Using Object Interactions to Improve Particle Filter Performance

Joe Marshall, Steve Mills, Steve Benford,
Mixed Reality Lab, University of Nottingham, NG8 1BB,UK
{jqm,smx,sdb}@cs.nott.ac.uk

Abstract

This paper describes and evaluates a novel set of approaches to handle situations where multiple distinct and visually differing objects are tracked, such as tracking of people and objects they are manipulating. Unlike tracking of multiple similar objects, visually different interacting objects can provide an opportunity to improve the tracking accuracy. These approaches are designed for use with Condensation/Particle Filter based algorithms, and allow drop-in replacement of tracker modules for each object type tracked. They use information about the relationships and interactions between objects to improve the tracking, rather than in order to distinguish between the objects, as in current algorithms. They are also designed to be highly efficient, for real time use. The approaches are tested on a challenging set of real data and achieve tracking performance similar to using a single very high dimensional tracker, but with vastly reduced complexity and hence much better time performance.

1 Introduction

The Condensation[4] algorithm is commonly used for single object tracking in video sequences. There are many generalizations of the algorithm to cope with the tracking of multiple similar objects[3, 5, 6]. These attempt to disambiguate the multiple objects, to allow association between each particular object over multiple frame sequences and to maintain tracking of the full set of objects. This paper addresses the class of situations where multiple distinct and visually differing objects are tracked at the same time, such as players balls and rackets etc. in games, or tracking of people manipulating objects. In these situations, interaction between objects, rather than being a problem, creates an opportunity to improve the tracking accuracy for the individual objects.

There are two main ways of extending single object tracking to tracking multiple objects, the first is to use a single object tracker which tracks over a multi-dimensional space representing the combined state of all objects tracked. This approach has serious performance problems, due to the high dimensionality of the tracking space. Ways to overcome this have been suggested[2], but these are still complex and performance intensive.

The second method of tracking multiple objects is to use a single tracker per object, and to add some way of taking the dependencies between the objects into account. This paper investigates various methods for taking these dependencies into account, without significantly increasing the complexity of the multiple object tracking process, in order to develop real-time suitable tracking methods for tracking multiple interacting objects. It

BMVC 2006 doi:10.5244/C.20.35

is possible to make use of the difference between objects in a way that current multiple tracking algorithms are not able to, thus allowing for a more efficient and accurate tracking process.

1.1 Existing Work

The Condensation algorithm is commonly used to detect object position from video data[4]. It uses a large number of particles each storing a single hypothesis about the possible state of the object, and an observation of the input data.

The model works in four stages. Firstly, each particle is scored as to how well it fits the observation data. From this set of scored particles, a new set of particles with the same number of elements as the current set is generated. This is generated by weighted random sampling, based on the score of each particle, so some high scoring hypotheses may be chosen multiple times and some hypotheses may not be propagated into the new set at all. After this sampling stage, the new set of items is subjected to drift, which uses a motion model to predict how this particle will have moved since the last frame, and diffusion, which adds a certain amount of randomness. This predicts a new position for the particle. Finally, in the measurement phase, each particle in the new set is scored against the video frame. An estimate of the current position of the object is derived from this set, usually by using a weighted mean.

There are several extensions of Condensation to multiple tracking of similar objects which effectively use a single tracker per object tracked. Khan et al.[5] use Markov Random Fields to model the interaction between several ants they are tracking. At points when the ants are far away from each other, their method works as a set of standard particle filters. Similarly, various methods, eg.[3, 9, 10], use one multi-modal particle distribution for all objects. These essentially try to solve the data association problem of how to assign observations to individual objects and to maintain the modality of the particle distribution when objects interact. These methods are not generally useful when objects are visually distinct, as the association problem does not exist.

