Decentralised Data Fusion with Particles

Lee-Ling Ong, Ben Upcroft, Matthew Ridley,
Tim Bailey, Salah Sukkarieh and Hugh Durrant-Whyte
ARC Centre of Excellence in Autonomous Systems (CAS),
The University of Sydney, Australia
www.cas.edu.au
s.ong@acfr.usyd.edu.au

Abstract

We aim to solve the problem of consistent Decentralised Data Fusion (DDF) with particle filters by a transformation of the sample statistics to a different representation that maintains an accurate summary of the particles. Two methodologies are proposed. The first method is a transformation of the particle representation to a Gaussian Mixture Model (GMM). The second algorithm approximates the particles by a Parzen representation. The two algorithms proposed differ in the accuracy of representing the particles as well as the accuracy of fusion methods and the bandwidth requirements. Our simulations results show that a transformation to GMMs requires less components and provides a more accurate summary compared to Parzen representations. However, the decentralised fusion solution for Parzen representations is more accurate than the solution for GMMs.

1 Introduction

This paper describes two methodologies for performing decentralised particle filtering conservatively in sensor networks. Particle filters are widely used for non-linear, non-Gaussian target tracking yet its application in multi-platform target tracking is limited.

A decentralised sensor network usually comprises of processing nodes each incorporating a sensor and communications capabilities. Each node runs its own local filter and communicates information to other nodes in the neighbourhood. Incoming information is fused with the local state to produce a global state of the world. There are three basic constraints to a decentralised data fusion (DDF) system [Grime, 1992] which are:

1. There is no single central fusion centre exists and no node is central to the operation of the network.
2. Communications are kept on a strictly node-to-node basis
3. There is no global knowledge of the network topology

Practical applications of DDF have focused on representing features with Gaussian noise and through the use of ranging devices such as laser and sonar. While such techniques have been successfully used in autonomous air, ground, and underwater vehicles, constructing accurate models of unstructured and complex environments is difficult.

However, our application aims to demonstrate DDF techniques for general non-Gaussian, non-point feature information. Such information includes observations of natural features and targets from both imaging and range sensors on flight and ground-based platforms such as in Figures 1 and 2.

Figure 1: Flight platforms

Vision sensors return rich feature information such as colour, texture and reflectivity. This information is difficult to model as the appropriate observation models chosen are non-Gaussian. Hence, general probabilistic representations and general filtering techniques such as particle filters must be considered. Other non-geometric probabilistic representations also include Gaussian Mixture Models (GMMs) [Alspach and Sorenson, 1972],

Figure 2: A ground platform

Parzen density estimates [Parzen, 1962] and grid based techniques [Stone *et al.*, 1999].

Although particle filters can represent arbitrary distributions there is no known method for consistent decentralised data fusion on particles directly as yet. As particles are discrete representations, samples from one set do not have the same support on the space as samples from another set as shown in Figure 3. In order to meet general DDF architecture constraints, particle representations require a transformation into a consistent representation for communication and fusion.

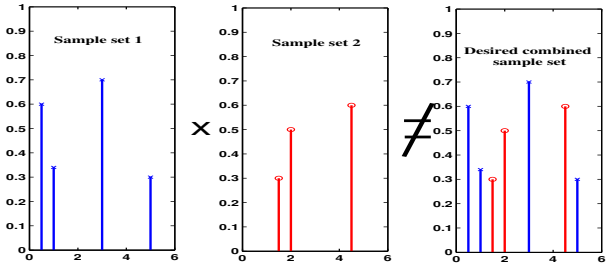

Figure 3: Samples from one particle set do not have the same support on the space as samples from another set

Among the different representations that particles can be converted to include grid-based techniques [Rosencrantz *et al.*, 2003a], GMMs and Parzen representations. Grid based representations are not compact and do not scale well with dimension and hence will not be considered. GMMs and Parzen representations on the other hand, are more compact than particles requiring less communication bandwidth, meet the constraints in a DDF system for fusion and can be converted back to particles after the fusion process. These two representations are considered in this paper.

Our simulation results show that less GMM components are required to summarise the sample statistics compared to Parzen components. Additionally, the accuracy of the approximation by GMMs is better than Parzen representations. However, the fusion algorithm for Parzen density estimates is more accurate than the algorithm for GMMs. This offsets the benefits of using GMMs.

The paper is organised as follows: After presenting some related work (Section 2), an introduction to particle filters is presented in Section 3. A generalised DDF node is described in Section 4, showing how common information is maintained in a node. Section 5 introduces the two methods of performing consistent fusion on Particle Filters. Section 6 shows some simulation results and section 7 concludes and presents future directions.

2 Related Work

Since the seminal paper by Gordon *et al.* [Gordon *et al.*, 1993], particle filters have been used widely especially in a centralised fashion [Arulampalam *et al.*, 2002],[Doucet *et al.*, 2001].

