Modeling and Optimizing Military Air Operations

Mariam Faied and Anouck Girard

Abstract—Dynamic programming has recently received significant attention as a possible technology for formulating control commands for decision makers in an extended complex enterprise that involves adversarial behavior. Enterprises of this type are typically modeled by a nonlinear discrete time dynamic system. The state is controlled by two decision makers, each with a different objective function and different hierarchy of decision making structure. To illustrate this enterprise, we derive a state space dynamic model of an extended complex military operation that involves two opposing forces engaged in a battle. The model assumes a number of fixed targets that one force is attacking and the other is defending. Due to the number of control commands, options for each force, and the steps during which the two forces could be engaged, the optimal solution for such a complicated dynamic game over all stages is computationally extremely difficult, if not impossible, to propose. As an alternative, we propose an expeditious suboptimal solution for this type of adversarial engagement. We discuss a solution approach where the decisions are decomposed hierarchically and the task allocation is separate from cooperation decisions. This decoupled solution, although suboptimal in the global sense, is useful in taking into account how fast the decisions should be in the presence of adversaries. An example scenario illustrating this military model and our solution approach is presented.

I. INTRODUCTION

Since 1916 when Lanchester developed the basic foundation of the erosion models for modern warfare [7], they have received significant attention [2], [12]. This mathematical analysis for calculating attrition of forces in an air mission is very useful in providing a mathematical representation of event probabilities, and operations. In the beginning and due to the fact that these models were not expressed using state space algorithms [1], [11], [6], they were not amenable for the application of optimal control and dynamic game theory. The emerging field of modeling and controlling such an extended complex enterprise [3] seems to provide an appropriate mechanism for analyzing and formulating command and control actions for modern military operations. An enterprise of this type is typically modeled as a highly complex nonlinear dynamical system that is controlled by two or more adversarial decision agents. The states are influenced by the environment and the objective functions. Recently, state space models of an extended complex military operation that involves two opposing forces were developed [5], [4]. Motivated mainly by military Unmanned Air Vehicle (UAV) operations and inspired by these state models, we propose this paper. This work is different from the previous models in the command constraints that control the game, the underlying process description, and also in the solution methodology. Along with our previous work in [9], [8], [10], this paper proposes one of the first UAV coordination algorithms that enable a number of UAVs to achieve autonomous battle management in an adversarial environment. In our previous work [9] adversarial UAV operations and a computationally effective algorithm were presented for a SEAD (Suppression of Enemy Air Defenses) type military mission. In [8], the algorithm was represented in the framework of Dynamic Network of Hybrid Automata (DNHA). This algorithm was further improved to incorporate MTL type constraints in [10].

In this paper, our problem includes both combinatorial and stochastic complexity, and represents the military uncertain and adversarial environment. More precisely, we consider a SEAD mission, in which the Blue force and the Red force engage in multiple stages. The Red force is composed of multiple targets which have to be engaged individually. It is assumed that these targets cannot be engaged in any order, but the set of feasible orderings of the targets will be calculated. In our model, we assume that the Red force can counterattack and destroy the Blue force with some known probability. The Blue force controller, however, can attack a target with multiple UAVs to increase the probability of destruction of the target which of course will introduce the extra cost and risk of employing more UAVs in the mission. Thus in the beginning of each stage a decision about which target to attack and which set of UAVs to attack with should be made. Then, the engagement between the two forces in this stage will take place according to the command constraints. In the next stage, and according to which target has been destroyed in the previous stage, the state of the game will be determined. This model is dynamic in nature with state variables whose evolution with time depends on the choice of control actions by each of the two forces. It is extended in the sense that the effects of each opposing force action, and of the environment are explicitly included in the model. In addition to the battle dynamics, we model the objective functions of each force and the control commands which will be calculated at each stage to optimize the state of the game. Consider a discrete-time system controlled by two independent decision-makers whose state vector evolves according to the equation:

\[ x_{k+1} = f_k(x_k, u^1_k, u^2_k), \quad k = 0, \ldots, N - 1 \]  

where \( x_k \) is the state vector, \( u^1_k \) and \( u^2_k \) are the independent control sequences of the two opposing decision makers and...
$x_0$ is the initial condition at $k = 0$. Each team decision maker is trying to optimize a performance index of the form:

$$J^i = \phi^i_N(N, x_N) + \sum_{k=0}^{N-1} L^i(x_k, u^i_k, u^2_k), \quad i = 0, 1, \ldots, N - 1$$

