Decentralised Data Fusion with Exponentials of Polynomials

Bradley Tonkes and Alan D. Blair

***Abstract*— We demonstrate applicability of a general class of multivariate probability density functions of the form $e^{-P(x)}$, where $P(x)$ is an elliptic polynomial, to decentralised data fusion tasks. In particular, we derive an extension to the Covariance Intersect algorithm for this class of distributions and demonstrate the necessary operations – diffusion, multiplication and linear transformation – for Bayesian operations. A simulated target tracking application demonstrates the use of these operations in a decentralised scenario, employing range-only sensing to show their generality beyond Gaussian representations.**

I. INTRODUCTION

Probabilistic methods lie at the heart of many current robotics applications. Descriptions of sensor data, expectations of sensor or actuator error and models of environmental features are all regularly couched in terms of probability distributions. The machinery to manipulate this information is typically based on the Bayesian framework.

The Kalman filter, for example, is used for many current robotics applications including tracking [1], simultaneous localisation and mapping [2] and decentralised data fusion (DDF) [3]. Its popularity can be attributed to its probabilistic grounding and its efficiency. In its unadorned form, however, the Kalman filter is limited by its linearity assumptions. With some success, these assumptions have been overcome with non-linear extensions such as the extended Kalman filter.

Central to the Kalman filter (and many other methods) is the premise that uncertain quantities are Gaussian in nature; that is, their uncertainty can be characterised by their mean and covariance. While this assumption leads to efficiency it comes at the expense of fidelity of representation. For some applications this burden can be overcome by assuming a Gaussian form of sufficient coverage (i.e., large covariance) with associated loss of precision.

Particle filter methods [4] overcome the Gaussian limitations inherent in the Kalman filter. For these methods, uncertainty is represented not in terms of a direct description of a probability distribution, but by a sample from the underlying distribution. A necessary trade-off here is between representational fidelity and computational efficiency.

A different approach has been to consider a more generalised description of distributions in the form of mixtures of Gaussians [5] and Parzen density estimators (with Gaussian kernels) [6], [7]. For both of these approaches, more complex distributions are represented via an agglomeration of Gaussian distributions. The major drawback to these techniques is that most operations between distributions, such as multiplication, are performed pair-wise between the sets of Gaussian components and produce an $O(n^2)$ number of components in the resulting distribution, requiring expensive resampling [5]. Particle filters also exhibit this behaviour, but in that case the resampling step is more efficient [8].

The bandwidth limitations of *distributed* scenarios impose an additional significant constraint, namely, that distributions must be capable of being transmitted efficiently. The particle filter approach suffers in these situations where the capacity of the communication channel may not be capable of supporting enough particles to represent distributions effectively. Resampling strategies have been proposed [9], [10] but they require compression, and hence distortion, of the distribution as well as additional computation. Mixture of Gaussian and Parzen density estimators obviate the need for resampling. Distributed applications may also require a channel filter for maintenance of common information between nodes [11]. In a Bayesian context, the channel filter involves a division between distributions which requires approximation for the three representations discussed here [7].

In a *decentralised* scenario (i.e., one which is both distributed and where there is no assumed knowledge of global network topology) as information flows through the network each node's estimate loses independence from the others [12]. It is therefore important when fusing information from other nodes not to assume independence. The most popular algorithm for conservative fusion is covariance intersect (CI) [13]. Variants of CI exist for mixtures of Gaussians [14] and particle filters, but require approximation and (sometimes expensive) resampling.

In previous work [15] we introduced a class of functions that can be used for representing non-Gaussian probability densities. This class, which we have dubbed ExPoly, comprises functions of the form $e^{-P(x)}$, where $P(x)$ is a multivariate polynomial of even degree which is also elliptic (that is, it approaches infinity in all directions). ExPoly extends the set of Gaussian functions to those which are multi-modal, skewed or otherwise exotic.

ExPoly has a number of properties that make it suitable for application to DDF applications. Multiplication of distributions, as required for integration of new sensor data, is straightforward. For distributed applications employing a channel filter, division is likewise simple. Importantly, representations of distributions, in the form of the coefficients of the polynomial, $P(x)$, are both compact and flexible and can be efficiently communicated around a network.

