2008 IEEE/RSJ International Conference on Intelligent Robots and Systems
Acropolis Convention Center
Nice, France, Sept, 22-26, 2008

Scalable Bayesian Human-Robot Cooperation in Mobile Sensor Networks

Frédéric Bourgault, Aakash Chokshi, John Wang, Danelle Shah, Jonathan Schoenberg,
Ramnath Iyer, Franco Cedano and Mark Campbell

Abstract— **In this paper, scalable collaborative human-robot systems for information gathering applications are approached as a decentralized Bayesian sensor network problem. Human-computer augmented nodes and autonomous mobile sensor platforms are collaborating on a peer-to-peer basis by sharing information via wireless communication network. For each node, a computer (onboard the platform or carried by the human) implements both a decentralized Bayesian data fusion algorithm and a decentralized Bayesian control negotiation algorithm. The individual node controllers iteratively negotiate anonymously with each other in the information space to find cooperative search plans based on both observed and predicted information that explicitly consider the platforms (humans and robots) motion models, their sensors detection functions, as well as the target arbitrary motion model. The results of a collaborative multi-target search experiment conducted with a team of four autonomous mobile sensor platforms and five humans carrying small portable computers with wireless communication are presented to demonstrate the efficiency of the approach.**

I. Introduction

This paper proposes an innovative scalable Bayesian approach for coordinating a network of humans and robots involved in information gathering type missions. The concept of a meta-node, to represent mobile robotic sensors and human-computer augmented systems, is introduced as a fundamental building block of a decentralized Active Sensor Network (ASN) architecture which couples decentralized communication, estimation and control.

In this approach the human-computer augmented node constitute a mobile sensor unit where the human provides both the sensors and their carrying "platform", while the portable computer runs both a decentralized Bayesian fusion node and a decentralized Bayesian control negotiation algorithm. The networked controller nodes iteratively negotiate anonymously in the information space to find cooperative search plans based on both observed and predicted information that explicitly consider the humans and robots motion models, their sensors detection functions, as well as the targets arbitrary motion model.

This type of decentralized architecture offers increased efficiency, reactivity, robustness and scalability by avoiding the overheads, bottlenecks and single points of failure associated with centralized structures. The proposed methodology enables synergistic human-machine interactions, with applications search and rescue, planetary exploration, mapping, environmental monitoring, disaster relief, urban warfare, surveillance and security systems.

This paper is organized as follows. First, Sec. II describes the Active Sensor Network architecture and reviews the decentralized Bayesian data fusion algorithm. Sec. III reviews the team utility structure and explains the decentralized iterative Bayesian control negotiation algorithm. Sec. IV covers the search problem. Sec. V presents the details of a search experiment and discusses some of the results. Finally, conclusions and ongoing research directions are outlined in Sec. VI.

II. Architecture

The general decentralized framework proposed for coordinating the search effort of a team of mobile robotic sensors and human-computer augmented units combines general hierarchical decentralized data fusion and decentralized hierarchical control into a coherent architecture. This type of approach is related to Active Sensor Networks (ASNs) [14].

Figure 1 depicts an example of a simple information driven decentralized mobile sensor network with three "meta-nodes". The *black* circles represent high level control nodes in different hierarchical control layers, while the *white* circles represent decentralized data fusion nodes that compile hierarchical levels of information used by the control nodes. For each meta-node, or mobile sensor unit, the *Platform* block represents the sensor carrying platform with its internal low-level feedback *Controller*, physical *Body* and associated mobility characteristics (i.e. robotic vehicle or human body), and payload *Sensor(s)*. This representation of mobile sensor nodes is general and may be applied to fully autonomous robots, remote-controlled platforms, manned vehicles as well as humans, with natural and/or artificial sensors. The strategic and tactical planning levels may be fully automated, or based on human decisions. For simplicity in this paper, a single layer of decentralized control, i.e. tactical trajectory planning, and low-level data fusion are implemented in fully autonomous fashion. Both functions are supported by a portable computers carried by the humans and the robots' onboard computers over a wireless had-hoc network.

A. Decentralized Bayesian Filtering

In Bayesian analysis any unknown quantity of interest is considered a random variable. The state of knowledge about such a random variable is entirely expressed in the form of a Probability Density Function (PDF). New information in the form of a probabilistic measurement or observation is combined with the previous PDF using Bayes' theorem in order to update the state of knowledge. This newly updated PDF, along with utility measures based on its prediction,

This work is partly supported by the U.S. Army Research Laboratory's Army Research Office (ARO) grant W911NF-07-1-0312, and the Air Force Office of Scientific Research (AFOSR) grant FA9550-06-1-0280.

F. Bourgault, A. Chokshi, J. Wang, D. Shah, J. Schoenberg, R. Iyer, F. Cedano and M. Campbell are with The Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY 14853, USA {fb42,adc32,jbw48,dcs45,jrs55,rri4,fac24,mc288}@cornell.edu

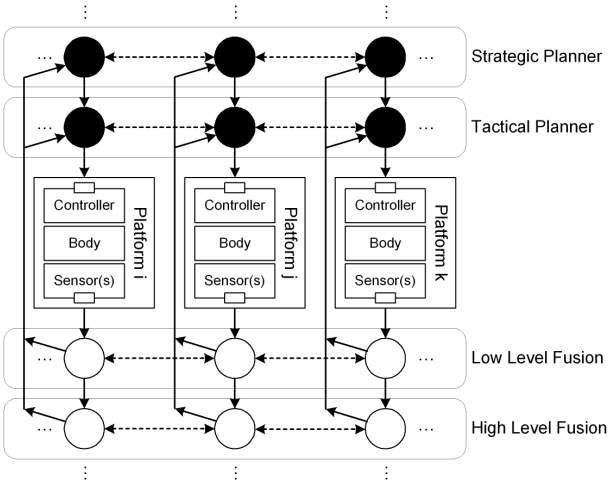


Fig. 1. Three meta-nodes used to notionally define a mobile Bayesian sensor network with human and robotic elements. The *white* circles linked by the dashed lines represent the general hierarchical decentralized data fusion network, while the *black* circles represent the nodes in the networked planning architecture.

forms the quantitative basis on which all inferences, or control decisions are made.

