Managing Particle Spread via Hybrid Particle Filter/Kernel Mean Shift Tracking

Asad Naeem[1], Tony Pridmore[1] and Steven Mills[2]

[1]School of Computer Science, University of Nottingham, Nottingham NG8 1BB, UK.

[2]Geospatial Research Centre (NZ) Ltd, Private Bag 4800, Christchurch 8140, New Zealand

Abstract

Particle filtering provides a well-developed and widely adopted approach to visual tracking. For effective tracking in real-world environments the particle set must sample widely enough that it can represent alternative target states in areas of ambiguity. It must not, however, become diffuse, spreading across the image plane rather than clustering around the object(s) of interest. A key issue in the design of particle filter-based trackers is how to manage the spread of the particle set to balance these conflicting requirements. To be computationally efficient, balance must be achieved with as small a particle set as reasonably possible. A number of hybrid particle filter/mean-shift trackers have recently been proposed. We believe that their strength lies in their ability to alternately disperse and cluster particles together, providing both a degree of balance and a reduced particle set. We present a novel hybrid of the annealed particle filter and kernel mean-shift algorithms that emphasises this behaviour. The algorithm has been applied to a wide variety of artificial and real image sequences. The method has performance and efficiency advantages over both pure kernel mean-shift and particle filtering trackers and existing hybrid algorithms

1 Introduction

The defining characteristic of the particle filter approach to visual tracking is its use of a set of discrete particles to represent multi-modal probability distributions that capture and maintain multiple hypotheses about target properties. Particle filtering is iterative. Particles are repeatedly selected, projected forwards using a motion model, dispersed by an additive random component, and evaluated against the image data. Many particle filter trackers have appeared since Blake and Isard [1] first introduced Condensation.

The ability of a set of particles to represent a wide variety of distributions is both the main strength and primary weakness of the particle filter. For effective tracking in real-world environments the particle set must sample widely enough to represent all reasonable alternatives in areas of ambiguity. It must not, however, become diffuse, spreading across the image plane rather than clustering around the object of interest. When this happens particles tend to migrate towards local maxima in their evaluation function, becoming caught on clutter and losing track of the target. Similarly, particles should not become too focused. Though it is encouraging to see a particle set coalesce when a single, clearly distinguishable target moves across the image, the tracker should not become irreversibly locked onto a single mode.

A key issue in the design of particle filter-based trackers is how to manage the spread of the particle set to balance these conflicting requirements. The variance of the posterior is simply and elegantly maintained by the Kalman filter, but particle filters cannot assume a Gaussian, or indeed any specific, distribution. Moreover, balance must be achieved with as few particles as reasonably possible. Increasing the particle set improves representational accuracy, but adds significantly to computational overhead.

Several works have addressed aspects of this problem. Some point out that, in practice, the advantages of the particle filter approach are often lost as particles cluster, sometimes very quickly, around one target hypothesis. They focus on maintaining a wider distribution. The Annealed Particle Filter [2] uses annealing to smooth out the evaluation function, making the global maximum clearer and allowing particles to be spread further, by increasing the process noise, without becoming caught on local clutter. Vermaak et al [3] explicitly model the particle distribution as a Gaussian mixture model, forcing the resulting filter to sample an appropriate number of particles from each model component. This prevents a single, slightly more highly weighted, mode from dominating the particle distribution.

Other workers consider standard algorithms to spread the particle set too thinly across the image and concentrate effort on forcing particles to coalesce, reducing the number needed and so computational expense. The Kernel Particle Filter [4] applies a mean shift operation to the particle set to pull the centre of the particle distribution towards the target centre. This is effective, but clusters weighted particles without further reference to the image data, taking no account of the actual shape of the evaluation function between the locations sampled by the particle set. Recently, Maggio and Cavallaro [5] used a Kernel Mean Shift tracker [6] to move particles towards local maxima of the evaluation function on each iteration of Condensation.

Kernel mean shift hill climbs towards the target, minimising the distance between target and model descriptions. A spatial kernel provides some robustness to noise and partial occlusion, and the algorithm provides fast and effective tracking as long as the target object does not move further than its own diameter between frames. A number of variations on the theme have been described; a variety of colour models and similarity measures have been used and arbitrary spatial weighting [7] has been incorporated to represent objects with arbitrary or changing shapes.

