# Case-based Team Recognition Using Learned Opponent Models

Michael W. Floyd[1], Justin Karneeb[1], and David W. Aha[2]

[1]Knexus Research Corporation; Springfield, VA, USA
[2]Navy Center for Applied Research in AI;
Naval Research Laboratory (Code 5514); Washington, DC, USA
{first.last}@knexusresearch.com
david.aha@nrl.navy.mil

**Abstract.** For an agent to act intelligently in a multi-agent environment it must model the capabilities of other agents. In adversarial environments, like the beyond-visual-range air combat domain we study in this paper, it may be possible to get information about teammates but difficult to obtain accurate models of opponents. We address this issue by designing an agent to learn models of aircraft and missile behavior, and use those models to classify the opponents' aircraft types and weapons capabilities. These classifications are used as input to a case-based reasoning (CBR) system that retrieves possible opponent team configurations (i.e., the aircraft type and weapons payload per opponent). We describe evidence from our empirical study that the CBR system recognizes opponent team behavior more accurately than using the learned models in isolation. Additionally, our CBR system demonstrated resilience to limited classification opportunities, noisy air combat scenarios, and high model error.

**Keywords:** Beyond-visual-range air combat, autonomous agents, team recognition, opponent modeling

## 1 Introduction

Beyond-visual-range (BVR) air combat is a modern style of air-to-air combat where teams of aircraft engage each other over large distances using long-range missiles [1]. This differs from the classic dogfighting combat of World Wars I and II, where aircraft used short-range weaponry in fast-paced, close-quarters combat. Whereas dogfighting lends itself well to reactive control strategies, BVR allows for longer-term strategic planning and reasoning. For an agent that engages in air combat, both styles offer similar challenges including an adversarial environment, imperfect information, and real-time performance constraints. While the large distance between aircraft provide BVR agents more time to reason than dogfighting agents, it also increases uncertainty when observing other aircraft.

One significant limitation of long distance observations is that they make it difficult to accurately identify the capabilities of opponent aircraft. Observations are made through various types of long-range sensors rather than being observed directly by a

pilot, making it difficult to sense opponents with sufficient precision to accurately detect their capabilities (e.g., maximum speed, maneuverability, flying range). For example, at close range it may be possible to visually differentiate the type of aircraft based on shape or defining characteristics (i.e., paint, materials, and engine type) but onboard sensors may be unable to provide information other than the aircraft's position, speed, and heading. Similarly, while it is possible to detect when an opponent fires a missile, it is difficult to determine the exact properties of an opponent's weapons (e.g., range, maximum speed, payload) through long-range sensors alone. An opponent's aircraft type and weapon capabilities could be provided as part of a pre-mission briefing, but given the adversarial nature of air combat, such information may be outdated (e.g., a last-minute aircraft change) or erroneous (e.g., deception by opponents). Having inaccurate opponent information in BVR combat can result in the agent wasting resources (e.g., firing a missile an opponent can easily evade), selecting sub-optimal goals or plans (e.g., based on incorrect assumptions about an opponent's possible actions), or putting itself in dangerous situations (e.g., underestimating an opponent's weaponry). BVR combat scenarios typically involve engaging with a team of opponents, thereby compounding the potential impact of incorrect assumptions about opponents.

Our work has two primary contributions. First, we describe an approach for learning models to predict the movement of aircraft and missiles in BVR scenarios. When encountering an unknown aircraft, these models can be used to classify the type of aircraft and its weapons capabilities. Second, we present a case-based reasoning (CBR) system that can use the classification of individual aircraft to determine the composition of an opposing team. Our approach requires only a small subset of aircraft or missiles to be correctly classified to perform accurate retrieval, making it resilient to classification errors (i.e., due to learning error or unexpected opponent behavior) and limited opportunities to classify opponents (i.e., when only certain observed behaviors can be used for classification).

In the remainder of this paper we describe our approach for opponent model learning and team recognition. Section 2 describes the BVR combat domain and motivates why accurate information about aircraft type and weapons capabilities are necessary. Our approach for learning aircraft and missile models is presented in Section 3, with a focus on how the models can be used for classification. Section 4 describes our case-based team recognition system, and how classifications of individual aircraft and missiles can be used to determine the composition of the entire team. In Section 5, we report evidence that our system improves team recognition performance in BVR scenarios. Related work is discussed in Section 6, followed by conclusions and topics of future work in Section 7.

## 2 Beyond-Visual-Range Air Combat

BVR scenarios occur in large airspaces (i.e., thousands of square kilometers) with opposing aircraft located tens or hundreds of kilometers from each other. Figure 1 shows a graphical representation of a BVR engagement between two opposing teams,

each of which has five aircraft. The objective of each team is to destroy their opponents or force them to retreat. Given the large distances involved, aircraft are equipped with active radar homing missiles that have ranges of approximately 50 kilometers.

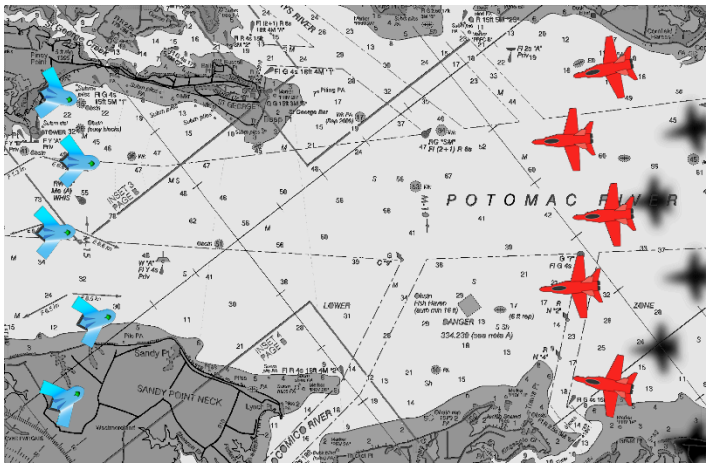


**Fig. 1.** Graphical representation of two teams of aircraft engaged in a 5 vs 5 beyond-visual-range air combat scenario (aircraft size is not shown to scale)

