A Behavior Optimization Method for Unmanned Combat Aerial Vehicles Using Matrix Factorization

JAESEOK HUH\textsuperscript{1}, JONGHUN PARK\textsuperscript{2}, (Member, IEEE), DONGMIN SHIN\textsuperscript{3}, (Member, IEEE), AND YERIM CHOI\textsuperscript{4}

\textsuperscript{1}Department of Business Administration, Korea Polytechnic University, Siheung-si 15073, South Korea
\textsuperscript{2}Department of Industrial Engineering and Institute for Industrial Systems Innovation, Seoul National University, Seoul 08826, South Korea
\textsuperscript{3}Department of Industrial and Management Engineering, Hanyang University, Ansan-si 15588, South Korea
\textsuperscript{4}Department of Industrial and Management Engineering, Kyonggi University, Suwon-si 16227, South Korea

Corresponding author: Yerim Choi (yrchoi@kgu.ac.kr)

This work was supported by the Gyeonggi-do Regional Research Center (GRRC) program of Gyeonggi province (Research on Industrial Data Analytics for Intelligent Manufacturing) under Grant GRRC KGU 2017-B01.

ABSTRACT

One of the fundamental technologies for unmanned combat aerial vehicles and combat simulators is behavior optimization, which finds a behavior that maximizes the probability of winning a battle. With the advent of military science, combat logs became available, allowing machine learning algorithms to be used for the behavior optimization. Due to implicit attributes such as the experience of an operator that are not explicitly presented in log data, existing methods for behavior optimization have limitations in performance improvement. Furthermore, specific behaviors occur with low frequency, resulting in a dataset with imbalanced and empty values. Therefore, we apply a matrix factorization (MF) method, which is one of latent factor models and known for sophisticated imputation of empty values, to the behavior optimization problem of unmanned combat aerial vehicles. A situation-behavior matrix, whose elements are ratings indicating the optimality of behaviors in situations, is defined to implement the MF based method. Experiments for performance comparison were conducted on combat logs, in which the proposed method yielded satisfactory results.

INDEX TERMS

Behavior optimization, unmanned vehicle, matrix factorization, reinforcement learning, situation-behavior matrix.

ABBREVIATIONS

AM Advantage matrix.
FOV Field of view.
GA Genetic algorithm.
LOS Line of sight.
MF Matrix factorization.
nDCG Normalized discounted cumulative gain.
RL Reinforcement learning.
SB Situation-behavior.
UV Unmanned vehicle.

I. INTRODUCTION

Recently, a significant problem arising in military science is the behavior optimization of unmanned combat aerial vehicles [1], [2]. Optimal behavior inference is identifying the most suitable behavior for a given situation, including a corresponding indicator. In the context of combat, the optimality of behavior is defined as the probability of winning a battle when a behavior for that situation is followed.

With the advent of data sensing and storing technologies, combat logs from actual combat or combat training become available for inference. These data are composed of state features describing a situation and behavior features collected by an unmanned vehicle (UV) operator involved in the situation. The state features are acquired from avionics equipment and include velocity, altitude, and the number of missiles.

Traditional optimization algorithms are difficult to use for the optimizing behavior of UV due to high computational complexity of those methods [3]. To overcome the limitations of the existing optimization algorithms, instead of attempting to obtain an exact solution, heuristic or learning-based algorithms were proposed in previous studies as summarized in Table 1.
TABLE 1. Previous studies on the inference of optimal behavior of UVs.

| Methods          | Characteristics                                                                 | References |
|------------------|---------------------------------------------------------------------------------|------------|
| Advantage matrix | Building the advantage matrix which represents the preference of behaviors       | [4], [5]   |
|                  | - Easy to interpret                                                            |            |
|                  | - High cost for building matrix                                                 |            |
| Genetic algorithm| Exploring the solution space to improve particular objectives                  | [6], [7], [8], [9] |
|                  | - Long time to obtain solutions                                                 |            |
|                  | - Local optimum                                                                |            |
| Reinforcement learning | Finding an optimal policy that maximize total future reward | [10], [11], [12] |
|                  | - Global optimum                                                               |            |
|                  | - Difficult to define reward for each behavior                                 |            |
| Matrix factorization | Estimating latent factors that explain the implicit attributes of state features | [13]       |
|                  | - Latent factor                                                                |            |
|                  | - Robust to the data sparseness                                                |            |

Advantage matrix (AM) based methods determine the optimal behavior by using a predefined situation-behavior (SB) matrix [4], [5], [14]. The results obtained by the methods are easy to interpret because optimal behavior is identified by comparing values of elements in the given AM. AM-based methods perform effectively when a sufficiently large simulation log is provided.

However, these methods require considerable costs to build the AM, and it is sometimes impossible to obtain information regarding optimal behaviors for every possible situation. Therefore, the SB matrix of the AM-based method contains several null elements. Moreover, the matrix becomes sparser as the quantities of situations and behaviors increase. For example, approximately one-quarter of the elements are null in an SB matrix with 50 situations and 6 behaviors, generated from the simulation log used in this study. Consequently, it is impossible to infer optimal behavior for such cases without information.