For the multi-target search problem, with a number N_s of search sensors and N_t of targets, the random variable of interest is the joint targets state vector, denoted $\mathbf{x}_k^t = \{\mathbf{x}_k^{t_j} : j = 1, ..., N_t\}$ where $\mathbf{x}_k^{t_j} \in \Re^{n_{x^t}}$ represents the individual state vector for the j^{th} target at time step k. In general, $\mathbf{x}_k^t$ describes the targets locations but could also include their attitudes, velocities and other properties. In this paper, the superscripts t_j and s_i indicate a relationship to the target j and the sensor onboard the search vehicle i respectively. The subscripts are used to indicate the time index. Hence the joint sensors state are denoted $\mathbf{x}_k^s = \{\mathbf{x}_k^{s_i} : i = 1, ..., N_s\}$ where $\mathbf{x}_k^{s_i} \in \Re^{n_{x^s}}$. For simplicity of the notation, it is assumed in this paper that only one payload/search/detection sensor and one onboard localization sensor are mounted on each individual mobile platform.

At each time step k, the targets and the sensors are in their respective unknown states $\mathbf{x}_k^t$ and $\mathbf{x}_k^s$ While the targets are transiting to their new states, $\mathbf{x}_{k+1}^t$, the sensors, based on their anticipation of both their own and the targets' future states, select and execute a control action $\mathbf{u}_k^s = \{\mathbf{u}_k^{s_i} : i = 1, ..., N_s\}$, arrive at their new states, $\mathbf{x}_{k+1}^s$, and receive a set of observations ${}^l\mathbf{z}_{k+1}^s = \{{}^{l_i}\mathbf{z}_k^{s_i} : i = 1, ..., N_s\}$ of their own states, and a set of observations ${}^s\mathbf{z}_{k+1}^t = \{{}^{s_i}\mathbf{z}_k^{t_j} : i = 1, ..., N_s, j = 1, ..., N_t\}$ of the targets' states and the associated payoff.

Let $\mathbf{I}_k$ be the vector of all information available at time step k as defined by

$$\mathbf{I}_k \equiv \{\mathbf{I}_0, {}^s\mathbf{z}_{1:k}^t, \mathbf{u}_{0:k-1}^s, {}^l\mathbf{z}_{1:k}^s\}, \qquad k = 1, ..., N_k. \quad (1)$$

where the terms ${}^s\mathbf{z}_{1:k}^t$, $\mathbf{u}_{0:k-1}^s$ and ${}^l\mathbf{z}_{1:k}^s$ represent the entire histories up to time step k of the targets observations, control actions and platforms localization observations, respectively. Notice that the global information vector may be written in terms of the partial information vectors as in $\mathbf{I}_k = \{\mathbf{I}_k^{t_j} : i = 1, ..., N_t\}$ where $\mathbf{I}_k^{t_j} \equiv \{\mathbf{I}_0^{t_j}, {}^s\mathbf{z}_{1:k}^{t_j}, \mathbf{u}_{0:k-1}^s, {}^l\mathbf{z}_{1:k}^s\}$ contains all the information available about target t_j up to step k.

Similarly, let $\mathbf{I}_k^{s_i}$ be the vector of all information available about sensor s_i at time step k as defined by

$$\mathbf{I}_k^{s_i} = \{\mathbf{I}_0^{s_i}, \mathbf{u}_{0:k-1}^{s_i}, {}^{l_i}\mathbf{z}_{1:k}^{s_i}\} \subset \mathbf{I}_k, \qquad k = 1, ..., N_k. \quad (2)$$

Also the joint sensor state information vector $\mathbf{I}_k^s$ may be written in terms of the above partial vectors, $\mathbf{I}_k^s = \{\mathbf{I}_k^{s_i} : i = 1, ..., N_s\}$.

The purpose of the Bayesian filter is to recursively produce an estimate for the targets joint probability density, $p(\mathbf{x}_k^t|\mathbf{I}_k) = p(\mathbf{x}_k^{t_1}, ..., \mathbf{x}_k^{N_t}|\mathbf{I}_k)$ given the current information vector $\mathbf{I}_k$. However, doing is intractable for large numbers of targets since the computational cost and memory usage rapidly become prohibitive as they increase exponentially with the number of targets and the number of states for each target [19]. In this paper, in an effort to limit the complexity of maintaining such a high dimensional distribution, the targets individual densities are assumed to be independent of each other. This implies that a different Bayesian filter may be instantiated to maintain a separate independent PDF, denoted $p(\mathbf{x}_k^{t_j}|\mathbf{I}_k^{t_j})$, for each target. Fortunately, in practice it is often the case that the targets are completely unrelated, e.g. two independent hikers lost in the bush. Even when the targets are loosely coupled by sharing the same process, e.g. drifting life-rafts exposed to the same wind environment, it may still be reasonable to assume independence between the individual target densities as the induced estimation error is conservative and often considered negligible.

The analysis starts by determining a prior PDF $p(\mathbf{x}_0^{t_j}|\mathbf{I}_0^{t_j}) \equiv p(\mathbf{x}_0^{t_j})$ for each target state at time 0, given the vector $\mathbf{I}_0^{t_j}$ of 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, $p(\mathbf{x}_k^{t_j}|\mathbf{I}_k^{t_j})$, 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 update for the j^{th} target, $p(\mathbf{x}_{k-1}^{t_j}|\mathbf{I}_{k-1}^{t_j})$, is available. This prior PDF is predicted forward to time step k using the following Chapman-Kolmogorov equation

$$p(\mathbf{x}_k^{t_j}|\mathbf{I}_{k-1}^{t_j}) = \int p(\mathbf{x}_k^{t_j}|\mathbf{x}_{k-1}^{t_j}) p(\mathbf{x}_{k-1}^{t_j}|\mathbf{I}_{k-1}^{t_j}) d\mathbf{x}_{k-1}^{t_j} \quad (3)$$

where $p(\mathbf{x}_k^{t_j}|\mathbf{x}_{k-1}^{t_j})$ is a probabilistic Markov motion model. Also referred to as the process model, it describes the probability of transition of the target states form a given prior state, $\mathbf{x}_{k-1}^{t_j}$, to a destination state, $\mathbf{x}_k^{t_j}$. Deriving the process model from the equations of motion of the target and the probability distribution on their inputs is out of the scope of this paper. Ref. [7], however, provides some examples of realistic process models with constraints. For more details on how to derive $p(\mathbf{x}_k^{t_j}|\mathbf{x}_{k-1}^{t_j})$ from the systems equations refer to [4].