(2)

Where $\phi^i_N(N, x_N)$ is the terminal cost incurred at the end of the process, and $L^i(x_k, u^i_k, u^2_k)$ is the cost incurred at time $k$. Over the interval $[0, N]$, obtaining a game-theoretic optimal solution for such a system is not only complex but also the complexity will rise exponentially with the length of the time horizon $N$. In order to overcome these difficulties, we represent our decoupled solution which will use an off-line general policy for the battle. This policy yields threshold solutions which can be used on-line with minimal computational effort.

This manuscript is organized as follows. In section II we introduce the state space model of this extended complex military operation. Then in Section III, we introduce the computational complexity for solving this model. Section IV describes our solution approach. Later in section V we propose an example scenario illustrating our approach. The paper ends with conclusions.

II. AN EROSION MODEL FOR A MILITARY OPERATION

In our model there are two forces, the attacking force (Blue force) and the defending force (Red force). The Blue force consists of “Bombers” as the Master UAV (BM) and “Jammers” as the slave UAV (BS). The “Bombers” are essentially units whose purpose is to attack and suppress the Red air defenses and the Red units. The “Jammers” can support the “Bombers” and combine with each other as a team according to the risk assessment during the battle. On the other hand, the Red force consists of Red Defense Units (RDU) such as surface-to-air missiles (SAMs) in addition to Red Fixed Targets (RFT) such as command center, bridges.

A. Unit State Space Representation

In this game, the Blue force is planning to attack and the Red force is planning to defend. Each unit is fully described by its state vector. That is:

$$z^m_i(k) = \begin{bmatrix} x^m_i(k) \\ y^m_i(k) \\ w^m_i(k) \end{bmatrix}, \quad m = \{BM, BS, RDU, RFT\}$$

(3)

where $i = 1, 2, \ldots, N^m$

Here, $x$ and $y$ correspond to the x-coordinate and the y-coordinate at time step $k$, respectively; $w$ corresponds to total number of weapons left at time step $k$ in that unit, and $N^m$ to the total number of units of $m$ type. Thus for each moving unit in the theater of operations, we define a three dimensional state variable. Combining all the state variables for each type of forces into one vector, we have:

$$z^m(k) = \begin{bmatrix} z^1_m(k) \\ \vdots \\ z^{N^m}_m(k) \end{bmatrix}.$$  

(4)

In a similar way, we can write the overall state vector corresponding to the Blue and Red forces as follows:

$$z^B(k) = \begin{bmatrix} z^{BM}(k) \\ z^{BS}(k) \end{bmatrix} \quad \text{and} \quad z^R(k) = \begin{bmatrix} z^{RDU}(k) \\ z^{RFT}(k) \end{bmatrix}$$

(5)

The RFT units don’t carry any weapons, so size of the combined state vector for the Blue force as well as the Red force will be a $3x(N^{BM} + N^{BS} + N^{RDU}) + 2x(N^{RFT})$.

B. Control Variables

The control variables for each unit are chosen from a finite set of choices. They are divided into three types:

1) Motion Control $d^{m}_i(k)$: We discretized the geography of the battlespace as a rectangular grid to provide a computationally efficient abstraction for motion control. Each unit can move to another adjacent point on the grid; that is, the maximum distance traveled at each step is given by:

$$d^{m}_i(k) = \begin{bmatrix} \Delta x^m_i(k) \\ \Delta y^m_i(k) \end{bmatrix}$$

(6)

where $\Delta x^m_i(k)$ corresponds to a step in the $x$-direction and $\Delta y^m_i(k)$ corresponds to a realignment in the $y$-direction. Because we restrict the units to be moving along the “streets” of the grid, there are four neighborhood locations that each unit can relocate to, and the option of no movement from the current location at this time step $k$.

