Optimal opponents search scheme in spatial games: Multi-attribute similarity analysis based on Euclidean distance

Jongwan Kim
Smith College of Liberal Arts, Sahmyook University,
815 Hwarang-ro, Nowon-gu, Seoul, 01795, Korea
kimj@syu.ac.kr
http://www.syu.ac.kr

Abstract - With the development of mobile and GPS technologies, gamers in spatial games based on a real map or a virtual game map play against each other to determine winner and loser of the game. To improve their winning rate, gamers want to play against weak opponents, but it is limited to a certain degree for gamers to choose weak opponents. In general, the game server shows the attributes of other gamers around the current gamer, such as their game level, power, and difficulty. This paper proposes an optimal opponent search technique that allows game servers to search and show possible game opponents around a gamer by comparing their attributes. For comparison of possible game opponents, gamers and their game opponents are evaluated by Euclidean distance. Experiments were performed focusing on searching for game opponents with a danger level equal to or lower than that of the gamer. We measured the similarity between gamers and neighboring opponents based on virtual data to provide an optimal game opponent evaluation that could increase the winning rate of gamers.

***Keywords*: Spatial Game; Euclidean Distance; Similarity; Dissimilarity.**

1. Introduction

With the development of GIS and mobile technology, online games are evolving to spatial games using real maps as well as existing game maps. There are many tries to upgrade the process of games for wining in engagement on GIS-based or spatial games [1][2][3]. In virtual spaces, gamers increase their winning rate by playing against various opponents using their game skills, weapons, power, and items. If gamers know information about all game objects in advance, they can increase their winning rate by choosing to play against weak opponents. However, it is difficult for a gamer to get the information about game opponents.

Gamers and game opponents have attributes such as game level, skills, weapons, power or items. Gamers who have higher attributes than those of opponents are considered to have a high winning probability. If the game server compares the attributes of game objects and gamers and then recommends opponents that gamers can win, gamers will select a game opponent based on this information. Furthermore, in order to form a team of gamers with similar skills in a game, it is important to evaluate the similarity of attributes between gamers.

Similarity analysis is a key technique of data mining [4]. It is used to search objects or web pages based on Euclidean distance, which measures the distance between two points. In object search, the one-dimensional attributes of two objects are compared by distance and if the result converges to zero, they are considered to be similar. For web pages, the numbers of searched words are compared to evaluate the similarity between pages.

In this study, we propose an optimal game opponent search technique to select a game opponent who is easy to win in spatial games by measuring the similarity between gamers and game objects that have multiple attributes by using the above characteristic of Euclidean distance.

The game server recommends game opponents to gamers by comparing the multi-attributes of game objects participating in the game with those of gamers and ranking the game objects. When game attributes are compared, gamers may have higher or lower values than those of game opponents. Therefore, opponents with a high winning rate cannot be determined only by similarity between two objects. Reflecting this characteristic of similarity, this study also analyzes the opponents to be avoided when the similarity is the same.

The optimal opponent search technique based on Euclidean distance proposed in this study has a high demand for application in games, but it is difficult to find similar study cases. Therefore, the present study is pioneering a new field of study in games and makes the following contributions from research and practical aspects.

- Comparison and search for opponents with high similarity: Euclidean distance is being used in various fields of research including data mining and artificial intelligence. This study introduces a new application method for Euclidean distance by applying it to games as a new method of comparing the attributes of gamers.

- Expansion of the research fields of Euclidean distance: This study expands the research field by applying the existing Euclid distance method for searching the optimal game opponent.
- Evaluation of winning rate according to the direction of similarity: This study implemented a method of reevaluating game opponents by converting the dissimilarity of Euclidean distance to dissimilarity and analyzing the difference in the sum of attributes when the similarities are equal.

This paper is organized as follows. Section 2 explains the Euclidean distance and the similarity conversion method. Section 3 proposes a similarity evaluation method for game objects to search the optimal game opponents. Section 4 performs an experiment to search the nearest game opponents with virtual game data, performs visualization for similarity and analyzes proximity. Section 5 discusses the intention and research contributions of this study.

2. Preliminaries

The similarity of two objects is based on the difference of attribute values. In this section, we discuss how to calculate the proximity of two objects using Euclidean distance and how to convert distance-based proximity to similarity.