2) **Update:** At time step k, a new set of observations ${}^s\mathbf{z}_k^{t_j} = \{{}^{s_1}\mathbf{z}_k^{t_j}, ..., {}^{s_{N_s}}\mathbf{z}_k^{t_j}\}$ becomes available where ${}^{s_i}\mathbf{z}_k^{t_j}$ is taken from platform s_i at state $\mathbf{x}_k^{s_i}$. For each sensor s_i, the

conditional detection probability density function for target t_j at time step k, referred to as the detection sensor model, is denoted $p({}^{s_i}\mathbf{z}_k^{t_j}|\mathbf{x}_k^{t_j},\mathbf{x}_k^{s_i})$ and is assumed to be known. It is a mapping of target t_j detection probability distribution given the target and the sensor states, $\mathbf{x}_k^{t_j}$ and $\mathbf{x}_k^{s_i}$.

Given a measurement outcome, ${}^{s_i}\mathbf{z}_k^{t_j} = {}^{s_i}z_k^{t_j} \in \{D_k, \overline{D}_k\}$, the corresponding slice in the observation model $p({}^{s_i}\mathbf{z}_k^{t_j} = {}^{s_i}z_k^{t_j}|\mathbf{x}_k^{t_j},\mathbf{x}_k^{s_i})$ is function of the platform and the target states alone and will be referred to as an observation *likelihood*, i.e. the *detection likelihood* for an event D_k, and the *miss likelihood* for an event $\overline{D}_k$. Since the sensor state platform state $\mathbf{x}_k^{s_i}$ is uncertain the Bayesian update may be performed as follows

$$p(\mathbf{x}_k^{t_j}|\mathbf{I}_k^{t_j}) = Kp(\mathbf{x}_k^{t_j}|\mathbf{I}_{k-1}^{t_j}) \prod_{i=1}^{N_s} \int p({}^{s_i}z_k^{t_j}|\mathbf{x}_k^{t_j},\mathbf{x}_k^{s_i}) p(\mathbf{x}_k^{s_i}|\mathbf{I}_k^{s_i})\, \mathrm{d}\mathbf{x}_k^{s_i} \tag{4}$$

where K is the normalization factor given by

$$K^{-1} = \int p(\mathbf{x}_k^{t_j}|\mathbf{I}_{k-1}^{t_j}) \prod_{i=1}^{N_s} \int p({}^{s_i}z_k^{t_j}|\mathbf{x}_k^{t_j},\mathbf{x}_k^{s_i}) p(\mathbf{x}_k^{s_i}|\mathbf{I}_k^{s_i})\, \mathrm{d}\mathbf{x}_k^{s_i} \mathrm{d}\mathbf{x}_k^{t_j} \tag{5}$$

and where the updated sensor state PDF for each sensor s_i is obtained from the output of an Extended Kalman Filter (EKF) implemented for localization based on fusing GPS and electronic compass measurements, i.e. $p(\mathbf{x}_k^{s_i}|\mathbf{I}_k^{s_i}) = \mathcal{N}(\hat{\mathbf{x}}_{k|k}^{s_i}, \mathbf{P}_{k|k}^{s_i})$, where $\hat{\mathbf{x}}_{k|k}^{s_i}$ and $\mathbf{P}_{k|k}^{s_i}$ are the latest state estimate and covariance matrix, respectively, following the control action $\mathbf{u}_k^{s_i}$ and ensuing localization observation ${}^{l_i}\mathbf{z}_k^{s_i} = {}^{l_i}z_k^{s_i}$. The integral term in (4) corresponds to the observation likelihood expectation and can be rewritten simply as $p({}^{s_i}z_k^{t_j}|\mathbf{x}_k^{t_j},\mathbf{I}_k^{s_i})$ leading to the following version of the update equation

$$p(\mathbf{x}_k^{t_j}|\mathbf{I}_k^{t_j}) = Kp(\mathbf{x}_k^{t_j}|\mathbf{I}_{k-1}^{t_j}) \prod_{i=1}^{N_s} p({}^{s_i}z_k^{t_j}|\mathbf{x}_k^{t_j},\mathbf{I}_k^{s_i}). \tag{6}$$

B. Active Bayesian Sensor Network

Packaging a physical sensor with its own Bayesian filtering processor is an attractive way of making the sensor mobile and modular. Fig. 2 depicts algorithmically the nodal Bayesian filter for a single target t and how it interacts with the Controller, the Platform, and the Sensor in a network with node-to-node communication. The Platform block represents the actuators and dynamics of both the sensor and the mobile vehicle on which the sensor is mounted. At time step $k-1$, based on the latest beliefs about the targets $p(\mathbf{x}_{k-1}^t|\mathbf{I}_{k-1}), \forall j \in [1,...,N_t]$ and the sensor state estimate $\hat{\mathbf{x}}_{k-1|k-1}^{s_i}$, the local 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 set of observations. When it comes in, the new observation likelihood expectation $p({}^{s_i}z_k^t|\mathbf{x}_k^t,\mathbf{I}_k^{s_i})$, is fused to the predicted PDF, $p(\mathbf{x}_k^t|\mathbf{I}_{k-1})$ (lower product node in Fig. 2), to form the new nodal PDF estimate based on an incomplete local information vector denoted ${}^{s_i}\hat{\mathbf{I}}_k$, as in $p(\mathbf{x}_k^t|\mathbf{I}_{k-1}, {}^{s_i}\mathbf{z}_k^t, \mathbf{u}_k^{s_i}, {}^{l_i}\mathbf{z}_k^{s_i}) = p(\mathbf{x}_k^t|{}^{s_i}\hat{\mathbf{I}}_k)$.