Though the authors focus on the computational savings made, Maggio and Cavallaro's [5] hybrid tracker can be viewed as attempting to manage particle spread by alternately diffusing the particle set using Condensation and clustering them with Kernel Mean Shift. The algorithm shows performance advantages over both Condensation and Kernel Mean Shift, but has some drawbacks. If Condensation tends towards an incorrect local maximum the mean shift step will accelerate the process.

Recognising that the weakness of the hybrid Condensation/Mean Shift tracker lies in the particle set generated by the Condensation component, Naeem et. al. [8] propose an alternative hybrid in which Kernel Mean Shift is the dominant technology. A small number of particles are generated, in a structured fashion, to explore further when confidence in Kernel Mean Shift becomes low. Naeem et. al.'s tracker makes explicit the iterative diffuse-cluster structure implicit in Maggio and Cavallaro's hybrid, and shows performance advantages over Condensation, Kernel Mean Shift and the Maggio and Cavallaro hybrid. The algorithm is similar in principle to the hybrid tracker of [9], which runs Kernel Mean Shift and Condensation algorithms in parallel and uses the highest confidence estimate to initialise mean shift at each time step. Naeem et. al.'s SOK tracker carries a lighter computational overhead, but requires the user to specify the conditions under which extra particles are spawned and the size of the region to be searched. This is irksome and open to error.

Here we take an alternative approach. Rather than shift control away from the particle filter component and towards the kernel mean-shift tracker we replace Condensation with a more powerful particle filter. We propose a hybrid particle filter/mean-shift tracking algorithm created by combination of the kernel Mean-Shift algorithm with Deutscher et. al.'s [2] Annealed Particle filter. We hypothesize that by smoothing out local maxima in the evaluation function the annealed particle filter will allow a greater spread in the particle set, while the Mean-Shift component will successfully pull particles back towards the true target.

The proposed Kernel Annealed Mean Shift (KAMS) tracking algorithm is presented in Section 2 and experimentally compared with Condensation [1], Kernel Mean Shift [6], annealed particle filtering [2], Maggio and Caravello's [5] condensation-based and Naeem et. al.'s [8] SOK hybrids in Section 3. Conclusions are drawn in Section 4.

2 The Kernel Annealed Mean Shift Tracker

Annealed particle filtering relies upon a series of particle weighting functions $w_0(\mathbf{Z},\mathbf{X})$ to $w_M(\mathbf{Z},\mathbf{X})$ where $\mathbf{Z}$ is a measurement vector extracted from the image and $\mathbf{X}$ is the current model state. A given weighting function w_m is obtained by raising the original weighting function $w(\mathbf{Z},\mathbf{X})$ to a power β_m, so that

$$w_m(\mathbf{Z},\mathbf{X}) = w(\mathbf{Z},\mathbf{X})^{\beta m} \tag{1}$$

where $\beta_0 = 1.0$ and $\beta_0 > \beta_1 > \beta_2 > \ldots > \beta_M$. As β_m increases, extrema in the weighting function become more pronounced. So $w_0(\mathbf{Z},\mathbf{X})$ is the raw weighting function while $w_M(\mathbf{Z},\mathbf{X})$ captures only the broad structure of the search space. In [2] $w(\mathbf{Z},\mathbf{X})$ is the sum of squared differences between the model and image data.

In annealed particle filtering each particle is evaluated at each time step using each $w_m(\mathbf{Z},\mathbf{X})$, starting with $w_M(\mathbf{Z},\mathbf{X})$ and moving to $w_0(\mathbf{Z},\mathbf{X})$. At a given time step t_k the process begins with a set of N unweighted particles

$$S_{k,M} = \{s_{k,M}^{(0)}, s_{k,M}^{(1)},\ldots s_{k,M}^{(N)}\} \tag{2}$$

Each particle $s_{k,M}^{(i)}$ is then assigned a weight $\pi_{k,m}^{(i)}$ where

$$\pi_{k,m}^{(i)} \propto w_m(\mathbf{Z_k}, s_k \tag{3}$$

and, in the first step, $w_m(\mathbf{Z_k}, \mathrm{s_k}^{(i)}) = w_M(\mathbf{Z_k}, \mathrm{s_k}^{(i)})$, resulting in a set of weighted particles $S^{\pi}_{k,M}$. N particles are now drawn randomly from $S^{\pi}_{k,M}$ with replacement and used to create a set of unweighted particles for evaluation using the next weighting function

