) = |f_{k+1}(x,y) - f_k(x,y)| \quad (6)$$

For two of the obtained difference image, setting an appropriate threshold T can obtain the binarized image $b_{(k-1,k)}(x,y)$, $b_{(k,k+1)}(x,y)$.

For each pixel (x,y), make the above two binary images do "and" operation to obtain the binary

image $B_{(k-1,k)}(x,y)$.

$$B_{(k-1,k)}(x,y) = b_{(k-1,k)}(x,y) \otimes b_{(k.k+1)}(x,y) \quad (7)$$

In order to remove the isolated error points in the image, morphological filtering process for the binary image is needed.

$$B_{(k-1,k)}(x,y) = Close(Open(B_{(k-1,k)}(x,y))) \quad (8)$$

Wherein, $Open(\cdot)$ is opening operation, $Close(\cdot)$ is closing operation.

This paper use the Otsu method to select the threshold T. The method adaptively determine the threshold by the variance within clusters between background and target. The principle is as follows:

For the image $I(x,y)$, the segmentation between the target and the background is denoted as T, the proportion about the pixels belonging to the target of the whole image is denoted as ω_0, and its average gray value is μ_1. The total average gray value of the image is μ, the variance within clusters is g. Assumed that the image size denote as $M \times N$, the number of the pixels that its gray value is smaller than the threshold T is N_0, the number of the pixels that its gray value is larger than the threshold T is N_1. Then:

$$\omega_0 = \frac{N_0}{M \times N}, \omega_1 = \frac{N_1}{M \times N}, N_0 + N_1 = M \times N, \omega_0 + \omega_1 = 1$$

$$\mu = \omega_0 \times \mu_0 + \omega_1 \times \mu_1, g = \omega_0(\mu_0 - \mu)^2 + \omega_1(\mu_1 - \mu)^2$$

According to the above equation to obtain the equivalent formula:

$$g = \omega_1\omega_0(\mu_0 - \mu_1)^2 \quad (9)$$

Using ergodic method to get the largest g, which is the threshold T.

Basic Camshift algorithm.

The core idea of Camshift algorithm is to use the color feature to find the target's position and size in the video image. In each frame of the video image, use the current position and size to initialize the search window, compute centroid within the window by the color probability distribution, and then move the center of search window to the centroid. After a few iteration, when the distance between center point and the centroid point is less than a certain threshold that means matching the target [5] [6]. Camshift algorithm mainly includes several of the following steps:

1) Histogram back projection. In the HSV color model, extract an image's H component, and then establish the H component color histogram of the target in the initial search window of the moving target. After the back projection, obtain the discrete $2D$ color probability distribution based on the original image of the target.
2) MeanShift algorithm processing. Use the MeanShift to match the centroid point in the color probability distribution, after several iterations to calculate the coordinates of the target's centroid.
3) Adaptively adjust the search window. Depending on the color probability distribution, calculate the size and centroid location of the target.

Set point (u,v) as the pixel coordinate in the search window. $l(u,v)$ is the corresponding gray value of the the point (u,v) in the projection. Respectively define the 0-order matrix M_{00} and the 1-order matrix M_{01} and M_{10} of the search window are as follows:

$$M_{00} = \sum_u \sum_v I(u,v),\ M_{10} = \sum_u \sum_v u \cdot I(u,v),\ M_{01} = \sum_u \sum_v v \cdot I(u,v)$$

So the centroid position of current search window is:

$$(u_c, v_c) = \left[\frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}}\right] \quad (12)$$

Resize the search window, and move the center of search window to the centroid until the distance between the centroid and the center of the window is smaller than the preset threshold or the number of operations reaches a preset value. That is considered to meet the convergence conditions. Then proceed new target search in the next frame. Comparing to the current frame, the length l and width w of search window in next frame are to updated as follows:

$w = \sqrt{\frac{(a+c)-\sqrt{b^2+(a-c)^2}}{2}}$, $l = \sqrt{\frac{(a+c)+\sqrt{b^2+(a-c)^2}}{2}}$

Wherein,

$a = \frac{M_{20}}{M_{00}} - u_c^2, b = 2\left(\frac{M_{11}}{M_{00}} - u_c v_c\right), c = \frac{M_{02}}{M_{00}} - v_c^2$

$M_{11} = \sum_u \sum_v uv \cdot I(u,v)$, $M_{02} = \sum_u \sum_v v^2 \cdot I(u,v)$, $M_{20} = \sum_u \sum_v u^2 \cdot I(u,v)$

Improved Camshift algorithm

Traditional Camshift algorithm has the following disadvantages:

At the beginning of target tracking, traditional Camshift algorithm needs to manually select the initial target and tracking window, and needs to be human-intervened. So it is low intelligence.

