HyPE: Hybrid Particle-Element Approach for Recursive Bayesian Searching-and-Tracking

Benjamin Lavis and Tomonari Furukawa

ARC Centre of Excellence for Autonomous Systems
School of Mechanical and Manufacturing Engineering
The University of New South Wales
Sydney 2052, Australia
Phone: +61-2-9385-4125
Email: benjamin.lavis@student.unsw.edu.au

***Abstract*—This paper presents a hybrid particle-element approach, HyPE, suitable for recursive Bayesian searching-and-tracking (SAT). The hybrid concept, to synthesize two recursive Bayesian estimation (RBE) methods to represent and maintain the belief about all states in a dynamic system, is distinct from the concept behind "mixed approaches", such as Rao-Blackwellized particle filtering, which use different RBE methods for different states. HyPE eliminates the need for computationally expensive numerical integration in the prediction stage and allows space reconfiguration, via remeshing, at minimal computational cost. Numerical examples show the efficacy of the hybrid approach, and demonstrate its superior performance in SAT scenarios when compared with both the particle filter and the element-based method.**

I. Introduction

Recursive Bayesian estimation (RBE) of the state of a dynamic system, under uncertain observation and state transition processes, forms the basis for a variety of autonomous estimation and control problems, including mobile robot localization, environment mapping and exploration, target tracking and optimal searching [1]. RBE techniques recursively update and predict a probability density function (PDF) over the system's state with respect to time. Recursive Bayesian searching-and-tracking (SAT) refers to those RBE problems involving incorporation of sensor data, both when the target is observed (tracking) and when it is not (searching) [2]. SAT may be applied to any searching or tracking tasks (in the general sense of the terms), or multi-objective tasks such as those requiring a lost target to be found and then subsequently tracked.

Many of the fundamental concepts of search theory were first posed by B. O. Koopman and colleagues in the Anti-submarine Warfare Operations Research Group (ASWORG) during World War II [3]. Since the search problem is primarily concerned with the area to be searched, initial studies simplified the search problem to an area coverage problem. The introduction of the probability of detection along with advances in computational hardware led to more optimal allocation of search effort [4], [5], [6]. Later years saw the implementation of RBE for manned search and rescue and anti-submarine search operations [7]. More recently, techniques have been formulated for decentralized search using multiple vehicles [8] and optimal autonomous search using the grid-based method for RBE [9].

On the other hand, target tracking, which initially consisted of simple feedback motion tracking, has evolved with the development of a variety of RBE techniques such as the Kalman filter (KF) [10], the extended Kalman filter (EKF) [11], sequential Monte Carlo (SMC) methods [12] and sequential quasi-Monte Carlo (SQMC) methods [13], [14], and their variants. These techniques for tracking seek computational efficiency in representing the sharp and often near-Gaussian PDF of an observable target, with little thought about representing the boundary of the target search space, which is an important consideration for search missions. While the KF and EKF represent the target PDF with a mean and a covariance matrix, the SMC and SQMC methods represent it with a set of particles (the particle filter, PF), which move freely with a resampling technique such as sequential importance sampling [15], [16].

Recently, the unified SAT approach was introduced using the grid-based method (GBM), and subsequently the element-based method (EBM) [17]. However, the maintenance of a large search space, necessary to include all the possible states of the moving targets, yields an excessively large amount of computational effort. Thus SAT approaches involving reconfigurable search spaces have been developed, using the EBM [18] or PF [19]. The element-based space reconfiguration guarantees the inclusion of the full extent of the target's motion, but introduces significant computational overheads in doing so. Alternatively, PFs inherently reconfigure themselves, but the representation of the full target space, desirable for searching, is limited because only a finite set of discrete samples are considered.

This paper presents HyPE, a hybrid particle-element approach, suitable for recursive Bayesian SAT. The hybrid concept, to *synthesize* two RBE methods to represent and maintain the belief about all states in a dynamic system, is distinct from the concept behind "mixed approaches", such as Rao-Blackwellized particle filtering [20], which use different RBE methods for different states. HyPE performs a *static conversion* between an element-based representation, used during

update and evaluation of the PDF, and a particle-based representation, used for Bayesian prediction, and thus eliminates the need for computationally expensive numerical integration. The static conversion process also allows reconfiguration, via remeshing of the target space, at minimal computational cost (the cost associated with remeshing only).

This paper is organized as follows. RBE and SAT are outlined in Sect. II along with a description of PFs and the EBM. Section III details the concept behind HyPE, and its implementation. Numerical examples are shown in Sect. IV and conclusions and future work are contained in the final section.

II. Recursive Bayesian Searching-and-Tracking

This section outlines the general form of RBE, and describes the formulation for the SAT class of problems. Various methods for representing PDFs, the belief metric generated with RBE, are also reviewed.

A. Recursive Bayesian Estimation

RBE seeks to estimate the state, $\mathbf{x}$, of a dynamical system by considering sensor data in light of all previously collected data. The belief about $\mathbf{x}_k$, the state of the system at a discrete time step k, is represented by $p(\mathbf{x}_k|\mathbf{z}_{1:k}, \mathbf{u}_{1:k-1})$, the posterior probability density over the state space $\mathcal{X}$. Here, $\mathbf{z}_{1:k} = \{\mathbf{z}1, \ldots, \mathbf{z}_k\}$ and $\mathbf{u}_{1:k-1} = \{\mathbf{u}1, \ldots, \mathbf{u}_{k-1}\}$ are, respectively, the sequence of all observations, including the current observation, and the sequence of all previous control actions.