Another line of research addresses the optimal behavior inference problem using a genetic algorithm (GA), which improves their objectives by sufficiently exploring the solution space [6], [7], [15], [16]. Although GA did not require building a matrix, chromosome and crossover/mutation schemes should be predefined to perform the algorithm [8], [16], [14]. One of the limitations with this method is that the schemes adjusted for a specific domain are often challenging to apply to similar problems [9].

Several studies investigating reinforcement learning (RL), which aims to find an optimal policy, have been conducted [11], [12], [17]. In particular, Wang et al. [10] proposed the quantum probability model to determine the travel paths of multiple UVs. However, the existing RL-based methods were inappropriate for combat situations because the associated research focused on improving the trajectory of UVs. Furthermore, in actual combat or combat training, it is difficult to define a reward for every behavior in all situations, which is highly likely to degrade the performance of RL.

These existing methods do not consider implicit attributes that are not explicitly presented in state features. However, the performance of optimal behavior inference can be improved using latent factors such as the experiences of the operator [13]. For instance, Choi et al. [13] obtained satisfactory accuracy of fighter pilot behavior prediction using matrix factorization (MF), which is one of the most popular latent factor models.

MF has been applied to various domains, including the identification of muscle synergies [18], the recognition of facial expressions [19], cancer epigenomics analysis [20], and the analysis of human mobility patterns [21]. Furthermore, it has successfully addressed the problem of recommending an item to a user because it assumes that there are latent factors that explain user preferences for items [22], [23].

Motivated by these findings, in this paper, we propose an MF-based method for the problem of inferring optimal behavior. The proposed method is designed to uncover the latent attributes that explain observed patterns from previous behaviors to infer optimal behavior. Furthermore, due to the characteristics of MF that manage data sparseness [24], the proposed method is capable of maintaining inference performance when the amount of logged data is insufficient [25].

The proposed method is composed of four steps. First, situations are defined by analyzing the values of state features using a k-means algorithm. Second, the SB matrix is built from combat logs based on win and loss information. Third, null elements in the SB matrix are filled using an MF. Fourth, using the situations and modified SB matrix, the optimal behavior for a new situation is determined.

The two major novelties this study has with existing studies are: First, we attempt to perform a prior study on the optimization of human behaviors in the military domain. It is a well-known fact that finding the optimal behavior of a human is a difficult task as the behaviors are complex and hard to define [26], whereas defining combat behaviors of UVs is relatively viable since the actions are already defined through military documents such as the rules of engagement. Second, in a more practical view, we sought to contribute to the development of a key technology that can build an agent acting optimally in a battle as UVs are started to be used in a real-world situation with the advancement of military technologies.

This paper is organized as follows. In Section II, the behavior optimization problem for UVs is formally defined, and the
proposed MF based method is explained in detail. Experiment settings and performance comparison results of the proposed method are presented in Section III. Finally, the paper is concluded in Section IV.

II. BEHAVIOR OPTIMIZATION FOR UVs

In this section, a method for inferring the optimal behavior of an operator is introduced. The optimality of a behavior is defined as the probability of winning a battle when the behavior is performed. Accordingly, the objective of the proposed method is to determine the behavior that is most likely to win a battle for a given situation by analyzing previous logs.

In this paper, we attempt to find the optimal behavior of an operator for a given situation during a battle. However, defining a behavior is a challenging task due to the complexity human behaviors [26]. Therefore, we choose six behaviors based on the military documents such as the rules of engagement, and in order to mimic human behaviors, we define them as a form of probability, called intention. Moreover, the proposed method is only focused on the optimization of behaviors which is different from trajectory optimization. When the proposed method and the trajectory optimization are used together, fully automated UVs can participate in an actual battle.

A. PROBLEM DEFINITION

A set of combat logs is denoted by \( L = \{l_t\} \) for \( t = 1, \ldots, T \), where \( T \) is the total number of instances and \( l_t \) is the \( t \)-th instance composed of \( \vec{s}_t \), \( \vec{b}_t \), and \( w_t \). \( \vec{s}_t \) is a state vector which describes the state that a UV is facing at time \( t \) with diverse attributes such as velocity of the UV, distance between the UV and its target. \( \vec{b}_t \) is composed of \( s_{t,n} \) for \( n = 1, \ldots, N \), a value of the \( n \)-th attribute, where \( N \) is the total number of attributes.

\( \vec{b}_t \) is a behavior vector which describes the behavior taken at time \( t \) such as firing and turning. \( \vec{b}_t \) is composed of \( b_{t,m} \) for \( m = 1, \ldots, M \), which are binary values that indicate whether the \( m \)-th behavior has taken place at the \( t \)-th moment, where \( M \) is the total number of behaviors. \( b_{t,m} \) is defined by (1).

\[
b_{t,m} = \begin{cases} 1, & \text{if } m \text{-th behavior is performed at } t \text{-th moment,} \\ 0, & \text{otherwise.} \end{cases}
\] (1)

\( w_t \) represents the result of a behavior at the \( t \)-th moment during a battle, where 1 represents a win and 0 represents a loss. Note that all behaviors of an operator that took place during a battle are labeled as a win (loss) when the operator won (lost) the battle as it is impossible to accurately measure the contribution of each behavior on the final result of the battle.

The proposed method attempts to find \( \vec{b}_t \) for a given \( \vec{s}_t \), using the collected \( L \).