Coates [Coates, 2004] introduced distributive particle filter algorithms which strived to maintain a common or centralised particle representation of the posterior distribution at multiple nodes in the network. Sheng *et al.* [Sheng *et al.*, 2005] proposed a technique of using distributed particle filters where the particles were approximated with Gaussian Mixture model(GMMs) via an EM algorithm. However, this algorithm is very computationally expensive and requires a large sample set. The GMMs are then communicated to obtain a common or centralised particle representation.

However, the particle filters we describe in this paper differ in purpose and implementation from Coates [Coates, 2004] and Sheng *et al.* [Sheng *et al.*, 2005]. The key difference is that we seek to use decentralised particle filters that satisfy DDF constraints. Rosencrantz *et al.* [Rosencrantz *et al.*, 2003] and Ihler *et al.* [Ihler *et al.*, 2004] demonstrated DDF using particle representations but the guarantee for conservative fusion updates is not considered. Rosencrantz *et al.* decentralised a standard particle filter by communicating and fusing the most informative subsets of samples. Ihler *et al.* on the other hand, introduced an approximate communication algorithm known as non-parametric belief propagation for non-parametric distributions.

3 Particle Filters

Particle filters are a Monte Carlo estimation method based on importance sampling, adapted to sequential filtering for dynamic systems. At a given moment in time k, a particle filter represents the probability distribution of the state as a set of weighted samples $\{\mathbf{x}_k^{(i)}, w_k^{(i)}\}_{i=1}^N$, such that the density is approximated by an empirical

estimate,

$$P\left(\mathbf{x}_k|\mathbf{Z}^k\right) \approx \sum_{i=1}^{N} w_k^{(i)} \delta\left(\mathbf{x}_k^{(i)}\right) \quad (1)$$

where $\delta(\cdot)$ is the Dirac delta function. The basic operation of a particle filter, as described in Gordon *et al.* [Gordon *et al.*, 1993], is to recursively estimate the posterior distribution at the next time-step via a sequence of sampling, importance weighting and resampling. The first step is to draw samples from a proposal distribution, which is simply the transition density $P(\mathbf{x}_{k+1}|\mathbf{x}_k)$.

$$\mathbf{x}_{k+1}^{(i)} \sim P\left(\mathbf{x}_{k+1}|\mathbf{x}_k^{(i)}\right) \quad (2)$$

The next step is to assign an importance weight to each sample so that the weighted sample set represents the posterior $P\left(\mathbf{x}_{k+1}|\mathbf{Z}^{k+1}\right)$. For the case where the proposal density is $P(\mathbf{x}_{k+1}|\mathbf{x}_k)$, the importance function is equal to the likelihood function $P(\mathbf{z}_{k+1}|\mathbf{x}_{k+1})$, and samples are weighted as

$$w_{k+1}^{(i)} = P\left(\mathbf{z}_{k+1}|\mathbf{x}_{k+1}^{(i)}\right) w_k^{(i)} \quad (3)$$

These weights are then normalised such that $\sum_{i=1}^{N} w_{k+1}^{(i)} = 1$. The third step is resampling. Resampling need not be carried out at every iteration, but is necessary at regular intervals as the sample weights quickly diverge such that many samples have negligible influence on the density estimate. A resampled particle set is obtained by sampling with replacement from the original set in proportion to the weights $w_{k+1}^{(i)}$.

$$\tilde{\mathbf{x}}_{k+1}^{(j)} \sim \{\mathbf{x}_{k+1}^{(i)}, w_{k+1}^{(i)}\}_{i=1}^{N} \quad (4)$$

The resulting sample set has all samples equally weighted, such that $w_{k+1}^{(j)} = 1/N$.

The basic particle filter has two key weaknesses which hinder efficient application to many estimation problems [Gordon *et al.*, 1993; Carpenter *et al.*, 1999]. The first is sample impoverishment, wherein, during resampling, certain particles are selected multiple times and others not at all, thereby reducing the total number of independent samples. In the worst case, for a system with no process noise, the filter will rapidly degenerate to having N copies of a single sample. Solutions to this problem include the auxiliary particle filter, MCMC move steps, and regularisation (see [Arulampalam *et al.*, 2002; Doucet *et al.*, 2001] for details). The second weakness is an inability to adequately explore the state-space if the support of the prior distribution has little overlap with the likelihood function. This problem can occur if a measurement is an outlier, if the likelihood function is highly peaked, or if the process noise is small. One solution is to fit kernels or mixture models to the samples [Musso *et al.*, 2001], which, being a form of regularisation, also addresses impoverishment. Mixture models and kernels are used in this paper as they additionally provide a means to perform DDF.

