JOURNAL OF AEROSPACE COMPUTING, INFORMATION, AND COMMUNICATION
Vol. 1, December 2004

Real-Time Planning for Multiple Autonomous Vehicles in Dynamic Uncertain Environments

Anawat Pongpunwattana[*] and Rolf Rysdyk[†]
University of Washington, Seattle, Washington, 98195-2400

This paper introduces a market-based cooperative planning system for a team of autonomous vehicles operating in a dynamic environment. The system combines the flexibility of evolution-computation techniques with the distributed nature of market strategy to compute task and paths plans. Optimization is based on a team utility function which accounts for uncertainty in knowledge of the environment. Multiple vehicles will cooperate on the same task if doing so increases the predicted team utility value. The team utility function and the associated stochastic model that predicts future system states are described. The minimum required information exchange among the vehicles is identified. Simulation results, using the Boeing Company developed Open Experimental Platform, demonstrate the effectiveness of the planning.

Nomenclature

A = task allocation matrix
$\tilde{B}_v^i$ = probability that the path of vehicle v intersects target location of task i
$\tilde{B}_j^v$ = probability that vehicle v collides with obstacle j
$\tilde{C}_v$ = function to compute the expected operation cost of vehicle v
D = team task plans
D_v = task plan of vehicle v
$\overline{d}^V$ = task assignment vector
f = state transition function
$\tilde{H}_v^i$ = probability that task i will be completed by vehicle v
$\tilde{H}_j^v$ = probability that vehicle v will be destroyed by obstacle j
J = team utility function
J_v = individual utility function of vehicle v
N_O = number of obstacles
N_S = number of sites
N_T = number of tasks
N_V = number of vehicles
Q = planned trajectories of all vehicles
Q_v = planned trajectory of vehicle v
$\tilde{R}_i$ = expected value of task score obtained by executing task i
x^E = states of the environment
x^F = task states

This paper is part of the December Special Section on Intelligent Systems. Received 18 August 2004; revision received 30 November 2004; accepted for publication 23 November 2004.

[*] Research Associate, Aeronautics and Astronautics, Box 352400. AIAA Member.

[†] Assistant Professor, Aeronautics and Astronautics, Box 352400. AIAA Member.

$\tilde{x}_i^F$	=	expected value of the state of task *i*
x^V	=	states of vehicles
u	=	control inputs to the system
z^E	=	site positions
$\tilde{z}^E$	=	expected values of site positions
$\dot{z}^E$	=	site velocities
$\tilde{\dot{z}}^E$	=	expected values of site velocities
z^V	=	vehicle positions
$\tilde{z}^V$	=	expected values of vehicle positions
$\bar{z}^V$	=	commanded vehicle positions
$\dot{z}^V$	=	vehicle velocities
$\tilde{\dot{z}}^V$	=	expected values of vehicle velocities
$\bar{\dot{z}}^V$	=	commanded vehicle velocities
α_i^F	=	score weighting factor of task *i*
α^Q	=	path cost weighting factor
α_v^V	=	vehicle cost weighting factor of vehicle *v*
η_j^O	=	effectiveness of the payload of obstacle *j*
η_v^V	=	effectiveness of the payload of vehicle *v*
ζ^E	=	health states of sites
$\tilde{\xi}^E$	=	expected values of health states of sites
ζ^O	=	health states of obstacles
$\tilde{\xi}^O$	=	expected values of health states of obstacles
ζ^V	=	health states of vehicles
$\tilde{\xi}^V$	=	expected values of health states of vehicles
π_i	=	price of task *i*
ρ_x	=	probability density function of the uncertainty in site positions
ρ_v	=	probability density function of the uncertainty in site velocities
σ_j^x	=	bound of the uncertainty in the position of site *j*
σ_j^v	=	bound of the uncertainty in the velocity of site *j*
τ_w^i	=	estimated execution time of task *i*

I. Introduction

In this work we investigate dynamic planning for a team of autonomous vehicles to cooperatively execute a set of tasks. Communication of the proposed solution benefits from dynamic presentation of the simulation results. Included with this work are representative examples based on simulations using the Boeing Company developed Open Experimental Platform (OEP).

Interest in the use of unmanned vehicles is increasing. The Department of Defense has a variety of research programs whose objectives include the development of the technologies that enable unmanned vehicles. The DoD envisions using various types of unmanned vehicles; unmanned ground vehicles (UGV), unmanned aerial vehicles (UAVs), and unmanned underwater vehicles (UUV) in its future combat systems.[1] Unmanned vehicles are also envisioned for many civilian applications such as weather forecasting, reconnaissance, search, and rescue missions, and observation during wildfires.

In what follows, a vehicle with the ability to independently plan, adapt, and execute its actions based on sensed or communicated information, is referred to as an *autonomous vehicle*. In complex applications, the environment in which a vehicle operates is dynamic and uncertain. Therefore, it is not practical to execute a plan that was computed prior to the start of the mission. Consequently, a planning system is needed that is capable of replanning online in real time to adapt the precomputed plan in response to changes. This concept essentially transforms the vehicles from passive command followers to active decision makers.

For certain missions, multiple autonomous vehicles may be required. Planning for multiple vehicles involves not only generating trajectories for the vehicles, but also the coordination of their task execution. The challenging aspect of this problem is to generate plans that obtain better performance by means of coordination and cooperation.

Intelligent Planning is among the key technologies for achieving increased autonomy of unmanned vehicles. Planning is an active research area in the Artificial Intelligence community and the modern gaming industry. Planning is a decision-making process requiring processing time, which limits the responsiveness of the system. Another approach to autonomous action is *reactive behavior-based control*, in which the vehicle is preprogrammed with a set of behaviors to react to particular events. With reactive behaviors, the vehicle can respond quickly, which may be desirable, e.g. in collision avoidance or tactical operations in dense urban terrain. However, purely reactive behaviors can result in inefficient actions or instability. Therefore, an intelligent autonomous vehicle must be able to plan its future actions continuously, while allowing for tactics which include reaction to sudden changes.

A. Problem Definition

The question we consider in this research is: how should a networked team of autonomous vehicles behave to complete a set of tasks in a dynamic environment? This poses a complex problem which includes coordination among the vehicles, and generation and assignment of paths that support the executions of these tasks. These paths must be constrained to the vehicle capabilities, traverse multiple target regions, and avoid collision with obstacles and other team members in a dynamic hostile environment. All computational processing and communication must be achieved in real time on board the aircraft. Our approach to this problem is based on the design of autonomous task and path planning algorithms for a system for networked UAVs. We refer to this combination as *planning-system.*

Dynamic uncertainty in the environment is a major aspect of the problem. The autonomous vehicles operate in an environment where the available information about the environment is known with limited level of certainty. Ideally, the planning-system reacts to unanticipated threats and opportunities efficiently and effectively. Generated plans should maximize the probability of future success of the mission based on the latest available information. The planning-system must have the ability to dynamically reallocate tasks among the vehicles and continually adapt their paths.

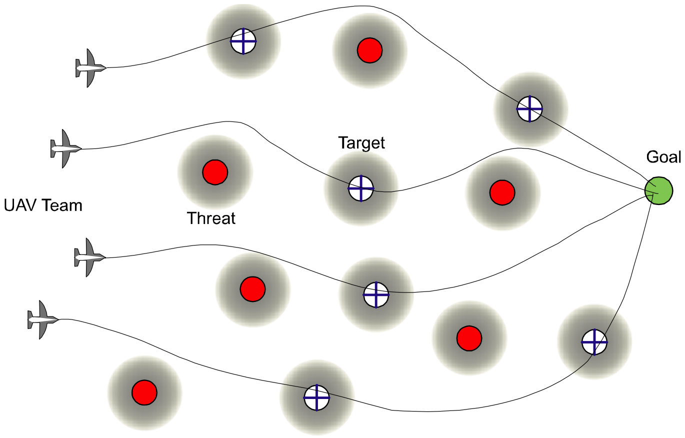

Fig. 1 Illustration of a reconnaissance mission.

Failures of system components are likely during complex missions, e.g. failures in communication links or a damaged team member. The planning-system must be able to adapt to unexpected failures.

The planning-system developed in this research can be applied to several types of missions. An example is a reconnaissance mission whose objective is to obtain information by visual observation or other detection methods in specific areas. In this example scenario, a team of autonomous vehicles is assigned to collect data of campsites. The vehicles have on-board sensors which can collect information and communication devices to transmit the data. The vehicles have to minimize exposure to threats. The exact locations of the camp sites and threats are unknown. What is known are the general areas where these camp sites and threats might be located. The camp sites and threats might move during the mission. Their future motions, however, are unknown at mission initiation. The information about the operating field is updated and shared among the vehicles during the operation. All the vehicles are required to reach a goal location at the end of the mission. This mission is illustrated in Fig. 1.

B. Assumptions

The planning-system is based on the following assumptions. First, each vehicle contains a guidance system capable of guiding the vehicle along its path. Second, each vehicle activates its payload according to the planned actions. Third, each vehicle communicates information with other vehicles and ground stations. Bandwidth and quality of these communication channels may be limited. Lastly, each vehicle merges data from its sensors and information from other vehicles to estimate system states.

C. Contributions

This article provides the following contributions. First, we develop a unique market-based distributed planning framework for coordinating multiple autonomous vehicles. This framework was designed to solve a class of complex planning problems posed by the DARPA's MICA program. Such problems have not been solved previously with the given constraints. Second, we introduce a stochastic dynamic world model and a team utility function both of which take into account the uncertain information of the environment and the coupling in the states of the system. Third, we develop a unique approach using evolution-based algorithms for allocating tasks and path planning simultaneously. We further extend the planning algorithms for dynamic replanning using the concept of Model Predictive Control. Lastly, we introduce a cooperative planning scheme which improves performance by considering probability of failure in prosecuting targets.

II. Background

A. Single Vehicle Planning

To move autonomously from an initial location to a desired task location, involves a large array of engineering aspects, including planning of each route segment and the guidance, navigation, and control laws to follow these segments.

1. Graph-Based Approaches

A popular approach to path planning is graph-based search, e.g. Murphy.[2] In graph-based search the environment is discretized and represented by a graph which is composed of a number of nodes linked together with arcs. Each node usually corresponds to a location and an arc links two nodes. There is a cost associated with each arc. A path is a series of connected arcs. The path planning problem is to find the path that minimizes the total cost. A graph search algorithm such as Dijkstra's Algorithm or the A*("A-star")-algorithm can be used to find the shortest path.[3] Thrun[4] investigated the problem of high-speed navigation of indoor mobile robots using a grid-based algorithm called *value-iteration*. The map of the environment was created autonomously from sonar and camera information and using Bayesian analysis techniques. Mata and Mitchell[5] proposed a new algorithm for path planning on planar polyhedral surfaces. The terrain is represented as a sparse graph called *pathnet*. They showed that the algorithm can provide highly competitive solutions. Mandow et al.[6] developed the PRIMO-A*-algorithm which extends the original A*-algorithm to multi-objective path planning problems. This is a solution to problems which require that resultant paths be optimal but also be within the limitations of the vehicle capabilities. Bander[7] developed an adaptive A*-algorithm which used a heuristic function to improve convergence of the A*-algorithm. This work also investigated mechanisms for incorporating sources of a-priori knowledge, and human inputs, to accelerate the search process.