The posterior is recursively *updated* using Bayes' Theorem,

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}, \mathbf{u}_{1:k-1}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1}, \mathbf{u}_{1:k-1})}{\int p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1}, \mathbf{u}_{1:k-1})d\mathbf{x}}, \tag{1}$$

where $p(\mathbf{z}_k|\mathbf{x}_k)$ is the likelihood of the observation, described by a sensor model, and $p(\mathbf{x}_k|\mathbf{z}_{1:k-1}, \mathbf{u}_{1:k-1})$ is the *predicted* probability density. Given a posterior, the prediction may be performed in light of a state transition probability, $p(\mathbf{x}_{k+1}|\mathbf{x}_k, \mathbf{u}_k)$, using the Chapman-Kolmogorov equation

$$p(\mathbf{x}_{k+1}|\mathbf{z}_{1:k}, \mathbf{u}_{1:k}) = \int p(\mathbf{x}_{k+1}|\mathbf{x}_k, \mathbf{u}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k}, \mathbf{u}_{1:k-1})d\mathbf{x}_k. \tag{2}$$

Note that when $k = 1$, the prior belief, $p(\mathbf{x}_0)$, is used in place of the prediction in (1).

This general form of RBE is a basis for probabilistic state estimation, however the implementation of RBE requires specific observation likelihood and state transition probability, $p(\mathbf{z}_k|\mathbf{x}_k)$ and $p(\mathbf{x}_k|\mathbf{z}_{1:k-1}, \mathbf{u}_{1:k-1})$, but also a method for representing the updated and predicted PDFs, $p(\mathbf{x}_k|\mathbf{z}_{1:k}, \mathbf{u}_{1:k})$ and $p(\mathbf{x}_{k+1}|\mathbf{z}_{1:k-1}, \mathbf{u}_{1:k-1})$.

B. Searching-and-Tracking

1) SAT observation likelihood: In SAT problems the observation likelihood for a sensor platform s making observations about a target t, is denoted $p({}^s\tilde{\mathbf{z}}_k^t|\tilde{\mathbf{x}}_k^t, \tilde{\mathbf{x}}_k^s)$, as it depends on not only the state of the target, $\tilde{\mathbf{x}}_k^t \in \mathcal{X}^t$, but also the state of the sensor platform, $\tilde{\mathbf{x}}_k^s \in \mathcal{X}^t$. Note that the tilde is used to signify an instance ($\tilde{\cdot}$) of a variable ($\cdot$).

Furthermore, the SAT observation likelihood also depends on ${}^sd_k^t \in \{0, 1\}$, where ${}^sd_k^t = 1$ signifies a detection event and ${}^sd_k^t = 0$ signifies a non-detection event. The SAT observation likelihood is therefore given by

$$p({}^s\tilde{\mathbf{z}}_k^t|\tilde{\mathbf{x}}_k^t, \tilde{\mathbf{x}}_k^s) = \begin{cases} l_d(\tilde{\mathbf{x}}_k^t|\tilde{\mathbf{x}}_k^s, {}^s\tilde{\mathbf{z}}_k^t), & {}^sd_k^t = 1 \\ l_{nd}(\tilde{\mathbf{x}}_k^t|\tilde{\mathbf{x}}_k^s), & {}^sd_k^t = 0 \end{cases} \tag{3}$$

where $l_d(\mathbf{x}_k^t|\tilde{\mathbf{x}}_k^s, {}^s\tilde{\mathbf{z}}_k)$ is the detection, or tracking, likelihood function, and $l_{nd}(\mathbf{x}_k^t|\tilde{\mathbf{x}}_k^s)$ is the non-detection, or searching, likelihood function. Note that for a sensor with a field of view, or 'detection space', ${}^s\mathcal{X}_k^d \subset \mathcal{X}^t$, there is a possibility of missing a detection or falsely detecting a target, that is $\Pr({}^sd_k^t = 0|\exists\tilde{\mathbf{x}}_k^t \in {}^s\mathcal{X}_k^d, \tilde{\mathbf{x}}_k^s) > 0$ and $\Pr({}^sd_k^t = 1|\nexists\tilde{\mathbf{x}}_k^t \in {}^s\mathcal{X}_k^d, \tilde{\mathbf{x}}_k^s) > 0$. Therefore the detection and non-detection likelihoods are typically defined to represent the uncertainty in both the positional observation error, $e \neq 0$, and the truth of the detection. Example detection and non-detection likelihoods and shown in figure 1. One consequence of considering such observation likelihoods is that the resulting update and prediction PDFs are typically non-linear, non-Gaussian and potentially multimodal. For this reason RBE methods which can accommodate arbitrary PDFs are generally preferred for SAT.

2) SAT state transition probability: Often the search vehicle cannot know the control actions of the target vehicle, however many SAT can be considered to be "one-sided" problems. One-sided problems describe SAT tasks where the target cannot deliberately act to aid nor avoid detection. An example of a one-sided SAT problem would be a search and rescue mission involving a powerless vessel adrift at sea. In such cases the state transition probability depends only on the target state, and may be written as $p(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t)$.

The SAT update and prediction equations are therefore given

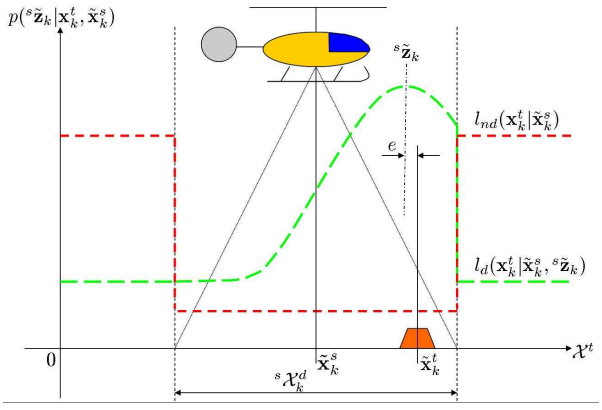


Fig. 1. SAT observation likelihoods.