Han et al.[2] represent the state of two tracked objects within one high dimensional particle, but store Gaussians representing the modes in the distribution, rather than individual point particles. This reduces the number of particles required for tracking in high dimensional spaces. However, while this increases efficiency, it is still exponential in the number of objects, and only increases the number of dimensions it is possible to track, rather than avoiding the underlying problems of using a high dimensional state to track multiple objects. This means it still requires large numbers of particles in order to track complex systems with more than two interacting objects. It is also inefficient at points when objects are not currently interacting, and could be tracked equally well individually. Wu et al.'s co-inference tracking[11] uses multiple cues in tracking a single object, however this does not simply extend to use multiple different objects as cues, as the cues do not always reinforce each other and the method fails.

1.2 The Example Tracking System

Situations where this kind of tracking occurs include tracking players and balls in sport[12], and tracking people manipulating objects during stroke rehabilitation[1]. A system tracking a person juggling is used here as an algorithm test-bed. Juggling, provides a challeng-

ing tracking situation, with a lot of interaction between the fast moving balls and hands, but is easy to set up and control. This uses three different trackers, all of which work based on the detection of particular colours and shapes in the image. The face tracker keeps track of the position of the persons face in order to determine where the centre-line between the two shoulders is located. The arm tracker detects two arms, on either side of this line. Finally the ball tracker detects the number, positions and velocity of multiple balls and tries to detect whether they are currently held by the juggler, or are in the air.

2 The Problem

It would be ideal to use a single Condensation tracker in which at time t each particle holds the combined state of a set of K objects $Object_{1..K}$. This tracker uses a scoring function to calculate a probability for a combined state $X_t = \{X_t^1, X_t^2 ..., X_t^K\}$, based on an observation Z_t, i.e. $P(X_t^1 \cap X_t^2 ... \cap X_t^K | Z_t)$.

It is desirable to approximate this single tracker with a set of individual trackers, with state vectors $X^1, ..., X^K$. The state space for each tracker will have a significantly lower number of dimensions than the single tracker and hence will require smaller numbers of particles to achieve the same tracking performance. However, the objects are not independent, so using completely independent trackers, without tracking the interactions between the objects, would fail to exploit the available information. Given this fact, the probabilities being calculated for tracker k must approximate $P\left(X_t^k | X_t^1 \cap X_t^2 ... X_t^{k-1} \cap X_t^{k+1} .. X_t^K, Z_t\right)$. It is not possible to directly calculate these conditional probabilities for each particle, as this would involve effectively calculating a high-dimensional tracker.

2.1 The Proposed Solution

The interactions affecting object X^1, based on $X^{2..K}$ are considered here, as the algorithm is identical for other objects.

2.1.1 Definitions

$\underline{X}_t^{1..K}$	This is a set of random variables that represent the part of the X^k which has an effect on the other X states. They are based on the value of each of the X states from the previous frame by using the prediction function inherent in the particle filter. $\underline{X}_t^k$ is only dependent on X_{t-1}^k, and is assumed conditionally independent of the observation and the other $\underline{X}$ variables. They are parameterized in a multi-dimensional space, which is discrete, or may be discretised.
$f_{1..K}(x)$	This is the probability density function of $\underline{X}_t^k \| X_{t-1}^k$, and must be computationally simple to calculate given a particular X^k state.
$E^k(X^k)$	This is the region of the $\underline{X}^k$ space where a given X^k has an effect. i.e. $x \notin E^k(X^k) \Longrightarrow P(\underline{X}^k = x \| X^k) = 0$, for a given X^k
$R^{1,k}(X^1)$	$P(X^1 \| \underline{X}^k)$ is dependent on only the region of $\underline{X}^k$ space defined by this function, i.e. for any given X^1, if $x \notin R^{1,k}(X^1)$ then $P(X^1 \| \underline{X}^k = x) = P(X^1 \| X^1 \notin R^{1,k}(X^1))$, which is constant for that X^1
T^k	This is an overall region containing all points where $P(\underline{X}_t^k \| X^k)$ may be non zero for all X^k. i.e. it is the union of all possible E^k regions.