The combined PDF estimate $p(\mathbf{x}_k^t|{}^{s_i}\hat{\mathbf{I}}_k \cup {}^{s_j}\hat{\mathbf{I}}_k)$ based on two incomplete but not mutually exclusive information vectors, ${}^{s_i}\hat{\mathbf{I}}_k$ and ${}^{s_j}\hat{\mathbf{I}}_k$, at node s_i and s_j is obtained from

$$p(\mathbf{x}_k^t|{}^{s_i}\hat{\mathbf{I}}_k \cup {}^{s_j}\hat{\mathbf{I}}_k) \propto \frac{p(\mathbf{x}_k^t|{}^{s_i}\hat{\mathbf{I}}_k) p(\mathbf{x}_k^t|{}^{s_j}\hat{\mathbf{I}}_k)}{p(\mathbf{x}_k^t|{}^{s_i}\hat{\mathbf{I}}_k \cap {}^{s_j}\hat{\mathbf{I}}_k)} \tag{7}$$

where $p(\mathbf{x}_k^t|{}^{s_i}\hat{\mathbf{I}}_k \cap {}^{s_j}\hat{\mathbf{I}}_k)$ is the estimate based on the common information. The common PDF estimate is referred to as the channel filter estimate.

So, as shown on Fig. 2, after fusion of the local likelihood expectation, the latest nodal estimate is then sent to neighboring nodes via Channel Filters whose purpose are to maintain a density estimate based on the common information shared between each two two nodes. In order to prevent double-counting, the Channel Filters use their estimates to remove the common information from the nodal estimate. The residual which corresponds to the new information accumulated by the emitting node since the last communication, through sensor observations and communication with other neighbors, is then communicated to the Channel Filter of the receiving node to update both the receiver's nodal and channel estimates. Likewise, the emitting node also receives new information from its other neighbors which it fuses (upper product node in Fig. 2) to its current local estimate, $p(\mathbf{x}_k^t|{}^{s_i}\hat{\mathbf{I}}_k)$.

The channel filter guarantees the nodes to converge to the global estimate given the network propagation delay. The details of the general channel filter are introduced in [5] where it is also proposed to evaluate the amount of estimation error using the following Hellinger affinity measure

$$D(p_i\|p_j) = 2\ln \int \sqrt{p_i(\mathbf{x}_k^t) p_j(\mathbf{x}_k^t)}\, d\mathbf{x}_k^t \tag{8}$$

which is a monotonic distance metric between two densities p_i and p_j [11]. The metric values range from 0, when the densities are identical, to $-\infty$, when they have nothing in common, i.e. $\int \sqrt{p_i(\mathbf{x}_k^t) p_j(\mathbf{x}_k^t)}\, d\mathbf{x}_k^t = 0$. The channel manager also uses this divergence measure to determine when to communicate on each channel [5]. For more details on the general decentralized data fusion algorithm, refer to Chapter 5 of [4].

A major advantage of implementing channel filters comes from the fact that if transmission packets are lost in the communication process, the filter does not update its estimate allowing the information to be recovered on the next communication step. Notice that since the nodes only know their immediate neighbors and are ignorant of the global network topology they cannot differentiate the source of the information they receive. One necessary condition to maintain proper accounting of the information is that the network connectivity must be acyclic [9]. In other words, no communication loops must exist between the nodes that would enable the information to cycle through multiple times.

III. Decentralized Cooperative Control

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

$$\mathbf{x}_{k+1}^{s_i} = \mathbf{f}_k^{s_i}(\mathbf{x}_k^{s_i}, \mathbf{u}_k^{s_i}) + \mathbf{w}_k^{s_i}, \tag{9}$$

subject to the following set of kino-dynamic constraints and control bounds

$$\mathbf{g}_k^{s_i}(\mathbf{x}_k^{s_i}, \mathbf{u}_k^{s_i}) \leq \mathbf{0} \tag{10}$$

$$\mathbf{u}_{LB}^{s_i}(\mathbf{x}_k^{s_i}) \leq \mathbf{u}_k^{s_i} \leq \mathbf{u}_{UB}^{s_i}(\mathbf{x}_k^{s_i}) \tag{11}$$

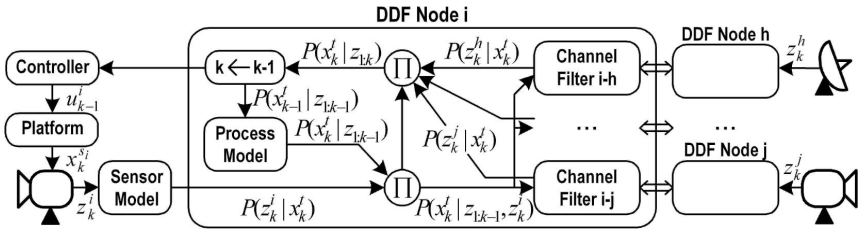


Fig. 2. General active Bayesian sensor node with channel filters for node-to-node communication.

where $\mathbf{u}_k^{s_i}$ is the active sensor control input vector, $\mathbf{w}_k^{s_i}$ is a zero-mean process noise of known covariance and $\mathbf{u}_{LB}^{s_i}$ and $\mathbf{u}_{UB}^{s_i}$ are the lower and upper bounds respectively on the control input in function of the state. The controller objective is to produce a command that will place the system in a desired state.