by

$$p(\mathbf{x}_k^t|{}^s\tilde{\mathbf{z}}_{1:k}^t, \tilde{\mathbf{x}}_{1:k}^s) = \frac{p({}^s\tilde{\mathbf{z}}_k^t|\tilde{\mathbf{x}}_k^t, \tilde{\mathbf{x}}_k^s)p(\mathbf{x}_k^t|{}^s\tilde{\mathbf{z}}_{1:k-1}^t, \tilde{\mathbf{x}}_{1:k-1}^s)}{\int p({}^s\tilde{\mathbf{z}}_k^t|\tilde{\mathbf{x}}_k^t, \tilde{\mathbf{x}}_k^s)p(\mathbf{x}_k^t|{}^s\tilde{\mathbf{z}}_{1:k-1}^t, \tilde{\mathbf{x}}_{1:k-1}^s)d\mathbf{x}_k^t} \quad (4)$$

and

$$p(\mathbf{x}_{k+1}^t|{}^s\tilde{\mathbf{z}}_{1:k}^t, \tilde{\mathbf{x}}_{1:k}^s) = \int p(\mathbf{x}_{k+1}^t|\mathbf{x}_k^t)p(\mathbf{x}_k^t|{}^s\tilde{\mathbf{z}}_{1:k}^t, \tilde{\mathbf{x}}_{1:k}^s)d\mathbf{x}_k^t, \quad (5)$$

respectively.

Furthermore, autonomous SAT may be achieved by utilizing RBE for SAT in the selection of search vehicle control actions. For an objective function J and a finite planning horizon of n_k time steps, a sequence of control actions can be found by solving

$$\arg\max J(\mathbf{u}_{k:k+n_k-1}| \tilde{\mathbf{x}}_k^s, \{p(\mathbf{x}_{k+\kappa}^t|{}^s\tilde{\mathbf{z}}_{1:k}^t, \tilde{\mathbf{x}}_{1:k}^s), \forall\kappa \in \{1, \ldots, n_k\}\}) \quad (6)$$

where $p(\mathbf{x}_{k+\kappa}^t|{}^s\tilde{\mathbf{z}}_{1:k}^t, \tilde{\mathbf{x}}_{1:k}^s)$ can be recursively predicted using the Chapman-Kolmogorov equation,

$$p(\mathbf{x}_{k+\kappa}^t|{}^s\tilde{\mathbf{z}}_{1:k}^t, \tilde{\mathbf{x}}_{1:k}^s) = \int p(\mathbf{x}_{k+\kappa}^t|\mathbf{x}_{k+\kappa-1}^t)p(\mathbf{x}_{k+\kappa-1}^t|{}^s\tilde{\mathbf{z}}_{1:k}^t, \tilde{\mathbf{x}}_{1:k}^s)d\mathbf{x}_{k+\kappa-1}^t. \quad (7)$$

C. Particle and Element Based PDF Representations

1) Particle Filtering: Particle filtering approximates the posterior distribution with a finite set of particles, $\mathcal{P}_k^t = \{{}^t\mathbf{p}_k^1, {}^t\mathbf{p}_k^2, \ldots, {}^t\mathbf{p}_k^M\}$, where each particle, ${}^t\mathbf{p}_k^m$, $m \in \{1, \ldots M\}$, represents a hypothetical state of the target t. The update is performed by sampling the M particles in the filter with a probability proportional to an importance weighting, corresponding to the latest observation. As a result, the particles are distributed according to the current posterior, ${}^t\mathbf{p}_k^m \sim p(\mathbf{x}_k^t|{}^s\tilde{\mathbf{z}}_{1:k}^t, \tilde{\mathbf{x}}_{1:k}^s)$. The prediction stage is carried out by taking M samples from the state transition probability and applying a single instance of the target's motion to each particle, thus avoiding any costly numerical integration in the prediction stage.

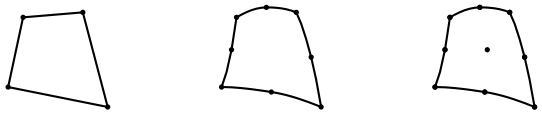
2) Element Based Method: The element-based method continuously approximates the target space and PDF using irregularly shaped elements described by shape functions. Generally the target space is first defined by a number of nodes which are then connected so as to create elements. For two-dimensional search spaces the simplest such elements are linear triangular elements generated via Delaunay triangulation. However elements need not be limited by shape or linearity; triangular or quadrilateral elements and higher-order elements with more nodes are all possible, as shown in Figs. 2 and 3.

Let an approximate target space $\mathcal{X}^e$, consisting of n_e elements, be described by

$$\mathcal{X}^e \equiv \{\mathcal{X}_1^e, \ldots, \mathcal{X}_{n_e}^e\} \approx \mathcal{X}^t, \quad (8)$$

(a) 3 nodes (b) 6 nodes

Fig. 2. Triangular Element Types



(a) 4 nodes (b) 8 nodes (c) 9 nodes

Fig. 3. Quadrilateral Element Types

where $\bigcup_{i=1}^{n_e} \mathcal{X}_i^e = \mathcal{X}^e$ and $\bigcap_{i=1}^{n_e} \mathcal{X}_i^e = \emptyset$. As such, any point in the target space, $\tilde{\mathbf{x}}^t \in \mathcal{X}^t$ may be located in one of the elements. For a point in the the ith element, $\tilde{\mathbf{x}}^t \in \mathcal{X}_i^e$, the point may be expressed in terms of the n_v nodes of the element. For nodes, $\check{\mathbf{x}}_{ij}^e = [\check{x}_{ij}^e, \check{y}_{ij}^e]^T$, $\forall j \in \{1, \ldots, n_v\}$, $\tilde{\mathbf{x}}^t$ may be expressed as

$$x^t = \varphi_x(\xi, \eta) \equiv \sum_{j=1}^{n_v} \check{x}_{ij}^e N_j(\xi, \eta)$$
$$y^t = \varphi_y(\xi, \eta) \equiv \sum_{j=1}^{n_v} \check{y}_{ij}^e N_j(\xi, \eta) \quad (9)$$

where $N_j(\xi, \eta)$ is the shape function, which must satisfy

$$0 < N_j(\xi, \eta) < 1$$
$$\sum_{j=1}^{n_v} N_j(\xi, \eta) = 1 \quad (10)$$

and $\xi \in \Xi = [\xi_{\min}, \xi_{\max})$ and $\eta \in H = [\eta_{\min}, \eta_{\max})$ are known as the natural coordinates.