2) Fire Controls $r^{m}_i(k)$, $e^{m}_i(k)$: Each unit that has $w^m_i(k) > 0$ has the option to fire or not to fire. When a unit decides to fire at time step $k$, that is $r^{m}_i(k) = 1$. Also at each time step, a unit can fire only at one target of the opposing forces. Let $e^{m}_i(k)$ represents the choice of target for unit $i$ of type $m$ at time step $k$.

3) Team Configuration Control $f^{m}_i(k)$: We assume that for each force, we have two configurations: Cooperative teams or Independent units. Every unit in each force has the option to decide between these two configurations.

$$f^{m}_i(k) = \begin{bmatrix} 1 \quad \text{if Cooperative team} \\ 0 \quad \text{if Independent unit} \end{bmatrix}$$

(7)

Combining all these command variables into one control vector, we have the following control vector for each unit:

$$u^m_i(k) = \begin{bmatrix} \Delta x^m_i(k) \\ \Delta y^m_i(k) \\ r^m_i(k) \\ e^m_i(k) \\ f^m_i(k) \end{bmatrix}$$

(8)

For each unit, we have 5-dimensional control vector. The composite control vector for all units in the Blue and Red forces is $5x(N^{BM} + N^{BS} + N^{RDU})$. Note that the Red Fixed Target doesn’t have this control command because they are neither moving nor shooting units.
C. Command Constraints

These control variables are subject to different types of constraints: Logic constraints, and Kinematic constraints. The Logic constraints are mainly trying to manage the firing maneuver, and the Kinematic constraints do the same thing with the moving maneuver. The Logic constraints can be summarized into:

- Units may not move and fire simultaneously.
  \[ \delta^m_i(k) + r^m_i(k) \leq 1. \]  
  (9)

In the above relation, \( \delta^m_i(k) \) is defined as follows:

\[ \delta^m_i(k) = \begin{cases} 
0, & \text{if } \Delta x^m_i(k) = 0 \text{ and } \Delta y^m_i(k) = 0 \\
1, & \text{otherwise} 
\end{cases} \]

(10)

That is, \( \delta^m_i(k) \) is equal to 1 if the unit decides to move to an adjacent location and is equal to 0 otherwise.

- A unit will not be able to fire at a unit of the other force unless they are both in range (this constraint is related to the range of weapons and requires that both units be \( m \) squares apart on the grid for the weapon from one side to reach the target on the other side).
  \[ |x^X_i - x^Y_j| + |y^X_i - y^Y_j| \leq l_i. \]  
  (11)

Eq.(11) represents that a unit \( i \) of type \( X \) cannot fire at unit \( j \) of type \( Y \) unless the distance between them is less than a predefined distance \( l_i \).

- Units on either force have a limited number of weapons and any unit can not fire at step \( k \) unless \( w^m_i(k) > 0 \).
  \[ r^m_i(k) = \begin{cases} 
1, & \text{if } w^m_i(k) > 0 \\
0, & \text{otherwise} 
\end{cases} \]

(12)

- Enemy units can be targeted only by the BM or a combination of the BM and BS.

- No two units of the same force can fire at the same target of the opposing force at the same time step \( k \):
  \[ \sum_{i=1}^{N_{BM}} r^{BM}_i(k) + \sum_{i=1}^{N_{BS}} r^{BS}_i(k) \leq 1. \]  
  (13)

for each Red target,

\[ \sum_{i=1}^{N_{RDU}} r^{RDU}_i(k) \leq 1. \]  

(14)

for each Blue target.

As we can see, some of the Logic constraints are naturally imposed and some we specified for the game. We use steerable unicycle Kinematics on the Manhattan grid for our example application. This level of abstraction is sufficient to evaluate the architecture and provides fast simulation results. Any unit will be constrained to move along the streets at a constant speed. We can describe the motion of these types of vehicles by the following discrete equations:

\[
\begin{bmatrix}
\Delta x^m_i(k) \\
\Delta y^m_i(k)
\end{bmatrix} = \begin{bmatrix}
v \cos \psi^m_i(k) \\
v \sin \psi^m_i(k)
\end{bmatrix}
\]

(15)

Here, \( \Delta x^m_i(k) \) and \( \Delta y^m_i(k) \) correspond to the x-displacement and the y-displacement on the grid, respectively; \( v \) is the linear forward velocity and \( \psi^m_i(k) \) is the orientation of the unit.

