An Improved Visual Tracking Algorithm for An Inertial Stabilization System

Sangveraphunsiri V. and Malithong K.

Department of Mechanical Engineering, Faculty of Engineering, Chulalongkorn University
254 Phyathai Patumwan Bangkok 10330 Bangkok, Thailand, 10330
Corresponding Author: Tel: 0-2218-6449, Fax: 0-2218-6583,
E-mail: viboon.s@eng.chula.ac.th

Abstract

This paper presents a visual tracking system for an inertial stabilization system. This system has a 2 - DOF gimbal which will be attached to an aviation vehicle. The camera is mounted at the center of the gimbals' inner joint. The proposed system consists of two modules. The first module is a motion control system. It can move the inertial stabilization system according to a reference command. The second module is a real-time image tracking system based on the Continuously Adaptive Mean-shift (CAMshift) and Kalman filter. We present an improved CAMshift algorithm for tracking a target in video sequences. The CAMshift iterations and finds the most probable target position in the current frame after the predicted location has been confirmed by Kalman filter. The improvement of the proposed algorithm is that when the moving target is largely occluded, the Kalman filter is updated by velocity vector which is estimated according to target locations in the prior frames. Experiments on various video sequences illustrate that the pan-tilt camera can automatically follow the moving target and the proposed algorithm performs better than the original CAMshift approach.

Keywords*:* Inertial stabilization system / CAMshift algorithm / Kalman filter

1. Introduction

At present, real-time object tracking is the critical task in the field of computer vision. Object tracking means detecting moving objects from video sequences, creating their matching relation and obtaining their motion parameters. Tracking algorithms can be classified into two groups, state-space approach and kernel based approach. State-space approaches are based largely on probability, estimation theory and stochastic processes, such as Kalman filter, Particle Filter.

The Mean Shift algorithm is a non-parametric method which belongs to the second group. Mean Shift is an iterative kernel-based deterministic procedure which converges to a local maximum of the measurement function under certain assumptions about the kernel behaviors.

CAMshift (Continuously Adaptive Mean Shift) is the basis for the tracking algorithm in OpenCV. It combines the Mean Shift algorithm with an adaptive region-sizing step. The kernel is a simple step function applied to a probability density. The probability of each pixel is based on color using a method called histogram back-projection. CAMshift is fast and appears on visual tracking to be moderately accurate. But it has some important disadvantages. Firstly, CAMshift is based on color histogram. It lacks robustness when object color is similar to background color, object is occluded and object scale is variable. Secondly, the algorithm may fail to track multi-hued targets or targets where hue alone cannot allow the target to be distinguished from other targets and the background. Thirdly, CAMshift, like the mean shift algorithm, can only be used to find local modes. It fails in tracking small and fast moving targets because it is captured in a local maximum. The last, sometimes it could lose motion objects in dynamic background.

Our previous capabilities include researching for gimbal structure and the controller design [4-6]. This paper is organized as follows: Section 2 presents basic principle of traditional CAMshift algorithm. The proposed tracking algorithm is developed and analyzed in Section 3. Experiments and comparisons are given in Section 4, and the conclusion is in Section 5

2. Basic principle of traditional CAMshift algorithm

2.1 Mean Shift Algorithm

The mean shift algorithm can be used for visual tracking. The simplest such algorithm would create a confidence map in the new image based on the color histogram of the object in the previous image, and use mean shift to find the peak of a confidence map near the object's old position.

2.1.1 Target Model

First, we need to initialize position and size of search window in the first frame, where x_0 is the center of the target area. Target model can be described as the probability density distribution of the pixel's feature value (color feature value) in the target area. The probability density function of feature value in Target model is:

$$\hat{q}_u = C\sum_{i=1}^{n} k\left(\left\|\frac{x_0 - x_i}{h}\right\|^2\right)\delta\left[c(x_i) - u\right] \quad (1)$$

Given that m-bin histograms are used, we define the n image pixel locations $\{x_i\}_{i=1...n}$ and the histogram $\{\hat{q}\}_{u=1...m}$. We also define a function $c:\Re^2 \rightarrow \{1...m\}$ that associates to the pixel at location the x_i histogram bin index $c(x_i)$,. $k\left(x\right)$ is the kernel profile with bandwidth h.

