Chapter 1

MULTI-TASK ALLOCATION AND PATH PLANNING FOR COOPERATING UAVS

John Bellingham
Department of Aeronautics and Astronautics
Massachusetts Institute of Technology
john_b@mit.edu

Michael Tillerson
Department of Aeronautics and Astronautics
Massachusetts Institute of Technology
mike_t@mit.edu

Arthur Richards
Department of Aeronautics and Astronautics
Massachusetts Institute of Technology
arthurr@mit.edu

Jonathan P. How
Department of Aeronautics and Astronautics
Massachusetts Institute of Technology
jhow@mit.edu

Abstract This paper presents results on the guidance and control of fleets of cooperating Unmanned Aerial Vehicles (UAVs). A key challenge for these systems is to develop an overall control system architecture that can perform optimal coordination of the fleet, evaluate the overall fleet performance in real-time, and quickly reconfigure to account for changes in the environment or the fleet. The optimal fleet coordination problem includes team composition and goal assignment, resource allocation, and trajectory optimization. These are complicated optimization problems for scenarios with many vehicles, obstacles, and targets. Furthermore, these problems are strongly coupled, and optimal coordination plans cannot be achieved if this coupling is ignored. This paper presents an approach to the combined resource allocation and trajectory optimization aspects of the fleet coordination problem which calculates and communicates the key information that couples the two. Also, this approach permits some steps to be distributed between parallel processing platforms for faster solution. This algorithm estimates the cost of various trajectory options using the distributed platforms and then solves a centralized assignment problem to minimize the mission completion time. The detailed trajectory planning for this assignment can then be distributed back to the platforms. During execution, the coordination and control system reacts to changes in the fleet or the environment. The overall approach is demonstrated on several example scenarios to show multi-task allocation and cooperative path planning prior to the mission and to show dynamic re-planning to account for changes in the environment during execution.

Keywords: Distributed coordination, Unmanned aerial vehicles, Task allocation, Trajectory design, Mixed-integer linear programming.

1. Introduction

The capabilities and roles of Unmanned Aerial Vehicles (UAVs) are evolving, and require new concepts for their control. Today's UAVs typically require several operators for control, but future UAVs will be designed to make their own tactical decisions autonomously and will be integrated into teams that coordinate to achieve high-level goals, thereby allowing one operator to control a fleet of UAVs [1]. This level of autonomy will require new methods in planning and execution to coordinate the achievement of goals between the UAVs in the fleet.

The simplest form of a mission for a fleet of UAVs (*e.g.* a Suppression of Enemy Air Defenses (SEAD) mission) can be generalized as visiting a set of N_W waypoints, while avoiding the "No Fly Zones". Further constraints can be added to this problem, including waypoint types that only a subset of the fleet is capable of visiting; simultaneous, delayed or ordered arrival at waypoints; and collision avoidance between UAVs. Numerous changes can also occur during the mission execution, such as movement, addition, or removal of waypoints and No Fly Zones, and the addition or loss of members of the fleet. An example of a fleet coordination scenario including capability constraints is shown in Fig. 1.1.

Designing a coordinated mission plan that satisfies these constraints can be viewed as three coupled decisions: (i) Teams are formed and group goals are assigned to each team; (ii) Tasks that achieve the group goals are assigned to each team member; and (iii) A path is designed for each team member that achieves their tasks while adhering to spatial constraints, timing constraints, and the dynamic capabilities of the aircraft. The coordination plan is designed to minimize some cost, such as the completion time or the probability of mission failure. The overall control system then monitors the execution of the coordinated plan, and reacts to changes in the fleet, the environment, or the goals. This work assumes that lower-level controllers are present on each vehicle that are capable of following the planned path and performing the activities required at each waypoint.

Even if each of the three fleet coordination decisions is considered in isolation, it is clear that they are computationally demanding tasks. For even moderately sized problems, the number of combinations of possible teams, task allocations, and waypoint orderings that must be considered for the team formation and task assignment decisions is very large, growing at a non-polynomial rate that is at least the number of permutations of N_W elements taken N_W at a time. The problem of planning kinematically and dynamically constrained optimal paths, even for one aircraft, is also a very high dimension nonlinear optimization problem [2]. The optimal path planning problem requires a trajectory of control inputs to be designed that guide the UAV to its destination in minimum time, subject to control input limits, the differential equations representing the aircraft dynamics, and kinematic constraints presented by the No Fly Zones.

As difficult as each of these three decisions is to make in isolation, they are in fact strongly coupled because the optimality of the overall coordination plan is strongly limited by the team partitioning and task allocation. However, it is not clear how to make these decisions optimally until detailed trajectories have been planned, because the cost to be minimized by these decisions is a function of the resulting detailed trajectories. This coupling has been handled in one approach [3] by forming a large optimization problem that simultaneously assigns the tasks to vehicles and plans corresponding detailed trajectories. This method is computationally intensive, but it is guaranteed to find the globally-optimal solution to the problem and thus can be used as a benchmark against which the techniques presented in this paper can be compared [3].