B. FINDING THE OPTIMAL BEHAVIOR USING MATRIX FACTORIZATION

We propose an MF based behavior optimization method to address the data sparseness and latent factor problems. The proposed method is composed of four steps as depicted in Fig. 1. First, the situation is defined by analyzing combat logs. Second, the initial SB matrix, which contains the number of observed winning instances for every combination of situations and behaviors, is built. Third, MF is conducted on the initial SB matrix to obtain a modified SB matrix. Finally, the optimal behavior of a new instance is determined by using the modified SB matrix.

1) SITUATION DEFINITION

To build an initial SB matrix, situations corresponding to the rows of the matrix need to be defined. In other words, it is necessary to discretize the state vectors of a battle, where diverse attributes describing a battle are continuously generated. We utilize a clustering approach to group the state vectors. Specifically, we adopt one of the most well-known method in clustering called k-means algorithm [27], which is widely used in diverse clustering problems [28]. The k-means algorithm partitions instances in a dataset into a given number of clusters using an Euclidean distance. In summary, a situation refers to a cluster of state vectors that are similar to each other.

In this study, the reason for using the k-means algorithm among various clustering methods is as follow. The process of the optimal behavior inference in the proposed method is designed based on the concept of clusters with a centroid and distances, meaning that the process assumes the existence of centroid in clusters and measures the distance between an instance and the centroids. Therefore, the most representative centroid based clustering algorithm, k-means algorithm, was adopted.

The objective function of the k-means algorithm is given in (2).

\[
J = \sum_{t=1}^{N} \sum_{k=1}^{K} i_{t,k} \delta(\vec{s}_t, \vec{c}_k),
\] (2)

where \( k \) indicate the situation index and \( K \) is the total number of situations. Moreover, \( i_{t,k} \) is an indicator variable describing
whether \( \tilde{s}_t \) is assigned to the \( k \)-th situation as defined by (3).

\[
\tilde{i}_{t,k} = \begin{cases} 
1, & \text{if } \tilde{s}_t \text{ is assigned to } k\text{-th situation}, \\
0, & \text{otherwise.}
\end{cases}
\]

(3)

\( \delta(\tilde{s}_t, \tilde{c}_k) \) measures the Euclidean distance between \( \tilde{s}_t \) and \( \tilde{c}_k \) as depicted in (4).

\[
\delta(\tilde{s}_t, \tilde{c}_k) = \sqrt{\sum_{n=1}^{N} (s_{t,n} - c_{k,n})^2},
\]

(4)

where \( \tilde{c}_k \) is a center vector of the \( k \)-th situation, which is the mean vector of \( \tilde{s}_t \) included in the \( k \)-th situation measured by Euclidean distance, whose element is \( c_{k,n} \) defined as (5).

\[
c_{k,n} = \frac{\sum_t i_{t,k} s_{t,n}}{\sum_t i_{t,k}}.
\]

(5)

The k-means algorithm attempts to minimize (2) by iteratively estimating \( i_{t,k} \forall t, k \) and \( \tilde{c}_k \forall k \) in the form of the expectation-maximization scheme.

Consequently, \( K \) situations are defined from the situation definition step based on \( \tilde{s}_t \forall t \). Situations are represented by \( \tilde{c}_k \forall k \), which indicate center vectors of situations. Specifically, \( \tilde{s}_t \) belongs to one of the situations, and the membership is represented by \( i_{t,k} \), which is set to 1 if \( \tilde{s}_t \) belongs to the \( k \)-th situation, and 0 otherwise.

2) SITUATION-BEHAVIOR MATRIX CONSTRUCTION

Using the defined situations, the initial SB matrix is developed. The initial SB matrix is denoted by \( R \) and composed of \( r_{k,m} \) called rating, which is the number of winning instances of the \( k \)-th situation and the \( m \)-th behavior. First, \( r_{k,m} \) is calculated using (6).

\[
r_{k,m} = \sum_{t=1}^{T} w_t b_{t,m} i_{t,k}.
\]

(6)

Note that to address the imbalance between the frequencies of behaviors, the rating is normalized across behaviors and situations.

Then, MF is applied to \( R \) to obtain the modified SB matrix denoted by \( \hat{R} \). The principle of matrix decomposition is illustrated in Fig. 2. As it is not possible to have combat logs for every situation and behavior, \( R \) contains many null values. MF decomposes \( R \) into a situation-factor matrix \( Q \) and a behavior-factor matrix \( P \) by using the known ratings, and the unknown ratings are estimated to build \( \hat{R} \) using \( Q \) and \( P \).

The estimation is performed by taking an inner product between \( \tilde{q}_k \) and \( \tilde{p}_m \) as in (7).

\[
\hat{r}_{k,m} = \tilde{q}_k \cdot \tilde{p}_m,
\]

(7)

where \( \tilde{q}_k \in \mathbb{R}^L \) and \( \tilde{p}_m \in \mathbb{R}^L \) are a situation-factor vector and a behavior-factor vector, respectively, and \( f \) is the number of latent factors considered in MF. The symbol ‘.‘ refers to the inner product between two input vectors.

3) OPTIMAL BEHAVIOR INFERENCE

In this section, we present how the proposed method detects the optimal behavior for a new state vector \( \tilde{s}_t' \) using \( \hat{R} \). Inference of the optimal behavior is composed of two steps: situation selection and behavior inference. In the situation selection step, multiple situations that are close enough to \( \tilde{s}_t' \) are selected for the inference in order to reduce the possibility of over-fitting, which might occur when only the closest situation is utilized for the inference. Then, the optimality of a behavior is measured and compared across the selected situations in the behavior inference step as depicted in Fig. 3.

