Proceedings of 2004 IEEE/RSJ International Conference on
Intelligent Robots and Systems
September 28 - October 2, 2004, Sendai, Japan

Decentralized Bayesian Negotiation for Cooperative Search

Frédéric Bourgault, Tomonari Furukawa and Hugh F. Durrant-Whyte
ARC Centre of Excellence for Autonomous Systems
The University of Sydney
Sydney, NSW 2006, Australia
{f.bourgault, t.furukawa, hugh}@cas.edu.au

Abstract—**This paper addresses the problem of coordinating a team of multiple heterogeneous sensing platforms searching for a single lost target. In this approach, the utility of a control sequence is a function of the probability density function (PDF) of the target state. Each decision maker builds an equivalent estimate of this PDF by communicating and fusing the information from the other sensor nodes. Coupled utilities incite the agents to collaborate and to agree on the next best set of actions. Decentralized cooperative planning is achieved via anonymous negotiation based on communication of expected observed information. Simulation results demonstrate the efficiency of the cooperative trajectories for a team of autonomous airborne search vehicles.**

I. Introduction

Interconnecting a number of active mobile sensors into a network may offer great potential for improved area coverage, robustness and responsiveness. The question remains of how to coordinate the sensing platforms effort in an optimal manner.

This paper explores the issue of optimal control decision making from a team perspective. Communication and coupled utility between decision makers are considered to be the fundamental mechanisms underlying cooperation. The focus is on obtaining, in a completely decentralized fashion, the optimal cooperative solution that considers the effects of everyone's control decisions on the global utility. Ref. [11] showed that this type of decentralized active sensing problem is equivalent to a distributed computation problem [3] as long as the individual decision makers have a way of evaluating the effect of the teammates predicted sensing actions on the global utility. For the searching problem, the information representation through which the individual utilities are coupled is in the form of the Probability Density Function (PDF) of the target state [7]. In the proposed negotiation algorithm, the decision makers maintain equivalent estimates of the future PDF. They anonymously exchange their expected observation likelihoods and iteratively adjust their reactions to the teammates previous actions until the best reaction, to the previous action, is itself, leading to the Nash equilibrium [14] for a given time horizon. This sort of decentralized architecture offers increased reactivity, robustness and scalability by avoiding the overheads, bottlenecks and single failure points associated with centralized structures.

The paper is organized as follows. First, Sec. II reviews the decentralized Bayesian filtering algorithm that accurately maintains and updates the information about the target state. Then Sec. III formulates the team decision problem and introduces the decentralized Bayesian negotiation algorithm. Sec. IV describes the searching problem. Sec. V demonstrates the efficiency of the cooperative search strategy for a team of unmanned air vehicles (UAVs) searching for a single stationary target. Finally, conclusions and ongoing research directions are highlighted.

II. Active Bayesian Sensor Network

This section reviews the decentralized Bayesian framework introduced in [7] for coordinating the search effort of a robotic team. The Bayesian approach is particularly suitable for combining heterogeneous non-Gaussian sensor observations with other sources of quantitative and qualitative information [2], [19].

A. Decentralized Bayesian Filtering

In the searching problem, the unknown variable of interest is the target state vector at time k, denoted $\mathbf{x}_k^t \in \mathbb{R}^{n_x}$ which in general describes the target location but could also include its attitude, velocity, and other properties. In this paper the superscripts t and s_i indicate a relationship to the target and the sensor i respectively. The subscripts are used to indicate the time index. The purpose of the analysis is to find an estimate for $p(\mathbf{x}_k^t|\mathbf{z}_{1:k})$, the PDF over $\mathbf{x}_k^t$ given the sequence $\mathbf{z}_{1:k} = \{\mathbf{z}_j^i : i = 1,...,N_s, j = 1,...,k\}$ of all the observations made from the N_s sensors on board the search vehicles, $\mathbf{z}_j^i$ being the observation from the i^{th} sensor at time step j. The analysis starts by determining a prior PDF $p(\mathbf{x}_0^t|\mathbf{z}_0) \equiv p(\mathbf{x}_0^t)$ for the target state at time 0, given all available prior information including past experience and domain knowledge. If nothing is known other than initial bounds on the target state vector, then a least informative uniform PDF is used as the prior. Once the prior distribution has been established, the PDF at time step k, $p(\mathbf{x}_k^t|\mathbf{z}_{1:k})$, can be constructed recursively using the prediction and update equations alternatively.