2. Probabilistic Roadmap Planners

Probabilistic roadmap planning (PRM) is an efficient method to compute collision-free paths for vehicles or robots with many degrees of freedom.[8] This method consists of two phases, a building and a query phase. The building phase is the construction of a graph called *roadmap*. The nodes in the roadmap are collision-free configurations and the edges linking the nodes are collision-free paths. The query phase is finding a path between an initial and goal configurations by connecting these nodes to the road map and searching them for a sequence of edges linking the two nodes. This method was originally developed for holonomic robots in a static environment. Overmars[9] applied PRM to holonomic and non-holonomic robots with constrained kinematics and high degrees of freedom, showing how this technique can be extended to handle kinematic constraints in car-like robots. Kavraki and Latombe[10] extended the PRM approach and applied it to several robots with 3 to 16 degrees of freedom operating in a known static environment. LaValle and Kuffner[11] proposed a randomized path planning technique related to PRM. This technique was used to compute collision-free *kinodynamic* trajectories for high degree-of-freedom robots with kinematic and dynamic constraints operating in a cluttered environment. Using a state space formulation, it transforms a *n*-dimension planning problem in configuration space into a 2*n*-dimension problem in state space. Song et al.[12] proposed a new method of building and querying PRMs. In this method, some of the validation checks in the building phase are postponed to the querying phase. A coarse roadmap is built during the building phase and further refined in the querying phase with focus on the area of interest, and customized to specific preferences such as maximum number of sharp turns.

3. Evolution-Based Approaches

Evolutionary Computation (EC) is a class of optimization methods which are inspired by the evolution processes found in nature. In EC-based path planning, optimization is approached through a stochastic search method. For initialization, a *population* of paths is generated at random. These paths are evaluated for fitness. The path with the highest fitness value is selected as the candidate solution of the current *generation*. The paths with high fitness values are selected and used to produce new candidate paths via *mutation* mechanisms. This evolution process continues until the candidate solution meets a certain stopping criterion. Fogel[13] applied EC to optimal routing of AUVs, and shows that EC can handle unexpected changes, multiple goal locations, detection avoidance, and cooperative goal observation. Solution to these complex problems was achieved by simply modifying a performance objective function. Xiao[14] presented an adaptive EC path planner for mobile robots. This approach combines off-line planning and on-line replanning in the same algorithm. A path is represented as a set of waypoints chosen at random connecting the initial and goal locations. The probability of selecting different mutation mechanisms is adapted during the search to improve performance. Potter[15] developed the cooperative coevolution algorithm for complex planning problems, dividing evolving solutions into several interacting co-adapted components. This work presents a case study involving the evolution of artificial neural networks and shows that this architecture can solve very complex problems which might not be possible with standard EC algorithms. Capozzi[16,17] presented an EC technique for path planning of a UAV in a simulated dynamic environment. The planning algorithm was tested in several complex scenarios with varying terrain, wind variations, dynamic obstacles, and moving targets. The simulation results show that the EC technique can efficiently search simultaneously in space and time to find feasible, near-optimal solutions. Hacaoglu and Sanderson[18] demonstrated EC-based planning using a multi-resolution path representation. Their approach does not require a map of the free configuration space. The use of the multi-resolution representation reduces the complexity and the computational cost. This work shows that the planning system is efficient for mobile robots or manipulators with many degrees of freedom. To further improve efficiency, they also proposed a multi-path planning algorithm that generates multiple alternative paths simultaneously. Rathbun[19] developed an EC-based path planner which explicitly accounts for uncertainties in the environment. This planning algorithm uses probability information of the obstacle locations, and suggests an approximation method to compute the probability of *inter-subsection* of the vehicle path with an obstacle. The probability of the inter-subsection is used in the fitness function to evaluate candidate paths. The present research builds upon these results.

B. Multi-Vehicle Planning

There is an increasing demand for applications where a single vehicle is not efficient, effective, or desirable. Examples include the desire for *distributed sensing*, small-scale or expendable vehicles and sensors, coordinated and cooperative sensing. This demand drives a need for planning methods applicable to multiple autonomous vehicles. For ease of discussion, we broadly categorize the associated efforts into three approaches: centralized, decentralized, and market-based.

1. Centralized Approaches

Centralized approaches are characterized by architectures in which only one *agent* manages the entire system. This agent can be one of the vehicles in the system or a command center. Adams[20] presented a hierarchy to control distributed teams of UAVs in military operation. The hierarchy consists of several levels which contain decision making nodes that exchange information and interact. It accounts for uncertainty in estimated states and the risk of losing team members during the mission. The proposed structure allows human operators to interact with the system at any level. Bellingham[21] presented a planning system for a fleet of UAVs using mixed-integer linear programming. The planning algorithm accounts for the probability of losing UAVs during the mission. The proposed system can improve the probability of success of the mission and the probability of survival of the vehicles. Maddula[22] assigns targets to UAVs based on the objective to minimize path length as well as target exposure in visiting all targets. The environment is represented by a *Voronoi diagram*, which is a graph of collision-free paths and waypoints assuming the environment is static. The planning algorithm computes target assignments using a semi-greedy heuristic. The target assignment is further refined using constrained exchange among the UAVs of subpaths in the Voronoi diagram. Capozzi[23] developed an EC-based planning system capable of generating and coordinating paths for multiple autonomous vehicles. He demonstrated this system in coordinated rendezvous and coordinated target coverage problems.

2. *Decentralized Approaches*

A common problem with centralized approaches to managing a large number of vehicles in complex missions is the lack of responsiveness to changes in the environment. The essence of a decentralized approach is the division a complex problem into subproblems which can be solved by components of the system. Estlin et al.[24,25] coordinated multiple *rovers* with a planning system developed to perform scientific tasks in a dynamic environment. The planning system is distributed and capable of coordinating activities among the rovers, monitoring plan execution, and performing replanning. Parker[26,27] developed a distributed behavior-based software architecture for fault tolerant cooperative control of teams of heterogeneous mobile robots. Each robot is autonomous and has ability to perform high-level functions and select appropriate actions. This system was demonstrated in a hazardous waste cleanup mission. Aicardi[28] presented a decentralized approach to coordinate motion of mobile robots based on team theory. The planned motion is derived from conservative force field techniques. Feddema[29] provides a control theoretic analysis of the problem of decentralized coordination of multiple vehicles. This work focuses on system properties such as stability, observability, and controllability. The analysis resulted in explicit limits on system parameters.

3. *Market-Based Approaches*

The concept of market-based approaches was introduced by Smith[30] in his work on the *contract net protocol* (CNP). This concept uses an economic model to coordinate multi-agent systems. Several researchers adopted the concept and extended its applicability. Sandholm[31] formalized the bidding and awarding decision process that was undefined in the original CNP. Each agent is self-interested, making decisions based on its own local criteria. This work also extends the CNP to allow trading of task-clusters. Fischer[32] developed a system for cooperative transportation scheduling. This is an extension of the CNP using task decomposition and allocation, providing increased flexibility for dynamic scheduling and execution. Wellman and Wurman[33,34] developed a market-oriented programming technique for solving distributed resource allocation problems. Autonomous agents in the system interact by offering to buy or sell commodities at fixed prices. The market system reaches an equilibrium point which is analytically demonstrated. Golfarelli[35-37] proposed a negotiation protocol based on the CNP in which the only type of contract is task swapping. The performance is improved by allowing tasks to be swapped in clusters. A clustering algorithm is presented, which considers both spatial and temporal distances between tasks. Dias and Stentz[38] presented an architecture for coordinating multiple robots based on the concept of free market systems. The proposed market architecture defines explicit revenue and cost functions for the computation of bid prices. The results show that the overall team profit can be maximized by allowing agents to be self-interested.

C. Dynamic Planning

In most applications, autonomous vehicles operate in dynamic uncertain environments, and planning systems must have the ability to dynamically replan when facing unexpected circumstances. Stenz[39,40] developed the D* algorithm, which is a dynamic variant of the A* algorithm. It generates motion plans for a mobile robot operating in a partially known environment. The algorithm handles situations where path cost parameters change during the search process. The planning algorithm is analytically shown to be optimal and efficient for sensor-equipped robots. Brumitt[41] developed a planning system for multiple mobile robots using the D* algorithm. The planning system is capable of dynamic reassignment of tasks in order to minimize mission completion time. In this approach, a set of dynamic planners are used to continually update the paths of all robots to all goals. Chien et al.[42] discussed the use of iterative repair techniques for continuous planning. This work presents an approach to integrate planning and execution in a feedback setting. This continuous planning framework is shown to improve the responsiveness of the on-board planning process to changes in the environment or mission objectives.

D. Cooperative Planning

The goal in *cooperative planning* is to achieve cooperative behavior in a system with multiple autonomous vehicles. Cooperative behavior in this context may be defined as coordinated action of multiple vehicles requiring communication between participating vehicles. Cao[43] presents a survey of early research in this field. Gillen[44] presented a system developed for finding and engaging targets using wide area search munitions in unknown environments, and methods to improve the cooperative behavior of the system. The cooperative engagement is controlled by a parameterized decision rule, and a study of the sensitivities of the parameters to the precision of autonomous target recognition is included. Bredenfeld[45] presented a framework for coordinating a team of mobile robots using a behavior-based scheme. The framework allows the system designer to implement cooperation policies using the concept of *Dual Dynamics*. Dual Dynamics combines the ideas from self-organizing systems and hybrid control in order to specify and constrain the behavior of a robot. McLain[46,47] presented a cooperative path planning approach for teams of multiple UAVs under timing constraints. This approach introduces the use of coordination variables and functions which define the cooperative strategy. The path planning problem is solved using a Voronoi diagram and Eppstein's *k-best paths* algorithm. The approach provides effective solutions to cooperative planning problems with three types of timing constraints: simultaneous arrival, tight sequencing, and loose sequencing. Polycarpou et al.[48-50] developed a distributed planning system for cooperative search by a team of autonomous vehicles. Vehicles are equipped with limited sensors and wireless communication devices. The proposed system is capable of on-line learning of the environment and generating a search map which is shared among the vehicles. Each vehicle uses the search map and the predicted states of the other vehicles to compute its collision-free trajectory that maximizes the team search coverage. The path planning algorithm is based on a *q-step* dynamic programming algorithm.