$$s_{k,M-1}^{(i)} = s_{k,M}^{(i)} + B_m \tag{4}$$

where B_m is a multi-variate Gaussian random variable with mean 0 and variance P_m. $S_{k,M-1}$ is then weighted using $w_{m-1}(\mathbf{Z_k}, s_k^{(i)})$. This is repeated until $S^{\pi}_{k,0}$ is produced

Annealing allows us to counteract the natural tendency of particle filters to cluster particles together by increasing the variance P_m, confident that the smoother weighting functions used in the early part of the annealing run will steer particles away from local extrema. Increasing the spread of the particle set, however, also increases the number of particles required to effectively sample the search area. To make explicit and accelerate the process of seeking the global maxima we apply a Kernel Mean Shift tracking step to each particle at each stage in the annealing run.

Kernel Mean Shift [6] maintains a single estimate of target position, hill climbing from the previous location estimate toward a local minimum in the Bhattacharya distance between normalized, kernel weighted color histograms representing the object model and local image data. Assuming a 3D colour histogram the Bhattacharya distance between model and candidate is:

$$bhata\,() = \sqrt{1 - \sum_{i}^{L}\sum_{j}^{L}\sum_{k}^{L} \sqrt{p(i,j,k)\times d(i,j,k)}} \tag{5}$$

where p and d are the object and the candidate models respectively. The iterative Kernel Mean Shift operation is as follows [6]:

$$x = \frac{\sum_{x=xi}^{xf}\sum_{y=yi}^{yf} x \times \sqrt{\frac{p[r_{(x,y)}, g_{(x,y)}, b_{(x,y)}]}{d[r_{(x,y)}, g_{(x,y)}, b_{(x,y)}]}}}{wt}$$

$$y = \frac{\sum_{x=xi}^{xf}\sum_{y=yi}^{yf} y \times \sqrt{\frac{p[r_{(x,y)}, g_{(x,y)}, b_{(x,y)}]}{d[r_{(x,y)}, g_{(x,y)}, b_{(x,y)}]}}}{wt} \tag{6}$$

where

$$wt = \sum_{i}^{L}\sum_{j}^{L}\sum_{k}^{L} \sqrt{\frac{p(i,j,k)}{d(i,j,k)}} \tag{7}$$

and x and y are the coordinates of the next estimate of the position of the centre of the object. In the current implementation the object and candidate are 10 x 10 x 10 bin histograms (L=10) recording RGB color values. The histogram is normalized to sum to 1. Experience has shown this to provide an effective compromise between descriptive power and ability to generalise. Though any suitable kernel could be employed, for simplicity and generality we use a linear kernel having maximum weight at the centre of the circular target area and zero weight at boundaries and beyond.

The original annealed particle filter used sum of squared difference as its base weighting function. The Kernel Mean Shift algorithm relied upon Bhattacharya distance. To allow comparison we employ Bhattacharya distance throughout. Kernel Mean Shift is run until Bhattacharya distance either falls below a small threshold or becomes stable. Experience has shown that this usually occurs within five iterations, so a limit on the number of iterations applied can reasonably be used, if needed, to reduce computation. The final KAMS algorithm is given in Figure 1

```
Kernel Based Annealed Mean Shift Tracker:
1. Acquire frame at time $t_k$, having a set $S_{k,M}$ of N unweighted particles from the
   previous time step,
2. Set weighting function index m = M
3. While (m>0)
     a. Assign each particle a weight $\pi_{k,m}^{(i)}$
     b. Select N particles with replacement and add Gaussian noise:
        $s_{k,m-1}^{(i)} = s_{k,m}^{(i)} + B_m$
     c. Apply Kernel Mean Shift to each particle until the Bhattacharya
        distance between the model and image measured by the weighting
        function $w_{m-1}(\mathbf{Z_k}, s_{km-1}^{(i)})$ becomes stable or minimum.
     d. m = m-1
Go to 1.
```

Figure 1: The Kernel Annealed Mean-Shift (KAMS) tracking algorithm

3 Experimental Evaluation

3.1 Algorithms

The proposed KAMS tracker has been experimentally compared with Kernel Mean Shift [6], Condensation [1], annealed particle filtering [2] and the hybrid tracking algorithms proposed by Maggio and Caravello [5] and Naeem et. al. [8].