1) Prediction: Suppose the system is at time step $k-1$ and the latest PDF update, $p(\mathbf{x}_{k-1}^t|\mathbf{z}_{1:k-1})$, is available. Then the predicted PDF of the target state at time step k is obtained from the following Chapman-Kolmogorov equation

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

0-7803-8463-6/04/$20.00 ©2004 IEEE

where $p(\mathbf{x}_k^t|\mathbf{x}_{k-1}^t)$ is a probabilistic Markov motion or process model which maps the probability of transition from a given previous state $\mathbf{x}_{k-1}^t$ to a destination state $\mathbf{x}_k^t$ at time k. The process model is a function of the equations of motion for the target and of the known distribution on their inputs. Various examples of process models with constraints can be found in [6].

2) Update: At time step k, a new set of observations $\mathbf{z}_k = \{\mathbf{z}_k^1, ..., \mathbf{z}_k^{N_s}\}$ becomes available. For each sensor i, the mapping of the target state observation probability, $\mathbf{z}^i \in \mathbb{R}^{n_z}$, for each given target state, $\mathbf{x}_k^t \in \mathbb{R}^{n_x}$, is denoted $p(\mathbf{z}_k^i|\mathbf{x}_k^t)$ and will be referred to as the observation likelihood, or sensor model. Assuming all the observations to be conditionally independent, the PDF update from the prediction stage (1), $p(\mathbf{x}_k^t|\mathbf{z}_{1:k-1})$, is performed using the following Bayes rule with the normalization coefficient K, also referred to in the literature as the "independent opinion pool"

$$p(\mathbf{x}_k^t|\mathbf{z}_{1:k}) = K\, p(\mathbf{x}_k^t|\mathbf{z}_{1:k-1}) \prod_{i=1}^{N_s} p(\mathbf{z}_k^i|\mathbf{x}_k^t) \tag{2}$$

where the normalization coefficient K is given by

$$K = 1/\int \left[p(\mathbf{x}_k^t|\mathbf{z}_{1:k-1}) \prod_{i=1}^{N_s} p(\mathbf{z}_k^i|\mathbf{x}_k^t)\right] d\mathbf{x}_k^t \tag{3}$$

B. Active Bayesian Sensor Node

Packaging a physical sensor with its own Bayesian filtering processor is an attractive way of making the sensor mobile and modular. Such a Bayesian sensor unit can be taken anywhere to take measurements about the world. Mounting the Bayesian sensor unto an actuated mobile Platform and coupling it to its own Controller makes it an active Bayesian sensor. Based on the latest belief about the world $p(\mathbf{x}_{k-1}^t|\mathbf{z}^{1:k-1})$ and the sensor state $\mathbf{x}_{k-1}^{s_i}$, the Controller sends a command $\mathbf{u}_{k-1}^{s_i}$ to the Platform to place the sensor in a desired position $\mathbf{x}_{k_{des}}^{s_i}$ with respect to the world to take the next observation. Fig. 1 depicts

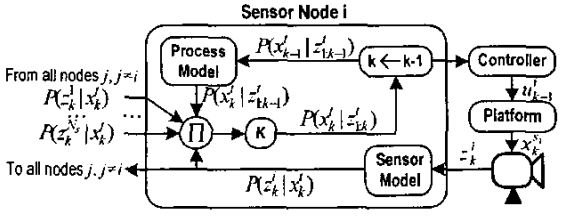

Fig. 1. General active Bayesian sensor node in a fully connected network with broadcast communications.

algorithmically the Bayesian filter and how it interacts with the Controller, the Platform, and the sensor to form a node in a fully connected network. To allow point-to-point communication between the nodes of the network and recovery from transmission failures the node may be augmented with an extra filter per communication channel called a channel filter [5]. The Platform block represents the actuators and dynamics of both the sensor and the mobile vehicle, if present, on which the sensor is mounted. Any number of sensors can be attached to a particular fusion node. For simplicity in this paper, each sensor is packaged with its own fusion node and controller.

For the searching problem, the process model, and especially the target PDF can be highly non-Gaussian and the complete description of the density function must be maintained. This prevent the use of parametric implementations of the Bayesian filter such as the Kalman filter. In this paper the prediction and update equations will be evaluated numerically using a grid based discrete approximation of the process model, the observation likelihood and the target PDF.

III. Decentralized Cooperative Control

Each of the N_s sensor platform is governed by its own dynamic model in the form

$$\mathbf{x}_{k+1}^{s_i} = \mathbf{f}_k^{s_i}(\mathbf{x}_k^{s_i}, \mathbf{u}_k^{s_i}, \mathbf{w}_k^{s_i}) \tag{4}$$

where $\mathbf{w}_k^{s_i}$ is the vector representing the process noise and the external forces acting on the system i, and where $\mathbf{u}_k^{s_i}$ is the corresponding control input vector at time k. The controller objective is to produce a command that will place the system in a desired state.