4 Decentralised Node Structure

The operations in a decentralised node is illustrated in Figure 4. In a DDF system, each sensor node processes

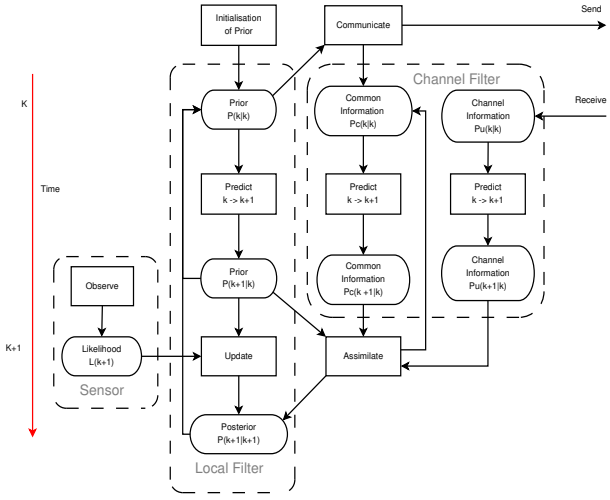

Figure 4: Generalised Decentralised Node operations

raw sensory data to generate a likelihood. This likelihood is updated in the particle filter. The local particle set is then transformed into a more compact representation and communicated to the other nodes in the network via the channel filters [Grime, 1992].

4.1 Channel Filters

Channel filters are used for maintaining an estimate of common information passed between two nodes [Grime, 1992]. The removal of common information between the communicated and local estimate is essential in order to avoid over-confident estimates due to "double counting". A channel filter maintains the common information $P\left(\mathbf{x}_k|\mathbf{Z}_a^k \bigcap \mathbf{Z}_b^k\right)$ between two nodes a and b. Should node b transmit its new state to node a then the Bayesian channel update [Bar-Shalom, 1990] at node a is:

$$P\left(\mathbf{x}_k|\mathbf{Z}^k\right) \propto \frac{P\left(\mathbf{x}_k|\mathbf{Z}_a^k\right) P\left(\mathbf{x}_k|\mathbf{Z}_b^k\right)}{P\left(\mathbf{x}_k|\mathbf{Z}_a^k \bigcap \mathbf{Z}_b^k\right)} \quad (5)$$

where $P\left(\mathbf{x}_k|\mathbf{Z}^{k-1}\right)$ is the prior belief of the state, $P\left(\mathbf{z}_k|\mathbf{x}^k\right)$ is the probabilistic method for combining ob-

servations $\mathbf{Z}^k$ of a state x_k, at time t_k and $P\left(\mathbf{z}_k|\mathbf{Z}^{k-1}\right)$ is the posterior distribution.

Equation 5 illustrates that a division operation is required in a channel update for removal of the common information held between communicating nodes. This division is the main problem encountered in generalised DDF.

5 Decentralised Particle Fusion Algorithms

Particles from one sample set in a local filter do not have the same support on the space as samples from a neighouring node as shown in Figure 3. Hence, fusion of these two particles sets cannot be performed directly. At least one set of particles has to be transformed to a continuous distribution to be sampled by the second set to obtain the new importance weights.

5.1 Conversion to a continuous distribution

The conversion to a continuous distribution is based on Musso [Musso *et al.*, 2001] where each sample is converted to a kernel $K_h(\mathbf{x})$:

$$K_h(\mathbf{x}) = h^D K(\mathbf{x}) \tag{6}$$

where D is the number of dimensions, $K(.)$ is the rescaled kernel density and $h > 0$ is the window or scaling parameter. The kernel selected is Gaussian and hence h is:

$$h = (\frac{4}{D+2})^e N^{-e} \tag{7}$$

where $e = \frac{1}{D+4}$, and N is the number of samples.

Communicating the continuous distribution in this form would be slightly worse than communicating the sample set itself. Hence, approximating this distribution by a more compact one is more desirable. Compact distributions approximated by GMMs and Parzen density estimates have the capacity for consistent fusion.

5.2 Algorithm 1 : GMM approximation and Fusion process

A Gaussian mixture model for a random variable $\mathbf{X}$ is:

$$P(x) = \sum_{i=1}^{n} \gamma_i G_i(x; \mu_i, \Sigma_i) \tag{8}$$

where x is in the domain of $\mathbf{X}$, G_i, is the ith Gaussian component, and γ_i are the weights where $\sum_{i=1}^{n} \gamma_i = 1$. The multivariate Gaussian distribution of the state $\mathbf{x}$ with mean μ and covariance $\mathbf{\Sigma}$ is defined as:

$$P(x) = \frac{1}{(2\pi)^{n/2}|\mathbf{\Sigma}|^{1/2}} \exp^{-\frac{1}{2}[\mathbf{X}-\mu]^T \mathbf{\Sigma}^{-1}[\mathbf{X}-\mu]} \tag{9}$$

Converting particles to GMMs

Two methods considered for reducing the sample set to the more compact GMM distribution are:

- via an Expected Maximisation (EM) algorithm [Dempster *et al.*, 1977] or
- via a joining algorithm [Salmond *et al.*, 1989]

The EM algorithm finds the maximum likelihood parameter estimation in statistical models with variables that are not observed, given initial parameters. For GMMs, the unobserved variables are the underlying mixture components and the observed variables are the data points or the sample set. The convergence of this algorithm can be very slow if the initial parameters are particularly bad compared to the true values and is susceptible to local minima. The Xmeans algorithm [Pelleg and Moore, 2000], based on k-means clustering [MacKay, 2003] is used to obtain reasonable initial parameters to reduce the number of iterations required for convergence.