Given a (linear) process model, an expected level of process noise and a distribution representing the current

Both authors are with the ARC Centre for Autonomous Systems and the School of Computer Science and Engineering, The University of New South Wales, 2052, NSW, Australia. {btonkes,blair}@cse.unsw.edu.au

likelihood, the prediction step seeks to find the expected likelihood at a future time. This operation can be considered in two steps: a (linear) transformation of the distribution and a diffusion (or spreading out). A major drawback for ExPoly is that the class is not closed under diffusion. For Gaussian and Gaussian-based representations this diffusion is performed simply by increasing the covariance (or each component's covariance), but for ExPoly distributions an approximation is necessary. Our former work [15] showed that this approximation was most accurate for distributions most resembling a Gaussian, and remained quite accurate before breaking down for extreme, boundary cases.

In this paper we demonstrate the applicability of ExPoly distributions to a simulated decentralised tracking task. We describe the operations necessary for DDF as applied to ExPoly. These include how CI can be extended to this class of distributions and the creation of sensor models.

II. OPERATIONS ON EXPOLY

It is important to highlight the dual co-ordinate systems that are typically used when considering Gaussian distributions. These co-ordinate systems are based on either the expectation parameters of the distribution (i.e., mean, μ, and covariance matrix, Σ) or the coefficient parameters, also known as the information parameters (i.e., $Y = \Sigma^{-1}$ and $y = \Sigma^{-1}\mu$) [16]. For Gaussian distributions, it is simple to translate back and forth between these co-ordinate systems. For example, the Information filter is simply the Kalman filter translated from expectation to information parameters.

While Gaussian distributions are completely parameterised by the mean and covariance, ExPoly distributions, being more flexible, require more parameters. Again, this parameterisation can be done in terms of either the expectations or the coefficients (of the polynomial $P(x)$). Unlike Gaussian distributions, there is no closed-form solution to the relationship. Consider an ExPoly distribution of n variables defined by its coefficients, $a_{[i]}$, $\exp(-\sum a_{[i]}x^{[i]})$ where $[i]$ represents a multi-index, $i_1, \ldots, i_n$, and $x^{[i]} = \prod_{k=1}^{n} x_k^{i_k}$. The corresponding expectation parameters (the higher-order moments) are given by

$$M^{[i]} = < x^{[i]} > = \int x^{[i]} e^{-P(x)} dx \ .$$

We can then approximate the transformation between the two co-ordinate systems with a linear map

$$M + \delta M \mapsto a + \mathrm{G}^{-1}\delta\mathrm{M}$$

where G is the Fisher information matrix given by

$$\begin{aligned} g(\tfrac{\partial}{\partial a_{[i]}}, \tfrac{\partial}{\partial a_{[j]}}) &= \int \tfrac{\partial P(x)}{\partial a_{[i]}} \tfrac{\partial P(x)}{\partial a_{[j]}} \, e^{-P(x)} dx \\ &= \int x^{[i]} x^{[j]} e^{-P(x)} dx = M^{[i+j]} \end{aligned}$$

and where $[i + j] = (i_1 + j_1), \ldots, (i_n + j_n)$.

In principle this technique defines an iterative procedure for finding a set of coefficients that define a distribution having the given expectation parameters. In practice we find that the process is numerically unstable, leading to non-elliptic polynomials. For univariate functions this problem is easily overcome by ensuring that the coefficient for the greatest (even) exponent is always positive, but for multivariate functions it is problematic to efficiently determine whether a polynomial is non-elliptic and it is not obvious how best to transform a distribution back into an elliptic form to allow the iterative procedure to continue.

Fortunately the reverse process of deriving the expectation parameters from the coefficients is straightforward, if time consuming. The moments can be simply found by integrating $x_{[i]}e^{-P(x)}$ over the domain on which the function is supported. For arbitrary distributions it is unclear where the supporting domain is located. A Metropolis sampler [17] can be used to provide an initial estimate.

The Metropolis sampler seeks to draw samples from an arbitrary distribution, $P(s)$. It uses a 'proposal' distribution, $f(s)$, from which samples can be generated (e.g., Gaussian or uniform), and maintains a state s_i. A state $\tilde{s}_{i+1}$ is proposed by perturbing s_i with a value drawn from f. This state is accepted as s_{i+1} if $P(\tilde{s}_{i+1})/P(s_i) > r$ where r is drawn from the uniform distribution on $(0, 1]$, otherwise $s_{i+1} = s_i$.