The shape function and the ranges of the natural coordinates vary according to the type of element. In general the shape function takes the form

$$N_j(\xi, \eta) = \sum_{k=1}^{n_v} a_{jk} b_k(\xi, \eta) \quad (11)$$

where a_{jk} is a coefficient determined by the constraints (10) and $b_k(\xi, \eta)$ is the basis function of monomials in the natural coordinates. $b_k(\xi, \eta)$ may be determined using the binomial theorem:

$$
\begin{aligned}
b_1(\xi,\eta) &= 1 \\
b_2(\xi,\eta) = \xi, b_3(\xi,\eta) &= \eta \\
b_4(\xi,\eta) &= \xi\eta \\
b_5(\xi,\eta) = \xi^2, b_6(\xi,\eta) &= \eta^2 \\
b_7(\xi,\eta) = \xi^2\eta, b_8(\xi,\eta) &= \xi\eta^2 \\
b_9(\xi,\eta) &= \xi^2\eta^2 \\
&\dots
\end{aligned} \tag{12}
$$

For triangular elements the ranges of the natural coordinates are $[\xi_{\min}, \xi_{\max}] = [0, 1]$ and $[\eta_{\min}, \eta_{\max}] = [0, 1-\xi]$, and for quadrilateral elements the ranges of the natural coordinates are $[\xi_{\min}, \xi_{\max}] = [-1, 1]$ and $[\eta_{\min}, \eta_{\max}] = [-1, 1]$.

In order to perform RBE using the EBM, one must be able to both evaluate a function at a point in the state space, and integrate a function over the search space. Using the EBM the evaluation of a function at a point and the integration of a function may be carried out by considering the natural coordinates. For a point in the ith element, $\tilde{\mathbf{x}}^t \in \mathcal{X}_i^e$, the natural coordinates of the point in the element may be determined using

$$
\left[\tilde{\xi}, \tilde{\eta}\right]^T = \varphi^{-1}(\tilde{\mathbf{x}}^t) \tag{13}
$$

where φ^{-1} is the inverse of the set of functions $\varphi = \{\varphi_x, \varphi_y\}$. The function value at $\tilde{\mathbf{x}}^t$ is then given by

$$
f(\tilde{\mathbf{x}}^t) \approx f^e(\tilde{\mathbf{x}}^t) = \sum_{j=1}^{n_v} f(\check{\mathbf{x}}_{ij}^e) N_j(\tilde{\xi}, \tilde{\eta}). \tag{14}
$$

Integration is performed with respect to the natural coordinates according to the transformation

$$
d\mathbf{x}^t = \det \mathbf{J}(\xi, \eta) d\xi d\eta \tag{15}
$$

where $\mathbf{J}$ is the Jacobian matrix

$$
\mathbf{J}(\xi,\eta) = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}. \tag{16}
$$

Integration over the target space is given by

$$
I = \int_{\mathcal{X}^t} f(\mathbf{x}^t) d\mathbf{x}^t \approx \sum_{i=1}^{n_e} I_i^e \tag{17}
$$

where I_i^e, the integral over an element, is

$$
I_i^e = \int_{\mathcal{X}_i^e} f^e(\mathbf{x}^t) d\mathbf{x}^t = \int_{\Xi, H} f^e(\varphi(\xi,\eta)) \det \mathbf{J} d\xi d\eta. \tag{18}
$$

Note that this integral is only analytically derivable for triangular elements with three nodes. In general, Gauss integration may be used to numerically calculate the integral over each element.

III. HyPE: Hybrid Particle-Element Approach

HyPE seeks to imbue the RBE process with the strengths of each of its constituent methods. The key idea being the synthesis of the two methods, in order to utilize the most appropriate representation at different stages of the estimation process. Through processes of *static conversion* between particle and element representations, HyPE is able to switch to either representation, without the need to maintain the other.

Before HyPE is described in detail, certain terms must be defined. The set of n_n nodes forming the element-based representation of the search space is given by,

$$
\mathcal{N} = \{\mathbf{n}_1, ..., \mathbf{n}_{n_n}\} = \{\check{\mathbf{x}}_{ij}^e | \forall i \in \{1, .., n_e\}, \forall j \in 1, ..., n_v\}\}. \tag{19}
$$

The sets of posterior and prediction values, evaluated at each node in the mesh, represent element-based beliefs, and are given by

$$
\mathcal{B}_k = \{p(\mathbf{x}_k^t = \mathbf{n}_i |^s \tilde{\mathbf{z}}_{1:k}^t, \tilde{\mathbf{x}}_{1:k}^s) | \forall i \in \{1, \dots, n_n\}\} \tag{20}
$$

and

$$
\overline{\mathcal{B}}_k = \{p(\mathbf{x}_k^t = \mathbf{n}_i |^s \tilde{\mathbf{z}}_{1:k-1}^t, \tilde{\mathbf{x}}_{1:k-1}^s) | \forall i \in \{1, \dots, n_n\}\}, \tag{21}
$$

respectively. Also, the set of observation likelihood values, evaluated at the nodes, is given by,

$$
\mathcal{Z}_k = \{p(^s \tilde{\mathbf{z}}_k^t = \mathbf{n}_i | \tilde{\mathbf{x}}_k^t, \tilde{\mathbf{x}}_k^s) | \forall i \in \{1, \dots, n_n\}\} \tag{22}
$$

The HyPE approach is described by Algorithm 1. The algorithm takes as input the prediction for the current state, the current observation likelihood, the state transition probability, the set of nodes and the number of particles. The update is performed using the element-based representation, whereas prediction is carried out using the particle-based representation. The algorithm returns the element-based representation of the prediction for the next time step.