A. Optimal Trajectory

For a multi-sensor system, 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 (11) and the constraints (10).

$$\mathbf{u}^* = \{\mathbf{u}_{1:N_k}^{s_1 *}, ..., \mathbf{u}_{1:N_k}^{s_{N_s} *}\} = \arg\max_{\mathbf{u}} J(\mathbf{u}, N_k) \qquad (12)$$

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 [8]) 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

$$J_k(\mathbf{u}, N_k) = \sum_{i=1}^{N_s} J_k^{s_i}(\mathbf{u}, N_k) = \sum_{i=1}^{N_s} J_k^{s_i}(\mathbf{u}^{s_i}, \overline{\mathbf{u}}^{s_i}, N_k) \quad (13)$$

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 $\overline{\mathbf{u}}^{s_i} = \{\mathbf{u}_{k:k+N_k-1}^{s_j} : j \neq i\}$. 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 [15]. That is

$$J_k^{s_i}(\mathbf{u}^*, N_k) \geq J_k^{s_i}(\mathbf{u}^{s_i}, \overline{\mathbf{u}}^{s_i *}, N_k), \quad \forall \mathbf{u}^{s_i} \in \mathbf{U}^{s_i}, \forall i \quad (14)$$

where $\overline{\mathbf{u}}^{s_i *} = \{\mathbf{u}_{k:k+N_k-1}^{s_j *} : j \neq i\}$. 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 Iterative Partial Optimizations for Control Negotiation

In this section it is proposed to approach the multi-vehicle multi-sensor optimization problem (12) as a distributed computation problem [2]. By taking advantage of the resources from multiple processors, distributed computation methods have a significant speed advantage over centralized methods. The control problem (12) 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 $J_k(\mathbf{u}, N_k) = J_k(\mathbf{u}^{s_i}, \overline{\mathbf{u}}^{s_i}, N_k)$ with respect to $\mathbf{u}^{s_i}$, and repeating for all components until convergence, i.e. until condition (14) 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 [2].

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

$$\mathbf{u}_{|l}^{s_i *} = \arg\max_{\mathbf{u}^{s_i}} J_k^{s_i}(\mathbf{u}^{s_i}, \overline{\mathbf{u}}_{|l-1}^{s_i}, N_k) = \arg\max_{\mathbf{u}^{s_i}} J_k(\mathbf{u}^{s_i}, \overline{\mathbf{u}}_{|l-1}^{s_i}, N_k) \qquad (15)$$

where $\overline{\mathbf{u}}_{|l-1}^{s_i}$ 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, $(\mathbf{u}_{|l}^{s_i *}, \overline{\mathbf{u}}_{|l-1}^{s_i})$ 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

$$\mathbf{u}_{|l}^{s_i} = \mathbf{u}_{|l-1}^{s_i} + \alpha_l(\mathbf{u}_{|l}^{s_i *} - \mathbf{u}_{|l-1}^{s_i}) = (1-\alpha_l)\mathbf{u}_{|l-1}^{s_i} + \alpha_l \mathbf{u}_{|l}^{s_i *} \tag{16}$$

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 [20] introduced a slightly different algorithm that exploits this idea of adding a stochastic element to the update equation. That algorithm was successfully applied in [10] 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 $\overline{\mathbf{u}}_{|l-1}^{s_i}$ 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.

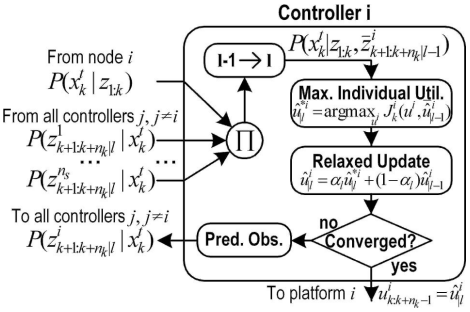


Fig. 3. Bayesian negotiator in a fully connected network with broadcast communications.

Fig. 3 graphically depicts the controller negotiation algorithm for a single stationary target t. At a given time step k and iteration l, the controller i fuses the latest density estimate $p(\mathbf{x}_k^t|\mathbf{I}_k)$ obtained from its fusion node with the predicted observation likelihoods $p(^{s_j}z_{k+1:k+N_k|l-1}^t|\mathbf{x}_k^t, \mathbf{I}_{k+N_k}^{s_j})$'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 $p(\mathbf{x}_k^t|\mathbf{I}_k, \mathbf{I}_{k+N_k \setminus k|l-1}^{s \setminus s_i})$, 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 $p(^{s_i}z_{k+1:k+N_k|l}^t|\mathbf{x}_k^t, \mathbf{I}_{k+N_k}^{s_i})$ is then broadcasted to the other controllers for another iteration. Otherwise, $\mathbf{u}^{s_i *} \approx \mathbf{u}_{|l}^{s_i} \rightarrow \mathbf{u}_{k+1:k+N_k}^{s_i}$ 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) [16].

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 $p(^{s_i}z_{k+1:k+N_k|l-1}^t|\mathbf{x}_k^t, \mathbf{I}_{k+N_k}^{s_i})$'s, $\forall i$, the set of all the predicted observation likelihoods from each controller i would need to be broadcasted separately, as in $\{p(^{s_i}z_{k+n|l-1}^t|\mathbf{x}_k^t, \mathbf{I}_{k+n}^{s_i}) : n = 1, ..., N_k\}$'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 [3], 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 [3]. 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. Multi-Target Search Utility

This section describes how, using the output from the filter equations from Sec. II-A, the performance of a multi-vehicle search plan may be evaluated by determining the 'cumulative' probability of detection for each target. An equivalent but different derivation is presented in [6]. Further details on the searching problem can also be found in [18] and [17] (Chap.9).

Let the detection likelihood expectation for target t_j by sensor s_i at time step k be given by $p(^{s_i}\mathbf{z}_k^{t_j} = D_k^i|\mathbf{x}_k^{t_j}, \mathbf{I}_k^{s_i})$ where D_k^i represents a 'detection' event by the sensor on vehicle s_i at time k. The likelihood of 'miss' by the same sensor is given by its complement $p(^{s_i}\mathbf{z}_k^{t_j} = \overline{D}_k^i|\mathbf{x}_k^{t_j}, \mathbf{I}_k^{s_i}) =$

$1 - p(D_k^i|\mathbf{x}_k^{t_j}, \mathbf{I}_k^{s_i})$. The combined 'miss' likelihood for all the vehicles at time step k is simply the multiplication of the individual 'miss' likelihood expectations for that target

$$p(\overline{D}_k|\mathbf{x}_k^{t_j}, \mathbf{I}_k^s) = \prod_{i=1}^{N_s} p(\overline{D}_k^i|\mathbf{x}_k^{t_j}, \mathbf{I}_k^{s_i}) \tag{17}$$

where $\overline{D}_k = \overline{D}_k^1 \cap ... \cap \overline{D}_k^{N_s}$ represents the event of a 'miss' observation by every sensor at time step k.

