Real Time Tracking Using Adaptive Window Techniques

{W.M. Yousf, O.M. Elmowafy, A.E.M. Fahmy, I.A. Abdl-Dayem}*

Abstract: Target tracking is an important field that attracted a lot of interest in computer vision. This paper describes an end-to-end technique for extracting moving targets from a real-time video stream and real sequence of live camera when the appearance of the object changes. Targets are classified into predefined categories according to image based properties, and then robustly tracking them. However, since the scale of the targets often varied irregularly, systems with fixed-size tracking window usually could not accommodate to these scenarios. In present paper, a modified approach of tracking algorithm with Self-Updating Tracking was introduced. The proposed algorithm divided into two stages, the first stage depending on detecting the sharp corner of moving target based on Lucas-kanade techniques. The second stage detecting the color of target being detected in the previous stage and able to track the detected color belonged to moving target with adaptive window size. Experimental results demonstrated that the improved algorithms could select the proper size of tracking window not only when the object scale increases but the scale decreases as well with minor extra computational overhead.

Keywords: Tracking, Self-Updating, Lucas-kanade techniques.

1. Introduction
Tracking is an important and challenging task within the field of computer science. It can be applied in a great number of different applications from various fields. Available techniques for object tracking use different approach such as background subtraction or modeling, particle and Kalman filtering, segmentation, supervised learning, etc. [1]. Many existing algorithms segment each video frame to determine the objects; this action can be computationally expensive, and it is not necessary if the goal is to determine the moving objects [2]-[3].

Alternatively, the previous algorithms concerned to corner detection techniques is used to establish the optical flow motion, several methods used a modified algorithms for face detection based on the Haar-classifier. Face detection is fast and robust to illumination condition but hardly work when face pose is too slanted. Shi and Thomasi techniques are used to extract feature points. These methods give good results. To track the facial feature points, Pyramidal Lucas-Kanade Feature Tracker (KLT) algorithm is used hence; optical flow is a useful tool for tracking objects in motion, [4]-[5].

Developing new algorithms for automatically tracking detected targets become necessary. In the case of moving target captured by camera the challenge become more important.

* Egyptian Armed Forces, Egypt
Therefore, we present a rule-based method to deal with these hard situations. The proposed tracking algorithm uses the first stage depending on Lucas-kanade algorithm which is the most popular method of tracking movement, using Shi and Tomasi technique to choose which features are “good” to track, in the second stage the continuously adaptive mean shift (CamShift) algorithm based on the color of target being detected in the previous stage using the first two eigenvalues, that is, length and width, depending on; the zero, first and second moment of image are generated to create the adaptive track window.

The rest of the paper is organized as follows: Section (2), we will review some necessary background on the optical flow; we will review the corner detector. Also we will review CamShift algorithm uses the multi-scale information for self tracking. In Section (3), we introduce the proposed tracking algorithm. In Section (4), the experimental results are shown. In Section (5), the conclusions are given. In Section (6) we will refer to the important references in this field of study.

2. Background

2.1. Optical Flow

This section describes several functions for calculating optical flow between two images. Most papers devoted to motion estimation use the term optical flow. Optical flow is defined as an apparent motion of image brightness. Let $I(x, y, t)$ be the image brightness that changes in time to provide an image sequence. Two main assumptions can be made:

1. Brightness $I(x, y, t)$ smoothly depends on coordinates $x, y$ in greater part of the image.
2. Brightness of every point of a moving or static object does not change in time.

Let some object in the image, or some point of an object, move and after time $dt$ the object displacement is $(dx, dy)$. Using Taylor series for brightness $I(x, y, t)$ gives the following:

$$ I(x + dx, y + dy, t + dt) = I(x, y, t) + \frac{\partial I}{\partial x} dx + \frac{\partial I}{\partial y} dy + \frac{\partial I}{\partial t} dt + ... , $$

(1)

According to assumption 2:

$$ I(x + dx, y + dy, t + dt) = I(x, y, t) $$

(2)

And

$$ \frac{\partial I}{\partial x} dx + \frac{\partial I}{\partial y} dy + \frac{\partial I}{\partial t} dt + ... = 0 $$

(3)

From the previous equations we obtain:

$$ -\frac{\partial I}{\partial t} = \frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v $$

(4)

Usually called optical flow constraint equation, where $u$ and $v$ are components of optical flow field in $x$ and $y$ coordinates respectively.
2.2. Lucas-Kanade
The Lucas-kanade algorithm is easily the most popular method of tracking movement. The LK tracker uses three assumptions, brightness constancy between the same pixels from one frame to the next, small movements between frames (requiring image pyramids to track larger movements), and spatial coherence, that’s points near each other are on the same surface. Then the basic concept of the tracker is to estimate the velocity of a moving pixel by the ratio of the derivative of the intensity over time divided by the derivative of the intensity over space.

2.3. Haar Classifier
The power of the Haar Classifier is that it will quickly reject regions that are highly unlikely to contain the object. It does this by making use of the cascade of classifiers. In this cascade, the early stages will quickly reject the majority of false regions and the object detection can move on to other regions. The later stages however require progressively more computational effort in order to reject the region.

2.4. CamShift Algorithm
1. Set the calculation region of the probability distribution to the whole image.
2. Choose the initial location of the 2D mean shift search window.
3. Calculate the color probability distribution in the 2D region centered at the search window location in the region of interest (ROI) slightly larger than the mean shift window size.
4. Run Mean Shift algorithm to find the search window center. Store the zeroth moment (area or size) and center location.
5. For the next video frame, center the search window at the mean location stored in Step 4 and set the window size to a function of the zeroth moment found there. Go to Step 3.