For ExPoly distributions the Metropolis sampler has two useful characteristics that result from the need to compute the ratio between distributions. Firstly, the distribution need not be normalised. Secondly, their are numerical benefits in using the ratio since it can be computed directly as $\exp(-(P(\tilde{s}_{i+1}) - P(s_i)))$. The ratio is usually well-defined (and non-zero) and the algorithm can quickly find a mode of the distribution irrespective of the initial state. The downside to the Metropolis sampler is that all modes may not be sampled, skewing estimates of the distributions properties. However, if the covariance of a distribution is approximately known, a conservative choice of the proposal distribution, f, can increase the likelihood that the sampler will traverse all modes of the distribution.

For functions with high dimensionality it may be more efficient to use the Metropolis sampler to directly estimate the expectation parameters but, at least for bivariate functions, we have found it more reliable and sufficiently fast to use the sampler to provide an initial estimate which is then refined via integration over the indicated region. Estimates of mean and covariance will prove necessary for the diffusion operator, discussed below.

A. Linear Transformation

Linear transformations of ExPoly distributions are one of the core operations necessary to support Bayesian analysis. We assume a (linear) process model,

$$x_{k+1} = F_{k+1}x_k + B_{k+1}u_{k+1} + w_{k+1}, \qquad (1)$$

where w_{k+1} is drawn from a Gaussian distribution of zero mean and covariance Q_{k+1}. The current estimate of the state is codified in terms of a probability density function $\exp(-P(\hat{x}_{k|k}))$ over states. Bayesian prediction is performed by transforming the distribution according to (1), and diffusing the distribution according to Q_{k+1} (see following section

for details). This (linear) transformation involves generating a new polynomial, $\hat{P}(\hat{x}_{k+1|k})$ where x_{k+1} is given according to (1) ignoring the process noise. Thus, the new polynomial becomes

$$\hat{P}(\hat{x}_{k+1|k}) = \sum_{[i]} a_{[i]}(F_{k+1}x_k + B_{k+1}u)^{[i]}. \qquad (2)$$

The determinant of F is used as a renormalising constant to maintain the volume of the distribution.

The general (multivariate) solution has been given in earlier work [15], but for a univariate distribution the coefficients of $\hat{P}$ become

$$\hat{a}_i = \sum_{j=i}^{n} a_j \binom{j}{i} F^{j-i}(-Bu)^j.$$

The number of coefficients for an lth degree polynomial of n variables increases as $O(\binom{n+l}{l})$. The linear transformation is quadratic in the number of terms, so $O(\binom{n+l}{l}^2)$. Consequently, the number of coefficients can be a limiting factor in practice and it may become important to compute this linear transformation quickly. Implementations can compute these coefficients by either solving the system for each new coefficient individually (as for the univariate example given above), or by expanding each term of (2) and summing the expanded terms as appropriate. Our implementation takes the first approach using a metaprogram to solve (2) and writing the appropriate source code to perform the transformation directly. While fast, this approach is practically limited to relatively simple functions (sixth order polynomials of three variables).

B. Diffusion

For Bayesian-based systems, prediction is governed by the Chapman-Kolmogorov equation which describes a convolution between the linearly transformed likelihood and the process noise distribution. For ExPoly we assume Gaussian process noise and consider the differential form of the diffusion equation rather than the integral form, which is given by the Laplacian Δ_x. For multivariate functions the Laplacian operator takes the form

$$\Delta = -\sum_{i,j} D_{i,j} \frac{\partial^2}{\partial x_i \partial x_j}$$

for a diffusion matrix, D. Note that Δ can be diagonalised by changing co-ordinates (linear transformation) to $u = V\Sigma^{-\frac{1}{2}}(x - \hat{x})$, where V diagonalises $\Sigma^{-\frac{1}{2}}D\Sigma^{-\frac{1}{2}}$ to Λ such that $\forall i, 0 < \lambda_{i,i} \leq \lambda_{i+1,i+1}$.