We use a high-fidelity BVR air combat simulator for our studies, the Advanced Framework for Simulation, Integration, and Modeling (AFSIM) [2]. AFSIM allows for control of a simulated aircraft using low-level control commands or high-level actions. Additionally, aircraft can be controlled programmatically (e.g., scripts or agents) or by human pilots using physical hardware. In AFSIM, each controller (i.e., script, agent, human) pilots a single aircraft. For the remainder of this paper, we assume that aircraft are controlled by intelligent agents.

At the start of a BVR mission, each agent receives a *mission briefing* that contains information about its teammates and its opponents. This information includes the number of aircraft per team, the type of each aircraft (i.e., the aircraft architecture, maximum speed, maneuverability), and each aircraft's weapons capabilities (i.e., the range and speed of its missiles). For teammates, this information can be assumed to be accurate. However, information about opponents may come from assumptions, intelligence reports, or previous encounters, so there is no guarantee that mission briefing data is accurate. As such, an agent that relies on this information will need to verify and update it during a mission. There are several reasons why information about an opponent's aircraft type and weaponry are vitally important. First, it directly impacts the attack ranges of the agent and its opponents. Underestimating an opponent's aircraft type will cause the agent to fire missiles that the opponent can easily evade, whereas overestimation will prevent the agent from firing in advantageous positions. Similarly, overestimating the opponent's weapons capabilities will cause the agent to engage from longer distances, possibly never entering a reasonable firing range, and underestimating may cause the agent to fly into dangerous positions. Second, an accurate model of each

opponent and their capabilities directly influences an agent's ability to perform long-term prediction, select appropriate goals, and generate appropriate plans.

Each agent receives sensory input at discrete time internals. The input includes the set of objects that are currently visible to the agent and positional information for each object. An object reading $o_i^t$ of object $i$ at time $t$ is a tuple $o_i^t = \langle lat_i^t, long_i^t, a_i^t, b_i^t, v_i^t, ac_i^t \rangle$ containing its latitude $lat_i^t$, longitude $long_i^t$, altitude $a_i^t$, bearing $b_i^t$, velocity $v_i^t$, and acceleration $ac_i^t$. The objects include aircraft and active missiles, but only a subset of objects are visible to each agent due to limited radar range. However, we assume that agents on the same team can communicate and share information (AFSIM provides such capabilities). If at time $t$ the entire team can observe $n_t$ unique objects $o_1^t, \ldots, o_{n_t}^t$ (i.e., the number of visible objects may change over time), each agent on that team receives as input a set $S_{team}^t$ that includes readings from all objects currently visible to the team ($S_{team}^t = \{o_1^t, \ldots, o_{n_t}^t\}$). The role of an agent is to use the mission briefing and sensory information to intelligently control the aircraft.

## 3 Opponent Model Leaning

In Section 2 we described why agents require accurate models of their opponents to operate efficiently in BVR scenarios but did not address what the models contain or how they are used. Our work focuses on models of an opponent's *maneuverability* and *weapon range*. The maneuverability is based on its aircraft type (e.g., F-16 Fighting Falcon, F/A-18 Super Hornet, Su-27 Flanker, MiG-29 Fulcrum) and incorporates velocity, acceleration, and turning radius. Similarly, the weapon range is based on the type of missiles an aircraft is equipped with and their effective range (e.g., short-range AA-11 Archer, medium-range AIM-120 AMRAAM, long-range AIM-54 Phoenix).

The primary challenge of using aircraft and missile models is that there are limited opportunities to differentiate between the possible models. Aircraft types differ based on their top-end performance but the majority of the time all aircraft will operate similarly. For example, aircraft use cruising speeds that are significantly less that their maximum speed, so all aircraft will appear identical when cruising. It is only when an aircraft operates at their top-end that they show noticeable differences. Similarly, the type of weapons an aircraft is equipped with can be determined only when a missile is fired.

We restrict our models to observations that can reliably differentiate between different aircraft and missiles. The following information is used:

- **Aircraft Models:** The most likely time for an aircraft to display its top-end performance is when it is threatened. As such, observations are collected while an aircraft is evading a missile. If at time $t$ a missile is fired at aircraft $i$, readings for the evading aircraft are added to the set $\mathcal{A}_i$ during a window of length $w_1$: $\mathcal{A}_i = \mathcal{A}_i \cup \{o_i^t, \ldots, o_i^{t+w_1}\}$. If the missile is destroyed before the end of the window (i.e., it reaches its maximum range and crashes, or collides with an object), any observations after destruction are not added to the set. This is because

the missile is no longer a threat so the aircraft will no longer evade it. Since each aircraft can be attacked multiple times, the set is extended during each attack. There is no guarantee that all observations in the set are of the aircraft actively evading a missile. For example, an aircraft could determine that its current cruising speed is sufficient to evade the missile, or be unaware that a missile has been fired at it. However, we assume that a sufficient number of observations will be of active evasion.

- **Weapons Models:** Missiles do not display the same level of agency as aircraft (i.e., they fly at maximum speed towards their target), so observations are collected as soon as a missile is detected. If at time $t$ missile $j$ is fired by aircraft $i$, readings for the missile are added to the set $\mathcal{W}_i$ during a window of length $w_2$: $\mathcal{W}_i = \mathcal{W}_i \cup \{o_j^t, \dots, o_j^{t+w_2}\}$. As with aircraft, observations are not added after the missile is destroyed (i.e., if the missile is destroyed before $w_2$). This groups together the observations of all missiles fired by an aircraft and assumes that each aircraft is equipped with a single type of missile (although we would like to relax that assumption in future work).