Another approach to this problem is to decouple the decisions to some degree in order to make the problem computationally tractable, while maintaining the essential aspects of the coupling in order to approach optimality. This paper presents such a partially-decoupled approach to the task allocation and trajectory optimization problems which calculates and communicates the key information that couples the two problems, and distributes the computational effort of some steps between paral-

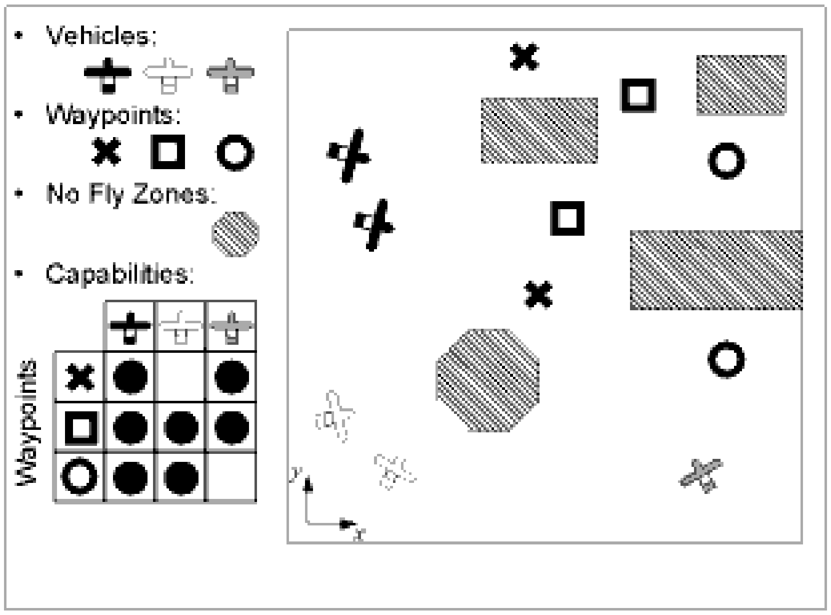

Figure 1.1. Schematic of a typical mission scenario for a UAV fleet with numerous waypoints, No Fly Zones and capabilities

lel processing platforms to improve the solution time. This partially-decoupled approach is shown to yield coordinated mission plans that are very close to the optimal solution [3].

Several previous studies have investigated methods of trajectory planning for coordination and control. Trajectory generation methods include the use of Voronoi diagrams [4], adaptive random search algorithms [5], model predictive control [6], and mixed-integer linear programming [3, 7]. However most of these methods are too computationally expensive for the multi-vehicle, multi-waypoint scenarios considered in this work. Thus, this paper investigates an approximate method that yields a fast estimate of the finishing times for the UAV trajectories which can then be used in the task allocation problem. It performs this estimation by using straight line path approximations. This method takes advantage of the fact that, for typical missions, the shortest paths for the UAVs tend to resemble straight lines that connect the UAVs' starting position, the vertices of obstacle polygons, and the waypoints. A more detailed trajectory generator is used to plan the UAV paths once the waypoints have been assigned.

The task allocation problem has also been considered in various applications. One method of determining the task allocation is through a network flow analogy [8]. This leads to a linear assignment problem that can be solved using integer linear programming. The task allocation problem has also been studied using a market based approach [9, 10]. In this approach, vehicles bid on each possible task presented. A central auction receives the bids and sends out the current price quote for each task. A decision is reached when no new bids arrive. The approaches discussed above handle some aspects of the task assignment, but they cannot easily include more detailed constraints such as selecting the sequence of the tasks and the timing of when tasks must be completed.

This paper presents a mathematical approach to solve the task allocation problem with the flexibility to include more detailed constraints through the use of mixed-integer linear programming (MILP). The resulting MILP problems can be readily solved using commercially available software such as CPLEX [11]. The combination of the approximate cost algorithm and task allocation formulation presented in this paper provides a flexible and efficient means of assigning multiple objectives to multiple vehicles under various detailed constraints.

2. Problem Formulation

The algorithms described here assume that the team partitioning has already been performed, and that a set of tasks has been identified which must be performed by the team. This paper presents algorithms that assign the tasks to team members and design a detailed trajectory for each member to achieve its tasks. The team is made up of N_V UAVs with known starting states and maximum velocities. The starting state of UAV p is given by the p^{th} row $[\ x_{0p} \quad y_{0p} \quad \dot{x}_{0p} \quad \dot{y}_{0p}\ ]$ of the matrix $\mathbf{S}_0$, and the maximum velocity of UAV p is given $v_{\max,p}$. The waypoint locations are assumed to be known, and the position of waypoint i is given by the i^{th} row $[\ W_{ix} \quad W_{iy}\ ]$ of the matrix $\mathbf{W}$. The application of the algorithms presented in this paper to No Fly Zones that are bounded by polygons is straightforward, but the case where the polygons are rectangles will be presented here for simplicity. The location of the lower-left corner is given by (Z_{j1}, Z_{j2}), and the upper-right corner by (Z_{j3}, Z_{j4}). Together, these two pairs make up the j^{th} row of the matrix $\mathbf{Z}$. Finally, the UAV capabilities are represented by a binary capability matrix K. The entry K_{pi} is 1 if vehicle p is capable of performing the tasks associate with waypoint i, and 0 if not.

This algorithm produces a trajectory for each vehicle, represented for the p^{th} vehicle by a series of states $\mathbf{s}_{tp} = [\ x_{tp} \quad y_{tp} \quad \dot{x}_{tp} \quad \dot{y}_{tp}\ ]$, $t \in [1, t_p]$, where t_p is the time at which aircraft p reaches its final waypoint. The finishing times of all vehicles make up the vector $\mathbf{t}$.