D. Model State Space Representation

The state equations for this military model are defined as

\[
\begin{bmatrix}
x^m_i(k+1) \\
y^m_i(k+1)
\end{bmatrix} = \begin{bmatrix}
x^m_i(k) \\
y^m_i(k)
\end{bmatrix} + \begin{bmatrix}
\Delta x^m_i(k)\beta^m_w(k) \\
\Delta y^m_i(k)\beta^m_w(k)
\end{bmatrix}
\]

(16)

\[
w^{m}_i(k+1) = w^{m}_i(k) - r^{m}_i(k) \]

(17)

\[
N^m_i(k+1) = N^m_i(k) - L^m_i(k)
\]

(18)

\[
L^m_i(k) = T^m_i(k)A^m_i(k).
\]

(19)

Eq.(17), (18) and (19) for weapons and unit losses are of the erosion type, where \( \beta^m_i(k), T^m_i(k), N^m_i(k), L^m_i(k) \) are weather effects, the number of targeted units, the total number of units, and the number of lost units respectively. The term \( A^m_i(k) \) in (19) is the percentage of units targeted at step \( k \) surviving the transition to step \( k+1 \) and \( T^m_i(k) \) represent the number of targeted units at step \( k \). For this model, we allow only one-on-one engagement at each time step. Once the identities of the attacking and the attacked units are determined from the choice of target controls, this percentage can be expressed as:

\[
A^m_i(k) = 1 - \beta^X_i \beta^Y_i P^{XY}_{ij}(k).
\]

(20)

In (20) the term \( \beta^X_i \beta^Y_i \) represent the probabilities that the \( j \)th unit of \( Y \) acquires the \( i \)th unit of \( X \) as a target. The other term in (20), \( P^{XY}_{ij} \), is called the “erosion factor” and can be computed from

\[
P^{XY}_{ij} = 1 - (1 - \beta^m_w(k) PK^{XY}_{ij}).
\]

(21)

Eq.(21) represents the probability of the \( i \)th unit of \( X \) being destroyed by a weapon fired from the \( j \)th unit of \( Y \). The term \( PK^{XY}_{ij} \) represents the probability that the \( i \)th unit of \( X \) is destroyed by a weapon fired from the \( j \)th unit of \( Y \) under ideal weather and terrain conditions. The factor \( \beta^m_w \) in Eq.(21) and Eq.(16) accounts for weather instability effects.

E. Objective Function

Although in the general model, every unit can have its own objective function, in this paper, we consider a global objective functions for all units of one force that take part in the operation. We define that aggregate objective function for the Blue force in the following way:
\[ J_k^B(x_k, u_1, u_2) = \max_{u_k} E \left[ \sum_{i=1}^{N_{BM}} \alpha_{BMi} P_{u_k}^{BM}(k) T_{BMi}^k(k) + \sum_{i=1}^{N_{BS}} \alpha_{BSi} P_{u_k}^{BS}(k) T_{BSi}^k(k) \right. \\
\left. - \sum_{i=1}^{N_{RFT}} \alpha_{RFTi} P_{u_k}^{RFT}(k) T_{RFTi}^k(k) \right. \\
\left. - \sum_{i=1}^{N_{RDU}} \alpha_{RDUi} P_{u_k}^{RDU}(k) T_{RDUi}^k(k) \right] \]

(22)

The Red force’s objective function can be written in a similar way. In this equation, Blue team tries to find the suitable configuration \( u_k \) to maximize its objective function. Each objective function is formed of the force’s reward (over all its units) minus a penalty over the enemy force’s units. Note that if the weighting coefficients \( \alpha_{Xi} = \beta_{Xi} \) for every \( k \) in eq.(22), we will have a zero-sum dynamic game.