III. Stochastic World Model and Team Utility Function

In real-world applications, environmental information often is known with a limited level of certainty. It is possible to explicitly account for this uncertainty in the planning algorithms by using a stochastic model to predict future states. The team utility function, which is used to judge the quality of a candidate plan, is defined as a function of the expected values of the random variables of the stochastic model. This section presents the team utility function and the stochastic model of the system.

To simplify our notation in the following equations, any sampled signal $s(t_k)$ at time t_k , where k can take on any non-negative integer value, is simply written as $s(k)$. The system considered here consists of a team of vehicles and their environment. We call this system the *world*.

There are N_V vehicles which are assigned to perform N_T tasks. Given a planning time horizon t_N, the world model used to predict future states during the time $t_k < t \le t_N$ can be written in a discrete form as

$$x(q+1) = f(x(q), u(q)),\ q = k, k+1, \ldots, N-1 \tag{1}$$

where f is the state transition function, u is the input vector provided by the planning system, x is the state vector of the system which includes states of the vehicles x^V, states of the environment x^E, and task states x^F, i.e. $x = [x^V \quad x^E \quad x^F]^T$. We assume that the information of all the states at time t_k is available to the planner except $x^E(k)$ which is known with a limited level of certainty.

The state of each task i, x_i^F , indicates whether the task is completed; $x_i^F = 1$ when the task is initially assigned to the team, and $x_i^F = 0$ if it is completed. The states of the vehicles consist of their positions z^V , velocities $\dot{z}^V$, and health states ξ^V, i.e. $x^V = [z^V \quad \dot{z}^V \quad \xi^V]^T$. The health state of vehicle v indicates if the vehicle is alive or destroyed; $\xi_v^V = 1$ if the vehicle is alive, and $\xi_v^V = 0$ if it is destroyed.

We define sites as any objects in the environment. The states of the environment are the states of all the *sites* in the environment. Let N_S denote the number of sites. *Obstacles* are special types of sites with ability to change states of vehicles if they make contact. A *target* is defined as a site in the environment associated with a task. Thus, a site can be a target and an obstacle simultaneously or it can be neither. The number of obstacles N_O plus the number of targets N_G is not necessary equal to the number of all the sites. The states of the environment are composed of the positions z^E , velocities $\dot{z}^E$, and health states ξ^E of all the sites, i.e. $x^E = [z^E \quad \dot{z}^E \quad \xi^E]^T$. The health state of site j indicates whether the site exists or not; $\xi_j^E = 1$ if the site exists, and $\xi_j^E = 0$ if it does not.

The inputs to the system, u, are commanded positions $\overline{z}^V$ and velocities $\dot{\overline{z}}^V$ of the vehicles, and task assignment vector $\overline{d}^V$, i.e. $u = [\overline{z}^V \quad \dot{\overline{z}}^V \quad \overline{d}^V]^T$. Given a set of planned trajectories $Q(s_{p-1})$ previously computed at time $t_{s_{p-1}} < t_k$, the commanded positions and velocities of the vehicles while $t_q \in [t_k, t_{s_p})$, are given by

$$\begin{aligned} \bar{z}^V(q) &= h_x(Q(s_{p-1}), q) \\ \dot{\bar{z}}^V(q) &= h_v(Q(s_{p-1}), q) \end{aligned} \tag{2}$$

where Q represents a set of parameters needed to define the planned trajectories. The mapping functions h_x and h_v, and Q, depend on how the trajectories are encoded. t_{s_p} is the time when a new plan computed at time t_k will be deployed. We call this point in time a *spawn point*. The commanded positions and velocities of the vehicles while $t_q \in [t_{s_p}, t_N)$ are given by

$$\begin{aligned} \bar{z}^V(q) &= h_x(Q(s_p), q) \\ \dot{\bar{z}}^V(q) &= h_v(Q(s_p), q) \end{aligned} \tag{3}$$

The task assignment vector $\bar{d}^V(q)$ can be expressed as

$$\bar{d}^V(q) = \begin{bmatrix} \bar{d}_1^V(q) \\ \bar{d}_2^V(q) \\ \vdots \\ \bar{d}_{N_V}^V(q) \end{bmatrix} \tag{4}$$

where $\bar{d}_v^V(q)$ is the task assignment vector of vehicle v with N_T elements. $\bar{d}_v^V(q)$ can be expressed as

$$\bar{d}_v^V(q) = D_v(s_{p-1}) \cdot \vec{1} \tag{5}$$

while $t_q \in [t_k, t_{s_p})$, and

$$\bar{d}_v^V(q) = D_v(s_p) \cdot \vec{1} \tag{6}$$

while $t_q \in [t_{s_p}, t_N)$. Here $\vec{1}$ is the vector all of whose elements are equal to one. D_v is the decision matrix for vehicle v with dimension $N_T \times N_T$. It is an element of team task plan $D = \{D_1, D_2, \ldots, D_{N_V}\}$. The element of D_v at row i column j is denoted by d_{ij}^v. $d_{ij}^v = 1$ if vehicle v plans to execute task i at sequence number j, otherwise $d_{ij}^v = 0$.

We assume that each vehicle's guidance system can follow its commanded trajectory. Therefore, the predicted positions and velocities of the vehicles are equivalent to commanded inputs:

$$\begin{aligned} \tilde{z}^V(q) &\equiv \bar{z}^V(q) \\ \dot{\tilde{z}}^V(q) &\equiv \dot{\bar{z}}^V(q) \end{aligned} \tag{7}$$

Assuming the location of each obstacle is independent of the location of all other obstacles, the dynamic propagation of the expected health state $\tilde{\xi}_v^V$ of vehicle $v \in \{1, 2, \ldots, N_V\}$ is given by

$$\tilde{\xi}_v^V(q+1) = \tilde{\xi}_v^V(q)\prod_{j=1}^{N_O}\left(1-\tilde{H}_j^v(q+1)\right) \tag{8}$$

for $q \in \{k, k+1, \ldots, N-1\}$. N_O is the number of obstacles in the environment, and $\tilde{H}_j^v(q+1)$ is the probability that the vehicle *v* will be destroyed by obstacle *j* during the time $t_q \le t < t_{q+1}$. $\tilde{H}_j^v(q+1)$ is given by

$$\tilde{H}_j^v(q+1) = \tilde{B}_j^v(q+1)\tilde{\xi}_j^O(q)\eta_j^O \tag{9}$$

where $\tilde{B}_j^v(q+1)$ is the probability that vehicle *v* collides or intersects with obstacle *j* during the time $t_q \le t < t_{q+1}$, $\tilde{\xi}_j^O$ is the expected value of the health state of obstacle *j*, η_j^O is the effectiveness of the obstacle *j* in destroying a vehicle if they make contact. The value of η_j^O is in the range [0, 1]. $\tilde{B}_j^v$ is computed using the probability field integral approximation technique.[51]

The position of site $j \in \{1, 2, \ldots, N_S\}$ at time t_k is a random variable which can be expressed by

$$z_j^E(k) = \tilde{z}_j^E(k) + \varepsilon_j^x \tag{10}$$

where $\tilde{z}_j^E$ is the expected value of the position and ε_j^x is a random variable with zero mean and a probability density function

$$\rho_x(x, \sigma_j^x) = \begin{cases} 1/(\pi(\sigma_j^x)^2), & \|x\| \le \sigma_j^x \\ 0 & , \ \|x\| > \sigma_j^x \end{cases} \tag{11}$$

σ_j^x is a given parameter specifying the uncertainty radius of site *j*. The area within the circle with center location $\tilde{z}_j^E$ and radius σ_j^x contains all possible locations of the site.

The velocity of site $j \in \{1, 2, \ldots, N_S\}$ at time t_k is also a random variable which can be written as

$$\dot{z}_j^E(k) = \dot{\tilde{z}}_j^E(k) + \varepsilon_j^v \tag{12}$$

where $\dot{\tilde{z}}_j^E$ is the expected value of the velocity and ε_j^v is a random variable with zero mean and a probability density function

$$\rho_v(x, \sigma_j^v) = \begin{cases} 1/(\pi(\sigma_j^v)^2), & \|x\| \le \sigma_j^v \\ 0 & , \ \|x\| > \sigma_j^v \end{cases} \tag{13}$$

σ_j^v is a given parameter specifying the bound of the uncertainty in the velocity.

In the model presented here, each site is assumed to maintain constant velocity at all times. Thus, the expected velocity of site $j \in \{1, 2, \ldots, N_S\}$ while $t_q \in (t_k, t_N]$ is given by

$$\tilde{\dot{z}}_j^E(q) = \tilde{\dot{z}}_j^E(k) \tag{14}$$

As a result, the dynamic propagation of the expected position of site $j \in \{1, 2, \ldots, N_S\}$ becomes

$$\tilde{z}_j^E(q+1) = \tilde{z}_j^E(q) + \tilde{\dot{z}}_j^E(k)\Delta t \tag{15}$$

for $q \in \{k, k+1, \ldots, N-1\}$ and $\Delta t = t_{q+1} - t_q$, and the dynamic equation of the uncertainty radius σ_j^x is given by

$$\sigma_j^x(q+1) = \sigma_j^x(q) + \sigma_j^v(k)\Delta t \tag{16}$$

Both of the expected future position of a site and its uncertainty radius are used to compute the probability that a vehicle will intersect with the site.

If site $j \in \{1, 2, \ldots, N_S\}$ is not a target associated with a task, its expected health state is assumed to remain constant at all times:

$$\tilde{\xi}_j^E(q) = \tilde{\xi}_j^E(k) \tag{17}$$

for $q \in \{k+1, k+2, \ldots, N\}$. If the site is a target associated with task $i \in \{1, 2, \ldots, N_T\}$ which is to destroy the site, then

$$\tilde{\xi}_j^E(q) = \tilde{x}_i^F(k) \tag{18}$$

The dynamic propagation of the expected value of the state of task *i* is described by

$$\tilde{x}_i^F(q+1) = \tilde{x}_i^F(q) \prod_{v=1}^{N_V} \left(1 - \tilde{H}_v^i(q+1)\right) \tag{19}$$

where $\tilde{H}_v^i(q+1)$ is the probability that task *i* will be completed by vehicle *v* during the time $t_q < t \leq t_{q+1}$. $\tilde{H}_v^i(q+1)$ is given by

$$\tilde{H}_v^i(q+1) = \tilde{B}_v^i(q+1) \tilde{\xi}_v^V(q) \eta_v^V \sum_{j=1}^{N_T} d_{ij}^v \tag{20}$$

where $\tilde{B}_v^i(q+1)$ is the probability that the path of vehicle *v* intersects the target location z_i^G associated with task *i*. η_v^V is the effectiveness of the payload of vehicle *v* in executing a task. The value of η_v^V is in the range [0, 1]. d_{ij}^v is the element at row *i* and column *j* of decision matrix D_v.