2.1. *Attributes of objects and Euclidean distance*

The proximity between two objects is measured by the relative difference of the other object from the position of one object. Not only the position, but also numerical attributes such as the frequency of texts and preferences can be also compared to determine proximity, thereby evaluating the similarity (s) of two objects [5]. Similarity is generally determined by agreement. If the attributes of two objects agree with each other, s=1; otherwise, s=0. Thus, similarity has the range of a closed interval of [0, 1] [6].

The distance between two points based on position is calculated by Euclidean distance. A larger difference between two objects indicates that they are farther away from each other. This is called dissimilarity (d) because it has an opposite meaning to similarity, which is higher when two objects are closer.

Unlike similarity, when two objects *p* and *q* are at the same position, *d(p,q)=0*. As the distance between the two objects become farther, the value of *d* becomes 1 or larger. Hence, it is indicated as an open interval [0, 1) [6]. As shown in Figure 1, Euclidean distance is based on the Pythagorean Theorem and indicates the distance between two points. In the Pythagorean Theorem, the length of the hypotenuse equals c2 = a2 + b2 [7]. Therefore, the distance between two points *p* and *q* can be expressed as Eq. (1), where d denotes distance.

$$d(p,q)=\sqrt{(x2-x1)^2+(y2-y1)^2}\,. \tag{1}$$

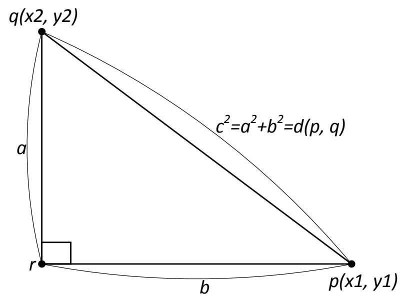


Fig. 1. Hypotenuse length of a right triangle.

In Figure 1, each of the two points *p* and *q* has two attributes *x* and *y*, and the distance between the two points can be calculated by the square of the difference between the attributes of the same dimension at each point.

Objects generally have multi-attributes, and can be categorized into two or higher dimensions. When Euclidean distance is extended to attributes of two or higher dimensions, it can be expressed by the sum of the same attributes of each object as follows:

$$d(p,q)=\sqrt{\sum_{i=1}^{n}(\mathrm{p}_i-\mathrm{q}_i)^2}\,. \tag{2}$$

2.2. *Similarity conversion*

Dissimilarity is calculated by the difference in distance, and a large value of dissimilarity between two objects indicates that they are not similar. Thus, a small value indicates that the two objects are close to each other or their attributes are similar. The similarity of two attributes can be understood more intuitively than dissimilarity because similarity between two objects is based on proximity [4].

Because similarity and dissimilarity are in inverse relationship with each other, similarity can be expressed as follows:

$$s = 1/\left(1+d\right)\ (s:\ similarity,\ d:\ dissmilarity). \quad (3)$$

3. Evaluation of Similarity of Game Objects to Search the Optimal Game Opponent

In this section, we explain the optimal opponent search technique, which finds and recommends the opponents with similar game level or skills around gamers to increase the winning rate of gamers.

3.1. *Multidimensional attributes and Euclidean distance*

The dissimilarity between gamers and their opponents is measured by comparing attributes. The ranges of game attributes do not include negative values, and larger positive values are associated with a higher winning rate. Therefore, game attributes follow Heuristic 1.

Heuristic 1. *Higher game attributes have a higher relative advantage in winning rate.*

The attributes of gamers such as level, power, skills, and strength are discharged or recharged through training and items during games. Since higher values of attributes indicate higher levels and skills, the higher values of attributes are more advantageous for winning rate.

Table 1 lists the attributes of gamer g and opponent obj1. Each attribute has a range of [1, 100]. Gamer g has a higher winning rate because he/she has a higher level attribute than that of obj1, but obj1 has a higher power than that of obj1. As shown in this example, gamers may have some attributes that are higher, or other attributes that are lower than those of opponents. Because a high winning rate cannot be guaranteed by any one attribute alone, all attributes must be evaluated.

Table 1. Attributes of gamer and opponent.