This work is concerned with coordination and control problems in which the cost is a function of the resulting trajectories. This is a broad category of coordination and control problems, and includes costs that involve completion time or radar exposure, and constraints on coordinated arrival or maximum range. While a cost function has been chosen that penalizes both the maximum completion time and the average completion times over all UAVs, the approach presented here can be generalized to costs that involve other properties of the trajectories. The cost used in this paper can be written as

$$\bar{t} = \max_{p} \; t_p \tag{1}$$

$$J_1(\bar{t}, \mathbf{t}) = \bar{t} + \frac{\alpha}{N_V} \sum_{p=1}^{N_V} t_p \tag{2}$$

where $\alpha \ll 1$ weights the average completion time compared to the maximum completion time. If the penalty on average completion time were omitted (*i.e.,* $\alpha = 0$), the solution could assign unnecessarily long trajectories to all UAVs except for the last to complete its mission. Note that, because this cost is a function of the completion time for the entire fleet, it cannot be evaluated exactly until detailed trajectories have been planned that visit all the waypoints and satisfy all the constraints. The minimum cost coordination problem could be solved by planning detailed trajectories for all possible assignments of waypoints to UAVs and all possible orderings of those waypoints, then choosing the detailed trajectories that minimize cost function $J_1(\bar{t}, \mathbf{t})$. However, the computational effort required to plan one detailed trajectory is large, and given all possible assignments and orderings, there exist a very large number of potential detailed trajectories that would have to be designed. For the relatively small coordination problem shown in Fig. 1.1, there are 1296 feasible allocations, and even more possible ordered arrival permutations.

3. Algorithm Overview

Clearly, planning detailed trajectories for all possible task allocations is not computationally feasible. Instead, the algorithm presented in this paper constructs estimates of the finishing times for a subset of the feasible allocations, then performs the allocation to minimize the cost function evaluated using the estimates. Next, detailed UAV trajectories are designed, and checked for collisions between vehicles. The main steps in the algorithm are shown in Fig. 1.2 and are described in the following.

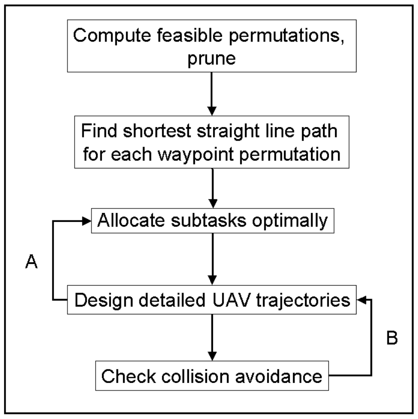
Compute feasible permutations, prune
Find shortest straight line path for each waypoint permutation
Allocate subtasks optimally
A
Design detailed UAV trajectories
B
Check collision avoidance

Figure 1.2. Steps in the task assignment and trajectory planning algorithm

First, a list of all unordered feasible task combinations is enumerated for every UAV, given its capabilities. Next, the length of the shortest path made up of straight line segments between the waypoints and around obstacles is calculated for all possible order-of-arrival permutations of each combination. The construction of these paths can be performed extremely rapidly using graph search techniques. The minimum finishing time for each combination is estimated by dividing the length of the shortest path by the UAV's maximum speed. Some of the tasks allocations and orderings have completion times that are so high that they can confidently be removed from the list.

With these estimated finishing times available, the task allocation problem can be performed to find the minimum of the estimated costs. MILP is well suited to solving this optimization problem, because it allows Boolean logic to be incorporated into constraints [12, 13]. These constraints can naturally express concepts such as "exactly one aircraft must visit every waypoint", and extend well to more complex concepts such as ordered arrival and obstacle avoidance [14]. Once the optimal task allocation is found using the estimated completion times, detailed kinematically and dynamically feasible trajectories that visit the assigned waypoints can be planned and checked for collision avoidance between UAVs [15]. If the minimum separation between UAVs is violated, the trajectories can be redesigned to enforce a larger separation distance (shown by loop B in Fig. 1.2). If desired, or if the completion time of the detailed trajectory plan is sufficiently different from the estimate, detailed trajectories can be planned for several of the task allocations with the lowest estimated completion times. The task allocation

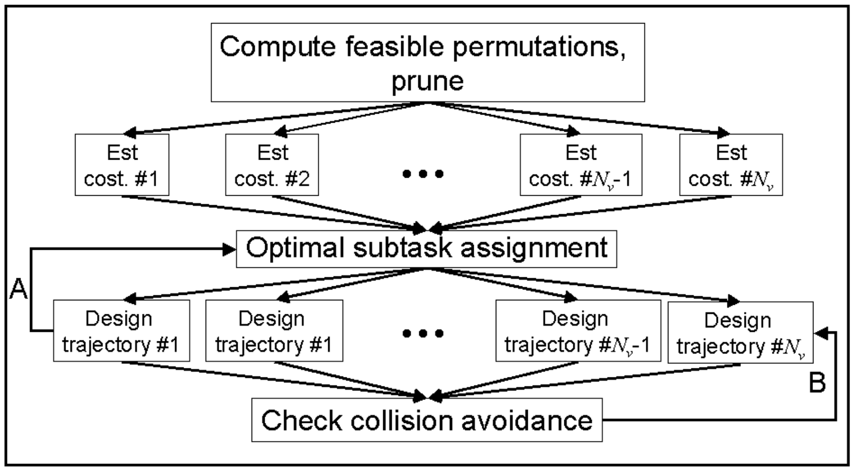

Figure 1.3. Steps in the distributed task assignment and trajectory planning algorithm

can then be performed using these actual completion times (shown by loop A in Fig. 1.2).