A. Optimal Trajectory

Optimality is defined in relation to an objective, or utility function [20]. For multiple systems, an optimal cooperative control solution must the group decision that is jointly optimal. Given N_k lookahead steps, the global utility function is denoted $J(\mathbf{u}, N_k)$, where $\mathbf{u} = \mathbf{u}_{1:N_k} = \{\mathbf{u}_{1:N_k}^{s_1}, ..., \mathbf{u}_{1:N_k}^{s_{N_s}}\}$ is the control action sequence for all platforms over a time horizon of length $T = N_k\, \delta t$. The optimal control trajectory $\mathbf{u}^*$ is the sequence that maximizes that utility subject to the control bounds $\mathbf{u}_{LB} \le \mathbf{u} \le \mathbf{u}_{UB}$ and the constraints $\mathbf{g}(\mathbf{u}, N_k) \le \mathbf{0}$.

$$\mathbf{u}^* = \{\mathbf{u}_{1:N_k}^{s_1 *}, ..., \mathbf{u}_{1:N_k}^{s_{N_s} *}\} = \arg\max_{\mathbf{u}} J(\mathbf{u}, N_k) \tag{5}$$

To be truly optimal, the trajectory should be evaluated for the entire duration of the mission. However, the computational cost for such optimal plans is subject to the "curse of dimensionality". With increasing lookahead depth and number of agents, the solution becomes intractable. In practice only solutions for a restricted number of lookahead steps are possible. One way to increase the lookahead without significantly increasing the cost of the solution is to have a piecewise constant control sequence (see [12] and [9]) where each control parameter is maintained over a specified number of time steps. Such control solutions are said to be *quasi-optimal* as they compromise the global optimality of the control solution for a lower computation cost, but nevertheless, depending on the problem at hand and because of their anticipative characteristic, often provide better trajectories than the ones computed with the same number of control parameters but with shorter time horizons. A *rolling time horizon* solution is when the planned trajectory is recomputed at short intervals to keep the lookahead constant as the agents progress forward. At time step k, the utility for a given horizon depth of N_k steps will be denoted $J_k(\mathbf{u}, N_k)$ with $\mathbf{u} = \mathbf{u}_{k:k+N_k-1}$ being the action sequence starting at step k.

B. Team Utility Structure

The global utility J_k can be decomposed into its individual components as in

$$J_k(\mathbf{u}, N_k) = \sum_{i=1}^{N_s} J_k^{s_i}(\mathbf{u}, N_k) = \sum_{i=1}^{N_s} J_k^{s_i}(\mathbf{u}^{s_i}, \bar{\mathbf{u}}^{s_i}, N_k) \quad (6)$$

where each $J_k^{s_i} : \mathbb{R}^{n_i} \mapsto \mathbb{R}$ corresponds to the reward received by the ith decision maker within a team for a control sequence $\mathbf{u}^{s_i} = \mathbf{u}_{k:k+N_k-1}^{s_i}$ given the teammates strategy $\bar{\mathbf{u}}^{s_i} = \{\mathbf{u}_{k:k+N_k-1}^{s_j} : j \neq i\}$. Such situation, where a decision maker's likely reward depends on the actions of others is called a game. In a game, a decision maker needs to choose a strategy that maximizes his utility not only based on his individual preferences alone, but on the likely actions of the other teammates as well. The optimal control sequence $\mathbf{u}^*$ satisfies the following *person-by-person optimality* condition, also referred to as the Nash equilibrium solution [13]. That is

$$J_k^{s_i}(\mathbf{u}^*, N_k) \geq J_k^{s_i}(\mathbf{u}^{s_i}, \bar{\mathbf{u}}^{s_i *}, N_k), \quad \forall \mathbf{u}^{s_i} \in \mathbf{U}^{s_i}, \forall i \quad (7)$$

where $\bar{\mathbf{u}}^{s_i *} = \{\mathbf{u}_{k:k+N_k-1}^{s_j *} : j \neq i\}$. At equilibrium, an individual cannot diverge from his Nash strategy without decreasing his utility. A game is inherently a distributed problem and as such is well suited to be solve by distributed computation approaches.

C. Distributed Computation Solution

In this section it is proposed to approach the multi-vehicle multi-sensor optimization problem (5) as a distributed computation problem [3]. By taking advantage of the resources from multiple processors, distributed computation methods have a significant speed advantage over centralized methods. The control problem (5) is broken down into smaller components. At every iteration, each processor optimizes its own component and communicate the result on the network. The proposed block-iterative non-linear algorithm is a coordinate descent type technique. It consists in iteratively fixing all the components of $\mathbf{u}$, except for the ith block-component and then maximizing $J_k(\mathbf{u}, N_k) = J_k(\mathbf{u}^{s_i}, \bar{\mathbf{u}}^{s_i}, N_k)$ with respect to $\mathbf{u}^{s_i}$, and repeating for all components until convergence, i.e. until condition (7) is satisfied.