Therefore line 1 of Algorithm 1 calls the element-based update function, `update`, to determine the posterior values

Algorithm 1: HyPE: Hybrid Particle-Element Approach

Input : $\overline{\mathcal{B}}_k$, $\mathcal{Z}_k$, $p(\mathbf{x}_{k+1}^t | \mathbf{x}_k^t)$, $\mathcal{N}$, M
Output: $\overline{\mathcal{B}}_{k+1}$

1 $\mathcal{B}_k =$ `update(`$\overline{\mathcal{B}}_k$, $\mathcal{Z}_k$`)`;
2 $\mathcal{W} =$ `calculate_weights(`$\mathcal{B}_k$, $\mathcal{N}$`)`;
3 $\mathcal{P}_k =$ `generate_particles(`$\mathcal{W}$, $\mathcal{N}$, M`)`;
4 $\mathcal{P}_{k+1} =$ `particle_prediction(`$\mathcal{P}_k$, $p(\mathbf{x}_{k+1}^t | \mathbf{x}_k^t)$`)`;
5 [Optional] $\mathcal{N} =$ `remesh(`$\mathcal{P}_{k+1}$`)`;
6 $\overline{\mathcal{B}}_{k+1} =$ `extract_prediction(`$\mathcal{P}_{k+1}$,$\mathcal{N}$`)`;

Function `update(`$\overline{\mathcal{B}}_k$, $\mathcal{Z}_k$`)`

Output: $\mathcal{B}_k$
forall *nodes*, i **do**
    $\mathcal{B}(i)_k = \alpha \overline{\mathcal{B}}(i)_k \mathcal{Z}(i)_k$;
end

at each of the nodes. The α which appears in the update stage is a normalization constant, used to ensure the PDF integrates to unity. Note that neglecting to normalize the PDF using α will not alter the performance of HyPE.

Line 2 of Algorithm 1 calls the function `calculate_weights` in order to determine a sampling weight for each of the nodes in the mesh. The set of sampling weights is denoted $\mathcal{W}$. The sampling weight for each node is its posterior density, given by the product of its posterior density value and its relative volume. Voronoi cells are used to calculate the relative volume associated with each node. Line 1 of `calculate_weights` computes $\mathcal{X}^v$ the Voronoi tessellation of $\mathcal{N}$, where $\mathcal{X}_i^v$ is the Voronoi cell corresponding to node i.

The Voronoi tessellation of a set of nodes randomly distributed in the plane is shown in Fig. 4. Each Voronoi cell $\mathcal{X}_i^v$ defines the space within which all points are closer to $\mathbf{n}_i$ than $\mathbf{n}_{j\neq i}$. Some Voronoi cells are unbounded, such as those shown unshaded in Fig. 4, and therefore have infinite volume. For that reason the relative volume, calculated in line 2, takes the intersection of the Voronoi cell and the search space $\mathcal{X}^t$. Note that if the nodes are equally spaced, such as in a regular mesh, then the relative volumes of all nodes are equal. In such cases, calculation of the relative volume, and also therefore the Voronoi tessellation, is unnecessary and can be neglected.

The M particles to be used for the prediction stage are then generated by calling `generate_particles` in line 3 of Algorithm 1. The particles are generated by sampling with replacement from the n_n nodes. The probability of a node i being selected is proportional to its weighting, $\mathcal{W}(i)$, resulting in the particle set being distributed according to $\mathcal{B}_k$. The standard particle filter prediction is then performed by calling `particle_prediction`, resulting in $\mathcal{P}_{k+1}$. All

Function `calculate_weights`($\mathcal{B}_k$, $\mathcal{N}$)

Output: $\mathcal{W}$

1 $\mathcal{X}^v(=\{\mathcal{X}_1^v,\ldots,\mathcal{X}_{n_n}^v\})=$ `voronoi`($\mathcal{N}$);
2 **forall** *nodes*, i **do**
3 $\quad v=$`volume`($\mathcal{X}_i^v \bigcap \mathcal{X}^t$)/`volume`($\mathcal{X}^t$);
4 $\quad \mathcal{W}(i)=v\mathcal{B}(i)_k$;
5 **end**

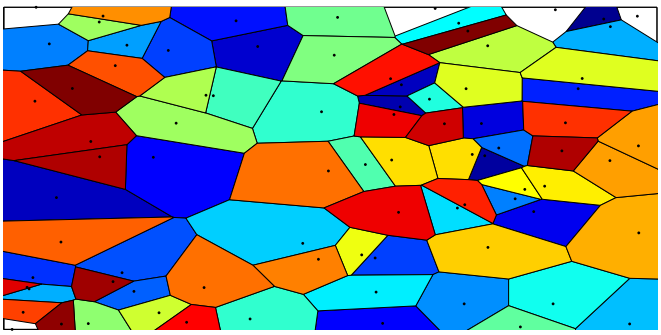


Fig. 4. 2D Voronoi Tessellation.

Function `generate_particles`($\mathcal{W}$, $\mathcal{N}$, M)

Output: $\mathcal{P}_k$

1 $\mathcal{P}_k=\emptyset$;
2 **for** $m=1$ *to* M **do**
3 $\quad$ draw i with probability $\propto \mathcal{W}(i)$;
4 $\quad$ add ${}^t\mathbf{p}_k^m=\mathbf{n}_i$ to $\mathcal{P}_k$;
5 **end**

Function `particle_prediction`($\mathcal{P}_k$, $p(\mathbf{x}_{k+1}^t|\mathbf{x}_k^t)$)

Output: $\mathcal{P}_{k+1}$

1 $\mathcal{P}_{k+1}=\emptyset$;
2 **for** $m=1$ *to* M **do**
3 $\quad$ sample ${}^t\mathbf{p}_{k+1}^m \sim p(\mathbf{x}_{k+1}^t|\mathbf{x}_k^t={}^t\mathbf{p}_k^m)$;
4 $\quad$ add ${}^t\mathbf{p}_{k+1}^m$ to $\mathcal{P}_{k+1}$;
5 **end**

that remains to complete the HyPE approach is to extract the density of the predicted particle set node locations by calling `extract_prediction`. Existing density extraction techniques which extract particle densities at certain points may be called in line 6. An example of such a technique is *kernel density estimation*, where each particle represents the center of a "kernel", with a known density function. The mixture (sum) of all kernel densities at a point in the search space, gives the overall density at that point.