2.1.2 Candidate Model

The candidate is the area possibly containing the moving object in the subsequent frames, where y is the center of the area. Candidate model can be described as the probability density distribution of the pixel's feature value in the candidate area. The normalized color distribution of a target candidate $p(y) = \left\{p_u(y)\right\}_{1,...,nh}$ centered in y can be calculated as

$$p_u(y) = C_h\sum_{i=1}^{n_h} k\left(\left\|\frac{y - x_i}{h}\right\|^2\right)\delta\left[c(x_i) - u\right] \quad (2)$$

where $\left\{x_i\right\}, i = 1,...n_h$ are the n_h pixel locations of the target candidate in the target area, and C_h is a normalization function defined as

$$C_h = \frac{1}{\sum_{i=1}^{n_h} k\left(\left\|\frac{y - x_i}{h}\right\|^2\right)} \quad (3)$$

2.1.3 Similar Function

In order to calculate the likelihood of a candidate we need a similarity function which defines a distance between the model and the candidate. A metric can be based on the Bhattacharyya coefficient defined between two normalized histograms $p(y)$ and q as

$$\rho\left[p(y), q\right] = \sum_{u=1}^{m}\sqrt{p_u(y), q_u} \quad (4)$$

The distance is defined as

$$d\left[\rho\left[p(y), q\right]\right] = \sqrt{1 - \rho\left[p(y), q\right]} \quad (5)$$

2.1.4 Mean Location Calculation for 2D Probability Distribution

The mean location (the centroid) within the search window of the discrete probability image is found using moments [1], [7]. Given that $I(x,y)$ is the intensity of the discrete probability image at position pixel (x,y) within the search window.

Compute the zeroth moment

$$M_{00} = \sum_x\sum_y I(x,y) \quad (6)$$

Find the first moment for x and y

$$M_{10} = \sum_x\sum_y xI(x,y),$$

$$M_{01} = \sum_x\sum_y yI(x,y) \quad (7)$$

Compute the mean search window location

$$x_c = \frac{M_{10}}{M_{00}};$$

$$y_c = \frac{M_{01}}{M_{00}} \quad (8)$$

2.1.5 The steps of Mean-Shift algorithm

The steps of Mean-Shift algorithm using window convergence iterations are shown as follows:

1. Initialize the size of the search window of the target.
2. Determine the initial location of the search window according to the known information.
3. According to the calculation method of the centriod, calculate the centroid x_c, y_c of the target search window.
4. Change the size of the window, and take the centroid obtained in step 3 as the new center of the search window.
5. Repeat step 3 and 4 until the moving distance of the window is shorter than the threshold or the number of the iterations reaches the set value.

2.2. CAMshift algorithm

All In 2003, Allen et al. use an algorithm known as the CAMshift algorithm for object tracking [2]. The principle of the CAMshift algorithm is given in [1-3]. For each video frame, the raw image is converted to a color probability distribution image via a color histogram model of the color being tracked. The center and size of the color object are found via the CAMshift algorithm operating on the color probability image. These new center and size are employed to place the search window in the next image. This process is then repeated for a continuous target tracking in the video sequence.

For discrete 2D image probability distributions, the mean location (the centroid) within the search window can be found by the

zeroth moment. The window size (s) can also be set by the zeroth moment. The 2D orientation of the probability distribution is also easy to obtain by using the second moments, and then length (l) and width (w) of the target can be calculated.

The process of the CAMshift algorithm is as follows:

1. Set the whole image as the search area.
2. Initialize the size and location of the search window.
3. Calculate the color probability distribution within the search window.
4. Run Mean-Shift algorithm, and get access to the new size and location of the search window.
5. In the next frame of video images, initialize the size and location of the search window with the values obtained in step 4; Jump to step 3 to continue.

3. An Improved CAMshift Algorithm

In this paper, we present an improved CAMshift algorithm to solve the problems in original CAMshift algorithm. Firstly, a background-weighted histogram which helps to distinguish the target from the background and other targets is introduced. Secondly, the approximate location of the initial search windows are determined by Kalman filtering Fig. 1 summarizes the algorithm. The proposed algorithm is based on the original CAMshift algorithm. To avoid the disadvantages we add two modules to improve the target tracking performance.