### 3.1 Model Training

Training the models requires obtaining a set of training observations for each type of aircraft and missile. However, in adversarial domains this can be challenging. The primary difficulty is collecting observations that represent actual engagements. Engagements are likely rare, so there are limited opportunities to collect training data. There is also the possibility of the opponent developing or deploying new aircraft or missiles (i.e., with no existing model).

To overcome these challenges, our models are trained on observations of friendly aircraft during training missions. The missions are simplified scenarios using simulated missiles (i.e., they will not damage the aircraft) where one aircraft pursues and simulates attack on another. Each training mission ends when the target aircraft is hit or successfully evades. The parameters for a mission are: *target's aircraft type*, *attacker's missile type*, *initial distance between aircraft*, *starting altitude of each aircraft*, *starting velocity of each aircraft*, and *relative heading of each aircraft*. This allows data to be collected for each aircraft and missile type using a variety of initial configurations (e.g., based on expert input or random sampling). Data collection is restricted only by the time and availability of training aircraft.

Uncertainty about possible opponent aircraft and missile types is handled by having friendly aircraft perform synthetic opponent behavior. For aircraft, this involves placing artificial limits on the training aircraft's *turning radius*, *acceleration*, and *maximum velocity*. For missiles, limits are placed on the training missile's *maximum range*, *acceleration*, and *maximum velocity*. Thus, modifying one of more of these parameters effectively creates a synthetic opponent that can be used to train a new model. It is possible that unrealistic models will be learned (i.e., the opponent does not use a similar aircraft or missile) or that it is not possible to replicate a particular aircraft or missile type (e.g., the opponent aircraft's maneuverability exceeds the training aircraft's top-

end performance). However, we anticipate the impact of superfluous or unobtainable models is offset by the performance benefits of learning valid models.

If $l$ synthetic aircraft types and $k$ synthetic missile types are used, $l$ aircraft models $M_{air}^1, \ldots, M_{air}^l$ and $k$ missile models $M_{mis}^1, \ldots, M_{mis}^k$ are learned. Each model is trained using all observations of that object type collected during training missions (i.e., the set $\mathcal{A}_i$ containing all observations of aircraft type $i$ and $\mathcal{W}_j$ containing all observations of missile type $j$). Input values are current observations (e.g., observed values at time $t$) and outputs estimate the expect rate of change (e.g., the rate of change between time $t$ and time $t+1$). If an observation is the last in a temporally related sequence (i.e., the last observation of an evasion or missile flight), it does not have a subsequent observation to calculate rate of change so is not used for training. The inputs and outputs are:

- **Aircraft**
  - **Inputs**: *bearing* (degrees), *velocity* (meters per second), *distance to attacking missile* (meters), *velocity of attacking missile* (meters per second)
  - **Outputs**: *rate of altitude change* (meters per second), *rate of separation from attacking missile* (meters per second, with positive values representing the aircraft distancing itself from the missile)
- **Missile**
  - **Inputs**: *altitude* (feet), *flight time* (seconds)
  - **Output**: *acceleration* (meters per second squared)

Models can be learned using any algorithm that can learn a mapping from continuous inputs to continuous outputs. However, for the remainder of this paper we use the M5′ algorithm [3]. M5′ is a decision tree induction algorithm where each leaf node contains a regression model. Training instances are first used to build the tree, and then all training instances that arrive at the same leaf node are used to train a linear regression model for that node. For an input instance, it traverses the tree to a leaf node and its outputs are calculated using the regression model at that node. Since there are two outputs for aircraft models, one decision tree is used per output.

### 3.2 Model-Based Classification

The learned models are used during scenarios to continuously predict the movement of aircraft and missiles. Since the models use values from time $t$ to predict the rate of change between $t$ and $t+1$, the output of a model can be evaluated at each subsequent time step. During an evasion, all aircraft models $M_{air}^1, \ldots, M_{air}^l$ are used to generate predicated outputs $p_{air_t}^1, \ldots, p_{air_t}^l$ (i.e., each prediction is a tuple containing the rate of altitude change and rate of separation from attacking missiles) at each time $t$. Similarly, during the flight of a missile, all missile models $M_{mis}^1, \ldots, M_{mis}^k$ are used at each time $t$ to generate predicted outputs $p_{mis_t}^1, \ldots, p_{mis_t}^k$ (i.e., each predication is the acceleration). At time $t+1$, the observed values $o_{air_t}$ and $o_{mis_t}$ are computed.

If the models have been used to predict values between time $t$ and $t + c$, the aircraft or missile is classified based on the model that minimizes the distance between predictions and observations:

$$class_{air} = \underset{i=1\ldots l}{\operatorname{argmin}}(dist^i_{air}), \qquad dist^i_{air} = \sum_{j=t}^{t+c} dist(p^i_{air_j}, o_{air_j})$$

$$class_{mis} = \underset{i=1\ldots k}{\operatorname{argmin}}(dist^i_{mis}), \qquad dist^i_{mis} = \sum_{j=t}^{t+c} dist(p^i_{mis_j}, o_{mis_j})$$