2.1.2 Algorithm

Taking the ideal tracker, with particle score equal to $P(X_t^1 \cap X_t^2 ... \cap X_t^K | Z_t) = P(X^2 ... \cap X^K | Z) P(X_t^1 | X_t^2 ... \cap X_t^K, Z_t)$, it is assumed that the states of the variables $X^{2..K}$ for the previous frame are known, and the score $P(X_t^1 | X^2 ... \cap X^K, Z)$ is to be calculated. This score can be rewritten in terms of $\underline{X}_t^{2..K}$, and as these are independent, in terms of $f_{1..K}$:

$$P(X_t^1 | X_t^2 ... \cap X_t^K, Z) \approx \left[\sum_{\{x_2 \in T_2, ... x_K \in T_K\}} \left(f_2(x_2) ... f_K(x_K) P(X_t^1 | \underline{X}_t^2 = x_2 ... \cap \underline{X}_t^K = x_K, Z) \right) \right]. \tag{1}$$

For two object interaction, this calculation is simple enough, however, for multiple object interaction, further simplification is required to avoid the combinatorial nature of this equation. An assumption is made that $P(X^1 | X_t^2, ... X_t^K)$ can be aproximated using a function of the pairwise interactions. For example, in the test tracker, the interaction function:

$$P(A | F, B_1, ..., B_C, Z) \approx [P(A | F, Z)] \left[\sum_{1..C} \frac{P(A | B_c, Z)}{C} \right], \tag{2}$$

where A=arm position,F=face position,B_c=cth ball position(of C balls), is used to score the arm. To use this approximation, Equation 1 is calculated for each two object case included in the equation. This approximation in terms of pairwise interactions has been shown to be relatively efficient in testing. For the rest of this section only pairwise interactions are considered, between X^1 and X^2.

By definition, $P(X^1 | \underline{X}^2)$ only needs to be evaluated within $R^{1,2}$ so

$$P(X^1 | X^2, Z) \approx \sum_{x_2 \in R^{1,2}} \left(f_2(x_2) P(X^1 | \underline{X}^2 = x_2, Z) \right) + \sum_{x_2 \notin R^{1,2}} \left(f_2(x_2) P(X^1 | \underline{X}^2 \notin R^{1,2}, Z) \right). \tag{3}$$

As $P(X^1 | \underline{X}^2 \notin R^{1,2}, Z)$ is not dependent on x_2, and thus can be taken outside the sum, and f_2 must sum to one, the dependency on values of f_2 outside of $R^{1,2}$ may be removed, giving

$$P(X^1 | X^2, Z) \approx \sum_{x_2 \in R^{1,2}} \left(f_2(x_2) P(X^1 | \underline{X}^2 = x_2, Z) \right) + P(X^1 | \underline{X}^2 \notin R^{1,2}, Z) \left(1 - \sum_{x_2 \in R^{1,2}} f_2(x_2) \right). \tag{4}$$

This is used as the input to the interaction function. Similar inputs are created for other interactions between the objects. In many cases, certain objects will be independent of others, in which case they will not be in the interaction function, and hence not require calculation. For example, in the test system the face tracker is independent, so does not require any extra inputs to be calculated in its interaction function. However, this does not directly allow the creation of a set of trackers as the $f_{1...K}$ pdfs rely on the state of individual particles, an approximation to the distribution of the $f_{2..K}$ values must be used as input to the X^1 tracker, giving a two step process:

1. Calculate probabilities for all $X^{1..K}$ particles based on the observation data and $f_{1..K}$ generated from the previous frame.
2. Calculate next frame $P(\underline{X}^k | X^k)$ and the pdfs f_k for all particles and construct approximations to the distributions of the pdfs, weighted by particle probability.