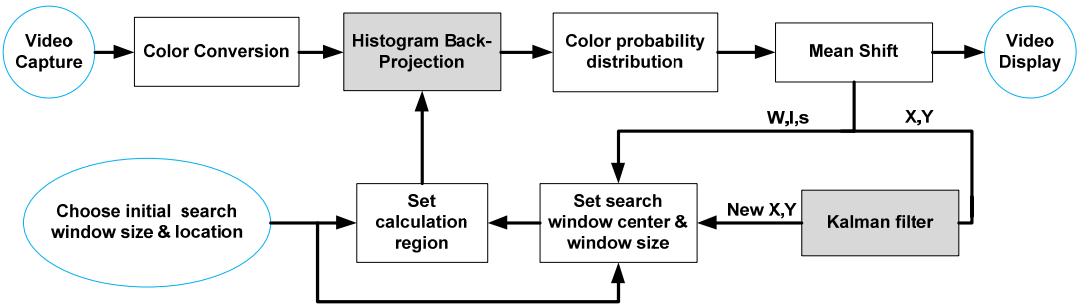

Fig. 1 Flowchart of the proposed algorithm

3.1 Background-weighted histogram

The background information is important for target tracking and often it is included in the detected target region. The background is represented as

$$\hat{O} = \left\{\hat{O}_u\right\}_{u=1\ldots m},\ \sum_{u=1}^{m}\hat{O}_u = 1 \qquad (9)$$

It is calculated by the surrounding area of the target [8]. Where $\hat{O}_u$ is the discrete representation (histogram) of the background in the feature space.

Denote by $\hat{O}^*$ is the smallest non-zero entry selected from the background model. The transformation between the representations of target model and target candidate model is defined as

$$\left\{\hat{w}_u = \min\left(\frac{\hat{O}^*}{\hat{O}_u},1\right)\right\}_{u=1\ldots m} \qquad (10)$$

These weights are employed to define a transformation for the representations of the target model and candidates. The transformation diminishes the importance of those features which have low $\hat{w}_u$, i.e., are prominent in the background. Compare with Eq. (1), the new target model representation is then defined by

$$\hat{q}_u = C\hat{w}_u\sum_{i=1}^{n}k\left(\left\|x_i^*\right\|^2\right)\delta\left[c(x_i^*)-u\right] \qquad (11)$$

with the normalization constant expressed as

$$C = \frac{1}{\sum_{i=1}^{n}k\left(\left\|x_i^*\right\|^2\right)\sum_{u=1}^{m}\hat{w}_u\delta\left[c(x_i^*)-u\right]} \qquad (12)$$

Compare with Eq.(2) and Eq.(3), similarly, the new target candidate representation is

$$p_u(y) = C_h\hat{w}_u\sum_{i=1}^{n_h}k\left(\left\|\frac{y-x_i}{h}\right\|^2\right)\delta\left[c(x_i)-u\right] \qquad (13)$$

where C_h is defined as

$$C_h = \frac{1}{\sum_{i=1}^{n_h}k\left(\left\|\frac{y-x_i}{h}\right\|^2\right)\sum_{u=1}^{m}\hat{w}_u\delta\left[c(x_i)-u\right]} \qquad (14)$$

3.2 Kalman filter

Kalman filter algorithm is that predict the most probable object location in the current frame according to the results of targets tracking in the previous frame, then search target location in the

neighbor area of the location. If there is a target existing in the search area, continue to process the next frame. The key of Kalman filter is prediction and update. This algorithm belongs to the state-space approach class of tracking algorithms. It solves the tracking problem based on the state-space equation and the measurement equation.

We define the state vector $X_k = \left[x,\ y,\ v_x,\ v_y\right]$, measurement vector $Z_k = \left[x_c,\ y_c\right]^T$, Where x, y is the centroid of the search window in the horizontal and vertical direction, x_c, y_c is the current measurement of the centroid in the horizontal and vertical direction. v_x, v_y is the velocity (displacement) of the target. Kalman Filter model is as follow:

Motion equation:

$$X_k = F \cdot X_{k-1} + W_k \tag{15}$$

Observation equation:

$$Z_k = H \cdot X_k + V_k \tag{16}$$

Where W_k and V_k are movement and measurement noise vectors which obey Gaussian distribution $p(w) \sim N(0, Q)$, $p(v) \sim N(0, R)$, F is state transition matrix and H is measurement matrix.