Although classifications can be made at any time, in practice we use only the classifications obtained by observing the entire sequence (i.e., entire evasion or missile flight). For missiles, the distance function $dist(p_{mis}, o_{mis})$ computes the absolute distance between the predicted and observed values (i.e., $|p_{mis} - o_{mis}|$). The distance function for aircraft $dist(p_{air}, o_{air})$ is slightly more complicated since each value is a tuple containing both the rate of altitude change $\Delta alt$ and rate of separation from attacking missile $\Delta sep$ (i.e., $p_{air} = \langle \Delta alt_p, \Delta sep_p \rangle$ and $o_{air} = \langle \Delta alt_o, \Delta sep_o \rangle$). The distance function computes the average absolute distance between the output: (i.e., $\frac{|\Delta alt_p - \Delta alt_o| + |\Delta sep_p - \Delta sep_o|}{2}$). The confidence in each of the $i$ models is also calculated, with values ranging from 0 to 1 (inclusive):

$$conf^i_{air} = \frac{\sum_{j=1\ldots l}(dist^j_{air}) - dis^i_{air}}{\sum_{j=1\ldots l}(dist^j_{air})}, \qquad conf^i_{mis} = \frac{\sum_{j=1\ldots k}(dist^j_{mis}) - dis^i_{mis}}{\sum_{j=1\ldots k}(dist^j_{mis})}$$

The confidence values are stored in the sets $\mathcal{CONF}_{air} = \{conf^1_{air}, \ldots, conf^l_{air}\}$ and $\mathcal{CONF}_{mis} = \{conf^1_{mis}, \ldots, conf^k_{mis}\}$. Thus, each classification outputs a class label (i.e., $class_{air}$ or $class_{mis}$) and the confidence in each possible label (i.e., $\mathcal{CONF}_{air}$ or $\mathcal{CONF}_{mis}$).

## 4 Case-Based Team Recognition

The learned models can be used to classify individual aircraft and missiles but, as we discussed in the previous section, the situations when classification can be performed are limited. When engaging a team of opponents, it is possible that some aircraft will never evade or fire missiles. To overcome the scarcity of classification opportunities, and therefore the scarcity of class labels, we use a case-based team recognition approach.

We assume the availability of a case base containing known compositions of opponent teams. Each case $C$ contains both the team composition $T$ and team properties $P$: $C = \langle T, P \rangle$. The team composition is a set containing the aircraft type and missile type of each member of the team: $T = \{\langle class'_{air}, class'_{mis} \rangle, \langle class''_{air}, class''_{mis} \rangle, \ldots\}$. The properties include additional information about the team including the team leader, base of operations, and records of previous encounters. The goal of the CBR process is to retrieve a case that is similar to the opponent observations. First, a target team $T_{tar}$ is created by merging the team provided by the mission briefing $T_{MB}$ and the observed team $T_{obs}$. While $T_{MB}$ contains a full, although possibly incorrect, team, $T_{obs}$ may

contain unknown values if only a subset of classifications have been performed (e.g., $class_{air} = \emptyset$, $class_{mis} = \emptyset$, or both are unknown). The method for merging the mission briefing and observations is show in Algorithm 1.

**Algorithm 1:** Merging mission briefing and observations

**Function:** *merge*$(T_{MB}, T_{obs})$ ***returns*** $T_{tar}$

1 $T_{tar} \leftarrow \emptyset$;
2 **foreach** $\langle class_{air}, class_{mis} \rangle \in T_{obs}$ **do**
3  **if** $\langle class_{air}, class_{mis} \rangle \in T_{MB}$ **then**
4  $T_{tar} \leftarrow T_{tar} \cup \langle class_{air}, class_{mis} \rangle$;
5  $T_{MB} \leftarrow T_{MB} \setminus \langle class_{air}, class_{mis} \rangle$; $T_{obs} \leftarrow T_{obs} \setminus \langle class_{air}, class_{mis} \rangle$;
6 **foreach** $\langle class_{air}, class_{mis} \rangle \in T_{obs}$ **do**
7  **foreach** $\langle class'_{air}, class'_{mis} \rangle \in T_{MB}$ **do**
8  **if** $class_{mis} = class'_{mis}$ **then**
9  $T_{tar} \leftarrow T_{tar} \cup \langle class'_{air}, class'_{mis} \rangle$;
10  $T_{MB} \leftarrow T_{MB} \setminus \langle class'_{air}, class'_{mis} \rangle$; $T_{obs} \leftarrow T_{obs} \setminus \langle class_{air}, class_{mis} \rangle$;
11  **break**;
12 **foreach** $\langle class_{air}, class_{mis} \rangle \in T_{obs}$ **do**
13  **foreach** $\langle class'_{air}, class'_{mis} \rangle \in T_{MB}$ **do**
14  **if** $class_{air} = class'_{air}$ **then**
15  $T_{tar} \leftarrow T_{tar} \cup \langle class'_{air}, class'_{mis} \rangle$;
16  $T_{MB} \leftarrow T_{MB} \setminus \langle class'_{air}, class'_{mis} \rangle$; $T_{obs} \leftarrow T_{obs} \setminus \langle class_{air}, class_{mis} \rangle$;
17  **break**;
18 **foreach** $\langle class_{air}, class_{mis} \rangle \in T_{obs}$ **do**
19  **foreach** $\langle class'_{air}, class'_{mis} \rangle \in T_{MB}$ **do**
20  $tar_{air} = class_{air}$; $tar_{mis} = clas_{mis}$;
21  **if** $tar_{air} = \emptyset$ **then** $tar_{air} = class'_{air}$ ;
22  **if** $tar_{mis} = \emptyset$ **then** $tar_{mis} = class'_{mis}$ ;
23  $T_{tar} \leftarrow T_{tar} \cup \langle tar_{air}, tar_{mis} \rangle$;
24  $T_{MB} \leftarrow T_{MB} \setminus \langle class'_{air}, class'_{mis} \rangle$; $T_{obs} \leftarrow T_{obs} \setminus \langle class_{air}, class_{mis} \rangle$;
25  **break**;
26 **return** $T_{tar}$;