We assume that the optimality of the \( m \)-th behavior in the \( k \)-th situation is proportional to \( r_{k,m} \), and it is inversely proportional to the distance between \( \tilde{s}_t' \) and \( \tilde{c}_k \).

MF is computed by minimizing (8).

\[
L = \sum_{k=1}^{K} \sum_{m=1}^{M} (r_{k,m} - \hat{r}_{k,m})^2 - \lambda \left( \sum_{k=1}^{K} |\tilde{q}_k|^2 + \sum_{m=1}^{M} |\tilde{p}_m|^2 \right),
\]

(8)

where \( \lambda \left( \sum_{k=1}^{K} |\tilde{q}_k|^2 + \sum_{m=1}^{M} |\tilde{p}_m|^2 \right) \) is a regularization term for preventing over-fitting and \( \lambda \) is a regularization parameter that depends on the dataset.

Parameters are learned using an alternating least squares with a weighted-lambda-regularization optimization process. This method factorizes \( R \) into \( Q \) and \( P \) by alternating between computing the situation-factor and the behavior-factor matrices. Further details are provided in [29].
First, situations are selected according to the Euclidean distance between $\vec{s}_t$ and $\vec{c}_k$ denoted by $d_{t,k} = \delta(\vec{s}_t, \vec{c}_k)$ as defined in (4). We denote the set of the indices of the selected situations as $I$. Specifically, the number of the selected situations is $\lfloor \rho K \rfloor$, where $\rho$ is the predefined selection ratio ranging from 0.0 to 1.0. Then, $o_{t,m}$, the optimality of the $m$-th behavior for $\vec{s}_t$, is calculated using (9).

$$o_{t,m} = \sum_{k \in I} r_{k,m} d_{t,k}.$$  
(9)

By comparing $o_{t,m} \forall m$, the index of the optimal behavior $\hat{m}$ is determined as the one with the highest value of $o_{t,m}$ obtained by using (10).

$$\hat{m} = \arg \max_m o_{t,m}.$$  
(10)

III. EXPERIMENT

A. DATASET

A simulation log from an air-to-air combat simulator was collected for performance evaluation of the proposed method. Fig. 4 illustrates snapshots of a simulation conducted on the simulator. The simulation log generated by the simulator was used because it is impossible to use actual combat logs for security reasons. For the same reason, we were not able to access any real-world data related to the actual UV. We assert that the performance of the proposed method on an actual log will be similar to the performance evaluated on the log collected from a simulator.

The simulator performs a simulation according to a scenario that contains information on the initial conditions of a battle. A scenario is composed of attributes such as the types of vehicles involved in a battle and the offensive and defensive armaments that the vehicles carry. The combat type was set to air-to-air and one-to-one; the field of view (FOV) was set to 60 degrees. Two aerial vehicles, a KF−16 and a MIG29, performed combat while carrying the same offensive and defensive armaments when the scenario was simulated. An example of the detail of a scenario is exhibited in Table 2, implying that the battle is between two fighters, KF−16 and MIG 29 with the same offensive and defensive weapons.

To obtain simulation logs from diverse situations, we used 113 scenarios, as depicted in Table 3. On average, each scenario was simulated for 100 seconds, with 37 behaviors performed by an operator. Note that the duration of each scenario is longer than the usual duration of a battle since we attempted to catch the moments before an actual battle begins such as the beginning of a flight, chasing, and approaching. This allows to consider the behaviors other than firing actions, such as turning direction and controlling velocity. The simulation log was composed of 54 state attributes, examples of which are presented in Table 4.

TABLE 2. Example of a scenario used to generate a simulation log.

| Combat type   | Air-to-air, one-to-one |
|--------------|------------------------|
| FOV          | 60                     |
| Radar range  | 4000                   |
| Fighter model| KF-16, MIG 29          |
| Offensive weapons | ABM120A, ABM120A |
| Defensive weapons | Chaff, Chaff |
| Results      | Our force finds enemy first and fires ABM120A |
|              | Our force evades missile |
|              | Both fighters fire AA11 |

TABLE 3. Summary of collected simulation log.

| Category     | Description                        | Value |
|--------------|------------------------------------|-------|
| Scenario     | Total number of scenarios          | 113   |
|              | Average duration of a scenario (s)  | 100.40|
|              | Average number of behaviors in a scenario | 37.59 |
| Attribute    | Total number of state attributes   | 54    |
| Behavior     | Total number of behaviors          | 6     |

TABLE 4. Examples of state attributes in terms of name and description.

| Attribute name     | Description               |
|--------------------|---------------------------|
| Aim intensity      | Aim intensity against a target |
| Altitude           | Altitude of vehicle       |
| Midrange missile measure | Quantity of midrange missiles |
| Move aim           | Aiming of a target        |
| Target altitude difference | Altitude difference from a target |
| Target distance    | Distance from a target    |
| Target speed difference | Speed difference from a target |
| Velocity           | Velocity of vehicle       |

Table 5 presents six behaviors, their behavior groups, and the number of occurrences of the behaviors in the collected dataset. The number of occurrences of behaviors in the turn group is smaller than in the fire and velocity groups. Specifically, turning vehicle means the change in the direction of a vehicle, and optimizations on both behavior and trajectory should be executed together for the actual implementation as the direction to change can be determined by a trajectory optimization method. Note that, unfortunately, we assumed that a situation has an effect on the selection of a behavior, while a behavior is not directly affecting a situation.
However, we believe that situations and behaviors are indirectly interacting as the changes in a situation is caused by the selection of the behavior occurred in the previous time.