We define a variable $R_i(q)$ as the predicted task score the team will have at time t_q through executing task *i*. This task score is used as a measure of success of the mission. The dynamic propagation of the expected task score is described by

$$\tilde{R}_i(q+1) = \tilde{R}_i(q) + \alpha_i^F(q)\tilde{x}_i^F(q)\left[1 - \prod_{v=1}^{N_V}\left(1 - \tilde{H}_v^i(q+1)\right)\right] \tag{21}$$

where $\alpha_i^F(q)$ is the time-dependent score weighting factor of task *i*. $\alpha_i^F(q)$ is used to define a time window for the vehicles to execute each task.

The objective of the planner is to maximize the predicted total score obtained by completing each task while minimizing the predicted total operation cost during the time $t_{s_p} < t \le t_N$ given the information about the world observed at time t_k. The objective function, also called the *team utility function*, can be written as

$$J = \sum_{i=1}^{N_T}\left(\tilde{R}_i(N) - \tilde{R}_i(s_p)\right) - \sum_{v=1}^{N_V}\tilde{C}_v(D(s_p), Q(s_p)) \tag{22}$$

where $\tilde{C}_v$ is the function used to compute the expected operation cost of vehicle *v* traveling along its planned path $Q_v(s_p)$ and executing the task plan $D_v(s_p)$ during time $t_{s_p} < t \le t_N$. Generally, operation cost of each vehicle is a function of all vehicle paths and task plans because of the coupling in the system states. Using Eq. (21), we can rewrite the team utility function as

$$J = \sum_{i=1}^{N_T}\sum_{q=s_p}^{N-1}\alpha_i^F(q)\tilde{x}_i^F(q)\left[1 - \prod_{v=1}^{N_V}\left(1 - \tilde{H}_v^i(q+1)\right)\right] - \sum_{v=1}^{N_V}\tilde{C}_v(D(s_p), Q(s_p)) \tag{23}$$

In this paper, we define the cost function $\tilde{C}_v$, $v \in \{1, 2, \ldots, N_V\}$ as

$$\tilde{C}_v(D(s_p), Q(s_p)) = \alpha_v^V\left(\xi_v^V(s_p) - \xi_v^V(N)\right) + \alpha^Q\left(1 - F_v(N)\right) \tag{24}$$

where α_v^V is the vehicle cost weighting factor and α^Q is the path cost weighting factor, and $F_v(N)$ is the ratio of the amount of fuel remaining in vehicle *v*'s fuel tank at time t_N to the full capacity of the fuel tank. The weighting factors α_i^F, α_v^V, and α^Q are parameters set based on the operator's assessment of cost of vehicle loss, task accomplishment, and fuel consumption. Specifically, α_i^F can also be time-dependent, which is a means to define task-execution time windows.

IV. Cooperative Planning

From the description of the stochastic world model described in Section 2, we can clearly see that the states of the world — task states, vehicle states, and states of the environment — are coupled. For example, the probability of success in executing a task at a time during the mission depends on the probability of survival of the vehicles at that time which, in turn, depends on the states of the obstacles intersecting with the vehicles along the path. Hence, to accurately evaluate the team utility function *J*, we need to run a simulation to predict the expected values of the world states at each discretized time step.

Given the team utility function in Eq. (23), the integrated task and path planning is an optimization problem which can be expressed as

$$\max_{D(s_p), Q(S_p)} J(D(s_p), Q(s_p)) \tag{25}$$

The optimal solution to the problem described above can only be obtained with centralized planning algorithms which take the coupling of all decision variables and system states into account, and is consequently considered very difficult. However, it can be shown that if every task assigned to the team does not involve changing states of obstacles in the environment, a distributed planning system can solve the said problem.[52] In this case, the probability that task i will be completed by vehicle v at time t_q, $\tilde{H}_v^i(q)$, can be predicted with only the knowledge of $D_v(s_p)$ and $Q_v(s_p)$ in addition to the current states of the environment. In non-cooperative planning where each task can be assigned to only one vehicle at a time,

$$\sum_{v=1}^{N_V}\sum_{j=1}^{N_T} d_{ij}^v = 1,\ \forall i \in \{1,2,\dots,N_T\} \tag{26}$$

The team utility function can then be simplified and given by

$$J = \sum_{v=1}^{N_V} J_v(D_v(s_p), Q_v(s_p)) \tag{27}$$

where

$$\begin{aligned} J_v\left(D_v(s_p), Q_v(s_p)\right) = &\sum_{i=1}^{N_T}\sum_{q=s_p}^{N-1} \alpha_i^F(q)\tilde{x}_i^F(q)\tilde{H}_v^i(q+1) \\ &-\sum_{v=1}^{N_V}\tilde{C}_v(D_v(s_p), Q_v(s_p)) \end{aligned} \tag{28}$$

We name the function J_v *individual vehicle utility function*. The team utility function in the form described above is suitable for distributed planning especially with market-based planning schemes. Because the team utility function J is now just a summation of each individual vehicle utility function J_v, the planner of vehicle v can determine whether receiving task i from vehicle w increases the team utility only with the knowledge of the difference in the individual utility of vehicle w if task i, T_i, is removed from its current set of tasks $\mathbf{T}^w$. This knowledge is essentially equivalent to the price in market-based planning systems. In this case, the price of the task i in the market is given by

$$\pi_i = -\left[J_w\left(\mathbf{T}^w - \{T_i\}\right) - J_w\left(\mathbf{T}^w\right)\right] \tag{29}$$

where $J_w\left(\mathbf{T}^w\right)$ is the individual utility function of vehicle w, and $J_w\left(\mathbf{T}^w - \{T_i\}\right)$ is the individual utility function of vehicle w when the task i is removed from its task plan.

In cooperative planning, each task is allowed to be assigned to more than one vehicle,

$$\sum_{v=1}^{N_V}\sum_{j=1}^{N_T} d_{ij}^v \geq 1,\ \forall i \in \{1,2,\dots,N_T\} \tag{30}$$

Therefore, Eq. (27) is no longer valid. In this case, each task state x_i^F depends on the plans of all vehicles cooperating on the task i. Assuming that each vehicle plans to execute each task only once,

$$\sum_{j=1}^{N_T} d_{ij}^v \leq 1,\ \forall i \in \{1,2,\dots,N_T\},\ \forall v \in \{1,2,\dots,N_V\} \tag{31}$$

the state of task *i* at time $t_l > t_{s_p}$ can be predicted by

$$\tilde{x}_i^F(l) = \tilde{x}_i^F(s_p)\prod_{v\in V_i}\left(1-\tilde{H}_v^i(\tau_v^i)\right) \ , \quad V_i = \left\{w \in \Lambda_i \mid \tau_w^i \le l\right\} \tag{32}$$

where τ_v^i is the estimated time that vehicle *v* will execute task *i*. Λ_i is the set of all vehicles which cooperate on task *i*, and $\Lambda_i \subset \left\{1, 2, \dots, N_V\right\}$. If $\tau_v^i \ne \tau_w^i$ for any $\left\{v, w\right\} \subset \Lambda_i$, the team utility function can then be written as

$$J = \sum_{v=1}^{N_V} J_v \tag{33}$$

where

$$\begin{aligned} J_v = &\sum_{i=1}^{N_T} \alpha_i^F(\tau_v^i)\tilde{x}_i^F(s_p)\prod_{w\in W_i}\left(1-\tilde{H}_w^i(\tau_w^i)\right)\tilde{H}_v^i(\tau_v^i) \\ &-\sum_{v=1}^{N_V}\tilde{C}_v(D_v(s_p), Q_v(s_p)) \end{aligned} \tag{34}$$

where $W_i = \left\{u \in \Lambda_i \mid \tau_u^i < \tau_v^i\right\}$

For optimization using market-based planning algorithms, the information that the planner of vehicle *v* needs from the other cooperating vehicles $w \in \Lambda_i$ to make trading decisions for each task $i \in \left\{1, 2, \dots, N_T\right\}$ includes

1) price (π_i)
2) estimated execution time (τ_w^i)
3) probability that task *i* will be executed successfully at time τ_w^i ($\tilde{H}_w^i(\tau_w^i)$)

We call this information *cooperation data set* of task *i*. Therefore, the minimum amount of information needed to be shared among the vehicles in order to make decisions comprises the cooperation data sets of all the tasks.

Using the team utility function given in Eq. (33), the integrated task and path planning problem can be simply considered as a three-level optimization problem. The top-level subproblem is the task allocation problem which can be written as

$$\max_{A(s_p)} \sum_{v=1}^{N_V} J_v(D_v^*(s_p), Q_v^*(s_p)) \tag{35}$$

where *A* is a $N_T \times N_V$ *task allocation matrix* whose element at row *i* and column *v* is given by

$$a_{iv} = \sum_{j=1}^{N_T} d_{ij}^{v*} \tag{36}$$

and d_{ij}^{v*} is the element of D_v^* at row *i* column *j*. D_v^* and Q_v^* are the solutions of the middle-level task scheduling problems which are given by

$$\max_{D_v(s_p)} J_v(D_v(s_p), Q_v^*(s_p)),\ \forall v \in \{1, 2, \ldots, N_V\} \tag{37}$$

with $D_v(s_p)$ whose element d_{ij}^v satisfies the condition

$$\sum_{j=1}^{N_T} d_{ij}^v = a_{iv}^\dagger,\ \forall i \in \{1, 2, \ldots, N_T\} \tag{38}$$

where $a_{iv}^\dagger$ is the element at row *i* and column *v* of a candidate solution of the top-level problem $A^\dagger$. Q_v^* is the solution of the bottom-level path planning problem which is given by

$$\max_{Q_v(s_p)} J_v(D_v^\ddagger(s_p), Q_v(s_p)),\ \forall v \in \{1, 2, \ldots, N_V\} \tag{39}$$

where $D_v^\ddagger$ is a candidate solution of the task scheduling problem defined in Eq. (37).

V. Planning System

To address the planning problem stated in the previous section, we developed a cooperative task and path planning system for multiple autonomous vehicles operating in dynamic uncertain environments. We name this system *Evolution-based Cooperative Planning System* (ECoPS).[52] ECoPS is capable of allocating tasks among the vehicles in a team and generating feasible paths that support the assigned tasks in distributed fashion. ECoPS is a real-time planning system in that it runs continuously and dynamically reallocates tasks and adapts the planned trajectories to the changes in the environment, and it can provide a solution by interrupting the computations at any time during the sampling period.