The algorithm starts with an empty team (line 1) and adds aircraft to the team using a priority-based merging method. First, aircraft are added if both the mission briefing and observations agree on the type of aircraft and missile (lines 2-5). Second, aircraft are added if the mission briefing and observations agree on the missile type (lines 6-11). Third, aircraft are added if there is agreement on aircraft type (lines 12-17). For all three previous merging steps, the aircraft is added using the labels stored in the mission briefing (although for the first merging method the labels are identical). This is done because the observations may be missing labels, so the information from the mission briefing is used to ensure a fully-defined team. Finally, any remaining aircraft that do not have a full or partial match between the mission briefing and the observations are

merged (lines 18-25). Priority is given to the observed labels, and only if there is a missing label is information from the mission briefing used (lines 21 and 22). The method used to fill in unknown values is uninformed; it uses the value from the first available aircraft in the mission briefing. After merging, the number of aircraft stored in $T_{tar}$ is equal to the number that were originally in $T_{obs}$ and $T_{MB}$ (e.g., if $T_{obs}$ and $T_{tar}$ both contained five aircraft, $T_{MB}$ will contain five aircraft).

Consider an example where $T_{MB} = \{\langle 1, B\rangle, \langle 3, A\rangle, \langle 2, C\rangle\}$ and $T_{obs} = \{\langle 2, C\rangle, \langle 2, A\rangle, \langle \emptyset, C\rangle\}$. $T_{tar}$ is initially empty (line 1). The first merger stage (lines 2-5) finds one perfect match $\langle 2, C\rangle$ that is added to $T_{tar}$ and removed from $T_{MB}$ and $T_{obs}$ ( $T_{tar} = \{\langle 2, C\rangle\}$ , $T_{MB} = \{\langle 1, B\rangle, \langle 3, A\rangle\}$ and $T_{obs} = \{\langle 2, A\rangle, \langle \emptyset, C\rangle$ ). The second merger stage (lines 6-11) matches $\langle 3, A\rangle$ and $\langle 2, A\rangle$ because they have identical missile types. They are removed from their respective teams and $\langle 3, A\rangle$ is added to $T_{tar}$ because priority is given to aircraft from the mission briefing ($T_{tar} = \{\langle 2, C\rangle, \langle 3, A\rangle\}$, $T_{MB} = \{\langle 1, B\rangle\}$ and $T_{obs} = \{\langle \emptyset, C\rangle$). The third merger stage (lines 12-17) does not result in any changes because $T_{MB}$ and $T_{obs}$ no longer contain any aircraft with matching aircraft types. The forth merging stage (lines 18-25) pairs the remaining aircraft $\langle 1, B\rangle$ and $\langle \emptyset, C\rangle$ and merges their class labels. Priority is given to $\langle \emptyset, C\rangle$ because it came from $T_{obs}$, but its missing value is filled in with the associated label from $\langle 1, B\rangle$. The merged aircraft $\langle 1, C\rangle$ is added to $T_{tar}$, and the other aircraft are removed from their teams. This results in a final merged team of $T_{tar} = \{\langle 2, C\rangle, \langle 3, A\rangle, \langle 1, C\rangle\}$, with $T_{MB}$ and $T_{obs}$ now empty.

After the mission briefing and observations are merged, the target team is used to retrieve from the case base the case containing the most similar team. Similarity between a target team $T_{tar}$ and a source team $T_{src}$ is computed using Algorithm 2. The similarity function performs a greedy matching where the labels for each aircraft in the source team are matched to the aircraft with the most similar labels in the target team. Since the algorithm is greedy, aircraft in the source case are iterated over based on order of occurrence (line 2) and their best match is determined without considering the optimal global match (lines 3-7). Once an aircraft from the target team has been found as the best match for an aircraft in the source team, it is not considered as a possible match for any other aircraft (line 8). The similarity between the labels of two aircraft (line 5) is calculated using the local similarity function $sim(\dots)$ (lines 11-13). The local similarity function first retrieves the confidence in each of the possible class labels (lines 11 and 12). Recall that these confidence values are computed after each classification, so any class labels that came as a result of observations will have these confidence values computed (i.e., any parts of $T_{tar}$ that came from $T_{obs}$). For class labels that originated from the mission briefing, all possible class labels are given an equal confidence. The labels from the source team are used to retrieve the confidence the target team has in those labels, and their average value is returned (line 13). Since the target team's classification labels are chosen by selecting the label with the highest confidence, similarity will be highest when all labels are identical (i.e., $class_{air} = class'_{air}$ and $class_{mis} = class'_{mis}$ ). However, the similarity function takes into account the relative similarity of class labels by also using the confidence of non-matching labels, although they will result in lower similarity than matching labels.

**Algorithm 2:** Similarity between teams

**Function:** *similarity*($T_{tar}, T_{src}$) ***returns*** *sim*

1 $sim = 0$;
2 **foreach** $\langle class_{air}, class_{mis} \rangle \in T_{src}$ **do**
3 $\quad bestMatch = \emptyset$; $bestSim = -1$;
4 $\quad$ **foreach** $\langle class'_{air}, class'_{mis} \rangle \in T_{tar}$ **do**
5 $\quad\quad localSim = sim(\langle class_{air}, class_{mis} \rangle, \langle class'_{air}, class'_{mis} \rangle)$;
6 $\quad\quad$ **if** $localSim > bestSim$ **then**
7 $\quad\quad\quad bestSim = localSim$; $bestMatch = \langle class'_{air}, class'_{mis} \rangle$;
8 $\quad T_{tar} \leftarrow T_{tar} \setminus bestMatch$;
9 $\quad sim = sim + bestSim$;
10 **return** *sim*;