Category	g(amer)	obj1	obj2	obj3	obj4	obj5	obj6
Level	50	27	16	83	72	28	72
Power	70	97	68	64	86	56	84
Skills	20	48	58	12	22	1	39
Strength	30	40	9	4	1	10	50

The dissimilarity of gamers g and obj1 is calculated by Eq. (4), which is 46.282. Table 2 lists the dissimilarity of each object. Proximity indicates a difference in the attributes of gamers and their surrounding objects. A lower dissimilarity value indicates an object with more similar attributes as those of the gamer.

$$d(g, obj1) = \sqrt{(50-27)^2 + (70-97)^2 + (29+48)^2 + (30-40)^2} = 46.282 \quad . \quad (4)$$

Dissimilarity is less intuitive than similarity in terms of the meanings of the term and value, and has an open range of [0, 1). Thus, dissimilarity can be a large value and has a lower consistency than similarity having a range of [0, 1]. Therefore, to maintain consistency regarding the proximity of objects, dissimilarity should be converted to similarity. For similarity conversion between gamers and opponents, intuitive conversion is guaranteed as shown in Heuristic 2.

Heuristic 2. *The similarity conversion for differences of attributes does not have meaning loss.*

If the attribute of the object for which dissimilarity was measured is an ordered sequence type or has a correlation with proximity, the order or interval depending on the size of the value or the sign for positive or negative correlation can be changed by similarity conversion. However, for proximity using the difference in game attributes, the importance of attributes is not changed even if it is converted to similarity. In other words, the attribute that converged to 0 in dissimilarity converges to 1 in similarity; thus, the characteristic of the closer to the gamer, the higher the similarity is maintained in the same way.

In Table 2, the dissimilarity between gamer g and opponent obj corresponds to the open interval [0, 1). Because a smaller value indicates a higher proximity, obj5 and obj6 are opponents with the closest attributes to g. When dissimilarity is converted to similarity with a range of [0, 1], it becomes the similarity s in Table 2. The similarity of obj5 with the lowest dissimilarity is 0.026, converging closer to 1 than other objects. Therefore, the opponent obj5 can be considered to have a skill similar to that of gamer *g*. However, obj6 with the same similarity is a disadvantageous opponent for g because obj6 has higher attributes than those of *g*. Similarity conversion is calculated by Eq. (3). This conversion does not cause meaning loss because the direction of difference between the gamer and each object stays the same after conversion.

Table 2. Dissimilarity and similarity of game opponents.

Category		obj1	obj2	obj3	obj4	obj5	obj6
Dissimilarity	d(g, obj$_n$)	46.282	55.182	43.186	39.812	37.961	37.961
Similarity	s=1/(1+d)	0.021	0.018	0.023	0.025	0.026	0.026

In Table 2, obj5 and obj6 with the highest similarity can be considered as the matchable opponents because they have similar attributes to those of gamer *g*. However, the result of *g's* game with obj5 or obj6 depends on the difference in skills or type of skills used. In particular, gamer g has a high probability of losing to obj6 because he/she has higher attributes than those of gamer g even though his/her similarity is the same as that of obj5.

The same similarity for different attributes can be explained by Heuristic3 because dissimilarity and similarity are based on differences in attributes,

Heuristic 3. *The winning rate of the same similarity varies by the difference in attributes.*

Because dissimilarity is calculated as the absolute value of a difference in two attribute values, if the absolute value is identical, the similarity is the same. However, if the difference in the attributes of the gamer and opponent is a positive value, the gamer has a higher winning rate; if it is a negative value, the opponent has a higher winning rate. Therefore, when searching for an opponent among opponents with the same similarity, it must be determined by the sign of the difference in attribute value.

In Table 2, obj5 and obj6 have the same similarity, 0.026. The opponent who should be avoided in the game can be determined by making a judgment about the relationships of the gamer with obj5 and obj6 based on the difference in the sum of attributes.

Table 3 shows differences in the attribute values between the gamer and opponents. Opponents obj2, obj3, and obj5 are weaker opponents than the gamer because they have positive differences with the gamer. However, obj6 is considered a stronger opponent than gamer because obj6 with the same similarity has the smallest negative value. Figure 2 lists the sums of four attributes of each game object in the order of size, and a larger value indicates a higher winning rate.