When the maximizations with respect to the different components, $\mathbf{u}^{s_i}$'s, are carried out simultaneously it may be referred to as a *Jacobi* type algorithm. When the maximizations are performed sequentially for each component, it is referred to as a *Gauss-Siedel* type algorithm [3].

Each optimization iteration l requires the parallel, or sequential, solution of the individual optimization problems

$$\mathbf{u}_{|l}^{s_i *} = \arg\max_{\mathbf{u}^{s_i}} J_k^{s_i}(\mathbf{u}^{s_i}, \bar{\mathbf{u}}_{|l-1}^{s_i}, N_k) = \arg\max_{\mathbf{u}^{s_i}} J_k(\mathbf{u}^{s_i}, \bar{\mathbf{u}}_{|l-1}^{s_i}, N_k) \quad (8)$$

where $\bar{\mathbf{u}}_{|l-1}^{s_i}$ corresponds to the information (possibly obsolete) controller i has about the other teammates decisions, and $\mathbf{u}_{|l}^{s_i *}$ is the corresponding best response solution of decision maker i at iteration step l. As such, $(\mathbf{u}_{|l}^{s_i *}, \bar{\mathbf{u}}_{|l-1}^{s_i})$ is a pareto-optimal solution. The algorithm has converged when the best response to everyone else's best response is fixed. In practice, the iterations are stopped when the optimization error has decreased under a desired accuracy threshold, i.e. when $\delta \mathbf{u}_{|l}^{s_i *} = \mathbf{u}_{|l}^{s_i *} - \mathbf{u}_{|l-1}^{s_i} < \varepsilon^i, \forall i$. It is important to notice however that such an equilibrium point might constitute a local maximum of J_k. Notice that the number of local maxima increases with the number of components in the planning sequence. In some situations, the optimization algorithm might get stuck oscillating between different equilibrium points. To prevent this sort of situation, a relaxation update equation is used

$$\mathbf{u}_{|l}^{s_i} = \mathbf{u}_{|l-1}^{s_i} + \alpha_l(\mathbf{u}_{|l}^{s_i *} - \mathbf{u}_{|l-1}^{s_i}) = (1-\alpha_l)\mathbf{u}_{|l-1}^{s_i} + \alpha_l \mathbf{u}_{|l}^{s_i *} \quad (9)$$

where $\alpha_l \in]0,1]$ is the stepping or relaxation parameter that specifies the amount by which the solution is moved towards the best response. The relaxation parameter may vary throughout the negotiation process. For example, the algorithm may start with a value close to one in order to converge quickly to a coarse estimate, and then progressively decreased towards zero to refine the solution. Experience has also shown that a random α_l is greatly useful to break the symmetries causing the algorithm to oscillate.

Reference [21] introduced a slightly different algorithm that exploits this idea of adding a stochastic element to the update equation. That algorithm was successfully applied in [11] to a multi-vehicle control problem for target identification. Simulated annealing is another optimization technique that uses stochastic updates to get out of local maxima. A version of the distributed evolutionary simulated annealing technique presented in [1] is also currently under investigation.

D. Bayesian Negotiator

As discussed in the above section, in a standard distributed computation approach, after each iteration l the controller processor i would need the components from the other the controllers $\bar{\mathbf{u}}_{|l-1}^{s_i}$ in order to evaluate the global utility. This implicitly means that processor i would require the knowledge of the other sensors' location or characteristics, e.g. observation model and vehicle model, and would raise scalability issues. Rather, each controller builds an estimate of the future target PDF based on the current estimate and the predicted contributions received from the teammates. Based on this estimate, an appropriate reaction is evaluated.

Fig. 2 graphically depicts the controller negotiation algorithm. At a given time step k and iteration l, the controller i fuses the latest density estimate $p(\mathbf{x}_k^t|\mathbf{z}_{1:k})$ obtained from its fusion node with the predicted observation likelihoods $p(\mathbf{z}_{k+1:k+N_k|l-1}^j|\mathbf{x}_k^t)$'s associated with the optimized components $\mathbf{u}_{|l-1}^{s_j}$'s from all the other processors $j \neq i$ computed on the previous iteration. Based on the resulting predicted density estimate $p(\mathbf{x}_k^t|\mathbf{z}_{1:k}, \bar{\mathbf{z}}_{k+1:k+N_k|l-1}^i)$, the best response $\mathbf{u}_{|l}^{s_i *}$ is computed and the component $\mathbf{u}_{|l}^{s_i}$ is updated. If the optimization has not converged, the corresponding observation likelihood $p(\mathbf{z}_{k+1:k+N_k|l}^i|\mathbf{x}_k^t)$ is then broadcasted