This strategy also casts the task allocation and detailed trajectory planning problems in a form that allows parts of the computation to be distributed to parallel platforms, as shown in Fig. 1.3. By making the processes of estimating the costs and designing detailed trajectories independent for each vehicle, they can be performed separately. The parallel platforms could be processors onboard the UAVs, or could be several computers at a centralized command and control facility. Having described how the steps in the algorithm are related, methods for performing them will be described next.

4. Finding Feasible Permutations and Associated Costs

This section presents a detailed analysis of the process for developing a list of feasible task assignments, finding approximate finishing times for each task assignment, and pruning the list. This step accepts the aircraft starting states $\mathbf{S}_0$, capabilities $\mathbf{K}$, obstacle vertex position $\mathbf{Z}$, and waypoint positions $\mathbf{W}$. The algorithm also accepts two upper boundaries: $n_{\max}$ specifies the maximum number of waypoints that a UAV can visit on its mission, and $t_{\max}$ specifies the maximum time that any UAV can fly on its mission. From these inputs this algorithm finds, for each UAV and each combination of $n_{\max}$ or fewer waypoints, the order in which to visit the waypoints that gives the shortest finishing time.

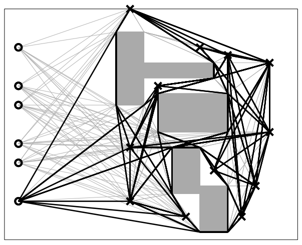
Figure 1.4. Visibility graph and shortest paths between UAV 6 and all waypoints.

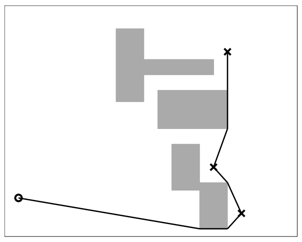
Figure 1.5. Shortest path for UAV 6 over one combination of waypoints.

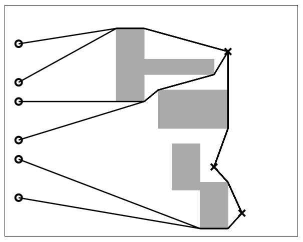
Figure 1.6. Shortest paths for all UAVs over same combination of waypoints.

The steps in this algorithm are depicted in Figs. 1.4–1.6, in which a fleet of UAVs (shown with $\circ$) must visit a set of waypoints (shown with $\times$). The first step is to find the visibility graph between the UAV starting positions, waypoints, and obstacle vertices. The visibility graph is shown in Fig. 1.4 with grey lines. Next, UAV 6 is considered, and the visibility graph is searched to find the shortest paths between its starting point and all waypoints, as shown in Fig. 1.4 with black lines. In Fig. 1.5, a combination of three waypoints has been chosen, and the fastest path from UAV 6's starting position through them is shown. The order-of-arrival for this path is found by forming all possible order-of-arrival permutations of the unordered combination of waypoints, then summing the distance over the path associated with each order-of-arrival from UAV 6's starting point. The UAV is assumed to fly this distance at maximum speed, and the order-of-arrival with the shorted associated finishing time is chosen. In Fig. 1.6, the fastest path to visit the same combination of waypoints is shown for each vehicle. Note that the best order-of-arrival at these waypoints is not the same for all vehicles.

The algorithm produces four matrices whose j^{th} columns, taken together, fully describe one permutation of waypoints. These are the row vector $\mathbf{u}$, whose u_j entry identifies which UAV is involved in the j^{th} permutation; $\mathbf{P}$, whose P_{ij} entry identifies the i^{th} waypoint visited by permutation j; $\mathbf{V}$, whose V_{ij} entry is 1 if waypoint i is visited by permutation j, and 0 if not; $\mathbf{T}$, whose T_{ij} entry is the time at which waypoint i is visited by permutation j, and 0 if waypoint i is not visited; and $\mathbf{c}$, whose c_j entry is the completion time for the j^{th} permutation. All of the permutations produced by this algorithm are guaranteed to be feasible given the associated UAV's capabilities.

All steps in this approach are shown in Algorithm 1. In this algorithm, finding the shortest distance between a set of waypoints and

```
1: Find shortest distances between all waypoint pairs $(i, j)$ as $D(i, j)$ using $\mathbf{Z}$ and $\mathbf{W}$.
2: **for all** UAVs $p$ **do**
3:     Find shortest distances $d(i)$ between start point of UAV $p$, and all waypoints $i$ using $\mathbf{S}_0$, $\mathbf{Z}$, and $\mathbf{W}$.
4:     **for all** combinations of $n_C$ waypoints that $p$ is capable of visiting, $n_C = 1 \ldots n_{\max}$ **do**
5:         **for** $j = 1 \ldots n_C P n_C$ **do**
6:             Make next unique permutation $P'_{1j} \ldots P'_{n_C j}$ of waypoints in the combination
7:             $c'_j = \frac{d(P'_{1j})}{v_{\max,p}}$
8:             $T'_{P'_{1j} j} = c'_j$
9:             **for** $i = 2 \ldots n_C$ **do**
10:                **if** $c'_j > t_{\max}$ **then**
11:                    go to next j
12:                **end if**
13:                $c'_j \leftarrow c'_j + \frac{D(P'_{(i-1)j}, P'_{ij})}{v_{\max,p}}$
14:                $T'_{P'_{ij} j} = c'_j$
15:            **end for**
16:        **end for**
17:        Append $p$ to $\mathbf{u}$
18:        Append a column to $\mathbf{V}$, whose $i^{\text{th}}$ element is 1 if waypoint $i$ is visited, 0 if not.
19:        $j_{\min} = \text{minarg}_j \; c'_j$
20:        Append $c_{j_{\min}}$ to $\mathbf{c}$
21:        Append column $j_{\min}$ of $\mathbf{T}'$ to $\mathbf{T}$ and column $j_{\min}$ of $\mathbf{P}'$ to $\mathbf{P}$
22:    **end for**
23: **end for**
```