As explained in Heuristic 3, when the sum of attributes of gamer g is 170, opponents obj5 and obj6 have the same similarity of 0.026. However, since the relationship of the sum of attributes is obj5 < *g* < obj6, obj6 has a higher winning rate than gamer *g*.

Table 3. Differences between gamer, *g* and opponents, *objs*.

Category	g(amer)	obj1	obj2	obj3	obj4	obj5	obj6
Sum of attributes	170	212	151	163	181	95	245
Difference between attributes	0	-42	19	7	-11	75	-75

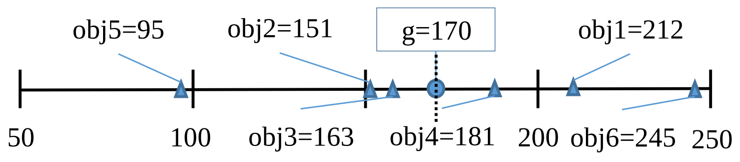


Fig. 2. Attribute positions of game objects.

3.2. *Dissimilarity and similarity algorithm*

To measure similarity in this study, we must first generate game opponents, an experimental data set, and calculate their dissimilarities. Next, we must convert the dissimilarities of gamer and opponents to similarities.

Because virtual data are used for the attributes of gamer and opponents, the gamer's attributes are specified as fixed values, whereas the attributes of the opponents are generated randomly using an object generator.

As shown in Figure 3, for the virtual data set, the attributes of each object are generated by using the *randrange()* function, which generates random numbers of standard distribution after receiving the number of objects to be generated (line 6). The algorithm manages objects as list type, and *lstObject* and *lstGameObjects* represent one opponent and all opponents, respectively. The line 9 and line 10 remove duplicate values so that each object would have unique attributes. The objects generated with unique attributes are stored in *lstGameObjects* in line 12 and this list object is returned finally.

```
Algorithm: Generate target game objects of 100K.
- Input: number, # of target gamers to be compared to gamer.
- Output: lstGameObjects, the target gamers as a list type.
01: def gen_gameobjects(number):
02: intLevel, intPower, intSkill, intStrength = (0,0,0,0) # initialize by 0
03: lstGameObjects, lstObject = (list(), list()) # Target game objects
04:
05: for count in range(0, number):
06:  intLevel, intPower, intSkill, intStrength = Initialize each variable
07:  with random.randrange(1, 100+1)
08:  lstObject = [intLevel, intPower, intSkill, intStrength]
09:  if(lstObject in lstGameObjects): #Discard the duplicated objects
10:  Re-generate the four properties and update lstObject.
11:  lstGameObjects.append(lstObject)
12:  return lstGameObjects
```

Fig. 3. Game opponent generation algorithm.

For the opponent objects generated randomly, dissimilarities are calculated using the Euclidean distance in order to determine proximity to the gamer. Figure 4 shows the dissimilarity calculation algorithm. The argument *arg* is used to receive two or more arguments in Python and one gamer and one opponent are received. The *zip()* function line 6 combines every two attributes of two objects that have been received as input for each dimension and are saved in the gamer and object. The objects delivered to arg have the same number of attributes. In line 7, the dissimilarity of two objects in each dimension is calculated and returned through distance.

```
Algorithm: Compute a dissimilarity between a gamer and a target object.
- Input: *args, gamer and target object.
- Output: distance, dissimilarity of gamer and target object
01: def euclidean_distance(*args):
02:   numberofargs = len(args)
03: if(numberofargs < 2 or numberofargs >=3): # Check the number of arguments
04:  print('Errors’)
05:  return -1
06:  for gamer, object in zip(*args) : # The first argument is a gamer.
07:  distance = math.sqrt(sum((gamer - object) ** 2))
08:  return distance
```

Fig. 4. Gamer and opponent dissimilarity algorithm.