**Function:** $sim(\langle class_{air}, class_{mis} \rangle, \langle class'_{air}, class'_{mis} \rangle)$ ***returns*** *sim*

11 $\mathcal{CONF}_{air} = retrieveConfidence(class'_{air})$;
12 $\mathcal{CONF}_{mis} = retrieveConfidence(class'_{mis})$;
13 **return** $\frac{confidence(class_{air}, \mathcal{CONF}_{air}) + confidence(class_{mis}, \mathcal{CONF}_{mis})}{2}$;

For an example of Algorithm 2, we consider when $T_{tar} = \{\langle A, 1\rangle, \langle B, 2\rangle\}$ and $T_{src} = \{\langle B, 1\rangle, \langle A, 2\rangle\}$. We assume $\langle A, 1\rangle$ came from observations (i.e., merged from $T_{obs}$ in Algorithm 1) and has known confidence values (calculated during classification): $conf^{A}_{air} = 0.7$, $conf^{B}_{air} = 0.3$, $conf^{1}_{mis} = 0.6$, and $conf^{2}_{mis} = 0.4$. We assume $\langle B, 2\rangle$ came from the mission briefing (i.e., merged from $T_{MB}$) so the confidence values are all equal: $conf^{A}_{air} = conf^{B}_{air} = 0.5$, and $conf^{1}_{mis} = conf^{2}_{mis} = 0.5$. The first iteration (lines 2-9) finds a match for $\langle B, 1\rangle$. The similarity between $\langle B, 1\rangle$ and $\langle A, 1\rangle$ (line 5) is calculated by first retrieving the associated confidence values of $\langle A, 1\rangle$ (lines 11 and 12). As we mentioned previously, the confidence values associated with $\langle A, 1\rangle$ are $conf^{A}_{air} = 0.7$, $conf^{B}_{air} = 0.3$, $conf^{1}_{mis} = 0.6$, and $conf^{2}_{mis} = 0.4$. The confidence in class labels $B$ and 1 are retrieved (i.e., since $\langle A, 1\rangle$ is being compared to $\langle B, 1\rangle$), resulting in the values $conf^{B}_{air} = 0.3$ and $conf^{1}_{mis} = 0.6$. These values are used to compute the similarity: $sim_{B1-A1} = 0.5 \times (conf^{B}_{mis} + conf^{1}_{air}) = 0.5 \times (0.3 + 0.6) = 0.45$. The similarity between $\langle B, 1\rangle$ and $\langle B, 2\rangle$ is calculated in a similar manner, but using the confidence values from $\langle B, 2\rangle$ : $sim_{B1-B2} = 0.5 \times (conf^{B}_{mis} + conf^{1}_{air}) = 0.5 \times (0.5 + 0.5) = 0.5$. Thus, $\langle B, 1\rangle$ is matched with $\langle B, 2\rangle$ because it has the higher similarity ($sim_{B1-B2} > sim_{B1-A1}$). During the second iteration $\langle A, 2\rangle$ is matched with $\langle A, 1\rangle$ as they are the only two remaining, resulting in $sim_{A2-A1} = 0.55$. The similarity returned by Algorithm 2 is $sim_{B1-B2} + sim_{A2-A1} = 1.05$.

## 5 Evaluation

In this section, we evaluate our claim that *our case-based technique improves team recognition*. Our evaluation tests the following hypotheses:

**H1**: The teams retrieved by the CBR system are similar to the opponent's actual team (i.e., are composed of similar aircraft).
**H2**: The team retrieved by the CBR system is more accurate than the team defined in the mission briefing.
**H3**: The team retrieved by the CBR system is more accurate than relying exclusively on observations.
**H4**: The observed team using the learned models is more accurate than the team defined in the mission briefing.

### 5.1 Data Collection and Model Training

Our evaluation uses three synthetic aircraft types and five synthetic missile types. As a result, three aircraft models and five missile models are learned. The default aircraft type has similar maneuverability to an F-16 fighter jet. The other two aircraft types are modifications of the default aircraft. One has a 35% increase in maneuverability (i.e., maximum velocity, acceleration and turn radius) and the other has a 35% decrease in maneuverability. The default missile type has similar properties to missiles used by an F-16. The additional missiles are variations of the default missile with their range and maximum velocity modified. The variations are: 20% decrease, 10% decrease, 10% increase, and 20% increase.

The training missions place each aircraft type and missile type in a variety of mission configurations. For collecting aircraft data, the initial configurations use a sampling of values that are expected to be encountered during actual encounters: altitudes of the attacked aircraft (feet) from the set $\{1000, 2000, \ldots, 20000\}$, velocities of the attacked aircraft (meters per second) from the set $\{200, 225, \ldots, 350\}$, bearings of the attacked aircraft (degrees) from the set $\{0, 30, \ldots, 180\}$, and distances between the two aircraft (kilometers) from the set $\{25, 50, 75\}$. Missile data is collected with a similar set of initial configuration values: altitudes of the attacking aircraft from the set $\{1000, 2000, \ldots, 20000\}$, velocities of the attacking aircraft from the set $\{200, 225, \ldots, 350\}$, and distances between the two aircraft from the set $\{25, 50, 75\}$. Aircraft are observed when evading a missile for a maximum of 60 seconds (i.e., $w_1 = 60$) and missiles are observed for a maximum of 40 seconds (i.e., $w_2 = 40$).

As we mentioned earlier, models are learned using the M5′ algorithm. Identical settings are used to train each model: a minimum branch size of 20 (i.e., a node must contain at least 20 training instances before branching) and a minimum error reduction of 0.5 (i.e., branching must reduce error by at least 0.5).