The computational complexity for Xmeans is $O(i \times N \log k_{max})$ where k is the number of cluster means, N is the number of samples and i is the number of iterations. The computational complexity for the EM algorithm for GMMs is $O(i \times ND^2)$ where D is the dimensionality of the state.

The operation of Salmond's joining algorithm is to merge pairs of components in the Gaussian sum (from Section 5.1), successively until the desired level of reduction has been achieved. The distance measure utilized to gauge the similarity of component i and component j of the GMM is a Mahalanobis-type distance measure:

$$d_{ij}^2 = \frac{\gamma_i \gamma_j}{\gamma_i + \gamma_j}(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{P}^{-1}(\mathbf{x}_i - \mathbf{x}_j) \tag{10}$$

where $\mathbf{x}$ is the state vector of the component, $\mathbf{P}$ is the mixture covariance matrix and γ is the component weight. The components in the model would be joined until a set number of components has been reached. The computational complexity of Salmond's joining algorithm is $O(N \log N)$ where N is the number of components, hence much less computationally demanding compared to EM.

We find that the EM algorithm requires a large set of at least 2000 particles to perform adequately whereas Salmond's joining algorithm can be applied regardless of the number of particles. As we also find that EM is considerably more computationally expensive compared to the joining algorithm, we chose to implement the latter to reduce the Gaussian sum to a more compact distribution.

Fusion and Removal of Common information

Fusion is performed by first converting the local particle set to a GMM. A generalised covariance intersect (CI)

update [Upcroft *et al.*, 2005] is then performed with the GMM transformed from the local particles and the communicated GMM to ensure conservative fusion of possibly correlated information. Illustrated in Equations 11,12 and 13 are the CI operations where Σ_{ij},μ_{ij} and γ_{ij} are the new covariance, mean and weight of the fusion between the ith component of the local estimate and the jth component of the communicated estimate. A CI weighting parameter ω is selected to minimise the determinant of the result.

$$\Sigma_{ij}^{-1} = \omega\Sigma_i^{-1} + (1-\omega)\Sigma_j^{-1} \quad (11)$$

$$\mu_{ij} = \Sigma_{ij}(\omega\Sigma_i^{-1}\mu_i + (1-\omega)\Sigma_j^{-1}\mu_j) \quad (12)$$

$$\gamma_{ij} = \omega\gamma_i + (1-\omega)\gamma_j \quad (13)$$

5.3 Algorithm 2 : Approximation via Parzen denstiy estimates for fusion

For Parzen density estimates, any type of kernel may be used to represent a probability distribution. However, Gaussian kernels are preferred, as most of its operations are closed form and therefore efficient. The Parzen density estimator is similar to a GMM except each component has the same covariance. The equation for a Parzen density estimate with a Gaussian kernel is similar to the mixture of Gaussians which is:

$$P(x) = \sum_{i=1}^{n} \gamma_i G(x; \mu_i, \Sigma) \quad (14)$$

where $G(x)$ is the Gaussian probability density on $\mathbf{x}$ and γ_i are the weights where $\sum_{i=1}^{n} \gamma_i = 1$.

Converting particles to Parzen density estimates
A variant of Salmond's joining algorithm [Salmond *et al.*, 1989], can be applied to the sums of Gaussian kernels (Section 5.1) to reduce the number of components. The difference between this joining algorithm and the joining algorithm for GMMs is that the kernel covariance is adjusted such that the ensemble covariance is preserved after the number of components have been reduced.

Fusion and Removal of Common information
As the particle set is summarised and communicated as a Parzen density estimate, a Bayesian channel update operation (a division operation) can be performed at the receiving node to remove common information. The Parzen density estimate division is shown and numerically justified by Ridley *et al.* [Ridley *et al.*, 2004].

In this operation, each Parzen component of the communicated estimate is divided by the same kernel. This kernel is an approximation of the previously communicated estimate. The result of this division is then updated or fused with the local estimate by sampling with the local sample set to obtain the importance weights for resampling.

6 Simulation Results

A decentralised bearing-only tracking example is simulated. In this example, a feature exhibiting random walk behaviour within the x-y plane is tracked by two stationary sensors as shown in Figure 5.