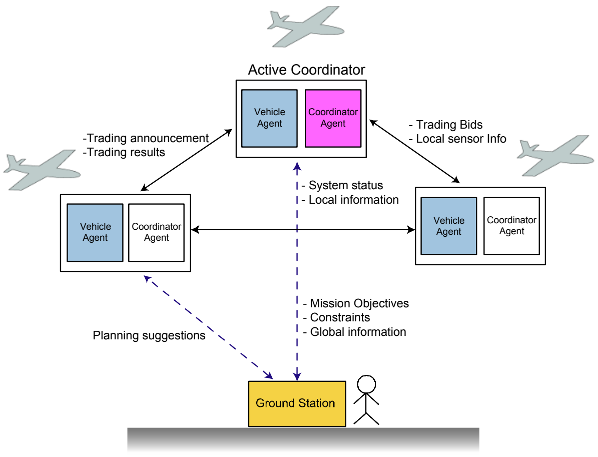

Fig. 2 Distributed structure of ECoPS

The overall structure of ECoPS is illustrated in Fig. 2. In this system, a market scheme and evolutionary algorithms are used to compute near-optimal paths and task plans. This concept is motivated by optimization processes in market economies and the evolution process in nature. Team composition, overall team objectives, and the tasks required in attaining these objectives are specified by the command center at the start of a specific mission. Individual vehicles are autonomous. Each vehicle has an on-board intelligent processing unit called *vehicle agent*. Vehicle agents perform task trading, communication, and planning. Under the market protocol, each vehicle can choose its own tasks and plan its path that will in turn benefit the team. Accomplishing tasks will bring reward to the team. However, there is cost associated with executing the tasks. The function of the task planning is to schedule and allocate tasks to those vehicles that can achieve them optimally.

This planning architecture considers the vehicles as if they are in a market where the trading items are tasks. In contrast to systems using the Contract Net Protocol (CNP),[30] this market is not a free market that allows anyone to trade directly to others. In this system, vehicles buy or sell tasks through one of the vehicles acting as the team coordinator. Each vehicle has another on-board processing unit called *coordinator agent*, but only one is active at any give time. The computational requirement for a running coordinator agent is small comparing to planning. The presence of the coordinator may weaken the notion of distributed systems. However, this planning system largely retains the same properties as those of a completely distributed one. Having a coordinator in the team enables organized and efficient trading in the market, as well as facilitates human operator interaction with the system. Another advantage of having the coordinator is monitoring of the team members and the current task allocation. Therefore, if the current coordinator finds that a vehicle is destroyed, it will put its tasks up for auction. Conversely, if the vehicle acting as the coordinator is damaged, a simple mechanism can be executed to elect another vehicle to be a new coordinator from the existing team members.

Key components in the ECoPS are vehicle agents. A vehicle agent contains a planner and a communication unit called *communicator* which interacts with the active coordinator agent and its local planner. In each vehicle, the planner is the brain of the vehicle agent. Each vehicle's planner has three components: task planner, path planner, and state predictor. The main function of the task planner is making trading decisions in order to obtain a set of tasks that the vehicle can do best for the team. The path planner determines the best task sequence and path that supports the actions required by the obtained tasks. In the planning process, the state predictor is used by both of the task planner and the path planner to predict future states of the world for a given set of task and a candidate path.

The task planner uses market price, shared information, and predicted states of the world to make trading decisions. An Evolutionary Computation (EC) based technique is used as the optimization engine of the task planner. One major beneficial property of EC-based techniques for on-line planning is the availability of an intermediate solution at any time during the optimization process. A viable solution is available and continues to be improved until its output is required. Therefore, the task planner can provide a trading decision at any time to meet a deadline for submitting sell or buy bids.

To achieve the optimal solution for a task allocation problem, the task planner needs to know the optimal path for the execution of each possible sequence of tasks. However, finding the optimal paths for all of these task sequences is subjected to the *curse of dimensionality*. For example, the number of all possible sequences of ten tasks is more than ten million. Finding the optimal path for just one of them is considered a hard problem. In our approach, the task planner also searches for the best task sequence in the process of making trading decisions. However, it does not search for the optimal path for the execution of the resultant task sequence. Given a task sequence, the task planner approximates the team utility based on the predicted states of the world provided by the state predictor. The details of this approximation method are provided in Ref. 52. This technique improves the speed of the response of the planning system, and can provide viable solutions within short planning time.

The task planner of each vehicle also interacts with its path planner. It provides the current set of tasks as an input to the path planner. The path planner computes the best path to execute the assigned tasks based on the predicted states of the world. The path planner then sends this resultant path to the vehicle guidance controller and to the task planner. As in task planning, an EC-based algorithm is used in the path planning. In general, the algorithm initializes its set of candidate solutions, a *population*, at random. These candidate solutions are feasible paths, i.e. the random elements of the paths are all within the capabilities of the aircraft. They are not required to be close to an optimal solution for the evolution to succeed. Through the evolution process (mutation and selection), the candidate solutions move toward the optimal solutions. Collision avoidance is achieved by modeling each teammate as a dynamic site with a defensive range and effectiveness, and requires that each vehicle has information about the position and velocity vector of its teammates. Detailed implementation of the path planner is provided in Refs. 52 and 53. In the case of a problem where the search space is large and there are several local optima, a parallel evolution approach[54] can improve the performance of the planning. In this approach, multiple populations evolve simultaneously and compete with each other periodically, reducing the likelihood of being trapped in local optimal solutions. This approach also improves the performance of the evolution process in the event that the *fitness landscape* is flat. The task planner and path planner run simultaneously and continuously, although they may be updated at different rates.

It has been shown that the ECoPS can provide effective solutions for off-line planning, and it can dynamically adapt the generated plans in response to unexpected changes in the environment.[55] However, the original ECoPS was designed for non-cooperative planning only. We modified our market-based planning system to be used in cooperative planning. In the modified ECoPS, a vehicle can cooperate on a task with other vehicles only if its current task plan is empty. Each vehicle is allowed to cooperate on only one task at a time. It also needs to bid for the right to cooperate on the chosen task. Once the cooperation right has been received from the coordinator, the cooperating vehicle must plan to execute the task after the vehicle which owns the task does. For each task, there are several levels of the cooperation right that a vehicle can bid on. Each vehicle must execute the cooperating task after the other vehicles which have the cooperation right at lower levels have done so. The larger the number of cooperation levels of a task, the larger the number of vehicles working together on the task.

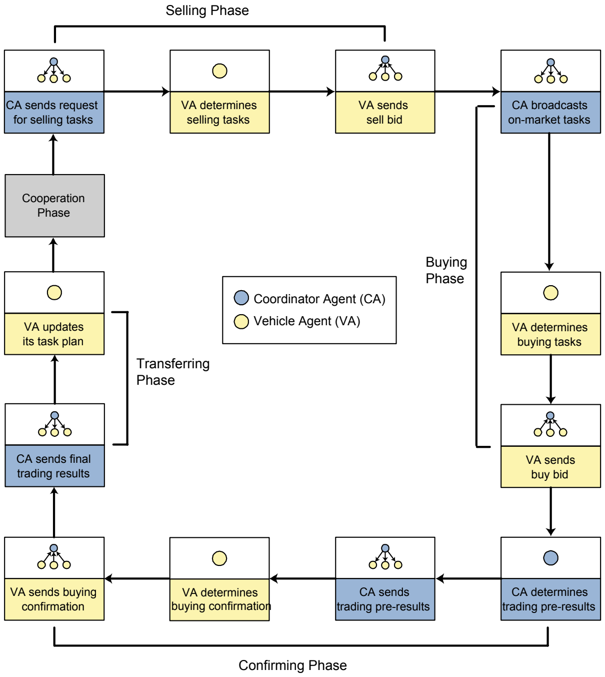
Selling Phase
CA sends request for selling tasks
VA determines selling tasks
VA sends sell bid
CA broadcasts on-market tasks
Buying Phase
VA determines buying tasks
VA sends buy bid
CA determines trading pre-results
CA sends trading pre-results
VA determines buying confirmation
VA sends buying confirmation
Confirming Phase
CA sends final trading results
VA updates its task plan
Transferring Phase
Cooperation Phase
Coordinator Agent (CA)
Vehicle Agent (VA)

Fig. 3 Cooperative task trading sequence.

Each vehicle selects a cooperation level of the task it plans to cooperate on and its execution time using the team utility function given in Eq. (33) where W_i in the equation is the set of the vehicles which cooperate on task i at lower levels. The individual vehicle utility of each cooperating vehicle given in Eq. (34) is called *cooperation utility*. To accommodate the cooperative planning process, we modified the task trading sequence by adding an additional phase called *cooperation phase* after the transferring phase. The modified task trading sequence is shown in Fig. 3. The cooperation phase comprises several steps which allow information sharing and the bidding for cooperation rights. In the other phases, the planner of each vehicle makes trading decisions using the team utility function given in Eqs. (27) and (28).

A vehicle can bid on an available cooperation right in the market. It can also bid on a cooperation right at a new higher level which no other vehicle possesses if its cooperation improves the team utility. In the cooperation phase, each cooperating vehicle must sell its cooperation right to the coordinator agent at a price. All selling cooperation rights are accepted. The following items describe the steps in the cooperation phase.

i. Each vehicle broadcasts the cooperation data sets of all the tasks in its current task plan once it received the final trading results from the coordinator agent.
ii. The coordinator agent broadcasts a request for cooperation data to the cooperating vehicles.
iii. Each vehicle sends the cooperation data set of its cooperating task to the coordinator agent.
iv. The coordinator agent broadcasts all the cooperation data sets received from the cooperating vehicles.
v. The coordinator agent broadcasts a request for sell prices of all cooperation rights.
vi. Each cooperating vehicle determines the sell price of the right to execute the cooperating task at the level it possesses. It also sends the updated cooperation data set of the task to the coordinator agent using the cooperation data received from other cooperating vehicles.
vii. The coordinator agent broadcasts the price of all the selling cooperation rights and the updated cooperation data sets received from the cooperating vehicles.

viii. The coordinator agent broadcasts a request for buy bids for cooperation rights.
ix. Each agent with empty task plan determines a buy bid for a selling cooperation right or a cooperation right of a task at a new higher level.
x. The coordinator agent determines and broadcasts the bidding results.

The price strategy of a selling cooperation right is the same as that used in the selling phase. Both the sell price and the buy price are equal to the cooperation utility given in Eq. (34). The planner of each cooperating vehicle iteratively searches for a task and its cooperation level to bid that most increases the team utility.