If the normalization factor K is neglected, the update equation (6) can be rewritten as the following pseudo-update equation.

$$p(\mathbf{x}_k^{t_j}|\mathbf{I}_{1:k}^{t_j})' = p(\mathbf{x}_k^{t_j}|\mathbf{I}_{k-1}^{t_j})' \prod_{i=1}^{N_s} p({}^{s_i}\mathbf{z}_k^{t_j}|\mathbf{x}_k^{t_j}, \mathbf{I}_k^{s_i}) \tag{18}$$

As shown in [6], the probability that a target gets detected for the first time at time step k, namely the probability of detection at time step k, is denoted $p_k^{t_j} = p(\overline{D}_{1:k-1}, D_k)$. It corresponds to the reduction in volume under the pseudo-density function when it is updated with the combined 'detection' likelihood expectation, denoted $p(D_k|\mathbf{x}_k^{t_j}, \mathbf{I}_k^s) = \left[1 - p(\overline{D}_k|\mathbf{x}_k^{t_j}, \mathbf{I}_k^s)\right]$, with $p(\overline{D}_k|\mathbf{x}_k^{t_j}, \mathbf{I}_k^s)$ given in (17), and is obtained as follows

$$p_k^{t_j} = \int p(\mathbf{x}_k^{t_j}|\mathbf{I}_{k-1}^{t_j}(\overline{D}_{1:k-1}))' \left[1 - p(\overline{D}_k|\mathbf{x}_k^{t_j}, \mathbf{I}_k^s)\right] d\mathbf{x}_k^{t_j} \tag{19}$$

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

$$P_k^{t_j} = \sum_{m=1}^{k} p_m^{t_j} = P_{k-1}^{t_j} + p_k^{t_j} \tag{20}$$

For this reason $P_k^{t_j}$ will be referred to as the 'cumulative' probability of detection to distinguish it from the payoff probability of detection function $p_k^{t_j}$.

A. Multi-Target Team Utility

The probability of detecting target j, given a series of observations generated by the control sequence over the planning horizon starting at time step k, is given by the engendered net variation in cumulative probability of detection

$$\Delta P_k^{t_j} = \sum_{l=k+1}^{k+N_k} p_l^{t_j} = P_{k+N_k}^{t_j} - P_k^{t_j} \tag{21}$$

with $p_l^{t_j}$ obtained from (19). Notice that $\Delta P_k^{t_j}$ can also be directly obtained from the corresponding reduction in volume under the pseudo-density function caused by the observations. This measure was used in [6] as the team utility function for the single target search problem. For the multi-target search problem, this paper uses the following weighted sum of the above measure

$$J_k(\mathbf{u}, N_k) = \sum_{j=1}^{N_t} w_k^{s_i} \Delta P_k^{t_j}, \quad \text{with} \quad \sum_{j=1}^{N_t} w_k^{s_i} = 1 \tag{22}$$

where the weights $w_k^{s_i}$'s correspond to the relative priority for each target at time step k.

B. Individual Utility

Hence the individual greedy version of the team utility in (22) is given by

$$J_k^{s_i}(\mathbf{u}^{s_i}, N_k) = \sum_{j=1}^{N_t} w_k^{s_i} \sum_{l=k+1}^{k+N_k} \int p(\mathbf{x}_l^{t_j}|\overline{D}_{1:l-1}^i)' [1 - p(\overline{D}_l^i|\mathbf{x}_l^{t_j})] \, d\mathbf{x}_l^{t_j} \tag{23}$$

which for a time-horizon of $N_k = 1$, reduces to the weighted sum of each target probability of detection, as in $J_k^{s_i}(\mathbf{u}^{s_i}, 1) = \sum_{j=1}^{N_t} w_k^{s_i} p_k^{t_j}$, with $p_k^{t_j}$ given in (19). It is Equation (23) that is used as a utility in the individual optimization problems for the negotiation algorithm presented in Sec. III-D where piecewise constant control solutions [12] are obtained for the by using a constrained non-linear programming technique called Sequential Quadratic Programming (SQP) [16].

V. Experiment

This section presents the implementation details and the results of the decentralized cooperative search framework implemented for a team of 5 human-computer augmented nodes and 4 autonomous mobile robots, searching for multiple stationary targets on Cornell University baseball field (see Fig. 4). For this paper, the necessary decentralized network components, i.e. controller/negotiator, sensor fusion node, and graphical interface, were a mix of Matlab implementations and C++ components developed at Cornell as part of multi-robot research facility that are compatible with the ORCA open-source robotics components repository [13].

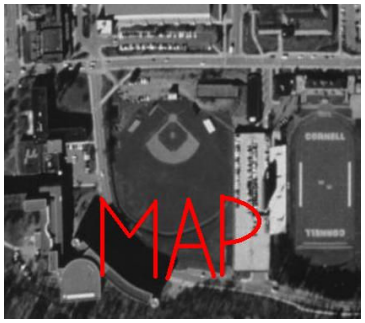


Fig. 4. Search Area : Cornell Campus.

A. Hardware

1) Rovers: The Segway Robotics Mobility Platform 50 (RMP-50) displayed in Fig. 5 is a sturdy, large payload capacity, long endurance and relatively low cost platform. Its large wheel radius makes it ideal for carrying sensors and testing algorithms in indoor as well as outdoor environments.

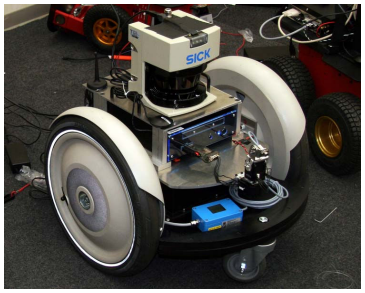


Fig. 5. Segway RMP-50 equipped with embeded computer, SICK laser range finder, GPS receiver, magnetometer, gimballed pyrometer and wireless communication.