Figure 5 shows the algorithm for converting dissimilarity calculated as the Euclidean distance to similarity. This algorithm calculates similarity using Eq. (3). dicSimilarity in line 3 is a dictionary object for storing similarity to the gamer by using the attributes of opponents as key. The attributes of the gamer are initialized to fixed values as shown in line 4. The number of opponents to be generated randomly is input and the object creation algorithm is called in line 6. In line 8, the gamer and opponents are delivered to the algorithm for calculating the Euclidean distance to calculate the dissimilarity for each opponent, which is converted to similarity in line 9. In line 10, the *list* type is converted to tuple type to store opponents in the dictionary type of Python. In line 11, the attributes and similarities of each opponent are stored in dictionary type as {*opponent*:*similarity*} which is a pair of key and value.

```
Algorithm: Compute a similarity between gamer and target objects.
- Input: none
- Output: a dictionary object structured in {target object:similarity}
01: similaty_two_objects()
02:  lstTargetObjects = list()
03:  dicSimilarity = dict()
04:  gamer = [50, 70, 20, 30] # Initialize a gamer’s properties.
05:  number = # of target game objects indicated by user
06:  lstTargetObjects = gen_gameobjects(number) # Generate target gamers
07:  for count in range(0, number):
08:  dissimilarity = euclidean_distance(gamer, lstTartgetObjects[count])
09:  similarity = 1/(1 + dissimilarity) # Convert to similarity
10:  TargetObject = tuple(lstGameObjects[count]) # Should be tuple for dict type
11:  dicSimilarity[TargetObject] = round(similarity, 5) # roundup from six position to fifth one.
12:  print(dicSimilarity)
```

Fig. 5. Dissimilarity to similarity conversion algorithm.

To evaluate the optical opponent, we implemented a game opponent generator, an algorithm for dissimilarity calculation using Euclidean distance, and a similarity conversion algorithm. As shown in these algorithms, the Euclidean distance determines proximity regardless of the number of dimensions in the attributes of two objects. The higher or lower position of winning rate with the same similarity in the attributes of game objects is clearly distinguished. Therefore, the difference in the sum of attributes must be determined besides similarity in order to determine whether or not a gamer can win an opponent.

4. Simulation of Euclidean Similarity

This study proposes a technique for searching optimal opponents by evaluating similarity between gamers with multiple attributes. In this section, we perform a simulation on the evaluation of opponents focusing on the differences of attributes and analyze the results.

4.1. *Virtual data and experiment environment*

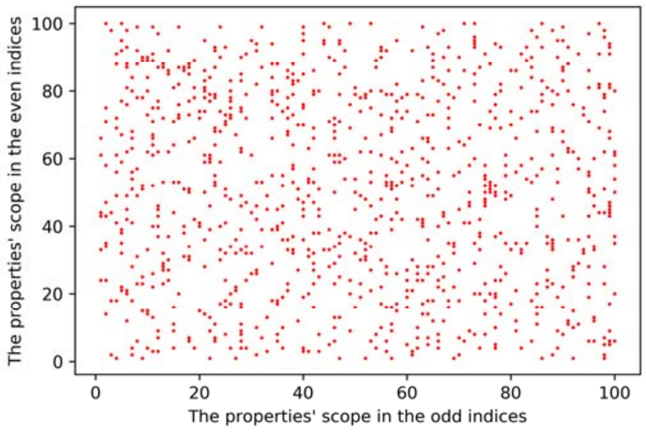
In this study, 100K opponents are evaluated for their similarity to the gamer. The attribute values of each object were generated randomly using an object generator. There are four attributes of objects: level, power, skills, and strength. The values of each attribute have a standard distribution of the integers from 1 to 100. In the comparison of similarity for attributes, a larger value has a priority over a smaller value.

The algorithms were implemented in Jupyter notebook [8] on Linux using Python. Each attribute was created as a list type object. Table 4 outlines the development environment for this experiment.

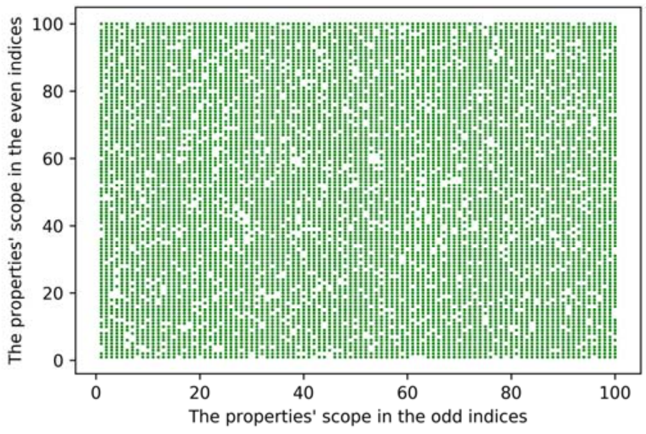
Table 4. Experiment environment.