The ECoPS was designed to solve the class of complex problems we have presented here. Such problems have not been solved previously with the given constraints. Evolutionary Computation techniques were chosen for their algorithmic efficiency for solving complex constrained non-linear optimization problems, recognizing the fact that they are not as computationally efficient in finding the optimal solution to simple convex problems as other, more traditional methods. The penalty paid is that the algorithms do not lend themselves to analytical assessment due to the complexity in the nature of the EC and Market-based planning techniques.

VI. Simulation Results

In this section we present two planning examples to demonstrate the performance of the planning system to generate off-line plans (solve the task and path planning problem prior to start of mission based on initial knowledge of the scenario), and dynamically replan in response to the changes in the environment once the mission is underway. The mission objective is to observe all the targets assigned to the team. The planning examples presented here are based on simulations using the Boeing Company developed simulation environment, the Open Experimental Platform (OEP). The computations for each of the examples include both the actual algorithms which generate plans for all vehicles as well as simulation of the environment. The algorithms were implemented in C++, and processed on one 2.0 GHz Pentium 4 PC. In an onboard system, the planner of each vehicle runs on a dedicated processor.

In these planning problems, the path planner of each vehicle runs the evolution-based optimization algorithm with a population size of 30. The tournament selection algorithm selects 15 of them to be parents for the next evolution step. During the simulation, the evolution-based path planner evolves one generation every 10 seconds simulation time. We intentionally set the number of evolution steps to be small to represent the situation where the on-board processing unit has limited computational power. The size of the execution time horizon (time difference between two consecutive spawn points) is 100 seconds. The sample period to run one round of trading is 40 seconds simulation time. The computation time for running each execution time period was less than one second.

All vehicles have complete knowledge of the environment as it changes. Each vehicle needs to estimate 1) the binary state of each of the tasks it has in its current task plan and those available in the market, 2) the position and velocity of each of the targets associated with the tasks, 3) the position, velocity, and binary health state of each dynamic obstacles, 4) the position, velocity, and binary health state of its teammates. Each vehicle has an on-board camera for observing targets. We assume that the camera can rotate 360 degrees. Each vehicle also has a sensor for detecting nearby sites. The sensors have a specified range. There are some obstacles in the field which the vehicles need to avoid, for these examples, the obstacles have defensive capabilities and the ability to destroy a vehicle when it comes into range. The vehicles are required to reach a goal location after executing their tasks. The score weighting function of each is shown in Fig. 4. The profile of this function is the same for both planning examples presented here.

In the plots showing simulation results, each vehicle is represented by a triangle with its vehicle number on it. The dashed circle around each vehicle represents the range of the on-board camera. The square markers represent actual locations of targets. Each of these square markers will have a vehicle number on it if the site is a target and assigned to that vehicle. A solid circle located near each square marker represents an area which covers all of the possible locations of the site represented by the marker known to the vehicles. The radius of the circle shrinks as a vehicle gets closed to the site. Each filled square marker with a dashed circle around it represents a site with defensive capabilities which can destroy or change the health states of vehicles if they are within the area marked by the dashed circle. The goal location where the vehicle is required to be at the end of the mission is represented by a hexagram in the plots.

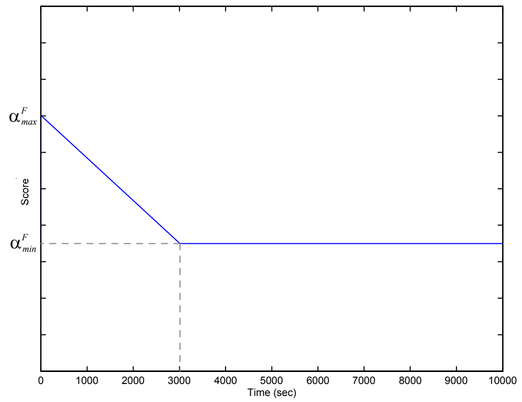

Fig. 4 Normal profile of score weighting function.

The first example is a simple planning problem with five vehicles and one target. The effectiveness of each vehicle's payload is 0.8 so a vehicle's encounter with the designated target does not automatically guarantee successful observation. There is also an obstacle near the target. Initially the target is not moving and is outside the effective protective range of the obstacle. Figure 5 shows the off-line planning result. The target is assigned to vehicle 1 and vehicle 2 decides to cooperate on the target. The other vehicles decide not to cooperate and plan their paths directly to the goal location. The results of dynamic replanning are shown in Figs. 6-8 which are snapshots from the simulation movie "Simple Cooperation". Once the simulation starts, the target begins moving toward the protective range of the obstacle. Figure 6 shows the simulation result at time 1400 seconds. At this time, the target has moved to be within the effective protective range of the obstacle. The increased difficulty of the task causes the predicted team utility to drop as shown in Fig. 9. Since there is less chance that vehicle 1 and vehicle 2 will executed the task successfully, vehicle 3 and vehicle 4 decide to cooperate on the target. Inspection of the individual vehicle utility shows that vehicle 1 missed the target and traded it to vehicle 2 which later execute the task successfully. This result can be seen in Fig. 7. Vehicle 1, vehicle 2, and vehicle 3 are destroyed quickly after the task was completed. Since vehicle 4 detects the completion of the task, it just adapts its path from going toward the obstacle to instead go directly to the goal which can be seen in Fig. 8.

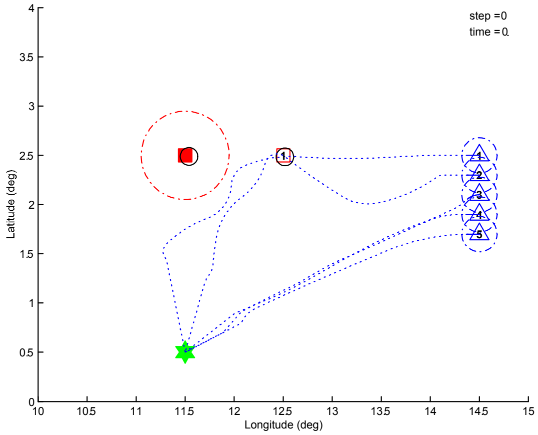

Fig. 5 Initial result of offline-planning.

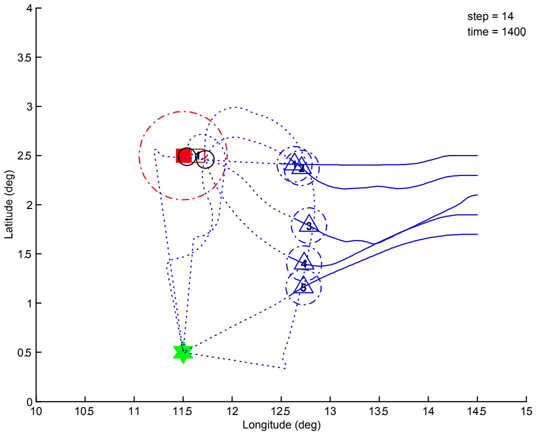

Fig. 6 Simulation result at time 1400 seconds.

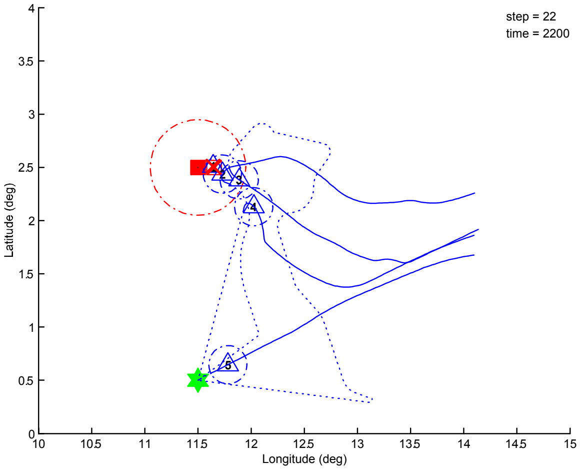

Fig. 7 Simulation result at time 2200 seconds.

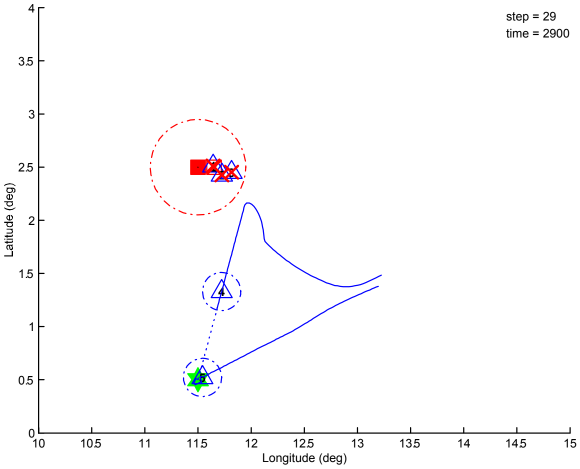

Fig. 8 Simulation result at time 2900 seconds.

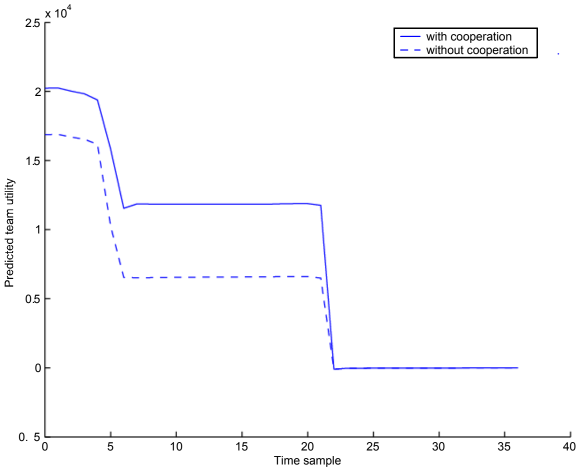

Fig. 9 Predicted team utility during the mission.

The second example is a more complicated scenario. In this scenario, there are 5 vehicles and 3 targets. The effectiveness of each vehicle's payload is 0.8. Initial knowledge is that all targets and obstacles are stationary. The score weighting factor of the target located at (10.5, 3) is two times of that of the other targets. The target located at (10.5,3) has $\sigma^F_{\min} = 10000$ and $\sigma^F_{\max} = 20000$. The other targets have $\sigma^F_{\min} = 5000$ and $\sigma^F_{\max} = 10000$. After 150 evolution steps of path planning and 30 rounds of task trading, the result of off-line planning is shown in Fig. 10. In this result, the three targets are assigned to vehicle 1, vehicle 2, and vehicle 3. Vehicle 4 and vehicle 5, however, decide to cooperate with vehicle 1 and vehicle 3.