Algorithm 1: Algorithm for finding shortest paths between waypoints

starting points is performed by finding the visibility graph between the points and vertices of obstacles, then applying an appropriate shortest path algorithm, such as the Floyd-Warshall All-Pairs Shortest Path algorithm [16]. Note that the iterations through the "**for** loop" between lines 2 and 23 of Algorithm 1 are independent, and can be distributed to parallel processors. The corresponding matrices from each processor can then be combined and passed onto the next stage in the algorithm, the task allocation problem.

5. Task Allocation

The previous section outlined a method of rapidly estimating completion times for individual vehicles for the possible waypoint allocations. This section presents a mathematical method of selecting which of these assignments to use for each vehicle in the fleet, subject to fleet-wide task completion and arrival timing constraints.

The basic task allocation problem is formulated as a *Multi-dimensional Multiple-Choice Knapsack Problem* (MMKP) [17]. In this classical problem, one element must be chosen from each of multiple sets. Each chosen element uses an amount of each resource dimension, but yields a benefit. The choice from each set is made to maximize the benefit subject to multi-dimensional resource constraints.

In the UAV task allocation problem, the choice of one element from each set corresponds to the choice of one permutation for each vehicle. Each resource dimension corresponds to a waypoint, and a permutation uses 1 unit of resource dimension i if it visits the i^{th} waypoint. The arrival constraints are then transformed into constraints on each resource dimension. The negative of the completion times in this problem is equivalent to the benefit. Thus the overall objective is to assign one permutation (element) to each vehicle (set) that is combined into the mission plan (knapsack), such that the cost of the mission (knapsack) is minimized and the waypoints visited (resources used) meet the constraint for each dimension. The problem can be written as

$$\begin{aligned} \min\ J_2 &= \sum_{j=1}^{N_M} c_j x_j \\ \text{subject to} \quad & \sum_{j=1}^{N_M} V_{ij} x_j \geq w_i \\ & \sum_{j=N_p}^{N_{p+1}-1} x_j = 1 \end{aligned} \tag{3}$$

where the permutations of vehicle p are numbered N_p to $N_{p+1}-1$, with $N_1 = 1$ and $N_{N_V+1} = N_M + 1$ and the indices have the ranges $i \in \{1, \dots, N_W\}$, $j \in \{1, \dots, N_M\}$, $p \in \{1, \dots, N_V\}$. c_j is a vector of costs (mission times) for each permutation. x_j is a binary decision variable equal to one if permutation j is selected, and 0 otherwise. The cost in this problem formulation minimizes the sum of the times to perform each selected permutation. The first constraint enforces that waypoint i is visited at least w_i times (typically $w_i = 1$). The second constraint prevents more than one permutation being assigned to each vehicle. The MMKP formulation is the basic task allocation algorithm. However, modifications are made to the basic problem statement to include additional cost considerations and constraints.

Modified Cost: Total Mission Time. The first modification for the UAV allocation problem is to change the cost. The cost in Eq. 2 is a weighted combination of the sum of the individual mission times (as in the MMKP problem) and the total mission time. The new cost is as follows,

$$J_3 = \bar{t} + \frac{\alpha}{N_V} \sum_{i=1}^{N_M} c_i x_i \tag{4}$$

The solution to the task allocation problem is a set of ordered sequences of waypoints for each vehicle which ensure that each waypoint is visited the correct number of times while minimizing the desired cost (mission completion time).

Timing Constraints. Solving the task allocation as a centralized problem allows the inclusion of timing constraints on when a waypoint is visited. For example, a typical constraint might be for one UAV to eliminate a radar site at waypoint A before another vehicle can proceed to a waypoint B. The constraint would then be that waypoint A must be visited t_D time units before waypoint B, which can be included in the task allocation problem. The timing constraint is met by either altering the order in which waypoints are visited, delaying when a vehicle begins a mission, or assigning a loitering time to each waypoint. The constraint formulation in which vehicle p starts at VT_p and then executes its mission without delay is presented below. To construct the constraint, an intermediate variable, WT_i, is used to determine the departure time of the vehicle that visits waypoint i. The constraint can be written as

$$\sum_{j=1}^{N_M} (T_{Bj} x_j - T_{Aj} x_j) + WT_B - WT_A \geq t_D \tag{5}$$

$$WT_A \leq VT_p + R(1 - \sum_{j=N_p}^{N_{p+1}-1} P_{Aj} x_j) \;\; \forall\, p \in \{1, \ldots, N_V\} \tag{6}$$

$$WT_A \geq VT_p - R(1 - \sum_{j=N_p}^{N_{p+1}-1} P_{Aj} x_j) \;\; \forall\, p \in \{1, \ldots, N_V\} \tag{7}$$

$$WT_B \leq VT_p + R(1 - \sum_{j=N_p}^{N_{p+1}-1} P_{Bj} x_j) \;\; \forall\, p \in \{1, \ldots, N_V\} \tag{8}$$

$$WT_B \geq VT_p - R(1 - \sum_{j=N_p}^{N_{p+1}-1} P_{Bj}x_j) \;\; \forall\, p \in \{1, \ldots, N_V\} \tag{9}$$

