An Adaptive Background Model for Camshift Tracking with a Moving Camera

R. Stolkin*, I. Florescu**, G. Kamberov***

*Center for Maritime Systems, **Dept. of Mathematical Sciences, ***Dept. of Computer Science
Stevens Institute of Technology,
Castle Point on Hudson, Hoboken, NJ 07030, USA,
E-mail: {*rstolkin@stevens.edu; **ifloresc@stevens.edu; ***george.kamberov@stevens.edu}

Continuously Adaptive Mean shift (CAMSHIFT) is a popular algorithm for visual tracking, providing speed and robustness with minimal training and computational cost. While it performs well with a fixed camera and static background scene, it can fail rapidly when the camera moves or the background changes since it relies on static models of both the background and the tracked object. Furthermore it is unable to track objects passing in front of backgrounds with which they share significant colours.
We describe a new algorithm, the Adaptive Background CAMSHIFT (ABCshift), which addresses both of these problems by using a background model which can be continuously relearned for every frame with minimal additional computational expense. Further, we show how adaptive background relearning can occasionally lead to a particular mode of instability which we resolve by comparing background and tracked object distributions using a metric based on the Bhattacharyya coefficient.

Keywords: CAMSHIFT; mean shift; ABCshift; tracking; adaptive; background model; robot vision

1. Introduction

Popular and effective approaches to colour based tracking include the CAMSHIFT,[1,2] Mean Shift[3] and particle filtering[4,5] algorithms. Of these, CAMSHIFT stands out as the fastest and simplest. CAMSHIFT was designed for close range face tracking from a stationary camera but has since been modified for a variety of other tracking situations.[6,7]

Robust and flexible tracking algorithms, requiring minimal training and computational resources, are highly desirable for applications such as robot vision and wide area surveillance, both of which necessitate moving cameras. Unfortunately CAMSHIFT often fails with camera motion, figure 3, since it relies on a static background model which is unable to adequately represent changing scenery.

We address these difficulties by modifying the algorithm to include a flexible background representation which can be continuously relearned. The resulting algorithm tracks robustly in two situations where CAMSHIFT fails; firstly with scenery change due to camera motion and secondly when the tracked object moves across regions of background with which it shares significant colours, figure 1.

It is observed that the adaptability of this approach makes it occasionally vulnerable to a special mode of instability, in which a shrinking search window can lead to the background being relearned as object. We detect and correct this error by comparing object and background distributions to check for convergence.

2. Bayesian mean shift tracking with colour models

For each frame of an image sequence, the CAMSHIFT algorithm looks at pixels which lie within a subset of the image defined by a search window. Each pixel in this window is assigned a probability that it belongs to the tracked object, creating a 2D distribution of object location over a local area of the image. The tracking problem is solved by mean shifting[3,8] towards the centroid of this distribution to find an improved estimate of the object location. The search window is now repositioned at the new location and the process is iterated until convergence.

By summing the probabilities of all the pixels in the search window, it is also possible to estimate the size of the tracked object region (in pixels). The search window can now be resized so that its area is always in a fixed ratio to this estimated object area.

The tracked object is modelled as a class conditional colour distribution, $\mathbf{P}(C|O)$. Depending on the application, 1D Hue, 3D normalised RGB, 2D normalised RG, UV or ab histograms may all be appropriate choices of colour model, the important point being that these are all distributions which return a probability for any pixel colour, given that the pixel represents the tracked object. These object distributions can be learned offline from training images,

or during initialisation, e.g. from an area which has been user designated as object in the first image of the sequence.

The object location probabilities can now be computed for each pixel using Bayes' law as:

$$\mathbf{P}(O|C) = \frac{\mathbf{P}(C|O)\mathbf{P}(O)}{\mathbf{P}(C)} \quad (1)$$

where $\mathbf{P}(O|C)$ denotes the probability that the pixel represents the tracked object given its colour, $\mathbf{P}(C|O)$ is the colour model learned for the tracked object and $\mathbf{P}(O)$ and $\mathbf{P}(C)$ are the prior probabilities that the pixel represents object and posesses the colour C respectively.

The denominator of equation (1) can be expanded as:

$$\mathbf{P}(C) = \mathbf{P}(C|O)\mathbf{P}(O) + \mathbf{P}(C|B)\mathbf{P}(B) \quad (2)$$

where $\mathbf{P}(B)$ denotes the probability that the pixel represents background.