2.5. Mass Center Calculation for 2D Probability Distribution
For discrete 2D image probability distributions [6], the mean location (the centroid) within the search window that is computed at step 3 above is found as follows:
Find the zeroth moment
\[
M_{00} = \sum_x \sum_y I(x, y) \tag{5}
\]
Find the first moment for x and y
\[
M_{10} = \sum_x \sum_y xI(x, y) ; \quad M_{01} = \sum_x \sum_y yI(x, y) \tag{6}
\]
Mean search window location (the centroid) then is found as
\[
x_c = \frac{M_{10}}{M_{00}} ; \quad y_c = \frac{M_{01}}{M_{00}} \tag{7}
\]
where \( I(x, y) \) is the pixel (probability) value in the position \( (x, y) \) in the image, and \( x \) and \( y \) range over the search window.
Our proposed tracking algorithm relay on the zeroth moment information, extracted as part of the internal workings of the algorithm, to continuously adapt its window size within or over each video frame.
Second moments are

\[ M_{20} = \sum_x \sum_y x^2 I(x, y) \quad ; \quad M_{02} = \sum_x \sum_y y^2 I(x, y) \] (8)

The first two eigenvalues, that is, length and width, of the probability distribution of the blob found by tracking algorithm may be calculated in closed form as follows:

Let

\[ a = \frac{M_{20} - x_c^2}{M_{00}}, \quad b = \frac{M_{11} - x_c y_c}{M_{00}}, \quad c = \frac{M_{02} - y_c^2}{M_{00}} \] (9)

Then length \( l \) and width \( w \) from the distribution centroid are

\[ l = \sqrt[2]{\frac{(a + c) + \sqrt{b^2 + (a - c)^2}}{2}} \] (10)

\[ w = \sqrt[2]{\frac{(a + c) - \sqrt{b^2 + (a - c)^2}}{2}} \] (11)

3. Tracking Model

The typical tracking paradigm comprises a repeating cycle of measurement process and adaptation as shown in figure (1). The measurement process, in turn, invokes both 1st stage represented in target detection using corner detection based on the Haar-like features then we applied Pyramidal Lucas-Kanade Feature Tracker (KLT) algorithm to track the features (points) in an image across multiple frames. In the 2nd stage CamShift algorithm based on target color was applied and the mass Center Calculation for 2D Probability Distribution to estimate the center of mass of the tracked target within fixed window, current size and location of the tracked object are reported and used to set the size and location of the search window in the next video image. The process is then repeated for continuous tracking.

4. Experimental Work

We have applied the proposed method to different sequence of real video capture image. The proposed method was developed using Open Source Computer Vision Library (OpenCV) under Microsoft visual studio 2008, running on Windows XP on a Pentium 4 platform and current implementation of the algorithm is capable of tracking target at 25 frames per second. The tracking results of the algorithm were visually confirmed.

Figure 2 shows a set of selected frames from a video capture sequence of moving tank where there is a high speed and variation in size due to its farness. The original image is treated with smoothing filter to reduce the noises associated with video image then Haar Classifier detector is applied to detect the sharp corner of the moving target, CamShift algorithm is
Fig. 1  Functional block diagram of tracking system
Fig. 2 Target tracking result for a sequence; (a) first frame from the 1st stage of target detection, (b) frame No 51 from the 2nd stage with desired target to be tracked, (c) Frame No 90 (d) frame No 276, (e) frame No 450 , (f) frame 680 showing the variation in the window size relative to the target size.

applied to track the feature color around the corner detected in an image across multiple frames, adaptive window is created depending on the mass Center Calculation for 2D Probability Distribution to suit with the target size through its flight. Target, is detected and enclosed with adaptive size window. The target tracker correctly tracked the target.

Figure 3 shows another set of selected frames from a video capture sequence of airplane where there is a high speed and variation in size due to its farness. The original image is treated with smoothing filter to reduce the noises associated with video image then Haar Classifier detector is applied to detect the sharp corner of the moving target, CamShift algorithm is applied to track the feature color around the corner detected in an image across multiple frames, adaptive window is created depending on the mass Center Calculation for 2D Probability Distribution to suit with the target size through its flight . Target, is detected and enclosed with adaptive size window. The target tracker correctly tracked the target.
5. Conclusions

In this paper, we have presented an adaptive window target tracking algorithm in different image sequence of real video capture, in order to use it in the target variation size recognition system. To detect the desired moving target in the image, we have used a modified detector based on the Haar-like features. This detector is fast and robust to illumination condition. For feature point’s extraction, we have used the algorithm of Shi and Thomasi to extract feature points. This method gives good results. To track the target feature points, Pyramidal Lucas-Kanade Feature Tracker KLT algorithm is used. We have not got a good result with a uniform distribution of feature points which explain that this algorithm needs strong points. So we applied the color phenomena to deal with these effects, using camshaft algorithm depending on progress stage used to detect the target motion corner. However, using detected points with the algorithm of Shi and Thomasi, we have got good results in video sequence and in real time acquisition. The obtained results indicate that the proposed algorithm can accurately extract target features points with color distribution. The future work will be able to track multi moving targets with adaptive track window enclosed them. This work will be used for real time target variation size recognition application.
6. References