2.2 Calculating the Approximate Distribution

It is important that an efficient way of calculating and storing these distributions be used. Otherwise, no efficiency will be gained over the higher dimensional calculation. Three ways of calculating and storing the intermediate distributions of $f_{1..K}$ are described here.

2.2.1 Gaussian Mixtures

This method attempts to approximate the overall state of the $f_{1..K}$ functions by using a set of N Gaussians for each pdf, representing the mean values of $f_{1..K}$ at each position, weighted by the particle score. First the generated probabilities $P(\underline{X}^k|X^k)$ are partitioned spatially, then their contribution to a single Gaussian for each partition can be estimated using an unbiased Maximum Likelihood Estimator. In the example system it is possible to generate these Gaussians directly from the X^k states and $P(X^k|Z)$ values without actually calculating all the f_k values at any point.

Samples from these Gaussians are used as input to the scoring function instead of sampling f_n. The sum of all Gaussians at position x is used as an estimate of the pdf at that position, and hence a pairwise score function is created as:

$$\sum_{x\in R^{1,2}} \left(P(X^1|\underline{X}^2 = x, Z) \sum_{n\in 1..N} [G_n(x)] \right) + P(X^1|\underline{X}^2 \notin R^{1,2}, Z) \left(1 - \sum_{x\in R^{1,2}} \sum_{n\in 1..N} [G_n(x)] \right). \tag{5}$$

The information from the Gaussians as to the distribution means that the calculation of this sum may be simplified to avoid having to calculate a large number of Gaussian values, by either using a single Gaussian sample as an estimate, or using the Gaussian distribution parameters directly. In this way, it is possible to avoid calculating many values from the Gaussians, making this technique relatively efficient.

This technique has advantages, in that the Gaussians are easily calculated and can be used directly in the second stage. However, it reduces the usefulness of the Condensation algorithm itself, as the modality of the data is reduced to the number of Gaussians used in the partitioning stage. It may be possible to detect when this is occurring, by detecting that a Gaussian with a large variance is being created, and thus split a partition automatically, however, this did not seem reliable in testing.

2.2.2 Sampling

For each X^1 state, a state from X^2, or a small set of states is sampled, to use as input to the X^1 state. This is done by using a probabilistic sampling stage similar to the Condensation sampling stage. The test system makes use of the Condensation sampling stage, by sampling uniformly from the distribution created by the sampling stage of the Condensation algorithm, thus avoiding doing two similar sampling stages. This method is appealing in that it uses a method very similar to the Condensation algorithm, and potentially allows for a full representation of the modality of the data. It is also the most simple conceptually of the methods described. A set of 10 states was used for the test version of this, as using higher numbers of states was too slow for real time use.

2.2.3 Mean Probability Density Functions

This combines aspects of the two methods above, to create a technique that allows multi-modality, but also allows all X^2 states to influence the result, unlike the sampling method. Instead of partitioning the particles as in the Gaussian model, a uniform partitioning over a fixed grid is used. At each point, the weighted mean of all the f_2 values is calculated:

$$M[x] = \frac{\sum_{1..count(particles)} P(X^2_{t-1}[particle]|Z_{t-1}) P(\underline{X}^2_t = x|X^2_{t-1}[particle])}{\sum_{1..count(particles)} P(X^2_{t-1}[particle]|Z)}. \quad (6)$$

The X tracker can then use $M[x]$ as input to its scoring function, defined as

$$P(X^1|X^2,Z) \approx \sum_{x \in R^{1,2}} P(X^1|\underline{X}^2 = x, Z) M[x] + P(X^1|\underline{X}^2 \notin R^{1,2}, Z) \left(1 - \sum_{x \in R^{1,2}} M[x]\right). \quad (7)$$

The values of $M[x]$ are the same over all particles, thus given that $P(\underline{X}|X) > 0$ only within a fixed range, all possible values of $M[x]$ can be precalculated. An extra stage in the Condensation algorithm is used to calculate M. This can be done by using a matrix to accumulate probability values for each X particle. For this to work efficiently, $E^2(X^2)$ and $R^{1,2}(X^1)$ must be relatively localised for states X^1 and X^2, so that each particle does not have to add to or read many accumulators. The process for each tracker is now:

1. Calculate tracker, using $M[x]$ matrices in scoring function.
2. Calculate the $M[x]$ matrix based on X^k distribution.