An alternative density extraction technique is described by the function `extract_prediction` (Voronoi Extraction). This technique may be considered as an element-based generalization of the grid-based technique in which a grid is superimposed on the search space and the number of particles which fall in each grid cell gives the density of the cell. In the Voronoi extraction approach the density is given by the number of particles which fall within a node's Voronoi cell, weighted by the cell's relative volume, (and normalized using a new α value if necessary). Again, if a regular mesh is employed the relative volume need not be calculated. Line 4 in `extract_prediction` makes use of the indicator function,

$$\delta_i({}^t\mathbf{p}_{k+1}^m-\mathbf{n}_i)=\begin{cases}1, & {}^t\mathbf{p}_{k+1}^m\in\mathcal{X}_i^v\\ 0, & {}^t\mathbf{p}_{k+1}^m\notin\mathcal{X}_i^v.\end{cases} \tag{23}$$

Function `extract_prediction`($\mathcal{P}_{k+1}$, $\mathcal{N}$) (Voronoi Extraction)

Output: $\overline{\mathcal{B}}_{k+1}$

1 $\mathcal{X}^v(=\{\mathcal{X}_1^v,\ldots,\mathcal{X}_{n_n}^v\})=$ `voronoi`($\mathcal{N}$);
2 **forall** *nodes*, i **do**
3 $\quad v=$`volume`($\mathcal{X}_i^v \bigcap \mathcal{X}^t$)/`volume`($\mathcal{X}^t$);
4 $\quad w=\sum_{m=1}^M \delta_i({}^t\mathbf{p}_{k+1}^m-\mathbf{n}_i)$;
5 $\quad \overline{\mathcal{B}}(i)_{k+1}=\alpha w/v$;
6 **end**

Both kernel density estimation and the Voronoi extraction technique can be used to extract the density at arbitrary points in the search space. For that reason the option of *remeshing* the search space has been included in line 5 of the HyPE algorithm, before the density extraction function is called. Doing so eliminates the need for extraneous interpolation at the new node locations, and enables the search space boundary and interior nodes to be reconfigured, in order to better capture the nature of the target state, at only the computational cost required to generate the new mesh.

Also, HyPE can perform the multistep prediction often required for the SAT control problem, simply by repetition of the particle prediction step, with density extraction as needed.

IV. Numerical Examples

Ten two-dimensional SAT scenarios were considered in order investigate the efficacy of HyPE and for comparison with the EBM and PF. In each scenario a single sensor platform searched for a single target. The sensor was assumed to have perfect detection capabilities (no false negatives or false positives), with a range of 30 meters, but observations of the target location were assumed to be noisy. The prior density used in all the scenarios was a mixture of two Gaussian distributions with means at $[250, 250]^T$ and $[450, 450]^T$ and covariances

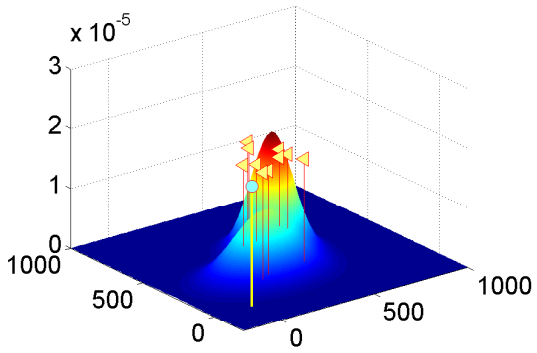
$$\Sigma_1 = \begin{bmatrix} 150 & 0 \\ 0 & 100 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 90 & 0 \\ 0 & 90 \end{bmatrix}.$$

The element and particle based representation of the prior density are shown in Figs. 5(a)&(b), along with the initial position of the sensor platform and the targets from all ten scenarios. Figure 5(c) shows the velocity map which the targets follow. The state transition probability considered in the RBE used a Gaussian distributions of the velocity at each point, with means given by the velocity map. The control limits of the sensor platform and targets are given in Table I. A time step of $\Delta k = 1$ second was used, and 210 instances of the sensor control actions (speed and steering angle) were used to evaluate a single step lookahead control strategy. The maximum allowable iteration time to complete the the observation, update, prediction and control was set to Δk.

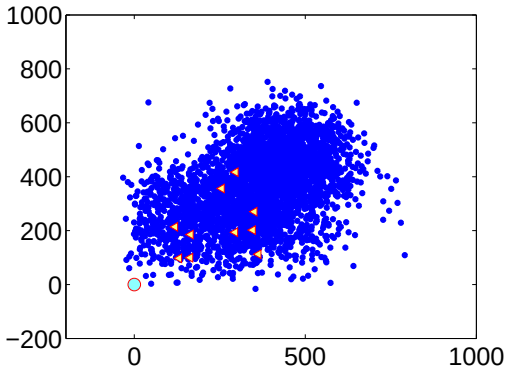
TABLE I
Vehicle Model Control Limits

	Sensor Platform	Target
Maximum Speed [knots]	100	50
Minimum Speed [knots]	10	0
Maximum Steer Angle [deg/s]	15	N/A