III. Computational Complexity

Discrete-time dynamic games in general suffer from the curse of dimensionality. If the decision-makers have a finite set of decision variables at each stage, then the number of all possible sequences choices over the entire game grows exponentially with the number of stages. In our problem, we have a finite set of control variables for every unit in the battle. This set is determined by the set of allowable moving controls \( d_{im}^m(k) \), the set of choice of target \( c_{im}^m(k) \), the set of choice of fire \( r_{im}^m(k) \), and the set of team configuration controls \( f_{im}^m(k) \). For motion control, every unit has four choices for the adjacent locations and the “no movement” control. So for motion control, we have \( 5 \times (N_{BM} + N_{BS} + N_{RDU}) \) possible actions. The number of target choices for any force at time \( k \) is equal to the number of the opposite force units. A unit \( m \)th of the Blue force has \( N_{RDU} + N_{RFT} \) choices for the next target and vice versa. Thus for the target choice control \( c_{im}^m(k) \), we have \( (N_{BM} + N_{BS}) \times (N_{RDU} + N_{RFT}) \) choices for the Blue force and \( (N_{RDU} \times (N_{BM} + N_{BS})) \) for the Red force. For the fire choice \( r_{im}^m(k) \), each unit in either force has the option to fire or not to fire, that is \( 2 \times (N_{BM} + N_{BS} + N_{RDU}) \) control choices. Our last command is \( f_{im}^m(k) \); for this one the command vector length is \( 2 \times (N_{BM} + N_{RDU}) \). Thus, the total number of choices for the \( m \)th unit in the Blue force at time step \( k \) is equal to the number of enemy units \( N_{RDU} + N_{RFT} \) that could be attacked. For each unit in the Red force, the \( m \)th unit in the Blue force should decide on a fire choice in a space that is \( 2 \times (N_{RDU} + N_{RFT}) \), and also on the team configuration control, which is another \( 2 \times (N_{RDU} + N_{RFT}) \) space choices. The total number of choices for the \( m \)th unit in the Blue force at time step \( k \) is therefore given by:

\[ n_{im}^B(k) = 5 + 4(N_{RDU} + N_{RFT}) \]

This number will be multiplied by the number of units in the Blue force, in order to get the total number of choices for the Blue force in one stage. These numbers of different control sequences for each force will be multiplied by the number of stages \( K \), in order to get the total control commands for the entire game. This exponential growth in the number of control sequences will lead to an equally huge number of states that need to be determined. It will therefore become computationally unfeasible to determine an optimal solution for the game over its entire duration of \( K \) time steps. In the next section, we will provide our solution approach for this complicated problem.

IV. Solution Approach

The optimal solution for such a composite game over all stages is difficult, if not impossible, to obtain. In order to reduce the computational requirements, instead of maximizing the objective function over the entire time horizon, we only consider locally optimal sub-solutions to the problem. We deal with each control command separately from the others. After choosing each control command individually, we used the whole set of control commands together to run the game. In the following, we will describe each control methodology. We specify our solution approach in a specialization of the scenario described above for readability and to provide a simulation example.

A. Motion Control \( d_{im}^m(k) \)

Assume that all the Blue force assets have the same speed and all the Red units move at a much lower speed so that effectively they appear stationary. The control and the state variables will be updated every time step. The path from a \( i \)th unit of type \( X \) in the Blue force to a \( j \)th unit of type \( Y \) in the Red force is generated using the shortest path cost function. That is

\[ J_{ij}^{XY} = \min_{ij} \sum \left| x_i^X - y_j^Y \right| + \left| y_i^X - y_j^Y \right| \]

(24)

So the \( i \)th unit of the Blue force will determine its \( \Delta x_i(k+1) \) and \( \Delta y_i(k+1) \) sequence according to the path that minimizes the cost function in eq. (24).

B. Fire Control

There are two control variables related to this command.

1) Target Choice \( c_{im}^m(k) \): The first one is \( c_{im}^m(k) \), the choice of the next target for the \( m \)th unit of type \( m \) at time step \( k \). The cost function we use for choosing the next target is a risk assessment cost function. An \( i \)th unit of type \( X \) of the Blue force chooses its next target \( j \) of type \( Y \) according to the cost function:

\[ c_{ij}^{XY} = \min_{j=1}^{N_Y} h_{ij}^{XY}(k) \]

(25)