### B. EXPERIMENT SETTING

For the performance comparison, we utilized two well-known methods for the optimization in similar problems, AM and RL. First, AM is widely utilized method for predicting the optimal behavior in the previous research, where a matrix is built by using logs of battles and the optimal behavior is derived by directly using the matrix. Specifically, we built a matrix for AM by the same way we built a matrix of MF. Second, RL is a method which is recently getting popular for inferring optimal behavior is not applicable to this problem because it requires labor-intensive tasks for adjusting the methods for different settings.

As a performance evaluation measure, the normalized discounted cumulative gain (nDCG) [31] was employed. nDCG is widely used for the evaluation of recommendation performance [32], [33]. Similar to the recommendation problem, multiple behaviors can be optimal for a situation. Therefore, rather than ignoring behaviors that are not ranked first, the ranking of behaviors based on optimality was considered for the performance evaluation using nDCG.

DCG is calculated using (11).

$$D CG = rel_1 + \sum_{i=1}^{M} \frac{rel_i}{log_2(i + 1)}, \quad (11)$$

where $rel_i$ is a relevance index calculated using (12).

$$rel_i = \begin{cases} 1, & \text{if the } i\text{-th ranked behavior occurs at the } t\text{-th moment,} \\ 0, & \text{otherwise.} \end{cases} \quad (12)$$

nDCG was modified not to assign the first- and second-ranked behaviors the same score. nDCG is obtained by dividing DCG by an optimal score for DCG; it varies from 0.0 to 1.0, with 1.0 representing the ideal ranking of the behaviors.

In addition to nDCG, we evaluated the time consumption of the behavior optimization in second. The behavior optimization require two steps for the optimization: training and prediction. The former indicates the process of learning the optimal matrix for behaviors and situations, and the latter means the process of finding the optimal behavior by analyzing the matrix. Therefore, the time consumption for the behavior optimization is the sum of training and prediction time.

To verify the effectiveness and robustness of the proposed method, performance comparison experiments between three methods, including the proposed method, were performed while varying $K$ and $\rho$. Note that $K$ indicates the number of clusters used for the situation definition. Moreover, $\rho$ indicates the ratio of situations utilized for the behavior prediction, making $\lfloor \rho K \rfloor$ the number of situations to be considered. We conducted experiments on $K$ ranged from 50 to 550 at an interval of 100 and $\rho$ ranged from 0.05 to 1.00 at an interval of 0.05. The number of latent factors for MF was fixed at 100.

Finally, the performance of the proposed method was evaluated by leave-one-out cross-validation for each scenario [34]. Given $n$ scenarios available in a dataset, behavior optimization methods are trained on $n - 1$ scenarios and then tested on the scenario that was excluded. This process is performed repeatedly $n$ times until every scenario in the dataset has been included once as a cross-validation instance [35]. All experiments were repeated five times to reduce the effect of randomness, and the performance results were averaged across the $n$ test scenarios.

### C. EXPERIMENT RESULT

1) **SITUATION DEFINITION**

In this section, the results of the situation definition are presented. State vectors in the simulation log were clustered into nine situations as an experiment example. Fig. 5 indicates the situation definition results by visualizing the center vectors of four situations with respect to seven state attributes.

In Fig. 5, Situation 2 exhibits high missile existence and very low missile distance, implying that a UV is being attacked in this situation. Comparing Situations 1 and 3, the values of the target-related attributes are large in Situation 1, whereas those in Situation 3 are very low, implying that Situation 1 is the beginning of combat, and Situation 3 is an engagement.

Meanwhile, all of the target-related values are very small, and all of the line of sight (LOS)-related values are large in Situation 4. LOS is defined as a straight line connecting the

### TABLE 5. Behavior groups and behaviors considered in experiments in terms of description and the number of occurrences.

| Behavior group | Behavior | Description | Number of occurrences |
|----------------|----------|-------------|-----------------------|
| Fire           | FM       | Raising intention of firing arms | 300                   |
|                | FL       | Reducing intention of firing arms | 1,243                 |
| Velocity       | VM       | Raising intention of increasing velocity | 2,039                 |
|                | VL       | Raising intention of decreasing velocity | 793                   |
| Turn           | TM       | Raise intention of turning vehicle | 124                   |
|                | TL       | Reducing intention of turning vehicle | 59                    |

VOLUME 8, 2020
instantaneous positions of the pursuer and evader represented by the angle and distance [36], [37]. Situation 4 indicates that the vehicle is attempting to evade the target vehicle by rapidly changing direction after engaging with the target vehicle.

2) SITUATION-BEHAVIOR MATRIX

Fig. 6 visualizes heatmaps representing the modified SB matrices estimated using RL and MF with the original matrix utilized for AM. Each box represents the evaluated rating of a behavior for a situation. The darker the color of the box, the closer the corresponding behavior is to the optimal behavior for the situation.