This is slightly less time efficient than using a small number of Gaussian mixtures. However, it can more accurately represent the distribution of the particles, thus making it work much better in situations where tracker distribution is noisy and the tracker is tracking several modes in the observation data.

2.3 Extensions

In the worst case scenario, all K objects will interact with each other. This will require K intermediate representations to be written, one per object, and $K-1$ to be read in each scoring function. There will still be a vastly smaller total number of particles compared to the single tracker, and thus it will be far more efficient, especially for large K. However, in many real cases optimisations will be possible where some objects are not affected by others, such as in our head tracker, which does not take input from any other trackers. It is also useful to use methods such as mixture tracking[10] within individual trackers, where there are multiple objects of one type. If it is unlikely that the multiple similar objects will overlap, or it is not relevant to the other scoring functions, a combined intermediate representation, representing the sum of the object pdfs may be used, rather than using a separate pdf for each object. This is the case in the ball and arm trackers described below.

In the case of juggling, the system is tracking balls, which are relatively predictable except when interacting with the hands, and hands, which are much less predictable. It is useful to reduce the reliance on the next frame predictions as much as possible for the less predictable object. In this case, the output from the less predictable tracker is used directly to create the probability density function, rather than using the next frame prediction. This is likely to be useful in other similar situations where a human interacts with one or more relatively predictable objects.

3 Evaluation

3.1 Implementation Details

Only the mean pdf based tracker is described in depth here, the implementation was similar for other tracking methods. The juggling system makes extensive use of tracker interaction as shown in Figure 1.

Face Tracker This has particles defining face position, size and angle of an ellipse representing the face shape. Each particle's score is added to a pdf for each position in the face ellipse, and normalized to create a representation of the distribution of the likely face position.

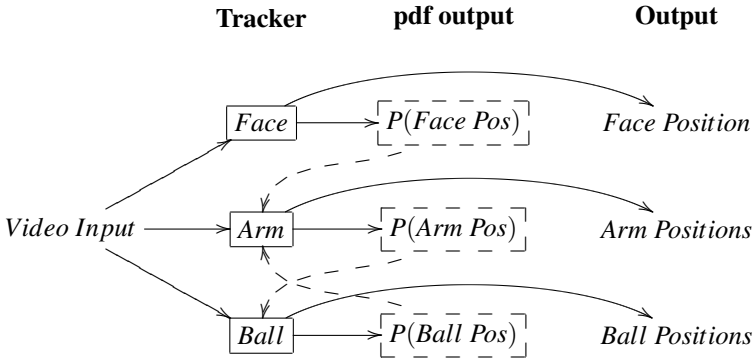
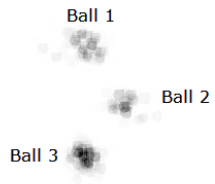


Figure 1: Use of pdfs in the juggling tracker

a)Input Frame



b)Next frame ball pdfs

c)Hand Position pdfs

d)Output - Arm and Ball tracking

Figure 2: The tracker state for a single frame

Arm Tracker When calculating a score for a particular arm position, this tracker disregards areas of skin colour which are part of the face by using the face probability map. Prior to this, the tracker was often misled by the face pixels. The arm tracker also uses a ball position probability map generated by the ball tracker in the previous frame. This is because when a ball is stopped over, or heading towards the arm, it generally defines the position of the hand. The arm tracker outputs a combined pdf for both hands, giving the sum of the probabilities for the left or right hand being centred on a particular position, as in Figure 2c. (annotated to show the modes of the left and right hand).