Bradski[1,2] recommends values of 0.5 for both $\mathbf{P}(O)$ and $\mathbf{P}(B)$. We find this choice difficult to justify since we take these terms to denote the expected fractions of the total search window area containing object and background pixels respectively. Hence we assign values to object priors in proportion to their expected image areas. If the search window is resized to be r times bigger than the estimated tracked object area, then $\mathbf{P}(O)$ is assigned the value $1/r$ and $\mathbf{P}(B)$ is assigned the value $(r-1)/r$.

Bradski[1,2] suggests learning the expression (2) offline (presumably building a static $\mathbf{P}(C|B)$ histogram from an initial image). While it is often reasonable to maintain a static distribution for the tracked object (since objects are not expected to change colour), a static background model is unrealistic when the camera moves. The CAMSHIFT algorithm can rapidly fail when the background scenery changes since colours may exist in the new scene which did not exist in the original distribution, such that the expressions in Bayes law will no longer hold true and calculated probabilities no longer add up to unity.

Particular problems arise with CAMSHIFT if the tracked object moves across a region of background with which it shares a significant colour. Now a large region of background may easily become mistaken for the object, figure 1.

3. Incorporating an adaptive background model

We address these problems by using a background model which can be continuously relearned. Rather than using an explicit $\mathbf{P}(C|B)$ histogram, we build a $\mathbf{P}(C)$ histogram which is recomputed every time the search window is moved, based only on the pixels which lie within the current search window. $\mathbf{P}(C)$ values, looked up in this continuously relearned histogram, can now be substituted as the denominator for the Bayes' law expression, equation (1), for any pixel. Since the object distribution, $\mathbf{P}(C|O)$, remains static, this process becomes equivalent to implicitly relearning the background distribution, $\mathbf{P}(C|B)$, because $\mathbf{P}(C)$ is composed of a weighted combination of both these distributions, equation (2). Relearning the whole of $\mathbf{P}(C)$, rather than explicitly relearning $\mathbf{P}(C|B)$, helps ensure that probabilities add up to unity (e.g. if there are small errors in the static object model).

Adaptively relearning the background distribution helps prevent tracking failure when the background scene changes, particularly useful when tracking from a moving camera, figure 4. Additionally, it enables objects to be tracked, even when they move across regions of background which are the same colour as a significant portion of the object, figure 2. This is because, once $\mathbf{P}(C)$ has been relearned, the denominator of Bayes' law, equation (1), ensures that the importance of this colour will be diminished. In other words, the tracker will adaptively learn to ignore object colours which are similar to the background and instead tend to focus on those colours of the object which are most dissimilar to whatever background is currently in view.

It is interesting to note that the continual relearning of the $\mathbf{P}(C)$ histogram need not substantially increase computational expense. Once the histogram has been learned for the first image it is only necessary to remove from the histogram those pixels which have left the search window area, and add in those pixels which have newly been encompassed by the search window as it shifts with each iteration. Provided the object motion is reasonably slow relative to the camera frame rate, the search window motion will be small, so that at each iteration only a few lines of pixels need be removed from and added to the $\mathbf{P}(C)$ histogram.

If the $\mathbf{P}(C)$ histogram is relearned only once every frame, the speed should be similar to that of CAMSHIFT. However, if the histogram is relearned at every iteration, some additional computational expense is incurred, since to properly exploit the new information it is necessary to recompute the $\mathbf{P}(O|C)$ values for every pixel, including those already analysed in previous iterations. Theoretically, updating at each iteration should produce more reliable tracking, although we have observed good tracking results with both options.

In practice, ABCshift often runs significantly faster than CAMSHIFT. Firstly, the poor background model can cause CAMSHIFT to need more iterations to converge. Secondly, the less accurate tracking of CAMSHIFT causes it to automatically grow a larger search window area, so that far greater numbers of pixels must be handled in each calculation.

4. Dealing with instability

Adaptively relearning the background model enables successful tracking in situations where CAMSHIFT fails, however it also introduces a new mode of instability which occasionally causes problems. If the search window should shrink (due to the object region being temporarily underestimated in size) to such an extent that the boundaries of the search window approach the boundaries of the true object region, then the background model will learn that background looks like object. This results in a negative feedback cycle with the estimated object region and search window gradually (and unrecoverably) collapsing.