In Maggio and Caravello's hybrid tracker (henceforth simply "Hybrid") Condensation provides a harness into which the Kernel Mean Shift tracker outlined in section 2 is slotted. In our implementation particles are evaluated at each time step by computing the Bhattacharya distance between the object and their candidate model. Particles are then selected with probability proportional to their measurement value and projected into the next image by a constant velocity motion model. A Kernel Mean Shift tracker is initialised at each particle location and run until its associated Bhattacharya distance becomes small or constant. A limit on the number of mean shift iterations may be imposed to reduce computation without significant degradation in performance.

Naeem et. al.'s [8] Structured Octal Kernel algorithm (henceforth "SOK") is a Kernel Mean Shift tracker augmented by a backup strategy triggered when confidence in the current location estimate is low. Confidence at time *t* is given by

$$C_t = (1.0 - bhata(t)) \tag{8}$$

A user-defined threshold, *T*, is applied to *C* at each time step. If C_t is below threshold a set of eight independent Kernel Mean Shift trackers are spawned, each with the same object model as the original but at locations designed to cover a search area around the current position estimate (Figure 2). When these additional trackers have also each converged, nine estimates of target location are available, each with an associated confidence level. The estimate with the highest confidence is selected and the process continues. In the original formulation the object model was a two-dimensional histogram of red/blue and green/blue. To allow comparison the implementation employed here uses a 10 x 10 x 10 RGB histogram.

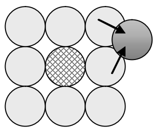

Figure 2: The SOK algorithm in operation. A hatched circle shows the primary Kernel Mean Shift tracker, light circles the secondary "particles", a dark circle the target.

3.2 Robustness

Quantitative, comparative analysis of the robustness of the proposed KAMS algorithm is achieved using McNemar's statistic [10]. McNemar's statistic is a form of chi-square test for matched paired data. Let N_{xy} give the number of times algorithm A produced result x and algorithm B produced result y, and f and s denote failure and success respectively. McNemar's statistic is then:

$$x^2 = \frac{(|N_{sf} - N_{fs}| - 1)^2}{N_{sf} + N_{fs}} \tag{9}$$

The Z score (standard score) is obtained as:

$$z = \frac{(|N_{sf} - N_{fs}| - 1)}{\sqrt{N_{sf} + N_{fs}}} \tag{10}$$

If the two algorithms give similar results then Z will tend to zero. As their results diverge, Z increases. Confidence limits can be associated with the Z value [10].

To apply McNemar's, a definition of success and failure is required. Focusing on robustness, and recognizing that any tracker will fail at some point, we consider algorithm A to have succeeded and algorithm B to have failed if algorithm A maintains tracking for a greater proportion of a given image sequence, from the same starting parameters. In effect we define success to be tracking as long as the more successful of the two trackers. McNemar's test was applied to a set of 36 assorted image sequences (available from http://www.cs.nott.ac.uk/~azn/kams_bmvc.htm) to provide quantitative comparison of the robustness of KAMS with Kernel Mean Shift, Condensation, Annealed particle filter, Hybrid and SOK. With Z scores shown in table 1, KAMS is significantly more robust than all the five algorithms with a confidence of 99.5%.

	Kams vs. Condensation	Kams vs. Mean Shift	Kams vs. Annealing	Kams vs. Hybrid	Kams vs. SOK
Z Scores	2.910428	5.126524	3.801316	4.828079	3.590662

Table 1: Z score comparisons of KAMS with the other five algorithms.

Figures 3-7 show selected frames from the results of applying the six algorithms to some of the sequences used in the McNemar's test. Figure 3 shows a tiger sprinting through dense jungle. The animal's motion is smooth, but quite fast, with frequent changes in head direction. Surrounding trees generate many partial occlusions and the dark stripes on the animal and the shadows caused by the leaves are similar, generating high levels of potentially confusing background clutter. All algorithms were manually initialised to the same point and (except SOK and Mean Shift) used 200 particles.