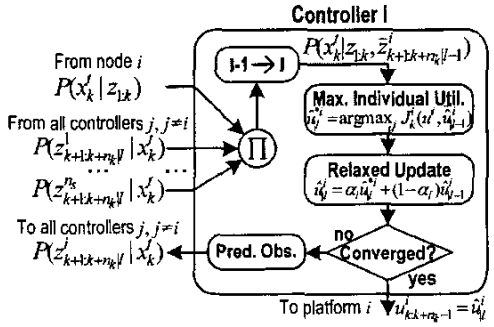

Fig. 2. Bayesian negotiator in a fully connected network with broadcast communications.

to the other controllers for another iteration. Otherwise, $\mathbf{u}^{s_i*} \approx \mathbf{u}^{s_i}_{|l} \rightarrow \mathbf{u}^{s_i}_{k+1:k+N_k}$ is sent to the Platform for execution. In this paper the individual optimization problems are solved using a constrained non-linear programming technique called Sequential Quadratic Programming (SQP) [15].

Notice that the negotiation algorithm as presented above is valid only for a stationary target, i.e. there is no process or prediction step involved. To take into account the process model, instead of being communicated in pre-combined blocks as $p(\mathbf{z}^i_{k+1:k+N_k|l-1}|\mathbf{x}^t_k)$'s, $\forall i$, the set of all the predicted observation likelihoods from each controller i would need to be broadcasted separately, as in $\{p(\mathbf{z}^i_{k+n|l-1}|\mathbf{x}^t_k) : n = 1,...,N_k\}$'s, $\forall i$, and stored so they could be fused at the right time in the optimization step. This would of course increase significantly the communication loads.

Also, the above iterative algorithm can be executed asynchronously. As discussed in [4], in an asynchronous implementation, the processors are allowed to iterate at their own pace on their respective component and are not required to wait to receive all messages generated during the previous iteration. If the latest predicted observation likelihood updated by some other processor is not available, then some outdated likelihood is used instead. Evidences suggest that asynchronous iterations converge faster than their synchronous counterparts [4]. The details of an asynchronous implementation of the above Jacobi algorithm robust to communication delays and transmission failures with point-to-point communication is out of the scope of this paper and will be the subject of an ulterior publication.

IV. The Search Problem

This section describes the equations for computing the probability of detection of a lost object referred to as the target by using the outputs of the prediction and update equations from Sec. II-A. An equivalent but different derivation is presented in [7]. Further details on the searching problem can also be found in [18] and [16] (Chap.9).

Let the target detection likelihood (observation model) of the i^{th} sensor at time step k be given by $p(\mathbf{z}^i_k = D^i_k|\mathbf{x}^t_k)$ where D^i_k represents a 'detection' event by sensor i at time k. The likelihood of 'no detection' by the same sensor is given by its complement $p(\overline{D}^i_k|\mathbf{x}^t_k) = 1 - p(D^i_k|\mathbf{x}^t_k)$. The combined 'no detection' likelihood for all the sensors at time step k is simply a multiplication of the individual 'no detection' likelihoods

$$p(\overline{D}_k|\mathbf{x}^t_k) = \prod_{i=1}^{N_s} p(\overline{D}^i_k|\mathbf{x}^t_k) \tag{10}$$

where $\overline{D}_k = \overline{D}^1_k \cap ... \cap \overline{D}^{N_s}_k$ represents the event of a 'no detection' observation by every sensor at time step k. Neglecting the normalization factor K in the update equation (2) gives

$$p(\mathbf{x}^t_k|\mathbf{z}_{1:k})' = p(\mathbf{x}^t_k|\mathbf{z}_{1:k-1})' \prod_{i=1}^{N_s} p(\mathbf{z}^i_k|\mathbf{x}^t_k) \tag{11}$$

The advantage of not normalizing the target PDF at every update is that the joint probability of failing to detect the target in all of the steps from 1 to k, denoted $Q_k = p(\overline{D}_{1:k})$, can be directly obtained from the integration of the pseudo PDF update (11)

$$Q_k = \int p(\mathbf{x}^t_k|\overline{D}_{1:k})' d\mathbf{x}^t_k = \int p(\mathbf{x}^t_k|\overline{D}_{1:k-1})' p(\overline{D}_k|\mathbf{x}^t_k) d\mathbf{x}^t_k \tag{12}$$