For diffusion we decompose our distribution into a Gaussian component, which can be diffused exactly, and a 'residual' whose diffusion is approximated. Thus, for $\exp(-P(u)) = \exp(-(Q(u) + R(u)))$ we have $Q(u) = \frac{1}{2}\sum_{i=1}^{n} u_i^2$ and $R(u) = P(u) - Q(u)$. In the case where $R(u)$ is zero (i.e., $\exp(-P(u))$ is purely Gaussian), the diffusion is given by

$$P_t(u) = \frac{1}{2}u^T(I + t\Lambda)^{-1}u.$$

This general pattern is extended to the residual term giving the full diffusion as

$$P_t(u) = \frac{1}{2}u^T(I + t\Lambda)^{-1}u + \frac{R((I + t\Lambda)^{-\frac{1}{2}})}{(1 + t\Lambda_n)^{\eta}} \qquad (3)$$

where η is a free parameter which experimentation suggests should be set at approximately $2 + \frac{l}{2}$ for an lth order polynomial.

A small extension to (3) chooses a different decay factor (here $(1+t\lambda_n)^{\eta}$) for each term of the residual. Details of this extension (and a more thorough description of the diffusion) are described in detail elsewhere [15], but it is the extended version we will use in the simulations in this paper.

The diffusion relies on a linear transformation of the polynomial based on its mean and covariance to diagonalise Δ. As described earlier, the mean and covariance cannot be directly inferred from the coefficients and must be estimated via either sampling or integration. This inference causes errors in the estimates to which the diffusion process may be sensitive. Simulations suggest, however, that the accuracy of the approximate diffusion process is relatively robust with respect to small errors in these estimates, and that it may even produce a marginally more accurate approximation when the covariance is slightly underestimated.

C. Covariance Intersect

In a DDF system, each node maintains a likelihood. New sensor data is integrated into this likelihood by multiplication of the distributions. For ExPoly distributions this multiplication of distributions becomes an addition of the polynomials' coefficients (as in the information filter). Likelihoods are also propagated around the network. An important issue is what happens when a node receives information: it is quite possible that a node's own previously communicated information is embedded within information coming from other nodes. Naive combination of an incoming likelihood with a node's own current estimate risks information re-use which can lead to overconfidence. To alleviate this problem, some distributed systems employ a known tree-architecture and channel filters so that nodes can filter out the information that they have previously sent [11]. These channel filters require division of distributions which, for ExPoly, are trivial involving simple subtraction of coefficients.

The covariance intersect algorithm (CI) [13] provides a mechanism for conservatively combining distributions with unknown cross-correlations. It provides a general solution for combining information from possibly self-affected sources which permits communication networks of arbitrary topology of which nodes have only local information.

Essentially, CI computes a convex combination of the two distributions using a weighting parameter, ω, chosen to optimise some property of the combined distribution such as the trace or determinant of the covariance. While CI is typically described in expectation co-ordinates,

$$\begin{aligned} \Sigma_c^{-1} &= \omega\Sigma_a^{-1} + (1-\omega)\Sigma_b^{-1} \\ \Sigma_c^{-1}\mu_c &= \omega\Sigma_a^{-1}\mu_a + (1-\omega)\Sigma_b^{-1}\mu_b \end{aligned}$$

its equivalent in coefficient co-ordinates is arguably simpler,

$$\exp(-Q_c(x)) = \exp(-(\omega Q_a(x) + (1-\omega)Q_b(x))). \quad (4)$$

This form of CI suggests an immediate extension to ExPoly distributions

$$\exp(-P_c(x)) = \exp(-(\omega P_a(x) + (1-\omega)P_b(x))). \quad (5)$$

We offer no proof that (5) guarantees consistency in the terms of [13], but experimentation suggests that it produces sensible results.

An alternative approach, which does guarantee consistency, would be to perform CI in terms of the mean and covariance of the distributions. That is

1) compute (estimate) the means and covariances of the two distributions, Σ_a, μ_a and Σ_b, μ_b
2) compute a new mean and covariance, Σ_c, μ_c according to CI and weighting parameter, ω
3) compute a new ExPoly distribution using this weighting parameter and (5)
4) compute (estimate) the mean and covariance of this new distribution, $\hat{\Sigma}_c, \hat{\mu}_c$
5) transform the new ExPoly distribution so that it has mean and covariance Σ_c, μ_c, that is $x = \Sigma_c \hat{\Sigma}_c^{-1}(\hat{x} - \hat{\mu}_c) + \mu_c$.