Where \( h_{ij}^{XY} \) represents the risk associated with the \( j \)th unit of type \( Y \) being targeted by the \( i \)th unit of type \( X \). Eq.(25) means that at each time step \( k \), an \( i \)th unit of type \( X \) will target the lowest risk factor \( j \)th unit of all the units.
of type $Y$. That risk factor is calculated for each $j$th unit of type $Y$ according to the equation:

$$ h_{ij}^{XY} = \sum (a_j^Y + \sum h_j^Y) \tag{26} $$

Here $a_j^Y$ is a factor specified a priori to represent the known shooting ability for the type $Y$ unit. So the risk factor for the $j$th unit of type $Y$ is this $a_j^Y$, which is a common factor between all units of the same type, added to the number of units of type $Y$ that can shoot the $i$th unit of type $X$ in its current location. Note that $U^Y \subset N^Y$, and this number can be determined according to eq. (11) which represents the fire distance condition.

2) Fire Choice $r^m_i(k)$: At every decision step $k$, only one unit in either force has the ability to shoot. That is:

$$ \sum_{i=1}^{N^{BM}} r_i(k) + \sum_{i=1}^{N^{BS}} r_i(k) + \sum_{i=1}^{N^{RDU}} r_i(k) \leq 1 \tag{27} $$

According to the above equation, shooting will occur sequentially. The Blue force will start the shooting sequence until all involved units of different types of the Blue force finish, then the Red force will have the ability to shoot back.

C. Team Configuration Control $f^m_i(k)$

All units in either force have two different team configurations to choose from: Cooperative team and Independent unit. For the Red force, we assume that all of the RDU will be located around the RFT trying to defend them. So all of the RDU and the RFT will work as a team until the RFT are rendered inoperative. When they work as a team, they have a higher shooting probability. When the RFT turn inoperative, the RDU automatically change to enter the “Independent unit” mode. For the Blue force, we define a team as a grouping of $n$ units which have a common mission, where $n$ is unity or greater. Note that it is possible for a team to be composed of only one unit. We assume that if we have only one unit, it will have more ability to shoot and hide. A team will have a higher collaborative shooting ability but it will also be easier to track and shoot it. We use Stochastic Dynamic Programming (SDP) to solve this trade-off game according to the equation (22). Here $\alpha_B$ is the weighting coefficient for the Blue force objective function, $P^{m}_{f^m}(k)$ is the shooting ability for the units of type $m$ according to their configuration $f^m_k$, and $T^{m}_{f^m}(k)$ is the number of units targeted by $i$th unit of type $m$ at time step $k$. Normally the SDP methodology yields a general strategy to control the entire game. Starting from time step $k = N$, going backward to $k = 0$, we find the maximum cost function at this step by comparing the “Cooperative team” cost function with the “Independent unit” cost function. Following these steps at each time step, we obtain a threshold solution. Over the entire time horizon $N$, this threshold strategy will control the choice of $f^m_i(k)$ at each time step. If the shooting ability $P^{RDU}_{f^m}(k)$ of the opposite force is higher than the threshold shooting probability, then the best solution is “Cooperative team”. Otherwise the “Independent unit” shooting will yield the maximum cost function.

V. SIMULATION EXAMPLES

Our scenarios take place on a 10x10 square grid. Each square on the grid corresponds to 40x40 square miles in dimensions. The Blue force consists of two Unmanned Air Vehicle (UAV) units: one Bomber and one Jammer. The Bomber can operate independently or cooperatively with the Jammer. The configuration will be adapted based on the Red force’s adversarial behavior. For all configurations, UAVs should fulfill and meet the cost function and the required command specifications. The mission of the Blue force is to destroy one RFT, which is a Command Center (CC), that is heavily defended by a number of RDUs. We consider a zero-sum game, in which the Blue force’s mission will be considered accomplished when the RFT and the RDUs are destroyed completely. After relaxing the weapons command, we assume that when engagement occurs, the forces continue optimizing their controls online until the goal is accomplished. We tried different configurations for the Red force in the simulation. In the next section, we present a heterogeneous scenario for the Red force.

A. heterogeneous distribution for the Red force assets

We use a configuration as in fig. 1, in which the SAM sites are located in a heterogeneous distribution around the command center. In this configuration, the UAVs can start shooting the command center first from the opening locations and then go to the SAM sites according to their risk factor. Since we will not be able to show any animation of the units’ movement or the unit erosion, the scenario’s initial conditions are shown in fig. 1.