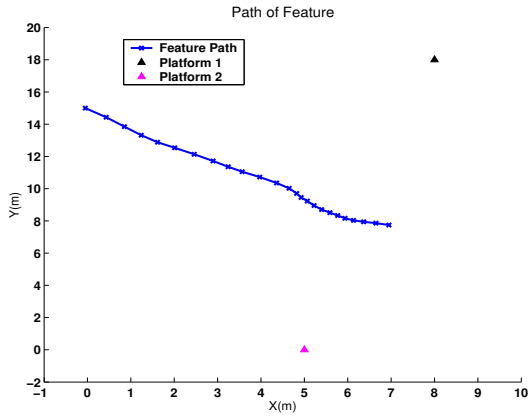

Figure 5: Path of the feature

6.1 Process Model

The feature is modelled based on the Integrated Ornstein-Uhlenbeck process [Stone *et al.*, 1999] which allows for bounding the Brownian velocity over time. This prevents excessively large velocities that can occur due to wayward measurements when the feature is not observed in an extended period. The process model for the state is:

$$\mathbf{x}_k = \mathbf{F}\hat{\mathbf{x}}(k-1|k-1) + \mathbf{G}\mathbf{Q}_{(k,k-1)} \quad (15)$$

where

$$\mathbf{x}_k = [x_k, \dot{x}_k, y_k, \dot{y}_k]^T \quad (16)$$

The state transition matrix for this system is given by:

$$\mathbf{F} = \begin{bmatrix} 1 & \Delta T & 0 & 0 \\ 0 & F_v & 0 & 0 \\ 0 & 0 & 1 & \Delta T \\ 0 & 0 & 0 & F_v \end{bmatrix} \quad (17)$$

where

$$F_v = e^{-\Delta T\gamma} \quad (18)$$

The process model for the covariances is:

$$\mathbf{P}_k = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^T + \mathbf{G}_k\mathbf{Q}\mathbf{G}_k^T \quad (19)$$

where

$$\mathbf{Q}_{(k,k-1)} = \begin{bmatrix} q_x & 0 \\ 0 & q_y \end{bmatrix} \quad (20)$$

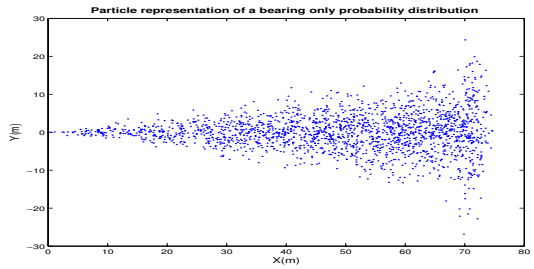

Figure 6: Particle representation of a bearing-only probability distribution

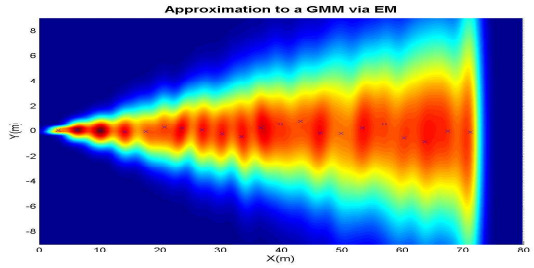

Figure 7: Approximation of the particle representation to GMMs via EM

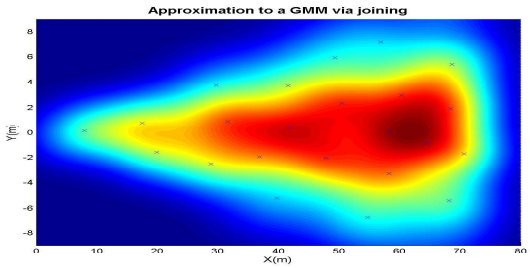

Figure 8: Approximation of the particle representation to GMMs via joining

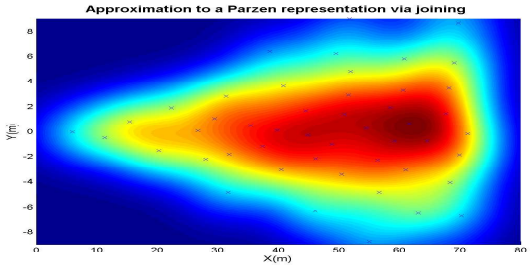

Figure 9: Approximation of the particle representation to a Parzen one

and

$$\mathbf{G}_k = \begin{bmatrix} \frac{\Delta T^2}{2} & 0 \\ \frac{(1-F_v)}{\gamma} & 0 \\ 0 & \frac{\Delta T^2}{2} \\ 0 & \frac{(1-F_v)}{\gamma} \end{bmatrix} \tag{21}$$

6.2 Likelihood Model

The observations are a sequence of bearing measurements:

$$z_k = \arctan(\frac{y_k}{x_k}) + v_k \tag{22}$$

where z_k is the target bearing and v_k is the measurement noise.

6.3 Accuracy of each particle summary

Figure 6 shows a particle sample set of 2000 particles representing a bearing only probability distribution. The transformation of this sample set to a Gaussian mixture model using EM is shown in Figure 7 while Figure 8 shows the equivalent transformation via Salmond's joining algorithm. The joining algorithm transformation by a Parzen representation is shown in Figure 9.

From these figures, one can see that using the EM algorithm (Figure 7) gives a better approximation of the particle set by GMMs, particularly between the ranges 0 and 20 meters, compared to approximations by joining (Figures 8 and 9).