where $\overline{D}_{1:k}$ corresponds to the set of observations $\mathbf{z}_{1:k}$ where every observation is a 'no detection', i.e. $\mathbf{z}_k = \overline{D}_k, \forall k$. Then, it can be shown that the probability the target gets detected for the first time on time step k, denoted p_k, is given by the volume under the surface resulting from the product of the combined detection likelihood, denoted $[1 - p(\overline{D}_k|\mathbf{x}^t_k)] = p(D_k|\mathbf{x}^t_k)$, with the predicted target PDF. This is equivalent to the reduction in volume $(-\Delta Q_k)$ of the pseudo PDF as in

$$\begin{aligned} p_k &= \int p(\mathbf{x}^t_k|\overline{D}_{1:k-1})' \left[1 - p(\overline{D}_k|\mathbf{x}^t_k)\right] d\mathbf{x}^t_k \\ &= Q_{k-1} - Q_k \end{aligned} \tag{13}$$

Assuming no false detection from the sensors, the probability that the target *has* been detected in k steps, denoted P_k, is obtained from the cumulative sum of the p_k's as in

$$P_k = \sum_{i=1}^{k} p_i = P_{k-1} + p_k \tag{14}$$

For this reason P_k will be referred to as the 'cumulative' probability of detection to distinguish it from the payoff probability of detection function p_k. Notice that plugging the expressions for p_k from (13) into (14) gives $P_k = 1 - Q_k$ since $Q_0 = \int p(\mathbf{x}^t_0) d\mathbf{x}^t_0 = 1$. This signifies that if the target PDF is not normalized after each update as in (11), then its volume, Q_k, represents the residual probability that the target is still present despite the search effort expended. Also, as k goes to infinity, Q_k decreases towards zero and P_k levels off towards one as it becomes harder to generate additional observation payoff, p_k, from hardly any probability mass left in the PDF.

The goal of a searching strategy could be to maximize the chances of finding the target given a restricted amount of time by maximizing P_k over a given time horizon [8]. For a lookahead depth of N_k steps as discussed in Sec. III, the global utility function at time step k is given by the

probability of detecting the target during the next series of observations, which corresponds to the net increase in P_k.

$$J_k(\mathbf{u}, N_k) = \sum_{i=k}^{k+N_k} p_i = P_{k+N_k} - P_k \qquad (15)$$

V. Application

The goal of the ongoing research effort is to demonstrate the cooperative search framework on a team of heterogeneous autonomous mobile platforms in various outdoor scenarios. A stepping stone towards this goal is to investigate the problem using simulation. The chosen simulation scenario involves a team of unmanned air vehicles (UAVs), such as the ones illustrated in Fig. 3a, equipped with downward looking millimeter wave radars and searching for a single lost target, a liferaft (Fig. 3b). More about the implementation details of the framework and the search problem can be found in [7] and [8].

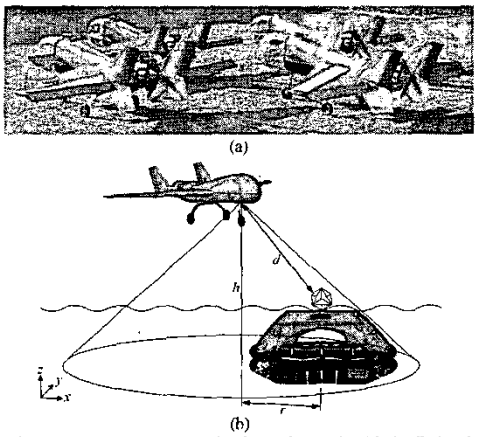

Fig. 3. Search scenario: (a) The fleet of Brumby Mark-III developed at ACFR. These UAVs have a payload capacity of up to 13.5 kg and operational speed of 50 to 100 knots; (b) Search sensor aperture cone and geometrical relationship between the search vehicle and the target.

Fig. 4 presents the two vehicles negotiation results for the algorithm presented in Sec. III. The prior target PDF is a Gaussian density with a standard deviation of 500m in each direction. Both UAVs are flying at 50 m/s and are trying to optimize their single control input, the turn rate in rad/s, for the next 30 seconds. As can be seen from the iteration results (Figs. 4a and b), relaxing the *Jacobi* algorithm, i.e. $\alpha_l < 1$, often improves the convergence rate. In this example, there is only one global maximum (Fig. 4c) and the algorithm is guaranteed to converge to it.