Figures 11-15 are snapshots from the simulation movie "Multi-Targets Cooperation" which shows the on-line replanning results. At time 300 seconds, all three targets have moved to new locations and are assigned to vehicle 2, vehicle 3 and vehicle 5 as shown in Fig. 11. Based on the value of cooperation utility, vehicle 1 and vehicle 4 are cooperating with the other vehicles. However, their planners have not been able to find paths to the targets they are cooperating on. At time 1600 seconds, the planner of vehicle 4 has found a path to the target which is owned by vehicle 3 at that moment. Vehicle 1 has moved faster toward the high-value target than vehicle 2, so the target is assigned to vehicle 1. Vehicle 2, however, keeps moving toward the target to cooperate with vehicle 1. Figure 13 shows the simulation result at time 2500 seconds.

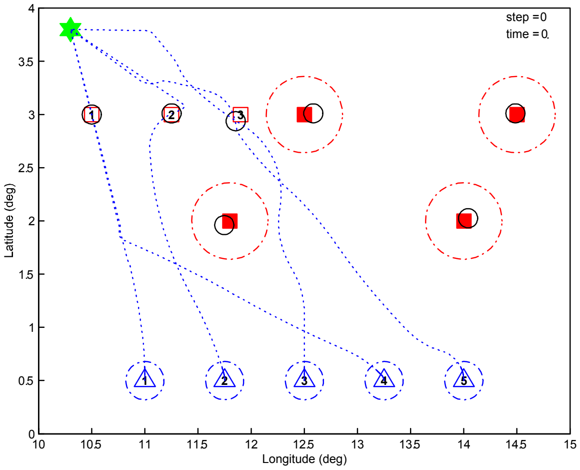

Fig. 10 Initial result of offline-planning.

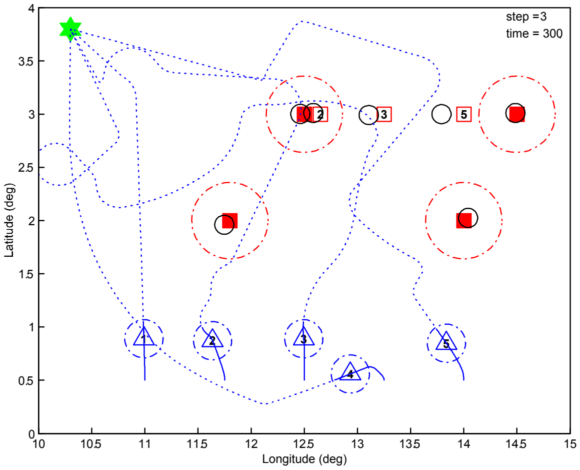

Fig. 11 Simulation result at time 300 seconds.

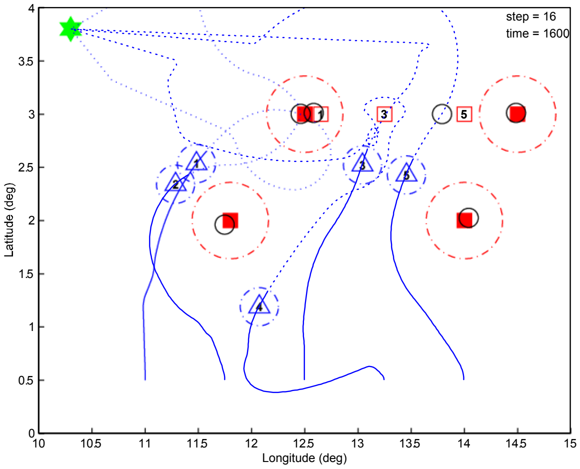

Fig. 12 Simulation result at time 1600 seconds.

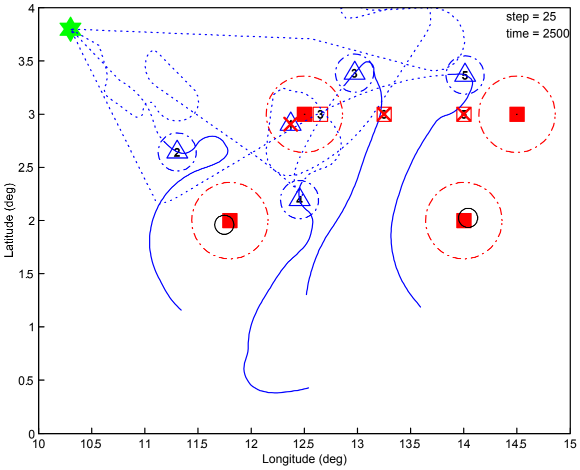

Fig. 13 Simulation result at time 2500 seconds.

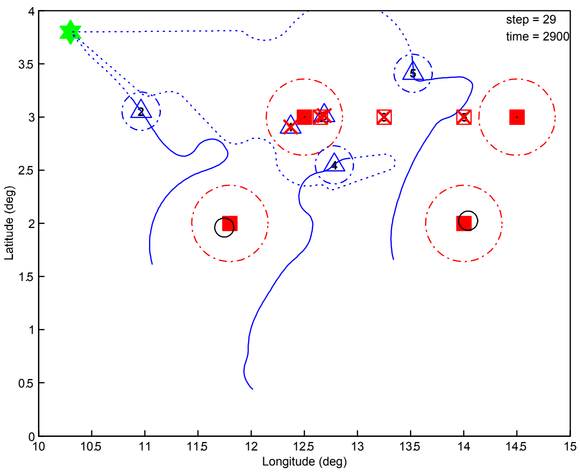

Fig. 14 Simulation result at time 2900 seconds.

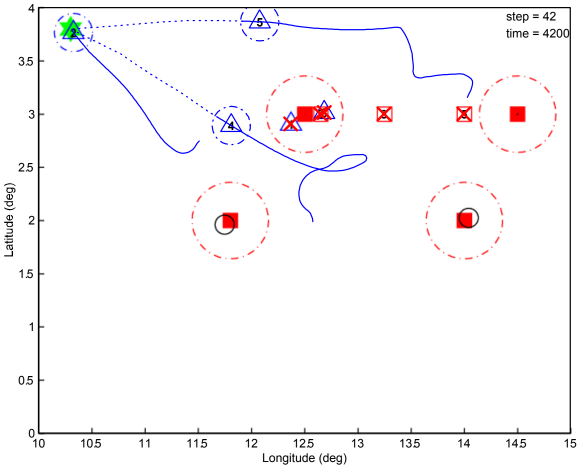

Fig. 15 Simulation result at time 4200 seconds.

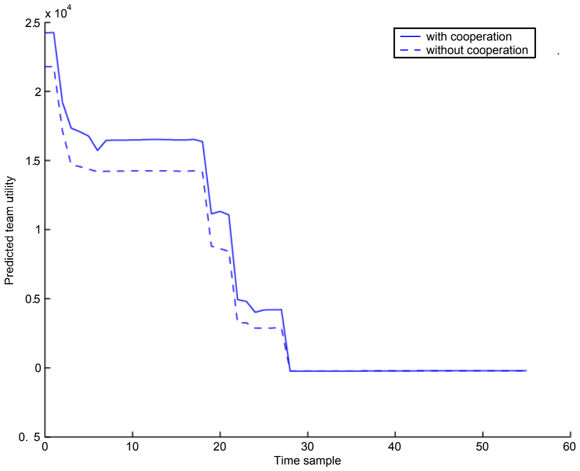

Fig. 16 Predicted team utility during the mission.

At this point in time, vehicle 3 and vehicle 5 have completed their previously assigned tasks. The only unfinished task is taking a picture of the target which stays within the range of the obstacle located at (12.5,3). Vehicle 1 attempted to reach the target, but it was destroyed in the process by the obstacle. At that moment, the task is assigned to vehicle 3 which is the closest vehicle to the target. Vehicle 4 and vehicle 5 decide to cooperate with vehicle 3 on the task because doing that gains the team utility as shown in Fig. 16. Vehicle 2 decides not to cooperate and is on its way to the goal location. At time 2900, all the tasks are completed, but the team lost another vehicle in executing the last task. Once the completion of the task is detected, vehicle 4 quickly adapts its path to avoid getting within the nearby obstacle range. Figure 15 shows that vehicle 4 successfully goes around the said obstacle. Both vehicle 4 and vehicle 5 are heading straight to the goal location.

VII. Conclusion

This article introduces a cooperative task and path planning system for a team of autonomous vehicles in a dynamic uncertain environment. We present an analytic dynamic world model and a team utility function both of which take into account the uncertain information of the environment and the coupling in the states of the system. Using the stochastic world model, we develop algorithms for allocating tasks and path planning simultaneously. By modifying the task trading sequence and sharing some information among the vehicles, the planning system computes cooperative plans which improve the performance of the team operation compared to that without the cooperation scheme. We identified the minimum amount of shared information needed for cooperation. The simulation results show that the planning system can be used effectively for both off-line planning and on-line replanning in a dynamic uncertain environment. During the simulation, the planning system ran continuously and dynamically reallocates tasks and adapts the planned trajectories to the changes in the environment.

Acknowledgment

The research presented here was supported in part by the DARPA Mixed Initiative Control of Automa-teams program and by the Washington Technology Center. The simulation software is provided by the Boeing Company. Juris Vagners provided direction for this research.

References

[1] Office of the Secretary of Defense, "Unmanned Aerial Vehicles Roadmap," Tech. Rept., Dept. of Defense, Dec. 2002.

[2] Murphy, R., *Introduction to AI Robotics*, MIT Press, Cambridge, MA, 2000.

[3] Nilsson, N., *Principles of Artificial Intelligence*, Tioga Publisher Company, Palo Alto, CA, 1980.

[4] Thrun, S., Bücken, A., Burgard, W., Fox, D., Fröhlinghaus, T., Hennig, D., Hofmann, T., Krell, M., and Schimdt, T., "Map Learning and High-speed Navigation in RHINO," *Artificial Intelligence and Mobile Robots*, MIT Press, Cambridge, MA, 1998.

[5] Mata, C., and Mitchell, J., "A New Algorithm for Computing Shortest Paths in Weighted Planar Subdivisions," *Symposium on Computational Geometry*, 1997, pp. 264-273.

[6] Mandow, A., Mandow, L., Munoz, V., and Garcia-Cerezo, A., "Multi-Objective Path Planning for Autonomous Sensor-Based Navigation," *Proceedings of the IFAC Workshop on Intelligent Components for Vehicles*, 1998, pp. 377-382.

[7] Bander, J., and White, C., "A Heuristic Search Algorithm for Path Determination with Learning," *IEEE Transactions of Systems, Man, and Cybernetics - Part A: Systems and Humans*, Vol. 28, January 1998, pp. 131-134.

[8] Kavraki, L. E., Svestka, P., Latombe, J., and Overmars, M. H., "Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces," *IEEE Transactions on Robotics and Automation*, Vol. 12, No. 4, 1996, pp. 566-580.

[9] Overmars, M., and Svestka, P., "A Probabilistic Learning Approach to Motion Planning," *Algorithmic Foundations of Robotics, The 1994 Workshop on Algorithmic Foundations of Robotics*, edited by Goldberg, Halperin, Latombe, and Wilson, 1995.