The two heatmaps in Figs. 6 (b) and (c) are estimated using the initial SB matrix in Fig. 6 (a), where the situations are ordered according to the optimality of VM by the initial SB matrix. As shown in Fig. 6 (b), although the matrix of RL contains more dark color boxes than those of AM, the results
generated by RL exhibited no significant difference from the initial distribution of behavior optimality. However, MF alters the initial values and provides a vastly different distribution. This phenomenon can be explained as the effect of latent factors that consider implicit interactions between the situation and behavior. For instance, the left parts of the behavior FM in the matrix in Fig. 6 (c) illustrate unique pattern. This pattern shows a different trend from the results of Fig. 6 (a) and is extremely dark compared to those in Fig. 6 (b). It indicates that MF was successful in mimicking the hidden pattern inherent in the SB matrix, where the color of elements in behaviors FM and FL tend to be reversed, whereas RL tends to strengthen the colors in the elements.

3) COMPARISON RESULT
Finally, the performance of the proposed method was evaluated and compared to those of AM and RL. Fig. 7 presents the comparison results for various values of \( K \). As shown in Fig. 7 (a), in terms of nDCG, MF outperformed RL and AM for every \( K \), and RL and MF exhibited similar levels of performance. Fig. 7 (b) presents the trend in the optimization time that indicates the sum of training and prediction time. The former and latter respectively mean the time consumption for learning the optimal matrix for behaviors and situations and finding the optimal behavior by analyzing the matrix. Although the optimization time of MF increased much more quickly than those of AM and RL as \( K \) increased,
the value remained sufficiently small for real-world usage, with a maximum of 180 seconds. Note that the prediction time of all methods was the same and very short, meaning that the methods are applicable to the real-world combat situation. The reason for the relatively small optimization time of RL is the adoption of simple Q-table approach as mentioned in Section III-B, which does not require the large amount of computation.

Furthermore, Fig. 8 illustrates the nDCG results of the proposed method, RL, and AM for various values of $K$ and $\rho$. AM and RL demonstrated the highest performance when $K = 150$, whereas MF demonstrated the highest performance when $K = 550$. When the number of situations was small, the three methods exhibited similar performance. However, the gap between MF and other methods widened as the number of situations increased because of the performance differences between methods when dealing with increased data sparseness.

As the value of $\rho$ becomes large, plots of all methods present increasing trends. In particular, the performance of MF was stable with respect to changes in the value of $\rho$ and exhibited a tendency to converge to a specific value. In contrast, severe fluctuations were observed in the results of RL, which suggests a significant deviation in the accuracy of the predicted rating for a behavior in each situation.

IV. CONCLUSION

For autonomous vehicles that operate without a human operator, developing automated operators that can perform as human operators is an important research topic. Specifically, inferring optimal behavior is an essential technology for performing goal-oriented tasks. Accordingly, a method for inferring optimal behavior using MF is proposed.

The optimality of a behavior is defined as its probability of producing a win during combat. Previously, AM-based methods were widely used for inference. However, the incompleteness of simulation logs that are unable to provide optimal behavior for every situation leads to the performance degradation of AM-based methods. Moreover, human behavioral decisions are based not only on explicit attributes such as velocity and missile measures but also on implicit attributes such as operator experience and expertise. Therefore, a well-established latent factor model, MF, is adopted for the problem.

The proposed method is composed of four steps. First, the situation is defined using a k-means algorithm. Second, the initial SB matrix is constructed by analyzing the simulation log. Third, the modified SB matrix is estimated using MF from the initial SB matrix. Fourth, the optimal behavior for the situation is calculated using the modified SB matrix. Experiments were performed on real-world datasets to evaluate the performance of the proposed method. Results demonstrated that MF successfully uncovers the latent factors that explain the observations from the modified SB matrix. Furthermore, MF outperformed AM and RL in terms of performance in most cases.

This study provides the following contributions. First, MF is adopted for the inference of the optimal behavior of UV operators. Second, the implicit decision made by the operator is modeled using latent factors based on MF, which is a well-established latent factor model.

We still leave some rooms for further improvements in the proposed method. First, the labeling process of the behavior optimality can be enhanced. We assumed that all behaviors occurred in a winning (losing) battle to be winning (losing) behaviors to provide training data for the proposed method. However, some sequences of behaviors during a winning battle may have a negative effect on the result of the battle. Therefore, instead of labeling all behaviors in a battle as win or loss, providing labels for a shorter sequence of behaviors is required to enhance the performances of the proposed method. Second, an unsupervised learning approach can be applied to discover the sequence of behaviors that is highly correlated to the result of a battle. Third, we will focus on developing a multi-view clustering method by designing more sophisticated distance measures and processes. Fourth, complicated combat situations where the number of UVs involved in a combat is larger than two will be investigated in future work. To achieve this, we plan to develop a multi-agent based framework for the inference of each agent’s optimal behavior. In addition, in order to practically validate the effectiveness of the proposed method, we hope to develop a new visualization method that can show the performances of the behavior optimization.