This approach would guarantee the consistency of the mean and covariance, but promises nothing with respect to the higher order moments, though step (3) suggests that some features should be inherited from the source distributions.

A clear problem with the alternative approach is apparent when considering a range-only example. Suppose that we have two range-only sensors. With a purely Gaussian representation, the sensor likelihood for a range reading, r, is a Gaussian centred on the sensor's location with a covariance of rkI for some constant k. When combining distributions from each sensor, one with covariance $r_1 kI$ the other with covariance $r_2 kI$, CI will always derive a copy of the distribution with the smallest covariance (smallest r). For a Gaussian representation, this may be the best possible outcome. Consider, however, an ExPoly sensor model as shown in Fig. 2. Since our alternative method performs CI only on the mean and covariance the result is identical to the Gaussian case, even though it seems likely that there is a better solution. Indeed, as seen in the next section, our original approach does yield attractive solutions.

For the original (Gaussian) CI, it was suggested that ω be optimised with respect to the size of the covariance (determinant or trace) [13]. Given the richer representations of ExPoly new possibilities become apparent. A clear candidate would be to take the most *informative* distribution as determined by the differential entropy or the Renyi entropy.

For distributions of many variables where numerical integration is impractical, it is still possible to use properties of the covariance as the optimisation criterion to obtain a result of the expected form. Unlike the Gaussian distribution in the range-only example, when combining ExPoly distributions with covariances of the form kI our extended version of CI is capable of producing distributions with covariances that are smaller than those of the source distributions.

As for CI, the optimised quantities of convex combinations of ExPoly functions appear also to be convex in the weighting parameter, ω. This property permits the use of more advanced optimisation strategies. Efficient optimisation is particularly important for ExPoly distributions since evaluation of each ω requires extensive sampling or integration of a distribution which is computationally expensive.

III. Range-Only Example

In this section we demonstrate the theory outlined above on a simple simulated tracking task with range-only sensors. The system is comprised of four nodes, $n_1, \ldots, n_4$ positioned at $(2,2), (1,-1), (-2,-2)$ and $(-2,1)$ respectively. The communication network for the nodes forms a directed ring: $n_1 \rightarrow n_2 \rightarrow n_3 \rightarrow n_4 \rightarrow n_1$. It is hence likely that information communicated from a node will at some future point return to that node, requiring use of our extended CI algorithm. The system aims to estimate the position, (x, y) of the target which moves with a constant velocity of $(0.5, 0.5)$ but with large random movements in position of the same magnitude as the velocity (i.e., the noise in the process model has a covariance of $0.5^2 I$).

A. Sensor Model

In the scenario we have described the key advantage of ExPoly distributions is in their ability to represent a much wider range of sensor likelihood models than their Gaussian counterparts. In a Bayesian data fusion context, with state x_k, state estimate $\hat{x}_k$ and measurement z_k, new sensor information is integrated by

$$P(\hat{x}_k|z_k) = \frac{L(z = z_k|x_k)P(\hat{x}_k|z_{k-1})}{P(z_k|z_{k-1})}$$

where the denominator can be disregarded as a normalising constant. The sensor model describes a likelihood distribution over the target's state for a given sensor reading. For a range-only sensor this likelihood function takes on the form of a ring where the probability mass is concentrated around the measured distance from the sensor. In practice, deriving accurate sensor models is a laborious task. For the purposes of this example we assume that the band cross-section takes on an approximately Gaussian shape, and that is constant irrespective of the measurement (an unlikely assumption). Since the function should be rotationally symmetric, we can consider the one dimensional case and extend it by substituting $\sqrt{x^2 + y^2}$ for x. Thus, for a given measurement z we use a sensor likelihood model (in one dimension) of the form $\exp(-\frac{a}{z^2}x^4 + 2ax^2)$ where a controls the variance of the cross-section (see Fig. 1). This function roughly approximates the summed Gaussian pair given by $\exp(-(4a(x \pm z)^2))$. That is, two Gaussians having variance $\frac{1}{8a}$ and means $\pm z$.