Fig. 5 illustrates the advantage of a cooperative solution over a flight formation flyby or the coordinated search solution obtained [7]. For the cooperative solution (Figs. 5a-c), a rolling time horizon of 30 steps, constituted of 3 control parameters, each maintained for 10 steps, is renegotiated every 10 steps. In the coordinated solution, each vehicle follows a greedy 1-step lookahead solution. By allowing a more efficient allocation of the search effort,

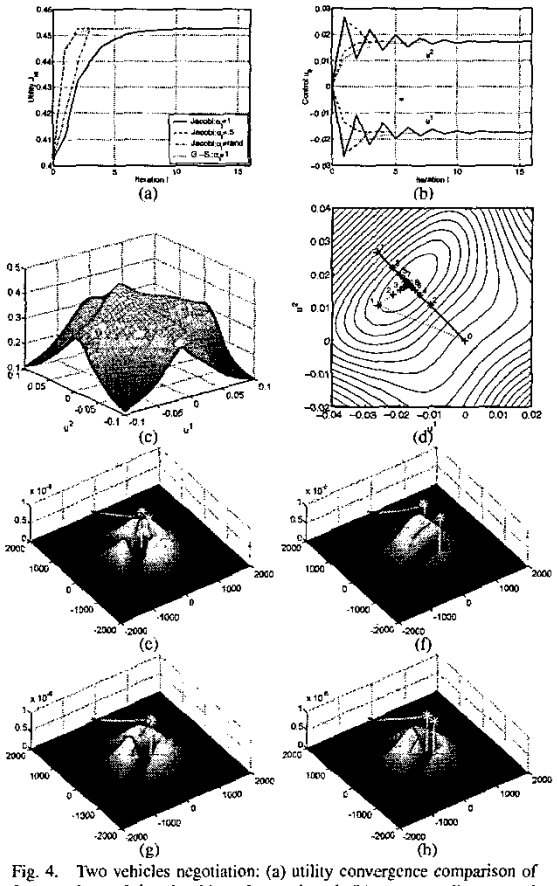

Fig. 4. Two vehicles negotiation: (a) utility convergence comparison of four versions of the algorithm, $J_{k|l}$ vs. iter. l; (b) corresponding control actions convergence, $\mathbf{u}_{k|l}$ [rad/s] vs. iter. l; (c) 3D view of the global utility function; (d) contours of the global utility function and convergence of the *Jacobi* and *Gauss-Siedel* best response ($\alpha_l = 1$) algorithms; (e)-(h) 3D views of the target PDF and vehicle trajectories for iteration steps l=0,1,2 and 17 respectively of the *Jacobi* best response algorithm.

the cooperative approach compares advantageously to both coordinated (Fig. 5g) and the flight formation (Fig. 5h) search strategies. In fact after 160s, the time needed for the formation to traverse the search area, the cooperating vehicles reach a final P_k of $P_{160} = .818$ vs. $P_{160} = .718$ for the coordinated solution and $P_{160} = .575$ for the flight formation, which correspond to an increase of a 13.9% and 42.3% respectively over the later solutions (Fig. 5d).

VI. Summary and Ongoing Work

A novel approach to cooperative control based on a decentralized negotiation algorithm that increases the time horizon of the decentralized search plans was presented. On an anonymous basis, decision makers interact to find cooperative search plans based on both observed and predicted information that explicitly consider the search vehicle kinematics, the sensor detection function, as well as the target arbitrary motion model.

With increased lookahead comes an increase added value

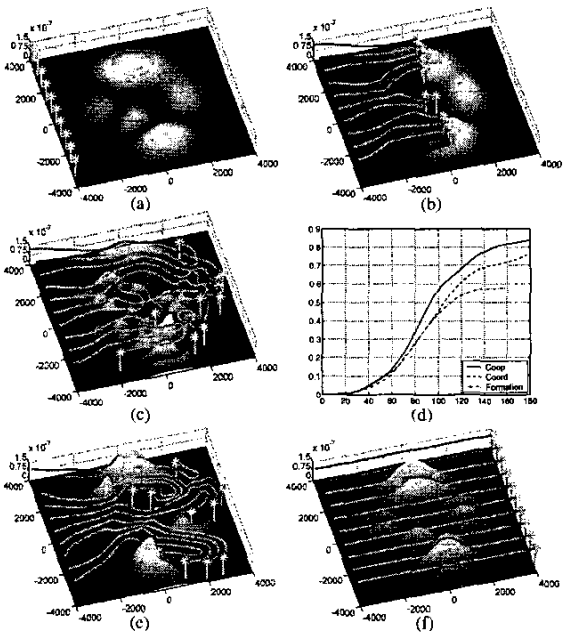

Fig. 5. Cooperative search for a static target with 10 vehicles: (a) to (c) 3D views of the target PDF and the cooperative trajectories at time steps k = 1, 90 and 180s respectively; (d) comparison of the cumulative probability of detection, P_k vs. k, for the cooperative, coordinated and the straight pattern search; (e) coordinated search at $k = 180$s, and (f) straight pattern flight formation search at $k = 160$s.

in cooperation. This gain in team utility however comes at the cost of increased communication and computation loads. The level of attainable optimality of the solution must therefore be traded-off against the system communication bandwidth, computing power and available time.