### 5.2 Experimental Setup

Our evaluation scenarios involve two teams of five aircraft engaged in BVR air combat. The *base scenario* arranges each team in a column with teammates spaced 5.5 nautical miles (approximately 10.2 kilometers) from each other and opposing teams at a distance of 40 nautical miles (approximately 74.1 kilometers). The aircraft start at an altitude of 17,000 feet and face in the direction of their enemies (i.e., east or west). The base scenario was used to generate 200 *random scenarios* where each aircraft's position

is modified by between -3 and 3 nautical miles (approximately 5.6 kilometers) according to a uniform random distribution in both the north/south and east/west directions. Additionally, each aircraft's altitude is modified between 0 and 2500 feet and its bearing between -15 and 15 degrees (according to a uniform random distribution). Figure 1 shows a graphical representation of one such random scenario. Similar to the training missions, the evaluation scenarios use simulated missiles so no aircraft are damaged or destroyed. Each scenario has a duration of 10 minutes.

The CBR system uses a case base composed of 10 expert-authored cases, with each of the cases containing a different team composition (i.e., the aircraft type and missile type of each aircraft). Before a scenario is run, each team is assigned a team composition based on a randomly selected case (according to a uniform distribution). This represents each team's *true composition*. Additionally, each team is given a mission briefing containing the assumed composition of their opponents. The *mission briefing composition* is also randomly selected from the teams defined in the case base (according to a uniform distribution). The CBR system operates as an external observer and performs team recognition on one team per run (i.e., either the left team or the right team). Each scenario is repeated twice so that the CBR system has to recognize both teams, resulting in 400 total runs. During each scenario, the models are used to classify the aircraft and those values are merged with the mission briefing (i.e., Algorithm 1) to create an *observed composition*. Both the observed composition and mission briefing composition are used by the CBR system to retrieve the *CBR composition* (i.e., using Algorithm 2).

To measure the effectiveness of team recognition, we use two metrics: *team recognition accuracy* and *average team distance*. Team recognition accuracy measures the percentage of scenarios where a predicted team composition (i.e., mission briefing, observed, or CBR) is identical to the true composition. Average team distance measures the distance between the predicted team and the true team. Since the models are ordered based on how much they differ from the default F-16 model (i.e., -35%, 0%, and 35% for aircraft, and -20%, -10%, 0%, 10%, and 20% for missiles), the distance between two models is measured by how their indexes in the sorted lists differ. Aircraft models have a maximum distance of 2, and missile models have a maximum distance of 4. For example, the default missile model differs from itself by a distance of 0, but a distance of 2 from both the -20% and 20% models. The team distance is the summation of all model distances, and that value is averaged over all scenarios.

### 5.3 Results and Discussion

Our results are shown in Table 1. The team recognition performance of our CBR system is a statistically significant improvement over mission briefing and observation-based compositions across all metrics (using a paired *t*-test with $p < 0.001$). This provides strong support for **H2** and **H3**. Additionally, the CBR system was able to identify the correct team nearly 90% of the time and had a low average distance from the team's true composition, providing support for **H1**. The observation-based team composition was a statistically significant improvement over the mission briefing composition using the average team distance metric, but a significant decrease using team recognition

accuracy. The reason for this is because the mission briefing and CBR team compositions are guaranteed to be valid (i.e., team compositions are selected from teams contained in the case base). However, the observations are not restricted in such a way, often leading to team configurations that cannot be used as true compositions. Even though this gives the observation-based composition a disadvantage over the mission briefing composition, and results for team recognition accuracy worse than random, its recognized teams are much closer to the true composition. This provides partial support for **H4**.

**Table 1.** Results of team recognition over 400 experimental runs

| Prediction Source | Team Recognition Accuracy | Average Team Distance | | |
|---|---|---|---|---|
| | | Aircraft Models | Missile Models | Total |
| *Mission Briefing* | 10.0% | 3.32 | 5.84 | 9.16 |
| *Observations* | 4.8% | 2.60 | 1.72 | 4.32 |
| *CBR* | 89.8% | 0.19 | 0.31 | 0.50 |

Our results also demonstrate that the opportunities to use the learned models for classification are relatively rare. On average, there are 3.6 aircraft and 4.5 missiles per run that performed behaviors that could be used to classify them (i.e., evading or firing a missile). Overall, only 12% of the scenarios had sufficient data to classify all 5 aircraft and missiles in the run. Additionally, the models are learned so there is a possibility of error during learning or classification (i.e., class labels may be incorrect). The CBR process helps reduce the impact of missing information and error by allowing for partial team matches during retrieval, resulting in improved team recogntion performance.

## 6 Related Work

Our previous work related to the BVR domain has primarily focused on discrepancy detection [4] and opponent behavior recognition [5]. Team recognition can be thought of as a form of both discrepancy detection (i.e., a discrepancy in the expected team composition) and behavior recognition (i.e., an aircraft's behavior is based on its aircraft and missile type), but our prior work reasons about opponents at a higher level of abstraction (i.e., actions, plans, and goals) and cannot detect variations in an aircraft's maneuverability or weapons capabilities. Similarly, single and multi-agent behavior recognition [6] has historically focused on identifying agents' actions, activities, and behaviors. Simultaneous Team Assignment and Behavior Recognition (STABR) identifies the behavior of agents in a multi-agent environment and determines the team to which they belong [7]. This differs from our work in that it focuses on team assignment (rather than determining the capabilities of each agent) and allows for dynamic team changes (rather than a static set of teammates and enemies).