We solve this problem by noting that as the search window shrinks and approaches the size of the object region, the learned background distribution, $\mathbf{P}(C|B)$, must become increasingly similar to the static distribution for the tracked object, $\mathbf{P}(C|O)$. If this increasing similarity can be detected, then both the object region and search window can be easily resized, figure 5, the correct enlargement factor being r, the desired ratio of search window size to object region size.

Several statistical measures exist for comparing the similarity of two histograms. We utilise a Bhattacharyya metric,[9] sometimes referred to as Jeffreys-Matsusita distance,[10] which for two histograms, $p = \{p_i\}_{i\in\{1,2,\ldots,K\}}$, and $q = \{q_i\}_{i\in\{1,2,\ldots,K\}}$ is defined as:

$$d(p,q) = \sqrt{\sum_{i=1}^{K} \left(\sqrt{p_i} - \sqrt{q_i}\right)^2}, \qquad (3)$$

$0 \leq d \leq \sqrt{2}$. Note that this metric can easily be shown to be the same, modulo a factor of $\sqrt{2}$, as that referred to elsewhere in the literature.[3–5,8]

At each iteration we evaluate the Bhattacharyya metric between the static object distribution, $\mathbf{P}(C|O)$, and the continuously relearned background distribution, $\mathbf{P}(C|B)$. If the Bhattacharyya metric approaches zero, we infer that the search window is approaching the true object region size while the estimated object region is collapsing. We therefore resize both by the factor r. In practice we resize when the Bhattacharyya metric drops below a preset threshold. Useful threshold values typically lie between 0.2 and 0.7.

We believe this is a novel application of the Bhattacharyya metric. It is common in the vision literature[3–5,8] to use this metric to evaluate the similarity between a candidate image region and an object distribution for tracking (i.e. comparing potential object with known object). In contrast, we use the metric to compare an object distribution with a background distribution, inferring an error if the two begin to converge.

5. Summary of the ABCshift tracker

The key differences between ABCshift and the conventional CAMSHIFT tracker are as follows:
1. The background is continuously relearned. In contrast CAMSHIFT uses a static background model which often fails with a moving camera. 2. Values for the prior probabilities, $\mathbf{P}(O)$ and $\mathbf{P}(B)$, are assigned based on the ratio, r, of search window size to estimated object region size. 3. Object and background distributions are compared using a metric based on the Bhattacharyya coefficient to check for instability and resize the search window if it is shrinking.

The ABCshift algorithm is summarised as:
1. Identify an object region in the first image and train the object model, $\mathbf{P}(C|O)$. *2. Center the search window on the estimated object centroid and resize it to have an area* r *times greater than the estimated object size. 3. Learn the colour distribution,* $\mathbf{P}(C)$*, by building a histogram of the colours of all pixels within the search window. 4. Use Bayes' law, equation* (1),

to assign object probabilities, $\mathbf{P}(O|C)$*, to every pixel in the search window, creating a 2D distribution of object location. 5. Estimate the new object position as the centroid of this distribution and estimate the new object size (in pixels) as the sum of all pixel probabilities within the search window. 6. Compute the Bhattacharyya metric between the distributions,* $\mathbf{P}(C|O)$ *and* $\mathbf{P}(C)$*. If this metric is less than a preset threshold then enlarge the estimated object size by a factor* r*. 7. Repeat steps 2-6 until the object position estimate converges. 8. Return to step 2 for the next image frame.*

6. Results

To test the enhanced tracking capabilities of the adaptive background model, we compare the performance of the ABCshift tracker with that of CAMSHIFT for a large number of extended video sequences, available at our website.[11] Some sample images are shown here in the figures.

ABCshift has succeeded in conditions of substantial camera motion, rapidly changing scenery, dim and variable lighting and partial occlusion. The strengths of ABCshift are particularly apparent in scenarios where the tracked object passes across regions of background with which it shares significant colours.

7. Conclusions

Adaptive background models, which can be continuously relearned, significantly extend the capabilities and potential applications of simple colour based tracking algorithms to include moving cameras and changing scenery. They also provide robustness in difficult situations where other algorithms fail, such as when the tracked object moves across regions of background with which it shares significant colours.