To accommodate comparisons among various representations, a divergence or distance measure is required to determine the accuracy of each approximation. The selected measure, the Bhattacharyya Coefficient [Comaniciu *et al.*, 2003] is defined as

$$\rho(x) \equiv \sum_{u=1}^{m} \sqrt{p_u(x) q_u} \tag{23}$$

where q is the represented distribution and p is the true distribution (represented by a fine grid). The minimum number of components was found by seeking the approximation that results in a coefficient of 0.95 where a value of 1 would indicate that $p = q$.

For this GMM approximation via Salmond's joining, a reduction to 20 components results in a Bhattacharyya coefficient of 0.95. A reduction to 50 Parzen components results in a coefficient of 0.945. Hence, less components are required for a GMM approximation compared to a Parzen window approximation and provide a more accurate summary. However, the approximation to 20 GMMs via the EM algorithm is more accurate with a coefficient of 0.985 but the EM algorithm is too computationally expensive for our requirements.

6.4 Bandwidth requirements

Table 1 shows the bandwidth requirements for communicating :

- the particle set directly
- a Parzen density approximation of the particles and

- a GMM approximation of the particles

Only the upper triangle of the symmetric covariance matrix of the Parzen and GMM representation need to be communicated.

20 GMM or 50 Parzen components were required to approximate a particle set of 2000. With a communication bandwidth of 500 floats per time interval, the maximum number of particles is 500 which could prove insufficient, whereas the mixture of Gaussian and Parzen approximations would exhibit better performances. The Parzen density estimate is the most compact especially for higher dimensions. For a dimension of 6, the reducing the particle representation to the required 19 GMM components for communication, may not be as accurate as reducing the same representation to the allowed 68 Parzen components.

Table 1: Bandwidth requirements

Represent -ation	Dimen -sion	Components for comms	Bandwidth available available
GMM approx	4	33	500
Parzen approx	4	98	500
Particle set	4	500	500
GMM approx	6	19	500
Parzen approx	6	68	500
Particle set	6	500	500

6.5 Results

In this simulation, the prediction step occurs every 0.5s. A local update occurs every second while each alternate node communicates a summary of its sample set every third second.

Figures 10 to 14 provide a snapshot of the distributions after the second decentralised fusion update to provide visual confirmation of a consistent DDF operation. In the first decentralised fusion update, Node 2 communicates its state to Node 1 which is fused. As there is no common information as yet, the fusion step is a direct update step. Three seconds later, Node 1 communicates its state to Node 2. It can be seen that the distributions at the decentralised nodes (Figures 12 and 13) are less compact compared to the centralised Node (Figure 14). This indicates that the fusion process is consistent as common information is accounted for.

The performance comparison used for this decentralised simulation is the optimal centralised solution as it provides the closest approximation to the 'true' solution. Here centralised, means that each node communicating to every other node in the network at every time step. A suitable measure would be one that measures the inefficiency of each distribution assuming that the centralised solution is the most efficient. An example of

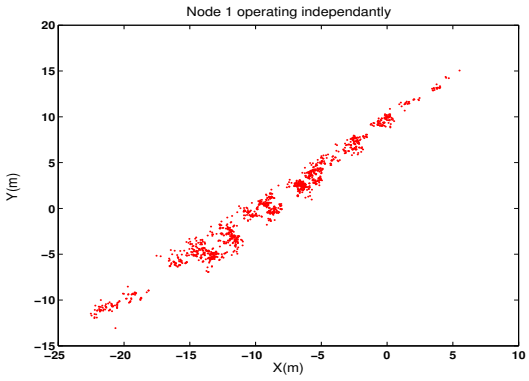

Figure 10: Node 1 operating independently : after the second decentralised fusion update

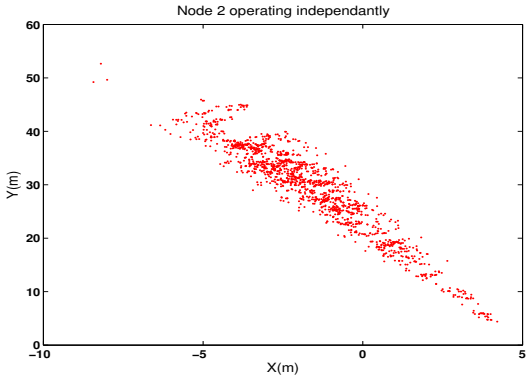

Figure 11: Node 2 operating independently : after the second decentralised fusion update

such a measure is relative entropy [Cover and Thomas, 1991].

The relative entropy or Kullback-Leibler(KL) divergence between two probability mass functions $p(x)$ and $q(x)$ is defined in Cover and Thomas [Cover and Thomas, 1991] as:

$$D(p||q) = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)} \tag{24}$$

The relative entropy is always non-negative and is zero if and only if $p = q$.

Figures 15 and 16 illustrate the KL-Divergence results for each node performing DDF and the standalone nodes (i.e. no communications). The results indicate that decentralised nodes exhibit performances better than the sensors operating alone. The final solutions for the decentralised nodes are similar but less compact that the centralised one.