Ball Tracker This uses the pdfs generated by the arm tracker to detect caught balls. This is both because the 'caught/not caught' status is a desired output, and because caught balls are often partly occluded by the hand, and may otherwise be lost by the ball tracker, or inaccurately tracked. The position and velocity of the ball is used to generate a combined pdf of the position of all balls in the next frame for the arm tracker as in Figure 2b. Multiple balls are detected by use of a segmentation algorithm on the output of the ball tracker, this segmentation is also used to ensure that one ball does not take over all the particles, in similar way to Milstein et al.[8]

3.2 Testing Results

Testing was run using a Pentium 4 2.6GHz test machine. Up to 4000 particles in total were used (3000 ball, 900 arm, 100 face), this number of particles allowed processing of the video at approx 30 frames per second, on all interaction methods.

In order to compare against the 'ideal' tracker (as described in section 2) a single 20 dimensional particle filter was used. This was only able to track 1 ball (plus two arms and a face), due to the vast number of particles required for multiple balls being too large to fit in memory/disk space. This took several minutes per frame to run. A set of independent particle filters, using only observation information were also used.

Three test sequences (available at http://www.mrl.nott.ac.uk/~jqm/juggling/bmvc) were used, firstly a 500 frame sequence (plus 100 frames of initialisation), of 1 ball being thrown from hand to hand, involving various complicated hand movements and different throws, secondly a 1400 frame sequence of mainly 3 ball juggling, with a short section of 2 and 1 ball, and finally a set of 200 frame sequences of juggling up to 4 balls.

In this tracker, the desired output is whether a ball is caught, and its position. When the balls are in flight and not nearing catch or release, all methods provide good tracking performance. In situations where the ball is being caught or held, there were several possible errors that occurred. These are defined as minor or serious, depending on their effect on the data being collected. The tracker is able to automatically reinitialize after an error and detect new objects, (it uses a small number of initialization particles, as in [7]), so no manual reinitialization was required after an error.

Serious Errors A ball is lost and the tracker has to reinitialize. The tracker detects a catch when no catch has occurred. A catch occurs and the tracker doesn't notice it.

Minor Errors A ball is temporarily lost but re-tracked without reinitialization, which may occur in some occlusion situations, and is not generally a problem for the system. An arm fails to be tracked - this may cause other errors to occur, but usually the next time a ball interacts with the missing arm, the tracker recovers.

Tracker Type	Particles	1 ball errors		3 Ball Errors	
		Minor	Serious	Minor	Serious
Ideal	120,000	4	0	n/a	n/a
	1,000,000	2	0	n/a	n/a
Independent	1,000	4	22	11	97
	4,000	1	28	15	56
Gaussian	1,000	5	4	8	13
	4,000	4	2	10	8
Sampling	1,000	8	2	14	11
	4,000	5	1	15	7
Mean pdfs	1,000	2	4	10	7
	4,000	2	0	5	4

Table 1: Comparative Tracker Performance

Using the one ball sequence, it was possible to compare the results from the ideal tracker against the methods described in section 2.2. For the 3 ball sequence, only the relative performance of the different types of tracker interaction could be compared. The testing results are shown in Table 1. In the 1 ball test 1,000,000 'ideal' particles were required in order to achieve the power of 4000 particles using the best performing mean pdf method. The independent trackers were very hard to tune in order to get a balance between false catches and missed catches, hence the large number of serious errors shown, which are mainly catch errors. Early attempts to fix these issues led to the algorithm presented here.

The best performing, pdf based tracker was also tested on the 200 frame sequences. For these sequences a ground truth hand position for one hand was marked and the distance from this point to the tracker hand circle was measured for each frame. Figure 3, shows for each number of balls, the percentage of frames within each error value, and the mean error. This was interesting, as with no balls, the arm tracker was very poor. Once balls were being juggled, the tracking accuracy went up greatly; also slightly improved tracking was observed as higher numbers of balls were juggled. These results demonstrate how the individual trackers are made more robust using feedback between the trackers.