Category	Contents
Gamer	Main gamer comparing with opponent objects. Gamer’s attributes = [50, 70, 20, 30]
Dataset(Opponent Object)	100K, Attributes’ scope: 1-100
Data Distribution	Uniform
Attributes of objects	Game Level, Power, Skills, and Strength
System	Fedora 30, Xeon 2 CPU, RAM 128 GB
Programming language and libraries	Python 3.x, Matplot, Numpy

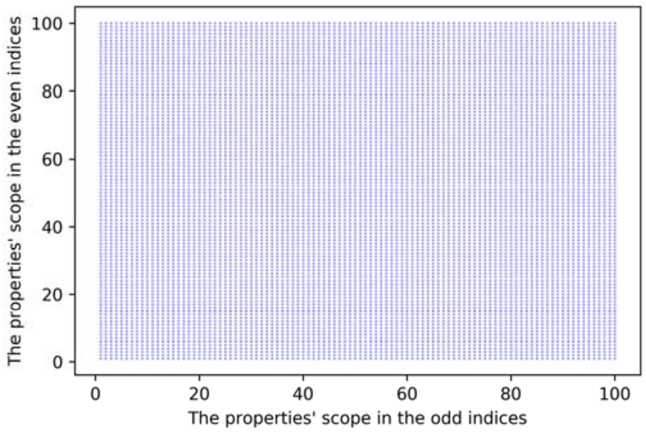
The attribute values were generated in a standard distribution using the *randrange()* function of Python. Figure 6 shows the distribution of game objects. To visualize the game attributes generated as one-dimensional values, a scatter graph was created by integrating the attributes of odd-numbered objects into one list, which was used as the y-axis, and using the attributes of the even-numbered objects as x-axis. Figure 6 (a), (b) and (c) show the standard distributions for 500, 10K, and 100K opponents, respectively.



(a) 500 opponents.



(b) 10K opponents.



(c) 100K opponents.

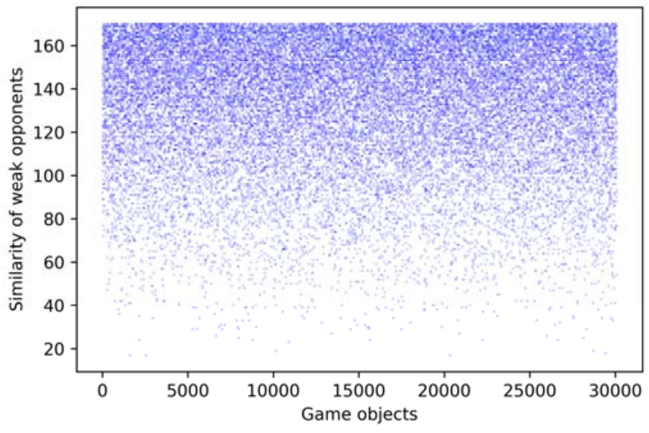
Fig. 6. Distributions of opponents' attributes.

4.2. *Similarity and winning rate*

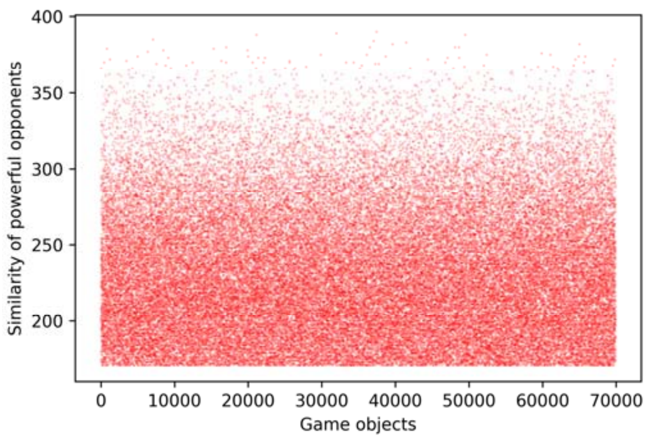
In this study, similarity indicates whether the opponent and gamer are matchable. A similarity closer to 1 means that the two objects have similar skills. However, as shown in Heuristic 3, the similarity between game objects have positive or negative direction. The opponent can be advantageous or disadvantageous for the gamer depending on the difference in the sum of attributes between the gamer and opponent. If the difference is a positive value, the gamer has a higher winning rate; if it is a negative value, the opposite is the case. A similarity value close to 1 does not always mean that the gamer is advantageous. In other words, it is difficult to determine the possibility of winning in a game based on similarity alone.