REFERENCES

[1] T. Oktay and S. Coban, “Simultaneous longitudinal and lateral flight control systems design for both passive and active morphing UAVs,” Elektronika ir Elektrotechnika, vol. 23, no. 5, pp. 15–20, Oct. 2017.
[2] O. O. Kanat, E. Karatay, O. Kose, and T. Oktay, “Combined active flow and flight control systems design for morphing unmanned aerial vehicles,” Proc. Inst. Mech. Eng. G, J. Aerosp. Eng., vol. 233, no. 14, pp. 5393–5402, Nov. 2019.
[3] C. Hartland and N. Bredeche, “Evolutionary robotics, anticipation and the reality gap,” in Proc. IEEE Int. Conf. Robot. Biomimetics, Dec. 2006, pp. 1640–1645.
[4] F. Austin, G. Carbone, M. Falco, H. Hinz, and M. Lewis, “Automated maneuvering decisions for air-to-air combat,” in Proc. AIAA, 1987, pp. 2387–2393.
[5] J. Gao and M.-A. Tong, “Modeling and solution of scheduling problem and fire allocations for multi-targets BVR attacking,” Fire Control Command Control, vol. 29, no. 3, pp. 12–16, 2004.
[6] T. Zhang, L. Yu, X.-Z. Wei, and Z.-L. Zhou, “Decision-making for cooperative multiple target attack based on adaptive pseudo-parallel genetic algorithm,” Huo li yu Zhihui Kongshe, vol. 38, no. 5, pp. 137–140, 2013.
[7] S.-C. Lee, H.-D. Kim, and J. Suk, “Collision avoidance maneuver planning using GA for LEO and GEO satellite maintained in keeping area,” Int. J. Aeronaut. Space Sci., vol. 13, no. 4, pp. 474–483, 2012.
[8] E. Edison and T. Shima, “Integrated task assignment and path optimization for cooperating uninhabited aerial vehicles using genetic algorithms,” Comput. Oper. Res., vol. 38, no. 1, pp. 340–356, Jan. 2011.
[9] Y. Zhang, C. Wang, X. Gu, and J. Chen, Cooperative Trajectory Planning for Multiple UAVs Using Distributed Receding Horizon Control and Inverse Dynamics Optimization Method. Cham, Switzerland: Springer, 2017, pp. 39–53.
[10] T. Wang, R. Qin, Y. Chen, H. Snoussi, and C. Choi, “A reinforcement learning approach for UAV target searching and tracking,” Multimedia Tools Appl., vol. 78, no. 4, pp. 4347–4364, 2018.
[11] M. De Paula and G. G. Acosta, “Trajectory tracking algorithm for autonomous vehicles using adaptive reinforcement learning,” in Proc. MTS/IEEE Washington (OCEANS), Oct. 2015, pp. 1–8.

[12] R. Yu, Z. Shi, C. Huang, T. Li, and Q. Ma, “Deep reinforcement learning based optimal trajectory tracking control of autonomous underwater vehicle,” in Proc. 56th Chin. Control Conf. (CCC), Jul. 2017, pp. 4958–4965.

[13] Y. Choi, S. Jeon, J. Park, and D. Shin, “A behavior optimization for fighter pilots using matrix factorization,” in Proc. Korean Soc. Aeronaut. Space Sci. Spring Conf., 2014, pp. 812–814.

[14] L. Fu, F. Xie, D. Wang, and G. Meng, “The overview for UAV air-combat decision method,” in Proc. 26th Chin. Control Decis. Conf. (CCDC), May 2014, pp. 3380–3384.

[15] L. Lin, Y. Seo, M. Gen, and R. Cheng, “Unusual human behavior recognition using evolutionary technique,” Comput. Ind. Eng., vol. 56, no. 3, pp. 1137–1153, Apr. 2009.

[16] L. Xunke, J. Huijun, and Z. Xun, “Human action tracking design of neural network algorithm based on ga-pso in physical training,” Cluster Comput., vol. 22, pp. 4149–4155, Jan. 2018.

[17] T. Zhang, Q. Li, C.-S. Zhang, H.-W. Liang, P. Li, T.-M. Wang, S. Li, Y.-L. Zhu, and C. Wu, “Current trends in the development of intelligent unmanned autonomous systems,” Frontiers Inf. Technol. Electron. Eng., vol. 18, no. 1, pp. 68–85, Jan. 2017.

[18] M. C. Tresch, V. C. K. Cheung, and A. d’Avella, “Matrix factorization algorithms for the identification of muscle synergies: Evaluation on simulated and experimental data sets,” J. Neurophysiol., vol. 95, pp. 2199–2212, Apr. 2006.

[19] R. Zhi, M. Flierl, Q. Ruan, and W. B. Kleinj, “Graph-preserving sparse nonnegative matrix factorization with application to facial expression recognition,” IEEE Trans. Syst., Man, Cybern. B, Cybern., vol. 41, no. 1, pp. 38–52, Feb. 2011.

[20] Z. Ma, A. E. Teschendorff, A. Leijon, Y. Qiao, H. Zhang, and J. Guo, “Variational Bayesian matrix factorization for bounded support data,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no. 4, pp. 876–889, Apr. 2015.

[21] C. Peng, X. Jin, K.-C. Wong, M. Shi, and P. Li, “Collective human mobility pattern from taxi trips in urban area,” PLoS ONE, vol. 7, no. 4, Apr. 2012, Art. no. e34487.