Because of Camshift algorithm uses the color probability histogram to determine and track the target, this algorithm is mainly applied to track the target that its color is obviously different with the background color. While the target's color and background color are same or similar, the traditional Camshift algorithm will be severely interfered, and even lead to loss of tracking.

This paper effectively solve the problems above. Algorithm flowchart is shown in Fig. 2.

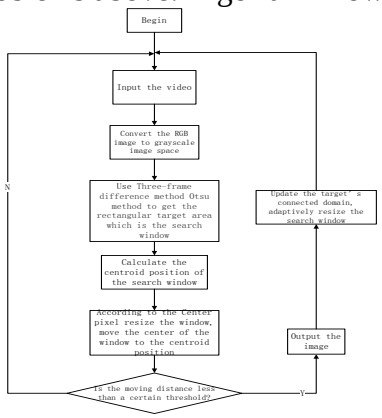


Fig.2: Improved algorithm flowchart

The implementation steps of the improved algorithm are as follows:

1) Input the video and convert the RGB image to grayscale image space.
2) Use the three-frame differencing method to detect the target area, and a rectangular target area will be used as the search window.
3) Calculate *0*-order matrix M_{00}, *1*-order matrix M_{10}, M_{01}, *2*-order matrix M_{11}.
4) Calculate the centroid of the search window (u_c, v_c), and use it as the center point of the search window of the next frame.
5) According to M_{00} resize the search window.
6) Move the search window center to the centroid position.
7) If the moving distance less than the threshold, go to step 9, else go to step 3.
8) Output matching results, update the target area connectivity, and adaptively adjust the window size.

Implement the algorithm on ARM

Build the cross compiling environment.

Cross compiling can generate executable code for another platform on one platform. Because of the limited static storage space on ARM processor, can not achieve the native compilation. So the cross-compiler is needed for getting the executable code which could run on the ARM processor. This paper uses arm-linux-gcc-4.3.2 for the cross compiler tools, specific steps of building the cross compiling environment are as follows:

1) To establish the installation directory of the cross compiler named arm in directory /usr/local/ .
2) Extract the downloaded cross-compiler package to the directory /usr/local/arm.
3) After extracting successfully, modify the PATH environment variable and add the path of the

cross-compiler at the end of the file profile.

4) Update the profile.

The transplant of the computer vision library OpenCV.

In this paper, the tracking system is implemented based on computer vision library OpenCV2. OpenCV (Open Source Computer Vision Library) is a cross-platform computer vision library and released of open source. OpenCV provides very rich of vision processing algorithms including over 500 functions for digital image processing and the common algorithms of computer vision. It is part of written by the C language, and it is open source. If handled properly, no need to add new external support can also complete the compilation and link to generate the executable program. So that it is easy to make the algorithm transplant. The steps of OpenCV transplant are as follows:

1) Download file OpenCV-2.0.0.tar.bz2, and extract it to the current directory.
2) Enter into the folder OpenCV-2.0.0, configure the options.
3) Compile the library files.
4) Install library files.
5) Transplant the library to the development board based on ARM.

3 The compiling on the Linux

The transplant in this paper is under the Linux and before that the source code need to be compiled, then copy the executable program to the ARM. Source code is compiled as follows:

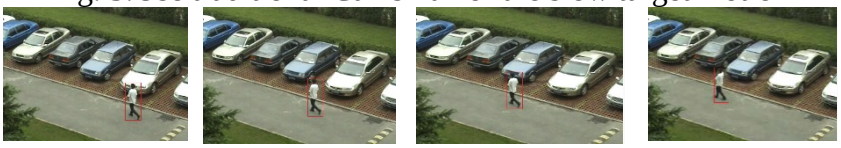
#arm-linux-gcc Test.c -o Test `pkg-config --cflags --libs opencv`

Analysis of experiment

In this paper, the experimental platform is S3C6410 which is based on ARM11. In the situations that the target motion is slow and the target motion is relatively fast, we separately use the traditional Camshift algorithm and the improved algorithm in this paper for target tracking. The result is shown below. Figure 4-5 is the actual target trajectory and the trajectories obtained by algorithms.