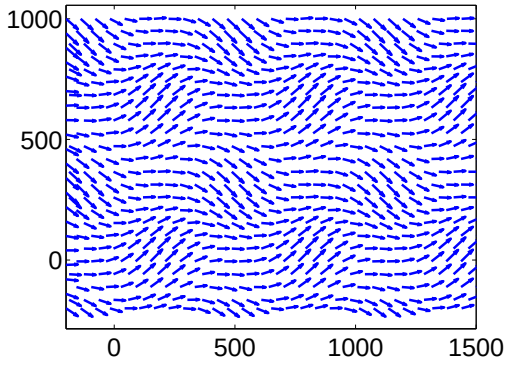
Two implementations of HyPE were evaluated, HyPE(a) and HyPE(b), both using regular meshes and the optional remeshing step, to resize the mesh based on the underlying particle distribution. The only difference in implementation between HyPE(a) and HyPE(b) was that HyPE(a) used fewer nodes in the mesh than HyPE(b). Despite running faster than HyPE(b), only one iteration of HyPE(b) (including control optimization) was performed per time step. The search space



(a) Element-Based Prior



(b) Particle Filter Prior



(c) Velocity Map

Fig. 5. Search and Tracking Scenarios

reconfiguration technique for the EBM was prohibitively slow for meeting the iteration time requirement, thus a static mesh was used for the EBM. A fixed sample size was used in the PF. The number of sample points used for each representation

are shown in Fig. 6. It can be seen that the median number of sample points used for HyPE(a) was 30% of the number used for the EBM and 14% of the number used for the PF. For HyPE(b) the percentages were 76% compared with the EBM and 35% compared with the PF.

Figure 7 shows the evolution of position errors over time for a single scenario. It can be seen that all approaches enabled the sensor to find the target (indicated by the distance between sensor and target falling below the 30m range line). Furthermore, despite using fewer sample points, the fast remeshing ability of HyPE allowed higher resolution of sample points during tracking, resulting in smaller estimation errors. This is demonstrated in terms of the error between the mode of the posterior density and true target state (note that the mean of the particle positions was used calculate this error for the PF due to the ease its computation and because, for tracking, the discrepancy between the two was negligible). Furthermore, whilst the EBM and PF approaches had trouble maintaining sensor contact with the found target, both implementations of HyPE remained well within sensor contact subsequent to finding the target.

Figure 8 shows the performance of each approach during tracking (subsequent to the first detection of the target). The boxplot represents the distribution of the average sensor and posterior position errors taken over the tracking periods of each scenario. It can be seen that the median sensor position error for HyPE(a) was 33% of the median error for the EBM and 25% of the error for the PF. For HyPE(b) the percentages were 29% compared with the EBM and 22% compared with the PF. The median posterior position error for HyPE(a) was 39% of the median error for the EBM and 26% of the error for the PF. For HyPE(b) the percentages were 26% compared with the EBM and 16% compared with the PF.

In terms of searching performance, the time taken to make the first detection of the target was recorded for each scenario and each approach. The detection times for each scenario were then ranked to determine in which order the approaches found the target. Figure 9(a) shows that on three occasions HyPE(a) was quicker than both the EBM and PF in detecting the target. In the other seven scenarios HyPE(a) was second behind either the EBM (6 times) or the PF (once). Figure 9(b) shows that in half of the scenarios HyPE(b) was quicker than both the EBM and PF in detecting the target. In in the other five scenarios HyPE(b) was second behind either the EBM (4 times) or the PF (once).

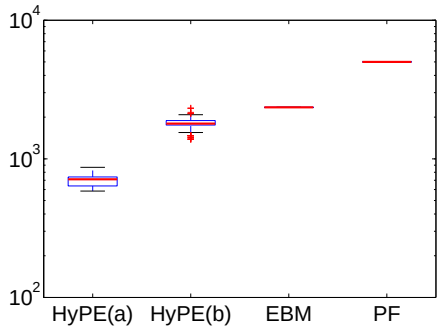
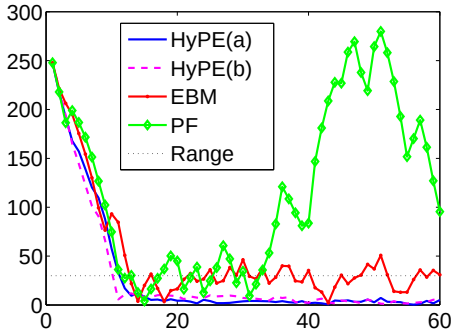
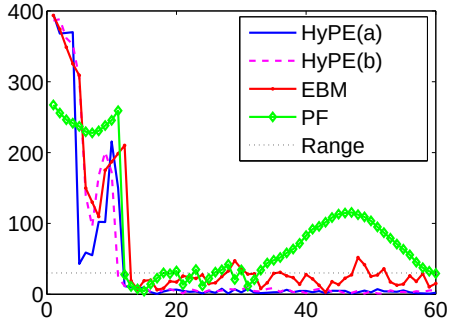


Fig. 6. Number of nodes used in each SAT implementation.

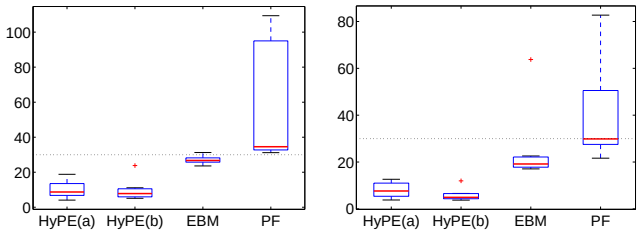


(a) 2D position error between sensor and target (m) vs. time step k.



(b) 2D position error between the posterior mode (mean for PF) and target (m) vs. time step k.

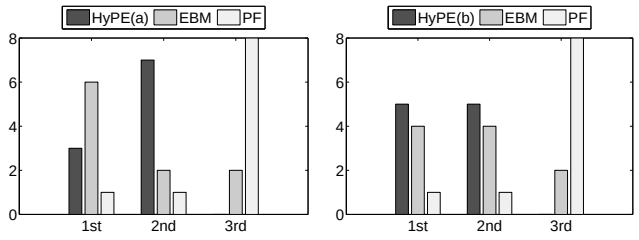
Fig. 7. Example of the evolution of position errors over time.