[10] Kavraki, L., and Latombe, J., "Probabilistic Roadmaps for Robot Path Planning," *Practical Motion Planning in Robotics: Current Approaches and Future Directions*, edited by K. Gupta and A. del Pobil, John Wiley, 1998, pp. 33-53.

[11] LaValle, S., and Kuffner, J., "Randomized Kinodynamic Planning," *Proceedings of IEEE International Conference on Robotics and Automation*, 1999.

[12] Song, G., Miller, S., and Amato, N., "Customizing PRM Roadmaps at Query Time," *Proceedings of IEEE International Conference on Robotics and Automation*, 2001.

[13] Fogel, D. and Fogel, L., "Optimal Routing of Multiple Autonomous Underwater Vehicles through Evolutionary Programming," *Proceedings of the 1990 Symposium on Autonomous Underwater Vehicle Technology*, 1990, pp. 44-47.

[14] Xiao, J., Michalewicz, Z., Zhang, L., and Trojanowski, K., "Adaptive evolutionary planner/navigator for mobile robots", *IEEE Transactions on Evolutionary Computation*, Vol. 1, April 1997, pp. 18-28.

[15] Potter, M., and Jong, K., "Cooperative Coevolution: An Architecture for Evolving Coadapted Subcomponents," *Evolutionary Computation*, Vol. 8, 2000, pp. 1-29.

[16] Capozzi, B., and Vagners, J., "Evolving (Semi)-Autonomous Vehicles," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Aug. 2001.

[17]Capozzi, B., and Vagners, J., "An Evolution-Based Alternative for Autonomous Motion Planning," *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, Aug. 2001.

[18]Hocaoglu, C., and Sanderson, A., "Planning Multiple Paths with Evolutionary Speciation," IEEE Transactions on Evolutionary Computation, Vol. 5, No. 3, June 2001, pp. 169-191.

[19]Rathbun, D., and Capozzi, B., "Evolutionary Approaches to Path Planning Through Uncertain Environments," *Proceedings of the 1st AIAA Unmanned Aerospace Vehicles, Systems, Technologies, and Operations Conference and Workshop*, May 2002.

[20]Adams, M., Hall, W., Hanson, M., Zacharias, G., McEneaney, W., and Fitzpatrick, B., "Mixed Initiative Planning and Control under Uncertainty," *Proceedings of the 1st AIAA Unmanned Aerospace Vehicles, Systems, Technologies, and Operations Conference and Workshop*, Portsmouth, VA, May 2002.

[21]Bellingham, J. S., Tillerson, M., Alighanbari, M., and How, J. P., "Cooperative Path Planning for Multiple UAVs in Dynamic and Uncertain Environments," *Proceedings of the 41st IEEE Conference on Decision and Control*, Vol. 3, December 2002, pp. 2816-2822.

[22]Maddula, T., Minai, A., and Polycarpou, M., "Multi-Target Assignment and Path Planning for Groups of UAVs," *Proceedings of the Conference on Cooperative Control and Optimization*, Gainesville, FL, 2002.

[23]Capozzi, B., "Evolution-Based Path Planning and Management for Autonomous Vehicles," Ph.D. Dissertation, Univ. of Washington, 2001.

[24]Estlin, T., Rabideau, G., Mutz, D., and Chien, S., "Using Continuous Planning Techniques to Coordinate Multiple Rovers," *the IJCAI99 Workshop on Scheduling and Planning meet Real-time Monitoring in a Dynamic and Uncertain World*, Stockholm, Sweden, August 1999.

[25]Estlin, T., Yen, J., Petras, R., Mutz, D., Castaño, R., Rabideau, G., Steele, R., Jain, A., Chien, A., Mjolsness, E., Gray, A., Mann, T., Hayati, S., and Das, H., "An Integrated Architecture for Cooperating Rovers," *Proceedings of the International Symposium on Artificial Intelligence Robotics and Automation in Space (ISAIRAS)*, Noordwijk, The Netherlands, June 1999.

[26]Parker, L., "On the Design of Behavior-Based Multi-Robot Teams," *Journal of Advanced Robotics*, Vol. 10, No. 6, 1996.

[27]Parker, L., "ALLIANCE: An Architecture for Fault Tolerant Multi-Robot Cooperation," *IEEE Transactions on Robotics and Automation*, Vol. 14, No. 2, 1998, pp. 220-240.

[28]Aicardi, M., "Coordination and Control of a Team of Mobile Robots," *Proceedings of the 1995 INRIA/IEEE Symposium on Emerging Technologies and Factory Automation*, Oct. 1995.

[29]Feddema, J. T., Lewis, C., and Schoenwald, D. A., "Decentralized Control of Cooperative Robotic Vehicles: Theory and Application," *IEEE Transactions on Robotics and Automation*, Vol. 18, No. 5, October 2002, pp. 852-864.

[30]Smith, R. G., "The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver," *IEEE Transactions on Computers*, Vol. 10, No. 12, Dec. 1980, pp. 1104-1113.

[31]Sandholm, T., "An Implementation of the Contract Net Protocol Based on Marginal Cost Calculations," *Proceedings of the 12th International Workshop on Distributed Artificial Intelligence*, 1993, pp. 295-308.

[32]Fischer, K., Müller, J. P., and Pischel, M., "A Model for Cooperative Transportation Scheduling," *Proceedings of the 1st International Conference on Multiagent Systems (ICMAS'95)*, San Francisco, CA, June 1995.

[33]Wellman, M., "Market-Oriented Programming: Some Early Lessons," *Market-Based Control: A Paradigm for Distributed Resource Allocation*, edited by S. Clearwater, World Scientific, 1996, Chap. 4.

[34]Wellman, M., and Wurman, P., "Market-Aware Agents for a Multiagent World," *Robotics and Autonomous Systems*, Vol. 24, Sept. 1998, pp. 115-125.

[35]Golfarelli, M., Maio, D., and Rizzi, S., "A Task-Swap Negotiation Protocol Based on the Contract Net Paradigm," Technical Report 005-97, Research Center for Computer Science and Telecommunication Systems, University of Bologna, Bologna, Italy, 1997.

[36]Golfarelli, M., Maio, D., and Rizzi, S., "Multi-Agent Path Planning Based on Task-Swap Negotiation," *Proceedings of the 16th UK Planning and Scheduling SIG Workshop*, Durham, England, 1997.

[37]Golfarelli, M., and Rizzi, S., "Spatio-Temporal Clustering of Tasks for Swap-based Negotiation Protocols in Multi-Agent Systems," *Proceedings 6th International Conference on Intelligent Autonomous Systems*, 2000, pp. 172-179.

[38]Dias, M., and Stentz, A., "A Free Market Architecture for Distributed Control of a Multirobot System*," Proceedings of the 6th International Conference on Intelligent Autonomous Systems*, July 2000, pp. 115-122.

[39]Stentz, A., "Optimal and Efficient Path Planning for Partially-Known Environments," *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '94)*, May 1994, pp.3310-3317.

[40]Stentz, A., "The Focussed D* Algorithm for Real-Time Replanning," *Proceedings of the International Joint Conference on Artificial Intelligence*, Aug. 1995.

[41]Brumitt, B., and Stentz, A., "Dynamic Mission Planning for Multiple Mobile Robots," *Proceedings of the IEEE International Conference on Robotics and Automation*, April 1996.

[42]Chien, S., Knight, R., Stechert, A., Sherwood, R., and Rabideau, G., "Using Iterative Repair to Improve the Responsiveness of Planning and Scheduling for Autonomous Spacecraft," *IJCAI99 Workshop on Scheduling and Planning meet Real-time Monitoring in a Dynamic and Uncertain World,* Stockholm, Sweden, August 1999.

[43]Cao, Y. U., Fukunaga, A. S., Kahng, A. B., and Meng, F., "Cooperative Mobile Robotics: Antecedents and Directions," Autonomous Robots, Vol. 4, 1997, pp. 1-23.

[44]Gillen, D., and Jacques, D., "Cooperative Behavior Schemes for Improving the Effectiveness of Autonomous Wide Area Search Munitions," *Proceedings of the Cooperative Control and Optimization Workshop*, Dec. 2000.

[45]Bredenfeld, A., and Kobialka, H., "Team Cooperation Using Dual Dynamics," *Lecture Notes in Computer Science*, Springer-Verlag, 2001, pp. 111-124.

[46]McLain, T., Beard, R., and Kelsey, J., "Experimental Demonstration of Multiple Robot Cooperative Target Intercept," Tech. Note, Brigham Yong Univ., 2001.

[47]McLain, T., and Beard, R., "Cooperative Path Planning for Timing-Critical Missions," *Proceedings of the American Control Conference*, June 2003.

[48]Polycarpou, M., Yang, Y., and Passino, K., "A Cooperative Search Framework for Distributed Agents," *Proceedings of the 2001 IEEE International Symposium on Intelligent Control*, Mexico City, Mexico, September 2001, pp. 1-6.

[49]Passino, K., Polycarpou, M., Jacques, D., Pachter, M., Liu, Y., Yang, Y., Flint, M., and Baum, M. "Cooperative Control for Autonomous Air Vehicles," *Proceedings of the Cooperative Control and Optimization Workshop*, Gainesville FL, December 2000.

[50]Flint, M., Polycarpou, M., and Fernandez, E., "Cooperative Path-Planning for Autonomous Vehicles Using Dynamic Programming," *Proceedings of the International Federation of Automatic Control 15th IFAC World Congress*, Barcelona, Spain, July 2002.

[51]Rathbun, R., and Capozzi, B., "Evolutionary Approaches to Path Planning Through Uncertain Environments," *Proceedings of the AIAA Unmanned Unlimited Conference*, May 2002

[52]Pongpunwattana, A., "Real-Time Planning for Teams of Autonomous Vehicles in Dynamic Uncertain Environments," Ph.D. Dissertation, Univ. of Washington, 2004.

[53]Rathbun, R., Kragelund, S., Pongpunwattana, A., and Capozzi, B., "An Evolutionary Based Path Planning Algorithm for Autonomous Motion of a UAV Through Uncertain Environments," *Proceedings of the AIAA Digital Avionics Systems Conference*, 2002.

[54]Jia, D., and Vagners, J., "Parallel Evolutionary Algorithms for UAV Path Planning," *Proceedings of the AIAA Unmanned Unlimited Systems Conference*, 2004.

[55]Pongpunwattana, A., Rysdyk, R., Vagners, J., Rathbun, D., "Market-based Co-evolution Planning for Multiple Autonomous Vehicles," *Proceedings of the AIAA Unmanned Unlimited Conference*, San Diego, CA, 2003.