In two dimensions, this sensor model yields a function of the form shown in Fig. 2. The distribution can no longer be approximated by combining a symmetric pair of Gaussian functions. While this form of the function describes a

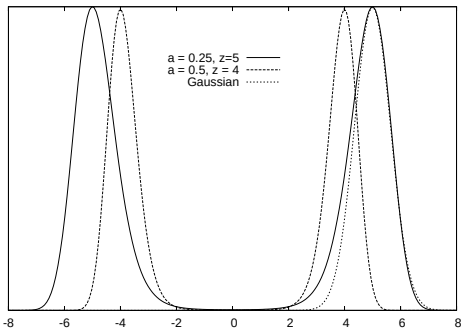


Fig. 1. Sensor model, in one dimension, for a range-only sensor. Shown are the (non-normalised) sensor likelihoods for two sensors, one more accurate (inner pair) than the other (outer pair) for measurements at two ranges, $z = 5$ (outer pair) and $z = 4$ (inner pair). For comparison, a Gaussian distribution has been overlaid on the rightmost mode.

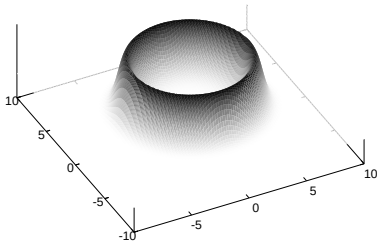


Fig. 2. Sensor model, in two dimensions, for a range-only sensor. For this function, $z = 5$ and $a = 0.25$.

distribution in sensor-centric co-ordinate space, a translation (section II-A) shifts it to the global co-ordinate system.

B. Simulation Results

The nodes in the network operated on slightly different schedules. All nodes performed a measurement every $0.2s$ and n_1 and n_3 communicated their distributions every $0.35s$ (to n_2 and n_4 respectively), whereas n_2 and n_4 communicated every $0.45s$. (Where measurements and communication fell at nominally the same time, measurements occurred first.) The target's initial position was $(-1, -2)$.

The sensor for the simulations was configured to use $a = 2.0$, and noise in the sensor was simulated by perturbing the actual range with Gaussian noise of variance $\frac{1}{8a}$ (approximately matching the sensor noise distribution). When fusing distributions our extended CI algorithm (5) was used optimising ω with respect to the trace of the covariance.

To gauge the performance of our representation we monitor the Renyi entropy ($\alpha = 2$) of each node during the course of the simulation (see Fig. 3) expecting that there should be an overall decrease and a convergence in the entropy of each node. Throughout the simulation, each node tracks the target as expected, and fusion between nodes results in plausible distributions (see Fig. 4).

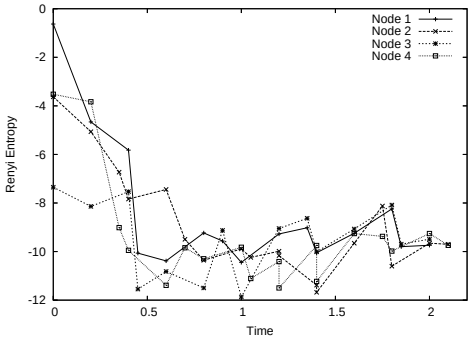


Fig. 3. Renyi entropy of each node in the DDF network. Note that, counter to expectations, for some fusion events the entropy increases. This effect is likely because for CI, ω was optimised with respect to the covariance and not the measure evaluated for this figure.

IV. CONCLUSIONS

In this paper we have considered the ExPoly class of probability density functions and outlined how to perform the operations on it necessary in a DDF context. ExPoly provides a natural extension to Gaussian representations providing more complex distributions as in our range-only example. ExPoly is aimed to sit alongside other non-Gaussian representations (particle filters, mixtures of Gaussians and Parzen density estimators) providing a different set of trade-offs for potential applications with specific constraints.

Unlike other representations, sensor integration is simple irrespective of the number of measurements taken (since resampling is not required). Fusion between nodes is somewhat more complicated, but is efficient in its use of bandwidth. Perhaps the major disadvantage to ExPoly is the need to estimate expectation parameters. Estimates of the mean and covariance are required for time-diffusion (prediction step) and fusion. For diffusion, the mean and covariance should change as for a Gaussian, so expectations can be accurately estimated from the previously estimated distribution. For fusion, efficient methods need to be employed to optimise ω with the fewest evaluations.