It is observed that the resulting flexibility can make the tracker prone to a particular mode of instability, however this can be corrected by an innovative application of a metric based on the Bhattacharyya coefficient.

Future work will examine alternative adaptive techniques for relearning models of both the background and the tracked object. We are also exploring automated initialisation of the tracker in surveillance scenarios and applications of these algorithms to visual servoing and robot vision.

References

1. G. R. Bradski, *Intel Technology Journal* (Q2 1998).
2. G. R. Bradski, Real time face and object tracking as a component of a perceptual user interface, in *Proc. 4th IEEE Workshop Applications of Computer Vision*, 1998.
3. D. Comaniciu, V. Ramesh and P. Meer, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **25**, 564 (2003).
4. L. V. G. Katja Nummiaro, Esther Koller-Meier, Object tracking with an adaptive color-based particle filter, in *Pattern Recognition : 24th DAGM Symposium, Proceedings*, Lecture Notes in Computer Science Vol. 2449 (Zurich, Switzerland, 2002).
5. P. Pérez, C. Hue, J. Vermaak and M. Gangnet, Color-based probabilistic tracking, in *Proceedings of the 7th European Conference on Computer Vision-Part I*, Lecture Notes In Computer Science Vol. 23502002.
6. J. G. Allen, R. Y. D. Xu and J. S. Jin, Object tracking using camshift algorithm and multiple quantized feature spaces, in *Proceedings of the Pan-Sydney area workshop on Visual information processing*, ACM International Conference Proceeding Series Vol. 100 (Australian Computer Society, Inc., Darlinghurst, Australia, 2004).
7. N. Liu and B. C. Lovell, Mmx-accelerated real-time hand tracking system, in *IVCNZ 2001*, (Dunedin, New Zealand, 2001).
8. D. Comaniciu and P. Meer, Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24(5)**, 603-619 (May 2002).
9. A. Bhattacharayya, On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of the Calcutta Mathematical Society* **35**, 99-110 (1943).
10. H. Jeffreys, An invariant form for the prior probability in estimation problems. *Proc. Royal Society (A)* **186**, 453-461 (1946).
11. `www.math.stevens.edu/~ifloresc/ABCshift.htm`.

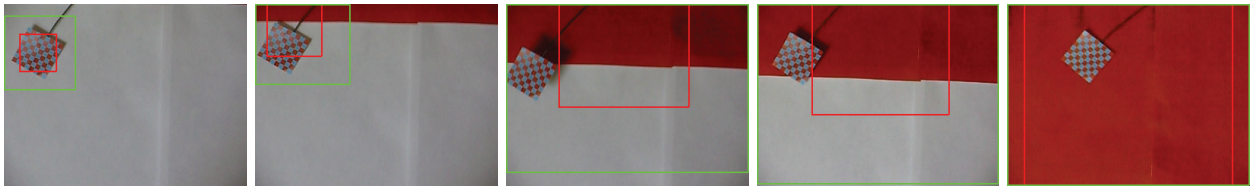

Fig. 1. A simple blue and red checkered object, moving from a region of white background into a region of red background. CAMSHIFT fails as soon as the object moves against a background with which it shares a common colour. Frames 350, 360, 380, 400, and 450 shown. Green and red squares indicate the search window and estimated object size respectively. This movie, RedWhite1CAMSHIFT.avi, can be viewed at our website.[11]

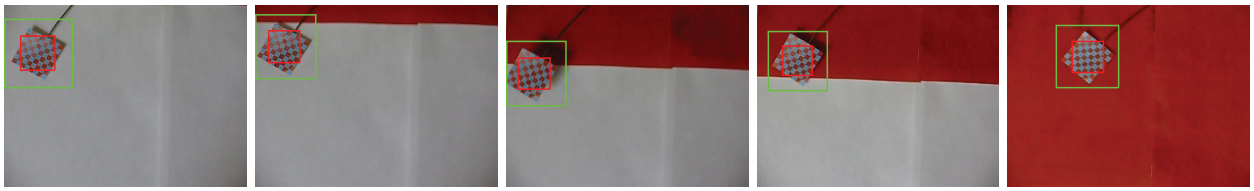

Fig. 2. ABCshift tracks successfully. Frames 350, 360, 380, 400, and 450 shown. Green and red squares indicate the search window and estimated object size respectively. This movie, RedWhite1ABCshift.avi, can be viewed at our website.[11]