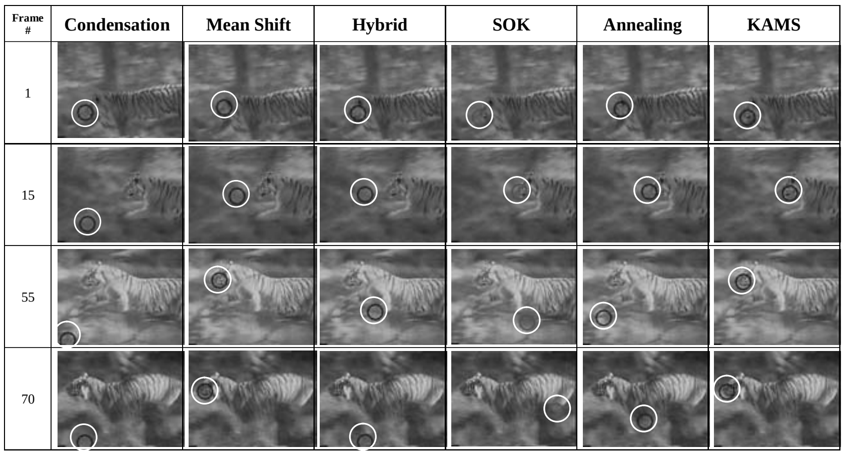

Figure 3: Six algorithms track a sprinting tiger. See supplementary material.

Condensation fails at the 15th frame; the particles are diffused and latch on to clutter resembling the tiger's head. Mean shift also fails around the 15th frame due to the high speed of movement, but latches back onto the head by chance around frame 41. Hybrid fails at frame 12 as the frequent changes in head velocity violate its motion model. SOK and the annealed particle filter fare better, and keep hold of the object until around the 50th frame, when changes in lighting conditions make clutter within their search areas appear more like the head model than the true head does.

KAMS successfully tracks to the end of the sequence. KAMS does not employ a motion model, and its combined use of particles and mean-shift allow it to use a large enough search space to keep the tiger's head within bounds, while at the same time focusing particles on the true target and so avoiding distractions. At nine times the area of the target the search area used by SOK is very large, its brute-force nature. KAMS uses more particles than SOK, but manages their spread very effectively.

Figure 4 shows the six algorithms tracking a ball moved by hand against a cluttered background. The hand moves at different velocities, sometimes partially occluding the ball. Condensation and annealed particle filtering both fail around the 5th frame as their particle sets are too dispersed and so attracted to very heavy, and very similarly coloured, clutter. Hybrid suffers the same diffusion problem, around the 55th frame. Its tight focus on the target allows the kernel mean shift to track until the 58th frame, when high target velocity throws it off. It does, however, regain the target around the 118th frame as the hand moves, by chance, underneath the wandering tracker. SOK's dominant mean-shift tracker guides it safely through the clutter around frame 5, but it also loses track around the 60th frame. High velocity disables the mean shift component and the particle set is too widely spread to avoid clutter. Again SOK reacquires the target by chance around frame 114. KAMS tracks the ball successfully throughout the sequence. Moreover, while the other particle-based algorithms failed using 200 particles, KAMS still succeeded when its particle set was reduced to only 50 particles.

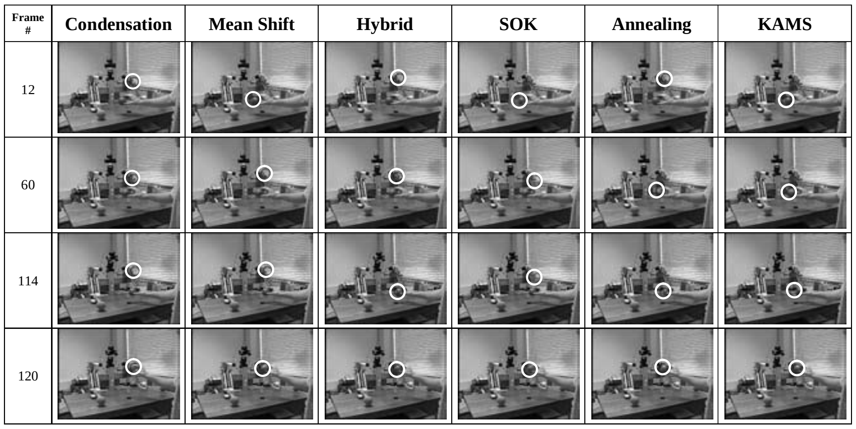

Figure 4: Six algorithms track a hand-held ball. See supplementary material.