Another difficulty with ExPoly distributions is in building sensor likelihood models. For other representations it is reasonably straightforward to infer a distribution matching observed data. While the model given here for a range-only sensor was relatively straightforward, other cases are more complicated. Further work needs to be done to make this task easier; either by rectifying the iterative procedure outlined in section II or by finding some entirely new approach.

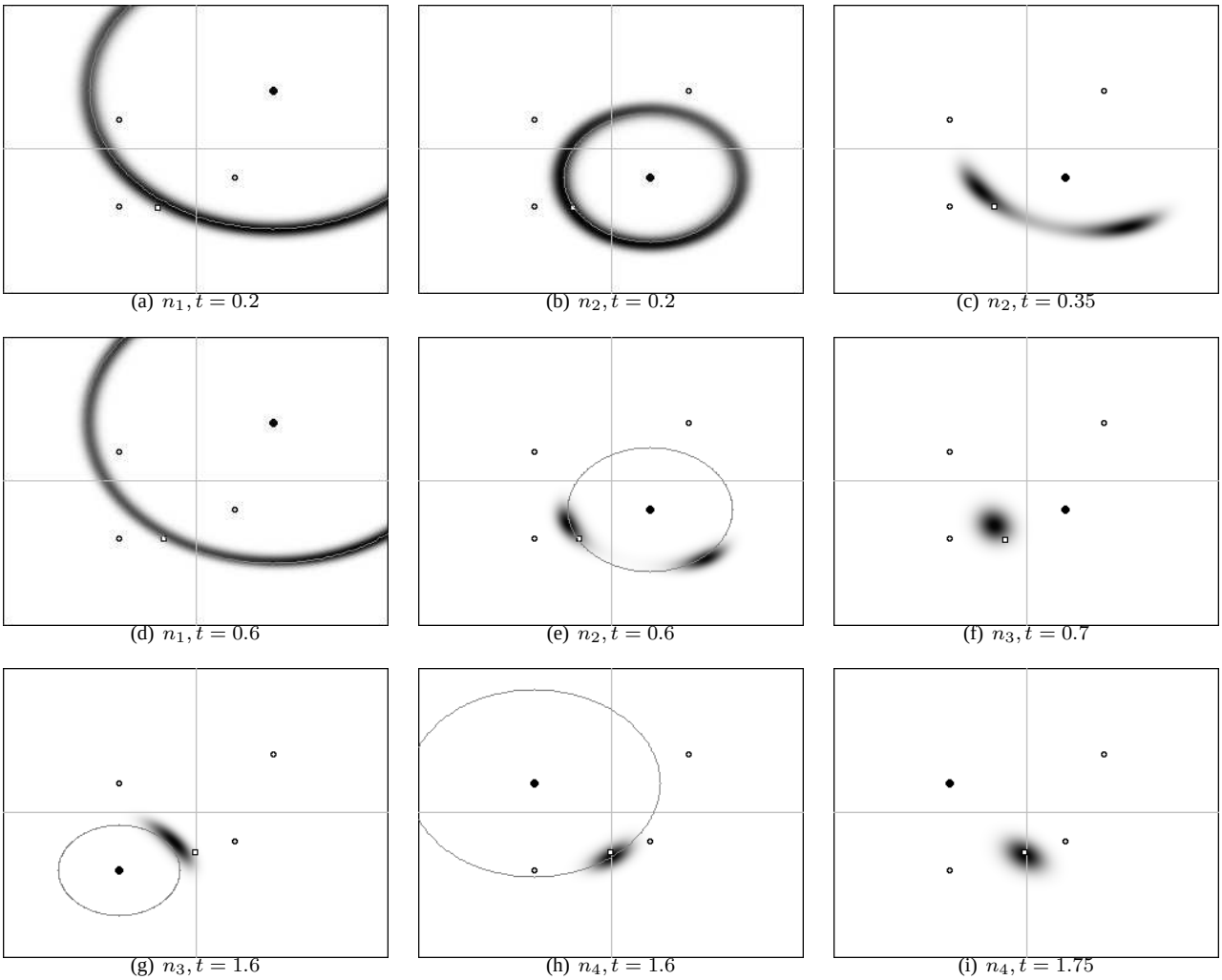
V. ACKNOWLEDGEMENTS

This work is supported by the ARC Centre of Excellence programme, funded by the Australian Research Council.