Among the 100K virtual opponents, the number objects having a lower winning rate than that of gamer g is 30,087 and the number of objects having a higher winning rate is 69,913. Figure 7 (a) and (b) show similarity distributions for opponents with lower and higher winning rates than that of the gamer, respectively. In Figure 7 (a), as the winning rate converges to 0.025, the sum of attributes becomes smaller than that of the gamer and their similarity becomes lower. Hence, the gamer has a high winning rate in actual games. As the similarity approaches 0.175, it means a high similarity for four attributes. Therefore, the opponent is a more difficult to play against than the opponent with a similarity of 0.025. In Figure 7 (b), when the similarity is 0.025, the winning rate of the gamer becomes very low because the opponents have a higher value than that of gamer g compared to the case when the similarity is 0.175. In other words, the opponents are stronger than the gamer because have a negative difference of attributes. Hence, they are more dangerous opponents.

In conclusion, if the game attributes have a high similarity, the value becomes close to 1, and as a result, the gamer and opponent have similar skills. If the attributes have a low similarity, the attributes of the opponent are higher or lower than those of the gamer; thus, they are opponents that the gamer can win or should avoid, respectively.



(a) Weak opponents (30,087 objects).



(b) Strong opponents (69,913 objects).

Fig. 7. Similarity distributions of opponents.

5. Conclusions

To determine the matchability of gamers with surrounding opponents in a spatial game, their game attributes must be compared with those of surrounding objects. Game attributes can help improve the fun and winning rate of games by comparing them in the server and providing the result to gamers. To raise the winning rate of gamers, the similarity must be determined by comparing various attributes of game objects.

In this study, we proposed a technique for selecting game opponents who have a lower winning rate than that of the gamer using the similarity of attributes between the gamer and nearby opponents. A higher or lower similarity does not always mean advantage over the opponent, because even if the similarity is low, the opponent may have a smaller or larger values in each attribute than the gamer. If the attribute value of the opponent is smaller than that of the gamer, the gamer has advantage, but otherwise, the opponent is stronger than the gamer. Therefore, it is difficult to determine advantage or disadvantage over an opponent by similarity alone

As discussed above, the similarity applied to games is different from the similarity of documents based on the frequency of words. In general, similarity is determined in the range of [0, 1], but even if the similarity is the same, the game result can vary by whether the difference between the gamer and opponent is positive or negative. This characteristic implies that the similarity must be differentiated depending on the corresponding problem domain. This study has significance in that we calculated similarity based on Euclidean distance and identified the difference of results depending on the direction of similarity

Acknowledgments

This paper was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korea Government (Ministry of Education) [NRF-2017R1D1A1B03035884].

References

[1] Jongwan, K. (2015): Targeting Enemies Using Reverse Skyline Query in Spatial Game, 10(10), pp. 27241–27248.
[2] Junbiao, G.; Kaihua, W. (2019): Towards pedestrian room evacuation with a spatial game, 347, pp. 492–501.
[3] Jongwan, K. (2018): An Application to Search for High-Priority War Opponent in Spatial Games Using Dynamic Skyline Query, 13(2), pp. 1496–1500.
[4] Jiawei, H.; Micheline, K.; Jian P. (2015): Data Mining: Concepts and Techniques, 1st edn, Elsevier.
[5] Harish, G. (2017): Distance and similarity measures for intuitionistic multiplicative preference relation and its applications, 7(2), pp. 117–133.
[6] Chaitanya, P. A.; Meena, A. (2017): Introduction to Data Mining, Education Publishing.
[7] Eli, M. (2007): The Pythagorean Theorem, Prinston University Press.
[8] Jupyter Notebook: http://jupyter.org.