The decentralised fusion update for GMMs, which is a generalised CI update, is more conservative than the Parzen fusion update resulting in a less compact particle sample set.

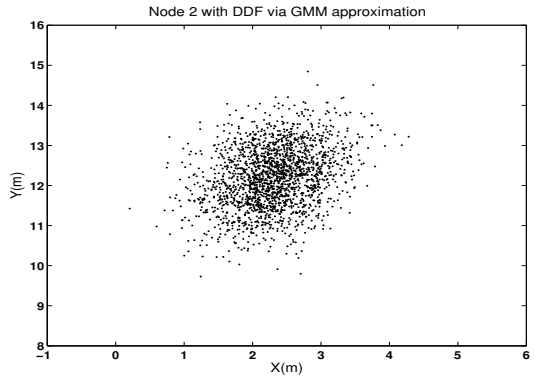

Figure 12: Node 2 with communicated GMMs fusion after the second decentralised fusion update

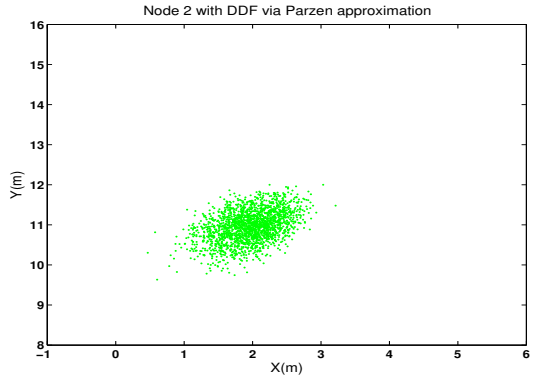

Figure 13: Node 2 with communicated Parzen representations fusion after the second decentralised fusion update

7 Conclusions

This paper has introduced two methodologies for performing consistent and efficient decentralised data fusion with particle filters which transform the particle set to either GMMs or Parzen estimates for communication and inter-nodal fusion. Summarising the sample set with a GMM requires less components and is more accurate than approximation by Parzen representation. However, the Parzen density estimate is more compact. Better fusion results as obtained using Parzen representations as the Parzen estimate division is more accurate than the generalised covariance intersect for GMMs.

One of the areas for future work is the development of different fusion methods for GMMs and particle representations. Future work will also include a demonstration of each of these representations using vision sensors on airborne vehicles, ground vehicles and stationary ground nodes.

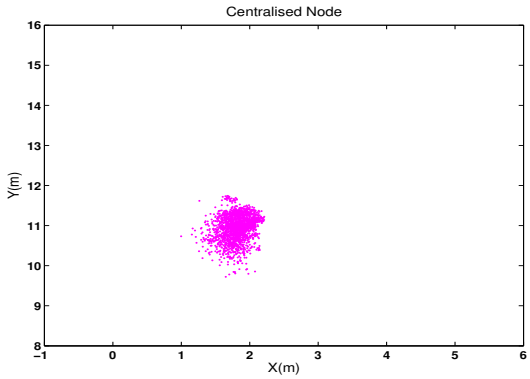

Figure 14: Centralised Solution after the second decentralised fusion update

Acknowledgments

This work is supported by the ARC Centre of Excellence programme, funded by the Australian Research Council (ARC) and the New South Wales State Government and is supported by BAE Systems, Bristol, UK.

References

[Alspach and Sorenson, 1972] D.L. Alspach and H.W. Sorenson. Nonlinear bayesian estimation using Gaussian sum approximations. *IEEE Transactions on Automatic Control*, 17(4):439–448, 1972.

[Arulampalam *et al.*, 2002] M.S. Arulampalam, S. Maskell, N. Gordon and T. Clapp. Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[Bar-Shalom, 1990] Y. Bar-Shalom, Ed. Multitarget-multisensor tracking: advanced appplications. Artech House, 1990.

[Carpenter *et al.*, 1999] J. Carpenter, P. Clifford and P. Fearnhead. Building Robust Simulation-based Filters for Evolving Data Sets. Oxford University, Department of Statistics, 1999.

[Coates, 2004] M. Coates. Distributed particle filters for sensor networks. *Information Processing in Sensor Networks (ISPN 2003)*, Springer, pages 99–107, 2004.

[Comaniciu *et al.*, 2003] D. Comaniciu, V. Ramesh and P. Meer. Kernel-Based Object Tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–575, 2003.

[Cover and Thomas, 1991] T.M. Cover and J.A. Thomas. Elements of Information Theory *Wiley Series in Telecommunications*,Wiley, New York, 1991

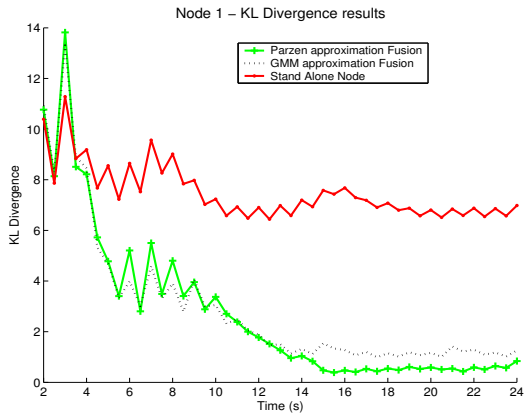

Figure 15: Node 1 - KL Divergence results for the Stand Alone Node, Node 1 communicating and fusing GMMs, Node 1 communicating and fusing Parzen representations.