Constraint Eq. 5 enforces waypoint A to be visited t_D time units before B. Constraint Eqs. 6 − 9 are used to determine the start time for the vehicles that are assigned the waypoints in the timing constraint. R is a large number that relaxes the constraint if vehicle p does not visit the waypoint in question. If vehicle p does visit waypoint A, R multiplies 0, and Equations 6 and 7 combine to enforce an equality relationship $WT_A = VT_p$. Note that using this formulation does not allow the same vehicle to visit both waypoints unless one of the original permutation met the timing constraint. This is because the start times WT_A and WT_B would be the same and cancel in Eq. 5. Again these constraints are added to the original problem in Eq. 3 to form a task allocation problem including timing constraints. The cost must also be altered to include the UAV start times as follows

$$J_4 = \bar{t} + \frac{\alpha}{N_V}\sum_{i=1}^{N_M} c_i x_i + \frac{\alpha}{N_V}\sum_{p=1}^{N_V} VT_p \tag{10}$$

The constraints presented here for delaying individual start times can be generalized to form other solutions to the timing constraint, such as allowing a UAV to loiter at a waypoint before going to the next objective.

6. Reaction to Dynamic Environment

The task allocation problem is used to assign a sub-team of vehicles to visit a set of waypoints based on the information (vehicle states, waypoint locations, and obstacles) known at the beginning of the mission. However, throughout the execution of the mission the environment can (and most likely will) change. As a result the optimal task allocation could be dramatically altered. Note that if the problem size is sufficiently small, it would be practical to perform a complete re-allocation using a new set of costs based on the updated environment. However for larger problems it would be beneficial to re-solve smaller parts of the allocation problem. Two smaller problems are presented in this section. One is a *local repair* where only one vehicle assignment is altered to meet the new environment. Another is a *sub-team allocation problem* where only those "directly influenced" by the change in environment are re-assigned.

Addition of Waypoint. As the mission is executed, it is possible that further reconnaissance will identify a new goal or waypoint. The set of waypoints could be re-allocated amongst the entire fleet, but several alternatives exist that can result in much smaller optimization problems.

The *local repair* method estimates the cost of adding the new waypoint to each vehicle's list of objectives using Alg. 1. The cost for each UAV is determined using the UAV's current state, the remaining waypoints assigned to the UAV from the original problem, and the new waypoint. The assignment of this waypoint that results in the smallest increase in the cost function is then chosen. The local repair can be solved very quickly, but it is a sub-optimal solution because it does not allow the vehicles to trade previously assigned waypoints.

A *sub-team* problem can be formulated which only considers those vehicles capable of visiting the new waypoint. These vehicles, their previously assigned waypoints, and the new waypoint are then considered as a smaller task allocation problem. This allows any waypoints within this group to be traded amongst the vehicles (subject to each vehicle's capabilities), but it is still sub-optimal because it does not consider the possibility that some waypoints of a type different than the new one could be traded to UAVs that are not in the sub-team considered. The *full re-allocation* problem would, of course, consider this coupling in the problem, but this would typically take longer to compute.

Addition/Removal of Obstacle. The addition or removal of an obstacle is considered by estimating the new cost for each vehicle given their current assigned waypoints with (and without) the obstacle in question. If the vehicle's cost estimate changes, then that vehicle is considered to be influenced by the obstacle.

If the vehicle is influenced by the obstacle, the *local repair* method does not change its assignment of waypoints, but redesigns its detailed trajectory to account for the change. The *sub-team* problem considers all vehicles that are influenced by the obstacle. The vehicles and their previously assigned waypoints are grouped into a new allocation problem and the re-assignment is performed for this subset of the fleet. The *full re-allocation* problem could also be solved for this situation.

The methods presented in this paper only consider unexpected changes in the environment, such as a waypoint suddenly appearing, or an obstacle appearing/disappearing. Another dynamic problem to consider is when knowledge of the environment includes a change that will occur at some future time or due to the actions of the vehicles, such as removal of an obstacle by a vehicle. Formulations of the cost estimation and task allocation steps that take advantage of expected changes are areas of current research.

7. Simulations

The problem formulation presented in this paper leads to cooperative path planning in the sense that the task assignments to each UAV are made in a centralized way. In particular, the tasks are divided amongst the vehicles in the fleet to minimize the overall cost. The problem formulation also allows multi-task assignments, possibly assigning a sequence of several tasks to a single UAV. The complete approach is illustrated in this section using several examples.