The 1st frame The 21st frame The 41st frame The 61st frame

Fig. 3:Use traditional Camshitf for the slow target motion



The 1st frame The 21st frame The 41st frame The 61st frame

Fig. 4:Use improved algorithm for the slow target motion

The 1st frame The 11st frame The 21st frame The 31st frame

Fig. 5:Use traditional Camshitf for the fast target motion

The 1st frame The 11st frame The 21st frame The 31st frame

Fig. 6:Use improved algorithm for the fast target motion

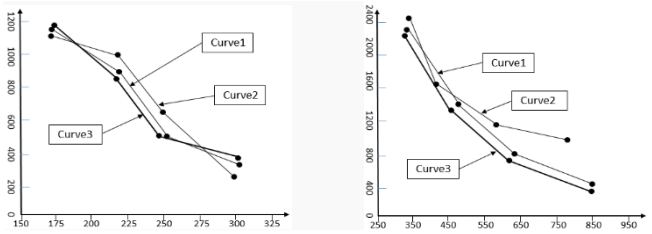


Fig. 7:Target trajectories (a)The target is slow;(b)The target is fast

As can be seen from Fig. 3 and Fig. 4, in the situation that the target is slow, the traditional Camshift algorithm can track the target at the 1st frame, but slight tracking divergence happened from 21st to 61st frame. Compared to traditional Camshift, this paper can accurately track the target during the whole process and have no divergence, the result of tracking is almost consistent with the actual trajectory of the target. To further validate the algorithm, as can be seen from Fig. 5 and Fig. 6, in the situation that the target is relatively fast, significant deviation and divergence happened at 1st frame by the traditional Camshift, and with the fast-moving of target, the situation is more serious. Compared to traditional Camshift, just very slight divergence deviation happened and still can accurately track the moving target by the improved algorithm.

Trajectories are shown in Fig. 7(a), from which curve 1 is the actual trajectory of the target, curve 2 and curve 3 respectively is the trajectory of the target obtained by the traditional Camshift and the improved algorithm. The same as Fig. 7(b). Can be concluded form Fig.7 that compared to traditional Camshift, this paper can accurately track the target, especially when the target moves relatively fast, the advantage is more obvious. In the situation that the target is slow, the traditional Camshift is acceptable, but when the target is moving fast, the deviation is big.

In order to reflect the advantages of this algorithm, this paper collected 1,000 consecutive frames as the sample, and respectively use the two algorithms for target tracking. Then calculated the average error, accuracy, arithmetic computation time. The average error is the distance between the track position and the actual position of the target, accuracy is the accuracy of target tracking, arithmetic computation time is the time cost to process each frame. The results are shown in Table 1.

Table 1: Comparison of two algorithms

algorithm	frame number	average error（pixel）	accuracy	computation time（s/frame）
Camshift（slow）	1000	50.889	89.47%	0.097
This aper（slow）	1000	25.478	95.94%	0.076
Camshift（fast)	1000	200.256	60.54%	0.097
This paper(fast)	1000	40.774	92.32%	0.076

As can be seen from Table 1, in the situation of the target motion is slow, the average error of this paper is 25.478 pixels, compared to traditional Camshift which is 50.889 pixels, this paper has a progress to some extent. In the situation of the target motion is fast, the average error of this paper is 40.774 pixels, compared to traditional Camshift which is 200.256 pixels, this paper has a significant progress; By calculation, the accuracy of traditional Camshift respectively is 89.47% and 60.54% in the two situations, this paper respectively has reached to 95.94% and 92.32%; For traditional Camshift, computation time for each frame is 0.097s, while this paper is 0.076s. Based on analysis of above, this paper has the best overall results, especially when the target moves fast, the advantages are obvious.

Concluding remarks

This paper proposed an improved algorithm for target tracking based on Camshift. Firstly, uses the three-frame differencing method for target detecting that overcomes the shortcoming of the traditional Camshift algorithm which needs to manually select the initial target and tracking window, and needs to be human-intervened. Secondly, uses the three-frame differencing method to determine the tracking area during the tracking process that overcomes divergence which easily happened in

the traditional Camshift. Then this paper give a brief introduction how to transplant the algorithm to the ARM platform that provides an effective way of using the computer vision system on the embedded platform.

Acknowledgment

This work is supported by National Natural Science Foundation of China (61162013, 61461003), the graduate innovation project of 2014 Beifang University of Nationalities and a project supported by National Natural Science Foundation pre-breeding program of Beifang University of Nationalities (No.2012QZP01).

References

[1] LI Zhi, XIE Qiang. Moving Target Tracking Based on Improved Particle Filter [J]. Beijing:Computer Science, 2014, 41(2):232-233

[2] LI Wenshu, ZHAO Yue. Digital Image Processing Algorithms and Applications [M]. Beijing:Peking University Press, 2012:239-244

[3] LIU Chao, HUI Jing. Object tracking algorithm based on improved CAMShift [J]. North China Institute of Computing Technology:Computer Engineering and Applications, 2014,50(11):149

[4] Lin Jiayi. A New Moving Detection Method Based on Background Differencing and Coterminous Frames Differencing [J]. Chinese Journal of Science Instrument, 2008, 29(4):112

[5] XU Xiaofeng. Design and Implementation of mobile intelligent video surveillance system based on Embedded and OpenCV [D]. The thesis of master degree of Taiyuan University of Science & Technology, 2014: 27-28

[6] YANG Jie. An Algorithm Research on Improved Camshift [J]. Computer Application and Software, 2014,31(2):169-170