To illustrate the key feature of the algorithm, figure 5 shows the particles generated during a single annealing run in KAMS. Each row shows the two particle sets created for a single value of *M*. The left image shows the particles after addition of Gaussian noise, the right after application of Kernel Mean Shift. Note the alternating expansion and contraction of the particle set. Note also that mean shift creates near-constant patterns of particles after only the second annealing step. Space restrictions prevent a detailed examination of the effect of varying *M*, but experience suggests that KAMS will require fewer annealing levels than pure annealed particle filtering.

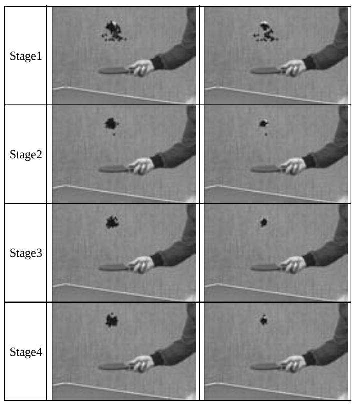

Figure 5. Dispersal and clustering of particles during an annealing run in KAMS.

Figure 6 shows a basketball player faking a pass and then passing the ball quickly from under his legs. Condensation, Mean Shift, Hybrid, SOK and pure annealing all fail around the 5th frame due to the player's high-speed and deliberately evasive movement of the ball. KAMS tracks successfully until frame 23, when the ball is totally occluded by the player's legs for 2-3 frames. Some particles briefly recquire the ball but, in the absence of a motion model, most are thrown off and the tracker loses its target.

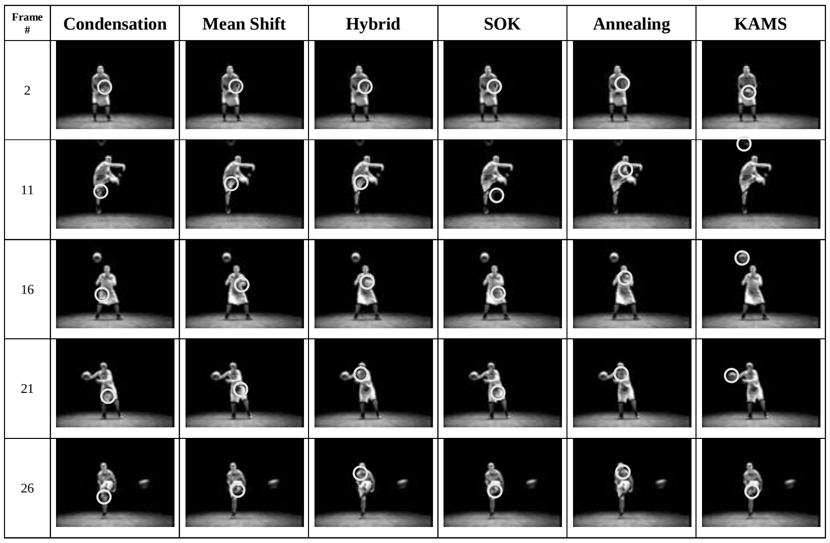

Figure 6: Six algorithms track a deliberately evasive basketball. See supplementary material.

3.2 Accuracy

Artificial sequences showing a multicolored circular target moving across a static background allow the trackers' positional estimates to be compared to ground truth in the presence of controlled amounts of measurement noise and clutter. Noise is simulated by perturbing the target's position in each frame with additive Gaussian noise. Clutter is added by randomly placing a user-defined number of similar circular objects on the otherwise white background (Figure 7). These distracting objects introduce local maxima into the evaluation function, while increased measurement noise raises the likelihood that a given tracker will come into contact with those maxima. All the artificial sequences used here consist of 140 (320x240 pixel) frames.