A small problem is first considered to show how the assignment changes when constraints are added. The basic scenario includes three UAVs and four waypoints to be visited. There are also two obstacles in the environment. The objective is to visit each waypoint once and only once in minimum time. Each vehicle also must take at least one waypoint. The costs for each vehicle to visit a sequence of waypoints is determined using the approximate cost algorithm presented earlier. The task allocation is then determined through the modified MMKP problem. The basic problem is formed using Eqs. 1–3.

The first solution, shown in Fig. 1.7, is the basic allocation problem. The UAVs are homogeneous so any vehicle can visit any waypoint. The solution for this problem is relatively straightforward, with each vehicle visiting waypoints that are close. The mission time in this case is 19.75. The second scenario is the same basic problem except now the vehicles are heterogenous. Each vehicle is capable of performing different tasks. UAV 1 can visit waypoints $1, 2, 4$, UAV 2 can visit any waypoints, and UAV 3 can visit waypoints $2, 3$. The vehicle capabilities were enforced by only considering waypoint permutations that were feasible for each vehicle. The result for this case is the slightly less obvious solution shown in Fig. 1.8. Because UAV 3 can no longer visit waypoint 4, the solution now assigns UAV 1 to waypoint 1 before going to waypoint 4. Waypoint 1 is assigned to UAV 1 because it can be achieved with little deviation in the route to get to waypoint 4. UAV 2 is assigned only waypoint 3 because it is the furthest objective. Note that two of the vehicle paths cross and a post check for collisions would be required. The mission time for this scenario increased to 19.90 time units. This problem was also solved using the approach described in [3], which is guaranteed to find the globally optimal coordinated mission plan. The allocation found in Fig. 1.8 was verified to be the global optimum, and was found much more rapidly by the approach presented here. The third scenario is the same as the second problem with the addition of the timing constraint that waypoint 4 must be visited 5 time units before waypoint 1. The timing constraint is included using Eq. 5. The solution to this problem

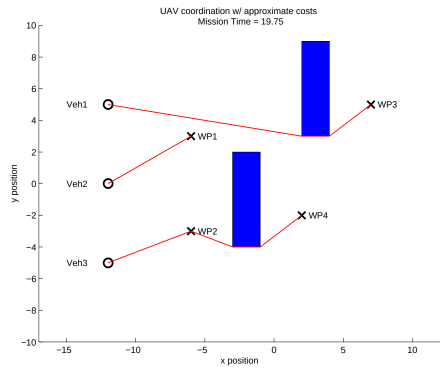

Figure 1.7. Scenario has 3 homogeneous vehicles, which is the basic task allocation problem.

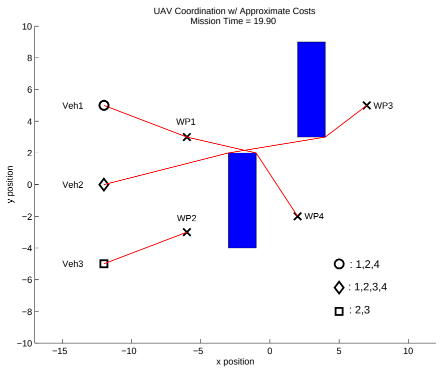

Figure 1.8. Same problem as Fig. 1.7, plus heterogenous capabilities shown in the legend.

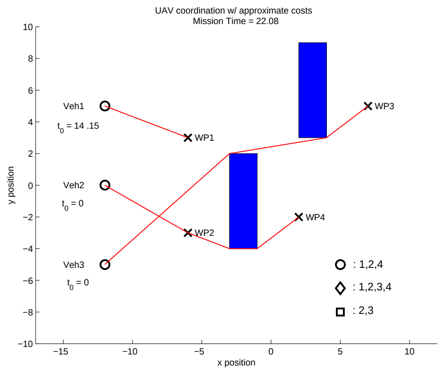

Figure 1.9. Same problem as Fig. 1.8, plus waypoint 4 must be visited 5 time units before waypoint 1.

(see Fig. 1.9) results in a very different set of assignments and UAV 1 delays the start time by 14.15 units before proceeding to waypoint 1. However, this results in only a small increase in mission time to 22.08 units.

The final simulation performed uses a much larger scenario that includes a fleet of 6 UAVs of 3 different types and 12 waypoints of 3 different types. The UAV capabilities are shown in Fig. 1.10. There are also several obstacles in the environment. Again the objective is to allocate waypoints to the team of UAVs in order to visit every waypoint once and only once in the minimum amount of time. There are no timing constraints in this scenario. The solution is shown in Fig. 1.10. All waypoints are visited subject to the vehicle capabilities in 23.91 time units.

In order to understand the difficulty of this problem, a "greedy" heuristic was applied to it. This heuristic performs allocation decisions one waypoint at a time. It calculates the increase in the cost function in Eq. 2 associated with allocating each waypoint to each capable vehicle. The vehicle-waypoint allocation with the smallest associated increase in the cost is selected. The allocated waypoint is removed from consideration, and the receiving vehicle is considered to be at the waypoint's location for the next allocation. This procedure is repeated until all waypoints are allocated. The greedy heuristic solved the large scenario shown in Fig. 1.10 with a maximum completion time of 28.20, an increase of 17.9%. In this coordination plan, the last waypoint to be allocated caused a large increase in the completion time. This clearly shows that locally justified decisions do not provide globally optimal fleet coordination plans, and that the MILP-based method presented here provides significantly better results than heuristics with local scope.