Table I summarize the destruction probabilities of each unit in the Blue force and the Red force respectively according to its configuration. We assume that the destruction probability $P^{m}_{f^m}(k)$ of each unit does not depend on the facing unit as much as it depends on the unit team configuration. In the simulation, we assume ideal weather conditions ($\beta_w = 1$) and equal weighting coefficients for $\alpha_{X_1}$ and $\beta_{X_1}$ (zero-sum game). All of these probabilities and coefficients can be adjusted by the top commander.

<table>
<thead>
<tr>
<th>Configuration</th>
<th>BM</th>
<th>BS</th>
<th>RDU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cooperative</td>
<td>0.6</td>
<td>0.6</td>
<td>-</td>
</tr>
<tr>
<td>Independent</td>
<td>0.5</td>
<td>0.5</td>
<td>-</td>
</tr>
<tr>
<td>CC alive</td>
<td>-</td>
<td>-</td>
<td>0.4</td>
</tr>
<tr>
<td>CC inoperative</td>
<td>-</td>
<td>-</td>
<td>0.1</td>
</tr>
</tbody>
</table>
we mean all the RDUs that see the Blue force according to (11) will shoot sequentially according to (14). Once the Command Center is rendered inoperative, each RDU will have lower shooting ability and will be defending itself only. For the Blue force, the game starts with the Target Choice control \( c_{i}^{u}(k) \) which will yield an ascending ordered list of the targets according to their risk factor as in (25). The target with the lowest risk factor is considered next. The Motion control returns the path from the current location for the bomber to its target according to (24). The jammer will join the bomber in this engagement if the risk factor for this target according to (25) is higher than the threshold shooting probability for this battle. Either the bomber only or the jammer and the bomber head to the target. They follow (24) in their path and (9). Once they arrive at the target, the fire maneuver starts according to (27), (13), and (14). As we can see from the figures, the UAVs working as a team engage the Command Center first in order to weaken the RDUs. One jammer is lost at time step \( k = 7 \), another one joins the attack at \( k = 8 \). After several additional time steps at \( k = 35 \), the mission is accomplished. The Red force lost all its units and the Blue force lost one jammer.

In such a dynamic environment, we assume that the game between the Blue and Red force takes place in several engagement stages. In each stage, a selected team of UAVs attacks certain selected targets. At the end of the stage, either the target is destroyed or one of the UAVs is destroyed. The decision commands try to maximize the cost function in (??), which includes a reward for destroying a target and a penalty for risking the UAVs. The risk of utilizing a UAV of a type increases as the number of UAVs of the same type decreases. The reward of destroying a specific target, on the other hand, is always the same.

This formulation enables a number of UAVs to achieve autonomous battle management in the presence of adversarial behavior. The algorithm can be extended to incorporate more configurations in either team (Red or Blue). For the Blue team, you can solve the problem off-line and specify threshold regions for each configuration. Then on-line, the controller will tune the team according to your threshold limits. For the Red team, we can propose different configurations. It was shown that this decoupled sub-optimal methodology was robust and quick enough to incorporate different variations relating to adversary actions and behavior models in military operations.

VI. CONCLUSION

The optimization of complex highly nonlinear multi-controller multi-objective systems requires the use of algorithms that mainly employ game theory, queueing theory, or dynamic programming. In general, the computational requirements in using these results could be extensive and hence their usefulness in a practical situation may become extremely limited. Because the design space is large and heterogeneous, in this paper we consider a sub-optimal structure solution as a possible mechanism for overcoming these limitations. We structure the space by decomposing it into task allocation and cooperation decision control. The task allocation procedure returns the next waypoint for the UAVs to target under dynamic command constraints.

Cooperation decision is made on-line in accordance to a given off-line policy. It is shown that the controllers implementation for each command are consistent with the system specification on the desired team behavior under dynamic adversarial changes. Computer simulation was done on a highly nonlinear dynamic model of a military air operation. Optimization and simulation results obtained for the sub-optimal decoupled solution illustrate the usefulness of this approach in accomplishing autonomous battle management in adversarial environment.

REFERENCES