Prediction equation and the update equation are as follows:

Prediction equation 1:

$$X_k^{'} = F \cdot X_{k-1} \tag{17}$$

Prediction equation 2:

$$P_k^{'} = F \cdot P_{k-1} \cdot F^T + Q \tag{18}$$

Kalman-gain equation:

$$K_k = P_k^{'} \cdot H^T \cdot \left(H \cdot P_k^{'} \cdot H^T + R\right)^{-1} \tag{19}$$

Update equation 1:

$$X_k = X_k^{'} + K_k \cdot \left(Z_k - H \cdot X_k^{'}\right) \tag{20}$$

Update equation 2:

$$P_k = P_k^{'} - K_k \cdot H \cdot P_k^{'} \tag{21}$$

Where $k \geq 1$, W_k is a white Gaussian noise with diagonal variance Q, V_k is a white Gaussian noise with diagonal variance R. The values of state transition matrix F, measurement matrix H, process noise covariance matrix Q and measurement noise covariance matrix R list as follow:

$$F = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},\ Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix},\ R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Where T is the interval between the frames.

4. Experiment and result

Tracking system hardware platform consists of two parts. The first part is a notebook with 2 GHz processor. We finish the video capture program based on Visual C++ 2005 and OpenCV to develop the tracking algorithm. The second part is formed by the two-axis gimbal configuration and camera installed on gimbal. Camera resolution is set to 640x320, image acquisition frame rate is 25 frames per second. To compare the results of the original CAMshift and the proposed algorithm, we experimented on various video sequences, face and car. In figures 2 and 3 the first row is the result of the original CAMshift and the second row is the result of the proposed algorithm.

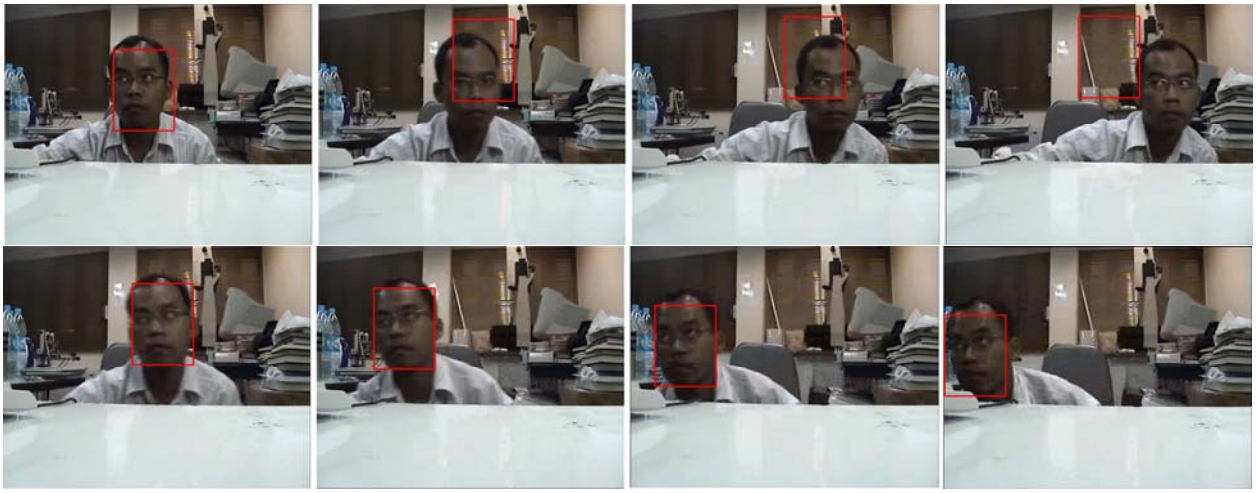

Fig. 2 Tracking the Face.