Case-based reasoning has been used for multi-agent behavior recognition in soccer [8]. Cases store environmental trigger conditions and behaviors the agents will take when the triggers occur. Similarly, plan recognition has been used as part of a case-based reinforcement learner to identify the plans of opponent teams in American Football [9]. Both of these approaches identify the coordinated behaviors of teams but cannot be used to identify changes in team composition. For example, if an elite player was substituted for a weak player, the systems could not identify the change. CBRetaliate responds to decreased mission performance using case-based reinforcement learning [10]. This allows it to respond to changes in the underlying strategies used by an opposing team. Their approach is similar to our own in that CBRetaliate detects discrepancies between the expected and observed behaviors of an opponent, but differs in that it identifies a team-level strategy rather than the composition of the team. Case-based multi-agent coordination in robotic soccer [11] is similar to our work in that cases are composed (in part) of information about agent teams. While soccer provides many similar challenges to BVR combat (e.g., noise, adversaries, non-deterministic actions), their prior work uses cases to control teammates rather than reason about opponents. Soccer is also similar to BVR combat in that it is a multi-agent environment which requires object matching due to partial observability, with greedy matching often preferable to optimal matching due to real-time constraints [12].

To the best of our knowledge, other applications of AI in BVR air combat have been restricted to expert-authored scripted agents [3] in high-fidelity simulators, and initial flight formation [13] and target assignment [14] in low-fidelity simulators. Unlike our approach, these systems do not consider the possibility that initial assumptions about opponents may be incorrect and should be continually assessed and revised as needed.

## 7 Conclusions

We presented a technique for case-based team recognition. Our approach uses learned models to classify an opponent's aircraft and missile types and utilizes that information during case retrieval. We tested our CBR system in simulated beyond-visual-range air combat scenarios and reported significantly increased team recognition performance compared to relying on the models or mission briefing data alone.

Our empirical results are promising but several areas of future work remain. We evaluated our CBR system as an external observer of BVR scenarios. We plan to incorporate the capabilities into individual agents so they can use the recognized teams to modify their own behavior. This will require evaluating both team recognition performance and influence on mission performance. Additionally, we plan to extend our approach to allow heterogeneous weapons systems (i.e., each aircraft can be equipped with multiple missile types). Finally, we plan to investigate team recognition countermeasures. A BVR agent could give the appearance of having different capabilities to influence their opponent's tactical decisions.

## Acknowledgements

Thanks to OSD ASD (R&E) for supporting this research.

## References

1. Shaw, R.L. (1985). Fighter combat: Tactics and maneuvering. Naval Institute Press.
2. Clive, P.D., Johnson, J.A., Moss, M.J., Zeh, J.M., Birkmire, B.M., and Hodson, D.D. (2015). Advanced Framework for Simulation, Integration and Modeling (AFSIM). *Proceedings of the 13th International Conference on Scientific Computing* (pp. 73-77).
3. Wang, Y., and Witten, I.H. (1997). Inducing model trees for continuous classes. *Poster Papers of the 9th European Conference on Machine Learning* (pp. 128-137). Prague, Czech Republic: Springer.
4. Karneeb, J., Floyd, M.W., Moore, P., and Aha, D.W. (2016). *Distributed discrepancy detection for BVR air combat*. In Proceedings of the IJCAI Workshop on Goal Reasoning. New York, USA.
5. Borck, H., Karneeb, J., Floyd, M.W., Alford, R., and Aha, D.W. (2015). Case-based policy and goal recognition. *Proceedings of the 23rd International Conference on Case-Based Reasoning* (pp. 30-43). Frankfurt, Germany: Springer.
6. Intille, S.S., and Bobick, A.F. (1999). A framework for recognizing multi-agent action from visual evidence. *Proceedings of the 16th National Conference on Artificial Intelligence* (pp. 518-525). Orlando, USA: AAAI Press.
7. Sukthankar, G., and Sycara, K.P. (2006). Simultaneous team assignment and behavior recognition from spatio-temporal agent traces. *Proceedings of the 21st National Conference on Artificial Intelligence* (pp. 716--721). Boston, USA. AAAI Press.
8. Wendler, J., and Bach, J. (2003). Recognizing and predicting agent behavior with case based reasoning. *Proceedings of the RoboCup Robot Soccer World Cup* (pp.729-738).
9. Molineaux, M., Aha, D.W., and Sukthankar, G. (2009). Beating the defense: Using plan recognition to inform learning agents. *Proceedings of the 22nd International Florida Artificial Intelligence Research Society Conference* (pp. 337-343). Sanibel Island, USA: AAAI Press.
10. Auslander, B., Lee-Urban, S., Hogg, C., and Muñoz-Avila, H. (2008). Recognizing the enemy: Combining reinforcement learning with strategy selection using case-based reasoning. *Proceedings of the 9th European Conference on Case-Based Reasoning* (pp. 59-73). Trier, Germany: Springer.
11. Ros, R., López de Mántaras, R., Arcos, J.L., and Veloso, M.M. (2007). Team playing behavior in robot soccer: A case-based reasoning approach. *Proceedings of the 7th International Conference on Case-Based Reasoning* (pp. 46-60). Belfast, Northern Ireland: Springer.
12. Floyd, M.W., Esfandiari, B., and Lam, K. (2008). A case-based reasoning approach to imitating RoboCup players. *Proceedings of the 21st International Florida Artificial Intelligence Research Society Conference* (pp. 251-256). Coconut Grove, USA: AAAI Press.
13. Luo, D.-L., Shen, C.-L., Wang, B., and Wu, W.-H. (2005). Air combat decision-making for cooperative multiple target attack using heuristic adaptive genetic algorithm. *Proceedings of the 4th International Conference on Machine Learning and Cybernetics* (pp. 473-478).
14. Mulgund, S., Harper, K., Krishnakumar, K., and Zacharias, G. (1998). Air combat tactics optimization using stochastic genetic algorithms. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics* (pp. 3136-3141).