The goal of the negotiation algorithm based on broadcasted communication as presented in this paper was to demonstrate the decentralized negotiation principles. Although such broadcasts communication facilities could be implemented, for some overhead costs, with *spanning-trees* [3], such implementation would not be quite realistic. In a physical multi-vehicle implementation, the processors sensor nodes may be far apart and their line-of-site may be obstructed. Communication delays may be unpredictable, and the communication links themselves may be unreliable. Other implementations issues such as the algorithm termination in an anonymous point-to-point communication system were also out of the scope. Such relevant implementation issues will be the subject of a further publication.

As part of the ongoing research effort, techniques such as Monte Carlo methods, or particle filters [10], as well as the so called kernel methods for density estimation [17] are being investigated to overcome the "curse of dimensionality" limitations of the grid based approach presented. As well, techniques to facilitate human interactions with the active sensor network are being investigated to enable an operator to enter observations in the network and influence the agents control decisions.

ACKNOWLEDGEMENTS

This work is partly supported by the ARC Centre of Excellence programme, funded by the Australian Research Council (ARC) and the New South Wales State Government, and by AFOSR/AOARD under contract 03-13.

REFERENCES

[1] M.E. Aydin and T.C. Forgarty. A distributed evolutionary simulated annealing algorithm for combinatorial optimisation problems. *Journal of Heuristics*, 10:269–292, 2004.

[2] J.O. Berger. *Statistical decision theory and Bayesian analysis*. Springer series in statistics. Springer-Verlag, New York, 2nd edition, 1985.

[3] D. Bertsekas and J. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Englewood Cliffs, NJ, 1989.

[4] D.P. Bertsekas and J.N. Tsitsiklis. Some aspects of parallel and distributed iterative algorithms - a survey. *Automance*, 27(1):3–21, 1991.

[5] F. Bourgault and H.F. Durrant-Whyte. Communication in general decentralized filters and the coordinated search strategy. In *The 7th Int. Conf. on Information Fusion*, Stockholm, Sweden, June 2004.

[6] F. Bourgault and H.F. Durrant-Whyte. Process model, constraints, and the coordinated search strategy. In *IEEE Int. Conf. on Robotics and Automation (ICRA'04)*, New Orleans, USA, April 2004.

[7] 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. on Intelligent Robots and Systems (IROS'03)*, Las Vegas, USA, October 2003.

[8] F. Bourgault, T. Furukawa, and H.F. Durrant-Whyte. Optimal search for a lost target in a bayesian world. In *Int. Conf. on Field and Service Robotics (FSR'03)*, Mt.Fuji, Japan, July 2003.

[9] T. Furukawa. Time-subminimal trajectory planning for discrete nonlinear systems. *Engineering Optimization*, 34:219–243, 2002.

[10] 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, April 1993.

[11] B. Grocholsky. *Information-Theoretic Control of Multiple Sensor Platforms*. Phd thesis, The University of Sydney, 2002.

[12] H.J.W. Lee, K.L. Teo, V. Rehbock, and L.S. Jennings. Control parametrization enhancing technique for time optimal control problems. *Dyn. Sys. and Appl.*, 6(2):243–262, April 1997.

[13] J.F. Nash. The bargaining problem. *Econometrica*, 18(2):155–162, 1950.

[14] J.F. Nash. Equlibrium in n-person games. *Proceedings of NAS*, 36(1):48–49, 1950.

[15] G.L. Nemhauser, A.H.G. Rinnooy Kan, and M.J. Todd. *Vol. 1: Optimization*. Handbooks in Operations Management Science. Elsevier Science Publishers B.V., Amsterdam, 1989.

[16] J.S. Przemieniecki. *Mathematical Methods in Defense Analyses*. AIAA Education Series, American Institute of Aeronautics and Astronautics, Inc., Washington, DC, 2nd edition, 1994.

[17] B.W. Silverman. *Density Estimation for Statistics and Data Analysis*. Monographs on statistics and applied probability; 26. Chapman and Hall, London; New York, 1986.

[18] L.D. Stone. *Theory of Optimal Search*, volume 118 of *Mathematics in Science and Engineering*. Academic Press, New York, 1975.

[19] L.D. Stone, C.A. Barlow, and T.L Corwin. *Bayesian Multiple Target Tracking*. Mathematics in Science and Engineering. Artech House, Boston, 1999.

[20] K.L. Teo, C.J. Goh, and K.H. Wong. *A Unified Computational Approach to Optimal Control Problems*. Longman Scientific and Technical, 1991.

[21] R. Wilson. Computing equilibria of n-person games. *SIAM Journal on Applied Mathematics*, 21(1):80–87, 1971.