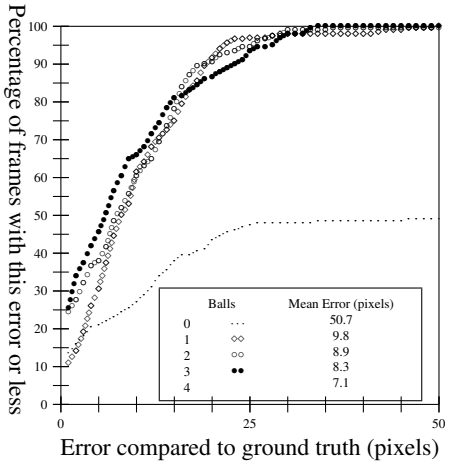


Figure 3: pdf based Arm tracker accuracy for different numbers of balls

4 Conclusions

The results shown in this paper demonstrate that these methods can greatly improve the tracking accuracy of individual object trackers. The final accuracy is similar to that of a single high-dimensional tracker, and far better than independent trackers. The proposed methods do not significantly increase the complexity, and hence perform only slightly slower compared to completely independent trackers, allowing real time use of these methods. This paper shows that where there are multiple visually distinct objects in a scene, this is no data association problem. Rather, there is an opportunity to improve the tracking performance for each individual object, to make it better than that of independent trackers. In addition, the improved tracking due to the interactions allows the use of simpler, faster individual trackers, as demonstrated by the very simple arm tracker, which performs poorly standalone, but has much better performance when interactions are taken into account.

These methods also offer a major advantage over existing algorithms in that they allow the use of individual trackers as black boxes, so that trackers for one class of object can easily be modified without needing to alter other trackers. For example in this system, several different versions of the arm and face trackers have been used, which may just be swapped in, without altering the ball tracker. They are also highly generalisable, and can be used in many situations where multiple trackers are used, simply by altering measurement functions and creating functions to generate the intermediate probabilities.

References

[1] A. Ghali, A. Cunningham, and T. Pridmore. Object and event recognition for stroke rehabilitation. In *VCIP*, 2003.

[2] B. Han, D. Comaniciu, Y. Zhu, and L. Davis. Incremental density approximation and kernel-based Bayesian filtering for object tracking. In *CVPR*, 2004.

[3] C. Hue, J.-P. L. Cadre, and P. Perez. A particle filter to track multiple objects. *IEEE Workshop on Multi-Object Tracking*, pages 61–68, 2001.

[4] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29(1):5–28, 1998.

[5] Z. Khan, T. Balch, and F. Dellaert. MCMC-based particle filtering for tracking a variable number of interacting targets. *PAMI*, 27(11):1805–1918, 2005.

[6] E. B. Koller-Meier and F. Ade. Tracking multiple objects using the Condensation algorithm. *Journal of Robotics and Autonomous Systems*, pages 93–105, 2001.

[7] E. B. Meier and F. Ade. Using the Condensation algorithm to implement tracking for mobile robots. In *Third European Workshop on Advanced Mobile Robots*, 1999.

[8] A. Milstein, J. N. Sánchez, and E. T. Williamson. Robust global localization using clustered particle filtering. In *AAAI/IAAI*, pages 581–586, 2002.

[9] D. Tweed and A. Calway. Tracking many objects using subordinated Condensation. In *BMVC*, pages 283–292, 2002.

[10] J. Vermaak, A. Doucet, and P. Pérez. Maintaining multi-modality through mixture tracking. In *ICCV*, pages 1110–1116, 2003.

[11] Y. Wu and T. S. Huang. A co-inference approach to robust visual tracking. In *ICCV*, 2001.

[12] F. Yan, W. Christmas, and J. Kittler. A tennis ball tracking algorithm for automatic annotation of tennis match. In *BMVC*, 2005.