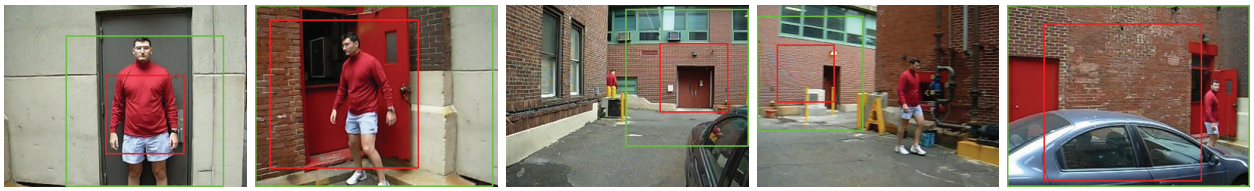

Fig. 3. Person tracking with CAMSHIFT from a moving camera in a cluttered, outdoors environment. Frames 1, 176, 735, 1631, and 1862 shown. Since the tracked person wears a red shirt, CAMSHIFT fixates on red regions of background, including brick walls and doors, and repeatedly loses the tracked person. Green and red squares indicate the search window and estimated object size respectively. This movie, PeopleTracking1CAMSHIFT.avi, can be viewed at our website.[11]

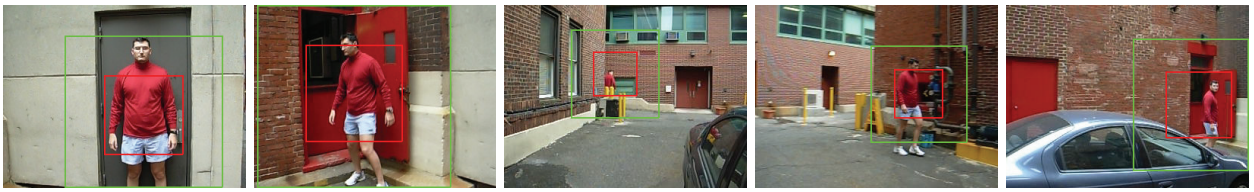

Fig. 4. ABCshift successfully tracks throughout the sequence and is not distracted by red regions of background, despite being initialised in image 1 which contains no red background. Frames 1, 176, 735, 1631, and 1862 shown. Green and red squares indicate the search window and estimated object size respectively. This movie, PeopleTracking1ABCshift.avi, can be viewed at our website.[11]

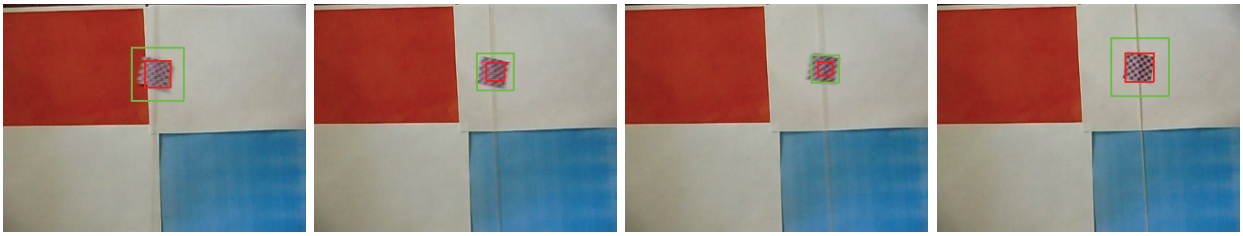

Fig. 5. Bhattacharyya resizing. A simple red and blue checkered object is tracked across red, white and blue background regions. Frames 180, 200, 205, 206 shown. Due to rapid, jerky motion from frames 180 to 205, the search window has shrunk until it falls within the object region, risking relearning that background looks like object. ABCshift has detected this instability using the Bhattacharyya metric, and automatically corrects the estimated object region and search window size in frame 206. Green and red squares indicate the search window and estimated object size respectively. This movie, BhattacharyyaRedWhiteBlue.avi, can be viewed at our website.[11]