(a) Average 2D position error between sensor and target during tracking.

(b) Average 2D position error between posterior mode (mean for PF) and target during tracking.

Fig. 8. Average position errors during tracking.



(a) HyPE(a). (b) HyPE(b).

Fig. 9. Search Performance: Number of scenarios ranked 1,2 or 3.

V. Conclusion

This paper presented HyPE, a hybrid particle-element approach, suitable for recursive Bayesian SAT. The hybrid concept, to *synthesize* two RBE methods to represent and maintain the belief about all states in a dynamic system, is distinct from the concept behind "mixed approaches", such as Rao-Blackwellized particle filtering, which use different RBE methods for different states. HyPE performs a *static conversion* between an element-based representation, used during update and evaluation of the PDF, and a particle-based representation, used for Bayesian prediction, and thus eliminates the need for computationally expensive numerical integration. The static conversion process also allows reconfiguration, via remeshing of the target space, at minimal computational cost (the cost associated with remeshing only). The efficacy of the proposed approach was shown through a number of simulated SAT scenarios. Furthermore, it was shown that the ability to quickly remesh the search space allowed HyPE to reduce estimation and position errors by over 39%, whist also using as little as 14% of the number of sample points used by existing methods for RBE.

It should be noted that the two static conversion processes necessarily introduce a degree of approximation error into the estimation which would not otherwise occur. Furthermore, the resolution of the mesh with respect to the range of motion in the target states must be considered, as within the GBM and EBM. If the mesh is too coarse, the predicted particles may never escape the influence of the nodes from which they are sampled.

As this paper represents a first proof of concept for the hybrid particle-element approach, investigation of some of the practical consequences of using the approach remains. For example, an investigation into how the time and memory requirements and transmissibility scale with the scope and dimension of the problem, has been left for future work. Also, the reduction in the number of sampling points required using HyPE may hold significance for the area of data communication in multi-robot cooperation and coordination missions. This will be a focus of future work.

References

[1] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA: MIT Press, 2005.

[2] T. Furukawa, F. Bourgault, B. Lavis, and H. F. Durrant-Whyte, "Recursive Bayesian search-and-tracking using coordinated UAVs for lost targets," in *Proc. IEEE Int. Conf. Robot. Autom.*, Orlando, Florida, May 2006, pp. 2521–2526.

[3] J. M. Dobbie, "A survey of search theory," *Operations Research*, vol. 16, no. 3, pp. 525–537, May 1968.

[4] L. D. Stone, "Search theory: a mathematical theory for finding lost objects," *Mathematics Magazine*, vol. 50, no. 5, pp. 248–256, Nov. 1977.

[5] ——, *Theory of Optimal Search*. Arlington, VA: Operations Research Society of America (ORSA) Books, 1989.

[6] ——, "What's happened in search theory since the 1975 Lanchester prize?" *Operations Research*, vol. 37, no. 3, pp. 501–506, May-Jun 1989.

[7] H. R. Richardson, L. D. Stone, W. R. Monach, and J. H. Discenza, "Early maritime applications of particle filtering," O. E. Drummond, Ed., vol. 5204, no. 1. SPIE, 2003, pp. 165–174.

[8] F. Bourgault, T. Furukawa, and H. F. Durrant-Whyte, "Coordinated decentralized search for a lost target in a Bayesian world," in *IEEE/RSJ Int. Conf. Intel. Robot. Sys.*, 2003, pp. 48–53.

[9] ——, "Process model, constraints and the coordinated search strategy," in *IEEE Int. Conf. Robot. Autom.*, vol. 5, 2004, pp. 5256–5261.

[10] D. Salmond, "Target tracking: introduction and Kalman tracking filters," in *Proc. IEE Target Tracking: Algorithms and Applications*, vol. Workshop, Oct. 2001, pp. 1/1–1/16 vol.2, (Ref. No. 2001/174).

[11] A. E. Nordsjo, "Target tracking based on Kalman-type filters combined with recursive estimation of model disturbances," in *Proc. IEEE Int. Radar Conf.*, May 2005, pp. 115–120.

[12] C. Hue, J.-P. Le Cadre, and P. Pérez, "Sequential Monte Carlo methods for multiple target tracking and data fusion," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 309–325, Feb. 2002.

[13] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Transactions on Automatic Control*, vol. 45, no. 3, pp. 477–482, Mar. 2000.

[14] D. Guo and X. Wang, "Quasi-Monte Carlo filtering in nonlinear dynamic systems," *IEEE Transactions on Signal Processing*, vol. 54, no. 6, pp. 2087–2098, Jun. 2006.

[15] D. Siegmund, "Importance sampling in the Monte Carlo study of sequential tests," *The Annals of Statistics*, vol. 4, no. 4, pp. 673–684, Jul. 1976.

[16] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, Jul. 2000.

[17] T. Furukawa, H. F. Durrant-Whyte, and B. Lavis, "The element-based method - theory and its application to Bayesian search and tracking," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, San Diego, California, Oct. 2007, pp. 2807–2812.

[18] B. Lavis, T. Furukawa, and H. F. Durrant-Whyte, "Dynamic space reconfiguration for Bayesian search and tracking with moving targets," *Int. Journ. Auton. Robot.*, Jan. 2008, in print.

[19] B. Lavis and T. Furukawa, "Particle filters for estimation and control in search and rescue using heterogeneous UAVs," in *Proc. 4th Int. Conf. Comp. Intel., Robotics and Auton Sys.*, 28-30 November 2007, pp. 217–222.

[20] C. Stachniss, G. Grisetti, and W. Burgard, "Information gain-based exploration using rao-blackwellized particle filters," in *Robotics: Science and Systems*, 2005, pp. 65–72.