REFERENCES

[1] Y. Bar-Shalom and X. Li. *Multisensor target tracking: Principles and techniques*. YBT Publishing, 1995.

[2] R. Smith, M. Self, and P. Cheeseman. Estimating uncertain spatial relationships in robotics. In I.J. Cox and G.T. Wilfong, editors, *Autonomous Robot Vehicles*, pages 167–193. Springer-Verlag, Berlin, 1990.



(a) $n_1, t = 0.2$ (b) $n_2, t = 0.2$ (c) $n_2, t = 0.35$

(d) $n_1, t = 0.6$ (e) $n_2, t = 0.6$ (f) $n_3, t = 0.7$

(g) $n_3, t = 1.6$ (h) $n_4, t = 1.6$ (i) $n_4, t = 1.75$

Fig. 4. These figures show the posterior distributions of different nodes after various events. In each figure the node under consideration is shown in black and the target is shown by an outlined box. The first two columns show the posterior distribution of a node after sensor integration with the measured range shown as a grey ellipse. The final columns show the posterior distribution after fusion with information propagating clockwise through the network. Each row forms a triplet: the first two columns show the source distributions and the rightmost column shows the result after the source distributions have undergone a prediction step and then been fused. Within each triplet no other events occur between measurements and fusion other than prediction.

[3] S. Sukkarieh, E. Nettleton, J.-H. Kim, M. Ridley, A Goktogan, and H. F. Durrant-Whyte. The ANSER project: Data fusion across multiple uninhabited air vehicles. *International Journal of Robotics Research*, 22(7/8):505–540, 2003.

[4] N. Gordon, D. Salmond, and A. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proceedings-F*, volume 140, pages 107–113, 1993.

[5] B. Upcroft, S. Kumar, M. Ridley, L. Ong, and H. F. Durrant-Whyte. Fast re-parameterisation of Gaussian mixture models for robotics applications. In *Australasian Conference on Robotics and Automation*, 2004.

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

[7] M.F. Ridley, B. Upcroft, L. Ong, S. Kumar, and S. Sukkarieh. Decentralised data fusion with parzen density estimates. In *Proceedings of the International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pages 161–166, 2004.

[8] L. Ong, M. F. Ridley, B. Upcroft, S. Kumar, T. A. Bailey, S. Sukkarieh, and H. F. Durrant-Whyte. A comparison of probabilistic representations for decentralised data fusion intelligent sensors. In *Sensor Networks and Information Processing*, 2005.

[9] M. Rosencrantz, S. Gordon, and S. Thrun. Decentralised data fusion with distributed particle filters. In *Proceedings of the Conference on Uncertainty in AI*, 2003.

[10] S. Challa, M. Palaniswami, and A. Shilton. Distributed data fusion system using support vector machines. In *Proceedings IEEE Conference on Information Fusion*, 2002.

[11] S. Grime and H. F. Durrant-Whyte. Data fusion in decentralised sensor networks. *Control Engineering Practice*, 2(5):849–863, 1994.

[12] J. Manyika and H. F. Durrant-Whyte. *Data Fusion and Sensor Management: A decentralized information-theoretic approach*. Ellis Horwood, 1994.

[13] S. J. Julier and J. K. Uhlmann. A non-divergent estimation algorithm in the presence of unknown correlations. In *Proceedings of the 1997 American Control Conference*, volume 4, pages 2369–2373, 1997.

[14] B. Upcroft, L. Ong, S. Kumar, M.F. Ridley, T.A. Bailey, S. Sukkarieh, and H.F. Durrant-Whyte. Rich probabilistic representations for bearing only decentralised data fusion. In *IEEE Conference Proceedings on Information Fusion*, 2005.

[15] A. D. Blair and B. Tonkes. Geometry and approximate diffusion for exponentials of polynomials. Submitted, 2007.

[16] Shun'ichi Amari. *Methods of Information Geometry*. Oxford University Press, 1993.

[17] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1), 1970.