8. Conclusions

This paper presents an approach to the task allocation and detailed trajectory design components of the optimal fleet coordination problem. The approach presented here partially decouples these problems. It efficiently estimates finishing times associated with the different allocation options, and provides this information to the allocation optimization. The allocation is an extension of the MMKP problem, and is solved as a MILP problem. A small number of detailed trajectories are then designed to perform the allocated tasks. The results from several examples clearly illustrate the impact of including various constraints in the fleet assignment problem. They also clearly illustrate that the partially-decoupled method presented here was capable of solving a large prob-

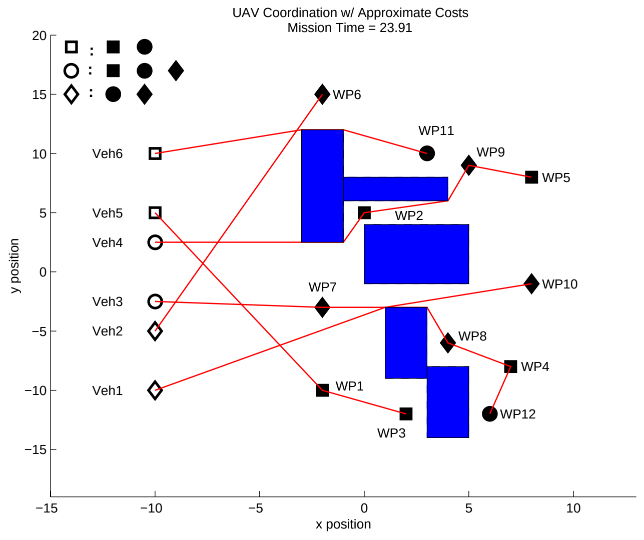

Figure 1.10. Scenario has three pairs of heterogenous vehicles with 12 waypoints (3 different types). Figure legend shows which vehicles can visit each waypoint. The scenario demonstrates task allocation for a large problem with heterogenous vehicles using the approximate cost method.

lem involving many obstacles and waypoints. Future work will compare these approximate results with the globally-optimal solution available from the full MILP solution.

Acknowledgments

Research funded in part under Air Force grant # F49620-01-1-0453.

References

[1] S. A. Heise, "DARPA Industry Day Briefing," available on-line at www.darpa.mil/ito/research/mica/MICA01mayagenda.html

[2] J. H. Reif, "Complexity of the Mover's Problem and Generalizations," in the proceedings of the *20th IEEE Symposium on the Foundations of Computer Science,* IEEE, Washington DC, pp. 421-427, 1979.

[3] A. Richards, J. Bellingham, M. Tillerson, and J. How "Co-ordination and Control of Multiple UAVs", submitted for publication at the 2002 *AIAA Guidance, Navigation, and Control Conference.*

[4] T. McLain, P. Chandler, S. Rasmussen, M. Pachter,"Cooperative Control of UAV Rendezvous," IEEE *American Control Conference,*

Arlington, VA, June 25-27, 2001, pp. 2309–2314.

[5] R. Kumar, D. Hyland, "Control Law Design Using Repeated Trials," IEEE *American Control Conference,* Arlington, VA, June 25-27, 2001, pp. 837–842.

[6] L. Singh and J. Fuller, "Trajectory Generation for a UAV in Urban Terrain, Using Nonlinear MPC," IEEE *American Control Conference,* Arlington, VA, June 25-27, 2001, pp. 2301-2308.

[7] J. Bellingham, A. Richards, J. How, "Receding Horizon Control of Autonomous Aerial Vehicles", to appear in the IEEE *American Control Conference*, May 2002.

[8] C. Schumacher, P. Chandler, and S. Rasmussen, "Task Allocation for Wide Area Search Munitions via Network Flow Optimization" AIAA *Guidance, Navigation, and Control Conference*, Montreal, Canada, Aug. 6-9, 2001.

[9] P. Chandler, M. Pachter, "Hierarchical Control for Autonomous Teams" AIAA *Guidance, Navigation, and Control Conference,* Montreal, Canada, Aug. 6-9, 2001.

[10] J. Tierno, "Distributed Autonomous Control of Concurrent Combat Tasks," IEEE *American Control Conference,* Arlington, VA, June 25-27, 2001, pp. 37-42.

[11] *ILOG CPLEX User's guide*, ILOG, 1999.

[12] C. A. Floudas, *Nonlinear and Mixed-Integer Programming – Fundamentals and Applications,* Oxford University Press, 1995.

[13] H. P. Williams and S. C. Brailsford, "Computational Logic and Integer Programming," in *Advances in Linear and Integer Programming,* Editor J. E. Beasley, Clarendon Press, Oxford, 1996, pp. 249–281.

[14] A. Richards, and J. P. How"Aircraft Trajectory Planning With Collision Avoidance Using Mixed Integer Linear Programming", to appear at the IEEE *American Controls Conference*, 2002.

[15] T. Schouwenaars, B. D. Moor, E. Feron, and J. P. How, "Mixed Integer Programming for Multi-Vehicle Path Planning" presented at the *European Controls Conference*, 2001.

[16] T. H. Cormen, C. E. Leiserson, R. L. Rivest. *Introduction to Algorithms,* MIT Press, 1990.

[17] M. Moser, D. Jokanovic, N. Shiratori, " An Algorithm for the Multidimensional Multiple-Choice Knapsack Problem" *IEICE Trans. Fundamentals,* Vol. E80-A, No.3 March 1997.