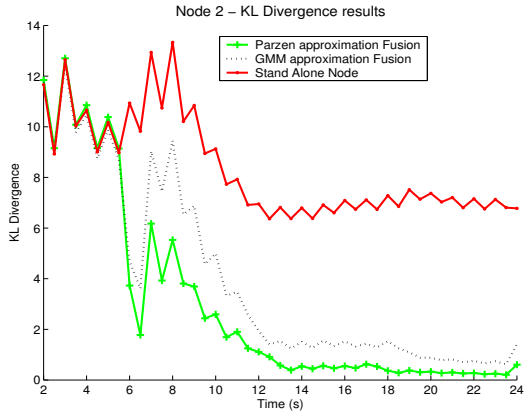

Figure 16: Node 2 - KL Divergence results for the Stand Alone Node, Node 2 communicating and fusing GMMs, Node 2 communicating and fusing Parzen representations.

[Dempster *et al.*, 1977] A.P. Dempster, N.M. Laird and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–37, 1977.

[Doucet *et al.*, 2001] A. Doucet, N. de Freitas and N. Gordon. An Introduction to Sequential Monte Carlo Methods. *Sequential Monte Carlo Methods in Practice*, pages 3–14, Springer-Verlag, 2001.

[Grime, 1992] S. Grime. Communication in Decentralised Sensing Architectures. *phD Thesis*, The University of Oxford, 1992.

[Gordon *et al.*, 1993] N.J. Gordon, D.J. Salmond and A.F.M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings-F*, 140(2):107–113, 2004.

[Ihler *et al.*, 2004] A.T. Ihler, E.B. Sudderth, W.T. Freeman and A.S. Willsky. Efficient Multiscale Sampling from Products of Gaussian Mixtures. *Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence*, 2004.

[MacKay, 2003] D. MacKay. Information Theory, Inference and Learning Algorithms. *http://www.inference.phy.cam.ac.uk/mackay/itila/*, 2003.

[Musso *et al.*, 2001] C. Musso, N. Oudjane and F. Le Gland. Improving Regularised Particle Filters. *Sequential Monte Carlo Methods in Practice*, pages 247–272, Springer-Verlag, 2001.

[Parzen, 1962] E. Parzen. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076, September, 1962.

[Pelleg and Moore, 2000] D. Pelleg and A. Moore. X-means: Extending K-means With Efficient Estimation of the Number of Clusters. *In Proceedings of the 17th International Conference on Machine Learning*, pages 277-, 2000.

[Ridley *et al.*, 2004] M. Ridley, B. Upcroft, L.L. Ong, S. Kumar and S. Sukkarieh. Decentralised Data Fusion with Parzen Density Estimates. *International Conference on Intelligent Sensors, Sensor Networks and Information Processing 2004 (ISSNIP '04)*, Melbourne, Australia, 2004.

[Rosencrantz *et al.*, 2003] M. Rosencrantz, G. Gordon and S. Thrun. Decentralised Data Fusion with distributed particle filters. *Proceedings of the Conference on Uncertainty in AI (UAI)*, Acapulco, Mexico, 2003.

[Rosencrantz *et al.*, 2003a] M. Rosencrantz, G. Gordon and S. Thrun. Locating Moving Entities in Dynamic Indoor Environments with Teams of Mobile Robots. *Proceedings of Autonomous Agents and Multi-Agent Systems*, Melbourne, Australia, 2003.

[Salmond *et al.*, 1989] D.J. Salmond, D.P. Atherton and J.A. Bather. Mixture reduction algorithms for uncertain tracking. *IFAC Proceedings Series*, 2(8):775–780, 1989.

[Sheng *et al.*, 2005] X. Sheng, Y.H. Hu and P. Ramanathan. Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network. *Fourth International Symposium on Information Processing in Sensor Networks, (IPSN)*, pages 181–188, 2005.

[Sorenson and Alspach, 1971] H.W. Sorenson and D.L. Alspach. Recursive bayesian estimation using Gaussian sums. *Automatica*, 7:465–479, 1971.

[Stone *et al.*, 1999] L.D. Stone, C.A. Barlow and T.L. Corwin. Bayesian Multiple Target Tracking. *Artech House*, 1999.

[Upcroft *et al.*, 2005] B. Upcroft, S. Kumar, L.L. Ong, M. Ridley, T. Bailey, S. Sukkarieh and H. Durrant-Whyte. Rich Probabilitstic Representation for Bearing Only Decentralised Data Fusion. In *Proceedings of the IEEE Conference on Information Fusion*, 2005