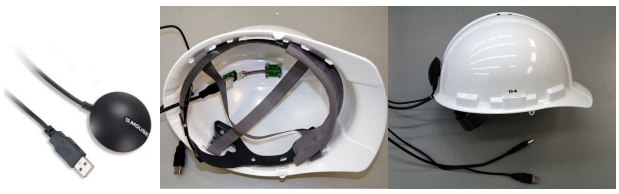
2) Computers: Each human searcher is equipped with a small portable Sony Micro-PCs as the shown in Fig. 6a, meet these requirements very well. Despite its ultra portable design, this micro-computer, with a 1.2 GHz processor running a full version of Windows XP, is capable of handling all the required processing tasks. Its wireless capabilities are similar to those of standard laptop. It also has other nice feature for distributed collaborative work such as imbeded webcams aimed at both the user and the environement away from the user.

(a) Portable Computer (b) Embeded Computer

Fig. 6. Computers: (a) The Sony Vaio VGN-UX280P Micro-PC features a 4.5 inch LCD touch screen, a hidden keyboard, a 1.2 GHz Ultra Low Voltage Intel processor, a 30 GB hard drive, 512 MB of RAM, a built-in webcam, a Memory Stick slot, Tri-mode Wi-Fi (802.11a/b/g), a USB 2.0 comm port and Bluetooth connectivity. (b) Small form factor Morex casing containing Mini-ITX board by Jetway with 2.0 Ghz VIA CPU, 1 GB of RAM, with 4 RS-232 serial and 6 USB connectors, power supply, 200 GB hard drive and wireless ethernet card (seen here mounted in custom modular support,

For the mobile robotic platform onboard computer, a Mini-ITX board with an integrated 2.0 Ghz processor and 1 GB of RAM was integrated into a small form factor case (Fig. 6b) with a regulated power supply unit, a 200GB hard-drive and a wireless communication card. The system can be connected to both the 12V DC from the robot's supply or 120V AC from a wall outlet making the perfect "plug-and-play" computing solution for the rovers and sensors.

3) Localization Sensors: Both the humans and robots are equipped with a small WAAS-enabled GPS receiver coupled with a 2D magnetometer to improve the position estimate estimate accuracy, especially in orientation. Both sensors communicate and receive power via a USB connection. This eliminates the need for extra battery supply. Fig. 7a shows the receiver with its USB connector. For the humans, as shown on Figs. 7b and 7c, the two sensors were mounted on helmets to obtain reliable measurements and good tracking of the head motion.



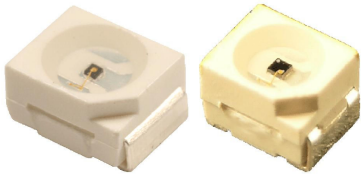
(a) (b) (c)

Fig. 7. Localization Sensors: (a) The GlobalSat BU-353 is a WAAS and EGNOS enabled GPS receiver incorporating the latest SiRF Star III GPS chipset and an active patch antenna for high degree of localization accuracy with waterproof casing and USB connector for interfacing with a portable computer; (b) SparkFun Electronics small 2-axis compass board with Honeywell HMC6352 compass chip mounted on top inside of the helmet; and (c) side view of human localization sensor helmet with GPS receiver mounted on the back.

4) Targets and Artificial Detection Sensors: In order for both the humans and robots to detect the same target, an upward-facing infrared light emitting diode (Fig. 8a), powered by a watch battery, was attached to a small object approximately the size of a golf ball. The infrared emitter is a GaAIAs infrared LED mounted in a plastic leadless PLCC-2 SMD package with a flat lens window that allows a half-power beam emission angle of 110 degrees.

The object on which the LED was attached is easy to see with the human eye at close range (approximately 0.7 meters), but visibility of the target drops off as it is obstructed by tall grass or shadow.

Each robot was equipped with three sensors (Fig. 8b) to detect the LED target(s). The NPN silicon phototransistors are mounted in miniature SMD packages. The devices have a flat window lens, which has a 50 percent acceptance angle of 100 degrees. The phototransistors are mechanically and spectrally matched to the infrared LEDs.



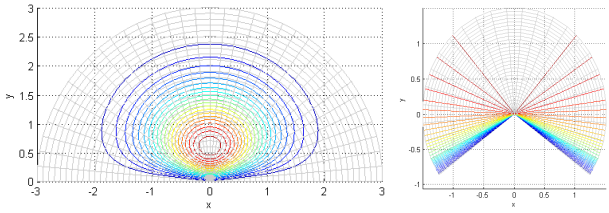
(a) (b)

Fig. 8. Target and detection sensor: (a) The OP280 is a GaAIAs infrared LED mounted in a plastic leadless PLCC-2 SMD package with a flat lens window that allows a wide beam angle. The PLCC-2 packaging is suitable for single device or array applications. The OP280 Series LEDs is mechanically and spectrally matched to OP580 series phototransistors; (b)The OP580 is an NPN silicon phototransistor mounted in a miniature SMD package. The device has a flat window lens, which enables a wide acceptance angle. It is packaged in a plastic leadless chip carrier that is compatible with most automated mounting equipment. The OP580 is mechanically and spectrally matched to the OP280.

B. Sensor Models

As mentioned earlier the target is easy to see with the human eye at close range (approximately 0.7 meters), but visibility of the target drops off as it is obstructed by tall grass or shadow. Fig. 9a shows the likelihood of target detection using the human vision model. This model was developed using actual measurements and instances of detection and no detection for several human subjects.

The model assumes a forward-facing human, where maximum target visibility is straight ahead at just under one meter from the human. As seen in Fig. 9a, detection drops off and is approximately zero when the target is located further than three meters from the human sensor, mostly due to tall grass, shadows, or other visual obstructions. Assuming a forward-facing human, target detection is highest straight-ahead, and drops as far peripheral vision makes it difficult for the human to see possible targets.



(a) (b)

Fig. 9. Detection Sensor Models: (a) Likelihood of Detection for Human Vision Model. The vertical axis (y) is along the human's center of gaze. The horizontal axis (x) is at the edge of the human's peripheral vision, where there is no target visibility. Red contour lines indicate high likelihood of detection. All units are in meters. (b) Likelihood of Detection for Robot Vision Model.