Only the three hybrid algorithms were included in this experiment. Hybrid completed the sequence of figure 8a with a mean error of 6.32 pixels, but failed around frame 20 when noise and clutter increased. SOK completed figures 8a. and b. with mean errors of 5.66 and 9.60 pixels, but failed thereafter. Only KAMS managed to track through all 4 sequences, with mean errors of 6.94, 18.17, 14.72 and 17.30 pixels. While the algorithms produced similar levels of accuracy (where comparable data is available), KAMS is noticeably more robust. Note also that Hybrid used 100 particles and KAMS only 40. KAMS manages its particles more efficiently and so needs significantly fewer.

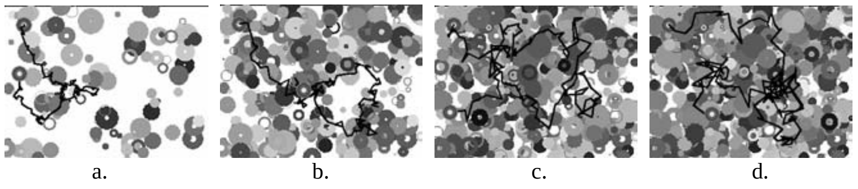

Figure 7. Artificial test sequences, a. σ = 4 with 100 background objects, b. σ = 8 with 300 objects, c. σ = 12 with 500 objects (See supplementary material), and d. σ = 14 with 600 objects. Black lines show target path, with the target displayed at either end.

4 Conclusion

Hybrid particle filter/mean shift tracking algorithms have been shown to have performance and computational advantages over their component parts. We believe the key feature of hybrid trackers to be their exploitation of the natural tension between the particle dispersal caused by the process noise of the particle filter and the clustering performed by Kernel Mean Shift. We suggest that this tension provides opportunities to better manage the spread of particles across the search space, providing higher performance with fewer particles. To test our hypothesis we have proposed a novel hybrid tracker (KAMS) that combines kernel mean-shift [6] with the annealed particle filter [2], allowing us to emphasise the iterative particle dispersal/clustering structure. The accuracies achieved by the various hybrid algorithms are comparable. The proposed algorithm, however, is significantly more robust than both previous hybrids tested and requires noticeably fewer particles than Maggio and Caravello's [5] algorithm.

References

[1] M. Isard and A. Blake, CONDENSATION – conditional density propagation for visual tracking, International. Journal of Computer Vision, 29(1) pp5-28, 1998.

[2] J. Deutscher, A. Blake, and I. Reid, Articulated body motion capture by annealed particle filtering, Proc. IEEE Conf. Computer Vision Pattern Recognition, 2000.

[3] J. Vermaak, A. Doucet and P.Perez, Maintaining multi-modality through mixture tracking, Proc. ICCV, pp 1110-1116, 2003.

[4] C. Chang and R. Ansari. Kernel particle filter for visual tracking, IEEE Signal Processing Letters, 12(3), pp. 242–245, 2005.

[5] E. Maggio and A. Cavallaro, Hybrid particle filter and Mean Shift tracker with adaptive transition model, Proc. Int. Conf. Acoustics, Speech, and Signal Processing 2005.

[6] D. Comaniciu, V. Ramesh, and P. Meer, Kernel-based object tracking, IEEE Trans. Pattern Analysis and Machine Intelligence, 25(5), pp. 564–577, 2003.

[7] A.P. Leung and S. Gong, Mean shift tracking with random sampling, Proc. BMVC 2005, pp.729-738, 2006.

[8] A. Naeem, S. Mills, and T. Pridmore, Structured Combination of Particle Filter and Kernel Mean-Shift Tracking, Proc. Int. Conf. Image and Vision Computing New Zealand, 2006.

[9] K. Deguchi, O. Kawanaka and T. Okatani, Object tracking by the mean-shift of regional colour distribution combined with the particle-filter algorithm, Proc. ICPR 2004, pp506-509, 2004.

[10] A Clark and C. Clark, Performance Characterisation in Computer Vision – A Tutorial, http://peipa.essex.ac.uk/benchmark/tutorials/essex/tutorial.pdf.