[22] B. Yi, X. Shen, H. Liu, Z. Zhang, W. Zhang, S. Liu, and N. Xiong, “Deep matrix factorization with implicit feedback embedding for recommendation system,” IEEE Trans. Ind. Informat., vol. 15, no. 8, pp. 4591–4601, Aug. 2019.

[23] L. Luo, H. Xie, Y. Rao, and F. L. Wang, “Personalized recommendation by matrix co-factorization with tags and time information,” Expert Syst. Appl., vol. 119, pp. 311–321, Apr. 2019.

[24] N. Chen, Y. Liu, and H.-C. Chao, “Overlapping community detection using non-negative matrix factorization with orthogonal and sparseness constraints,” IEEE Access, vol. 6, pp. 21266–21274, 2018.

[25] F. Mosayed, Z. Azimifar, and R. Boostani, “Human action recognition: Learning sparse basis units from trajectory subspace,” Appl. Intell. Intell., vol. 30, no. 4, pp. 297–317, Apr. 2016.

[26] N. Li, H. Kong, Y. Ma, G. Gong, and W. Huai, “Human performance modeling for manufacturing based on an improved KNN algorithm,” Int. J. Adv. Manuf. Technol., vol. 84, no. 1–4, pp. 473–483, Apr. 2016.

[27] J. A. Hartigan and M. A. Wong, “Algorithm AS 136: A k-means clustering algorithm,” Appl. Statist., vol. 28, no. 1, pp. 100–108, 1979.

[28] R. J. Kuo and P. S. Li, “Taiwanese export trade forecasting using firefly algorithm based K-means algorithm and SVR with wavelet transform,” Comput. Ind. Eng., vol. 99, pp. 153–161, Sep. 2016.

[29] Y. Hu, Y. Koren, and C. Volinsky, “Collaborative filtering for implicit feedback datasets,” in Proc. IEEE Int. Conf. Data Mining, Jan. 2008, pp. 263–272.

[30] B. H. Abdel-Algami, S. K. Chalup, F. A. Henskens, and D. J. Paul, “A multi-agent cooperative reinforcement learning model using a hierarchy of consultants, tutors and workers,” Vietnam J. Comput. Sci., vol. 2, no. 4, pp. 213–226, Nov. 2015.

[31] C. D. Manning, P. Raghavan, and H. Schütze, Introduction to Information Retrieval. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[32] M. He, B. Wang, and X. Du, “H2REC: Exploring knowledge in heterogeneous information for movie recommendation,” IEEE Access, vol. 7, pp. 30276–30284, 2019.

[33] Y. Jiang, Y. Zhu, X. Du, and T. Jin, “Improved metric-based recommender by historical interactions,” IEEE Access, vol. 7, pp. 125969–125975, 2019.

[34] G. Gong, “Cross-validation, the jackknife, and the bootstrap: Excess error estimation in forward logistic regression,” J. Amer. Stat. Assoc., vol. 81, no. 393, pp. 108–113, Mar. 1986.

[35] S. Y. Teh, A. R. Othman, and M. B. C. Khoo, “Dichotomous logistic regression with leave-one-out validation,” World Academy Sci., Eng. Technol., vol. 62, pp. 994–1003, Feb. 2010.

[36] V. Sunkara and A. Chakravarty, “Collison avoidance laws for objects with arbitrary shapes,” in Proc. IEEE 55th Conf. Decis. Control (CDC), Dec. 2016, pp. 5158–5164.

[37] M. Pelosi, C. Kopp, and M. Brown, “Range-limited UAV trajectory using terrain masking under radar detectability,” Appl. Intell., vol. 26, no. 8, pp. 743–759, Sep. 2012. 

JAESEOK HUH received the B.S. and Ph.D. degrees in industrial engineering from Seoul National University (SNU), South Korea, in 2013 and 2019, respectively. He is currently an Assistant Professor with the Department of Business Administration, Korea Polytechnic University, South Korea. His current research interests include semantic data mining, machine learning, and deep reinforcement learning.

JONGHUN PARK (Member, IEEE) received the B.S. degree in industrial and systems engineering with a minor in computer science from the Georgia Institute of Technology, Atlanta, GA, USA, in 2000. He was an Assistant Professor with the School of Information Sciences and Technology, Pennsylvania State University, University Park. He was an Assistant Professor with the Department of Industrial Engineering, KAIST, Daejon. He is currently a Professor with the Department of Industrial Engineering, Seoul National University (SNU), South Korea. His research interests include combinatorial optimization and deep learning applications.

DONGMIN SHIN (Member, IEEE) received the B.S. degree in industrial engineering from Hanyang University, Seoul, South Korea, in 1994, the M.S. degree in industrial engineering from the Pohang University of Science and Technology (POSTECH), Pohang, South Korea, in 1996, and the Ph.D. degree in industrial engineering from Pennsylvania State University, University Park, in 2005. He is currently a Professor with the Department of Industrial and Management Engineering, Hanyang University, Ansan, South Korea, where he is also the Director of the Smart Manufacturing Learning Center. His research interests include human–automation interactions, smart manufacturing systems, and machine learning.

YERIM CHOI received the B.S. and Ph.D. degrees in industrial engineering from Seoul National University (SNU), South Korea, in 2010 and 2016, respectively. She is currently an Assistant Professor with the Department of Industrial and Management Engineering, Kyonggi University, South Korea. Her current research interests include human modeling, smart manufacturing, AI applications, and deep learning.

* * *