The robot vision using three phototransistors is modeled as a single bearing-only sensor with a viewing angle up to 270 degrees and range up to 1.5 meters. The likelihood of target detection using the robot vision model is shown in Fig. 9b, where red indicates high likelihood of detection and blue is low likelihood of detection.

C. Results

Fig. 10 illustrates the decentralized search results for the active Bayesian sensor network algorithm presented in Sec. II. The advantage of a cooperative solution over the typical parallel search patterns usually implemented during search and rescue operations on foot or the coordinated search solution presented in [6] is made

clear on Fig. 10d. For the cooperative solution (Figs. 10a-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 searcher follows a greedy 1-step lookahead solution. By allowing a more efficient allocation of the search effort, the cooperative approach compares advantageously to both coordinated (Fig. 10e) and the parallel track formation (Fig. 10f) search strategies. In fact after 160 steps, the time needed for the formation to traverse the search area, the cooperating vehicles reach a final P_k of $P_{160}^{coop} = .818$ vs. $P_{160}^{coord} = .718$ for the coordinated solution and $P_{160}^{parallel} = .575$ for the flight formation, which correspond to an increase of a 13.9% and 42.3% respectively over the later solutions (Fig. 10d).

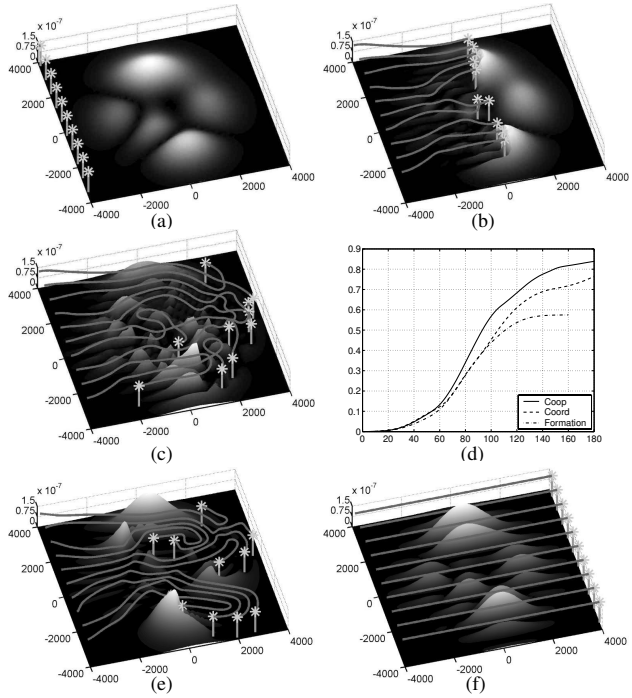


Fig. 10. Cooperative search for a static target with 10 human-computer augmented nodes: (a) to (c) 3D views of the target PDF and the cooperative trajectories at time steps k = 1, 90 and 180, respectively (i.e. .667, 60 and 120 minutes); (d) comparison of the cumulative probability of detection, P_k vs. k, for the cooperative, coordinated and the straight pattern search; (e) coordinated search at $k = 180$ (120 mins.), and (f) straight pattern flight formation search at $k = 160$ (106.67 mins).

VI. Conclusion

In this paper, scalable collaborative human-robot systems for information gathering applications are approached as active sensor networks. Peer-to-peer collaboration between human-computer augmented nodes and autonomous mobile sensor platforms happens by sharing information via wireless communication network. For each node, a computer (onboard the platform or carried by the human) implements both a decentralized Bayesian fusion algorithm and a decentralized Bayesian control negotiation algorithm.

The decentralized Bayesian control techniques demonstrated in this paper are a natural extension of the general decentralized Bayesian data fusion algorithm. The individual controllers iteratively negotiate anonymously in the information space to find cooperative search plans based on both observed and predicted information that explicitly consider the human motion model, its sensors detection functions, as well as the target arbitrary motion model. Applying the current technique for teaming human nodes with multiple autonomous robotic platforms into a large heterogeneous network in various application domains, i.e. exploration, mapping, environmental monitoring, surveillance, and others, is part of the ongoing research effort. The practical outcomes of these decentralized algorithms will be to increase situation awareness, redundancy, reliability and responsiveness useful for time critical missions such as in crisis situations where human life may be at stake.

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

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

[4] F. Bourgault. *Decentralized Control in a Bayesian World*. PhD thesis, Australian Centre for Field Robotics, Aerospace, Mechanical and Mechatronic Engineering, The University of Sydney, 2008.

[5] F. Bourgault and H.F. Durrant-Whyte. Communication in general decentralized filters and the coordinated search strategy. In *Proceedings of The 7th Int. Conf. on Information Fusion*, pages 723–730, Stockholm, Sweden, June 2004.

[6] F. Bourgault, T. Furukawa, and H.F. Durrant-Whyte. Coordinated decentralized search for a lost target in a Bayesian world. In *Proceedings of the 2003 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'03)*, pages 48–53, Las Vegas, Nevada, October 2003.

[7] F. Bourgault, T. Furukawa, and H.F. Durrant-Whyte. Process model, constraints, and the coordinated search strategy. In *Proceedings of the 2004 IEEE Int. Conf. on Robotics and Automation (ICRA'04)*, pages 5256–5261, New Orleans, LA, April 2004.

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

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

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

[11] A.O. Hero, B. Ma, O. Michel, and J. Gorman. Alpha-Divergence for Classification, Indexing and Retreival (Revised 2). Technical Report CSPL-328, Communication and Signal Processing Laboratory, The University of Michigan, 48109-2120, USA, June 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] A. Makarenko, A. Brooks, and T. Kaupp. Orca: Components for robotics. In *Proceedings of the 2006 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'06)*, Beijing, China, October 2006.

[14] A. Makarenko, A. Brooks, S.B. Williams, H.F. Durrant-Whyte, and B. Grocholsky. A decentralized architecture for active sensor networks. In *Proceedings of the 2004 IEEE Int. Conf. on Robotics and Automation (ICRA'04)*, volume 2, pages 1097–1102, New Orleans, LA, USA, 2004.

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

[16] 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.

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

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