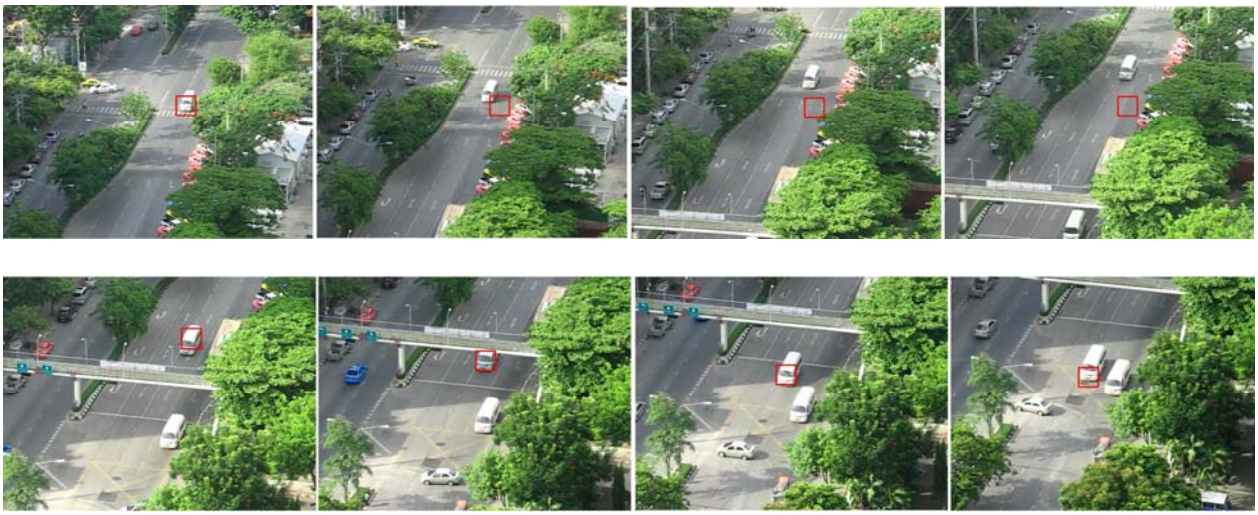

Fig. 3 Tracking the car, using the position controller with image tracking.

The first experiment describes the results obtained from experiment with the fixed camera gimbal. The face moves rapidly and venetian blind with a similar hue to the face disturbs the tracking. We can see in the first row, the original CAMshift loses the face. However the proposed algorithm can track the face in the second row of Fig. 2 because of background-weighted histogram.

The objective of the second experiment is to maintain the camera position in order to keep the target in the center of the camera image. In the road sequence, the car is difficult to distinguish from the background. It moves rapidly and is small so the displacement of this target is rather large. This large displacement results in the tracking failure with the original CAMshift tracker as can be observed in the first row of Fig. 3. But the proposed algorithm successfully tracks the car as can be seen in the second row of Fig. 3. The tracker can track the car through occlusion because of the prediction of the Kalman filter and the pan-tilt camera can automatically follow the moving target

5. Conclusion

In this paper we propose an improved CAMshift algorithm. Firstly, a background-weighted histogram is introduced, so the target can be easily distinguished from the background and other targets. Secondly, Kalman filter algorithm is that predict the most probable object location in the current. By combining the CAMshift and Kalman Filter, the proposed algorithm enhances the robustness to occlusion and it can track the target accurately despite its shape and orientation change. Compared with the original CAMshift algorithm, the improved CAMshift algorithm shows its higher performance in video sequences.

6. Acknowledgement

Part of this work is sponsored by Chulalongkorn University under Chulalongkorn University Centenary Academic Development Project.

7. References

[1] Intel Corporation (2001). Open Source Computer Vision Library Reference Manual, ISBN 123456-001.

[2] Allen, J. G., Xu, Richard, Y. D.and Jin, J. S. (2004). Object tracking using CamShift algorithm and multiple quantized feature spaces, paper presented in *VIP '05 Proceedings of the Pan-Sydney area workshop on Visual information processing*, Darlinghurst, Australia.

[3] Bradski G. R. (1998). Computer vision face tracking for use in a perceptual user interface, *Intel Technology Journal*, 2nd Quarter.

[4] Wongkamchang, P. and Sangveraphunsiri, V. (2008). Control of Inertial Stabilization Systems Using Robust Inverse Dynamics Control and Adaptive Control, *Thammasat International Journal of Science and Technology*, vol 13(2), pp. 20 – 32.

[5] Sangveraphunsiri V. and Malithong K.(2009) Robust Inverse Dynamics and Sliding Mode Control for Inertial Stabilization Systems, *Asian International Journal of Science and Technology*

in Production and Manufacturing Engineering, vol. 2(4), pp. 33 – 45.
[6] Malithong K., and Sangveraphunsiri V.(2010) Control of Inertial Stabilization Systems Using Image Tracking of Non-Rigid Objects, *Asian International Journal of Science and Technology in Production and Manufacturing Engineering*, vol. 3(4), pp. 1 – 11.
[7] Horn, B. K. P. (1986): Robot vision, MIT Press, Massachusetts.
[8] Comaniciu D., Ramesh V. and Meer P. (2003): Kernel-Based Object Tracking, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25(2), pp. 564-577.