Proc. of the 3nd IEEE Global Conference on Consumer Electronics
(GCCE 2014), Tokyo, Japan, Oct. 7-10, 2014.

Online Adjustment of the AI's Strength in a Fighting Game Using the *k*-Nearest Neighbor Algorithm and a Game Simulator

Yuto Nakagawa
Intelligent Computer Entertainment Laboratory, HCI Dept., ISE
Ritsumeikan University
Shiga, Japan
is0127kh@ed.ritsumei.ac.jp

Kaito Yamamoto
Intelligent Computer Entertainment Laboratory, GSISE
Ritsumeikan University
Shiga, Japan
is0093fh@ed.ritsumei.ac.jp

Ruck Thawonmas
Intelligent Computer Entertainment Laboratory, HCI Dept., ISE
Ritsumeikan University
Shiga, Japan
ruck@ci.ritsumei.ac.jp

Abstract— **This paper proposes a method for online adjustment of the AI's strength in a fighting game. The adjustment aim is that of maintaining the AI's strength slightly above its opponent's level. The proposed method adopts the *k*-nearest neighbor algorithm for predicting the opponent's next action and uses this information together with a game simulator for determining the next action of the AI, leading to slightly winning scores against the opponent. The proposed method is evaluated on a fighting-game used at the fighting game AI competition organized by the authors since 2013. Evaluation results confirm the effectiveness of the proposed method.**

Keywords— fighting games, AI's strength, online adjustment, k-nearest neighbor

I. INTRODUCTION

In most fighting games, the opponent's strength influences a lot the games' interestingness and difficulty [1]. It can thus be stated that the player will have more fun playing against an opponent that matches well with the player's skill level or experience level. However, different players have different such levels.

Most fighting games have been mainly designed for human versus human matches, but also allow their players to play against a non-player character or AI. Such AIs, however, are designed to have different pre-defined strength and thus do not adapt their strength to the encountering human player. A number of approaches have been proposed that attempt to increase the AI's strength [2] or mimic the human player [3]. However, those approaches did not aim at adapting the AI's strength to fit that of the human player. Such adaptability is required for making matches against an AI more fun to the player.

In this paper, we propose a method for online adjustment of the AI's strength in a fighting game. The adjustment aim is that of maintaining the AI's strength slightly above the player's level. The proposed method adopts the *k*-nearest neighbor algorithm for predicting its opponent's next action and uses this information together with a game simulator for determining the next action of the AI, leading to slightly winning scores against the opponent.

II. GAME RULES

Here we describe the game rules in the competition, the Fighting Game AI Competition [1] organized by the authors' laboratory since 2013 for game-AI research purposes, whose game platform is used in this research. In this competition, a game consists of three rounds, each having 60 sec. The winner of a match is the AI whose sum of the scores in each round, *self.score*, given below, is the larger.

$$self.score = \frac{opp.roundDamage}{self.roundDamage + opp.roundDamage} * 1000, \quad (1)$$

where *x.roundDamage* is the amount of accumulated damages the AI (*x* = "*self*") or the opponent (*x* = "*opp*") has received since the beginning of the corresponding round.

III. PROPOSED METHOD

In order to adjust the AI's strength, the proposed method uses the AI's scores, *acc.score*, accumulated from the beginning of each game and uses the *k*-nearest neighbor algorithm for predicting its opponent's next action *predictAct*, the description of which is given in Section IV. In addition, once such an action is predicted, it uses a game simulator to evaluate each of the AI's actions a_i (i=1, 2,…, n) selected in advance for this task, where $e[i]$ is the evaluation value for a_i and n is the number of such actions. The evaluation value $e[i]$ is defined by the difference between the opponent's amount of damages and the AI's amount of damages when the AI conducts a_i against *predictAct*. It has a minus value when a_i led to an inferior outcome and a plus value to a superior outcome

The AI changes the way to determine its own action based on the situation it is currently facing. We, therefore, consider two situations: inferior situation and superior situation. In addition, we employ a threshold τ and compare it with *acc.score* in order to determine which situation the AI is currently facing. If *acc.score* is smaller than τ, it is in the inferior situation and the action with the highest evaluation

[1] http://www.ice.ci.ritsumei.ac.jp/~ftgaic/

Algorithm 1 ActDecision($self, opp, predictAct, game$)

```
e[i] ⇐ 0 for all i = 1, 2,…, n
acc.score ⇐ 1000*(opp.gameDamage/
    (self. gameDamage + opp.gameDamage))
e ⇐ simulate(self, opp, predictAct, game)
if acc.score ≧ τ then
    repeat
        d ⇐ argmin (|e|)
        if e[d] ≥ 0 then
            e[d] ⇐ ∞
        end if
    until e[d] < 0
else
    d ⇐ argmax (e)
end if
return a_d
```

value will be conducted to make the AI have more strength. This mechanism is given in detail in Algorithm 1, where *game* represents the current game-status data, *x.gameDamage* is the amount of accumulated damages the corresponding character. In addition, simulate(*self*, *opp*, *predictAct, game*) uses the game simulator that simulates the game for about one-second-game time from the current time, and it returns the evaluation value for each of the AI's actions against *predictAct*.

IV. K-NEAREST NEIGHBOR ALGORITHM

For the predicting task, first, the opponent's attack data are recorded each time the opponent performs an attack action. The recorded information is the attack type and the relative position between the opponent and the AI. Next, when the AI needs to decide its action, it first checks whether the opponent is going to perform an attack action. When the number of the opponent's recorded attack actions within a specific distance from the current relative position, between the opponent and the AI, is larger than a pre-defined number, the AI will consider that the opponent is going to attack. If so, the AI will refer to the opponent's recorded attack actions that are k nearest to the current relative position and will predict the action that the opponent is going to perform, i.e., *predicAct*, by majority voting among those k actions.

V. PERFORMANC EVALUATION

In order to validate the effectiveness of the proposed method, we compared two types of AI: the one with the proposed method and the one designed to be a strong AI with $\tau = 1001$. Note that the latter AI is of the same behavior as a sample AI called MizunoAI available in the aforementioned competition site. We tested the proposed method with two values of τ: 550 and 500. The value of k was set to 3, as done in MizunoAI. The top three AIs in the 2013 competition, T, SejongAI, and Kaiju were individually used as the opponent. For each opponent, 100 games were conducted.

From the results shown in Tables 1-3, MizunoAI ($\tau = 1001$) outperformed all of the opponents while the proposed AI with $\tau = 550$ scored slightly above each of its opponents and the proposed AI with $\tau = 500$ had scores below the opponents.

TABLE I. AVERAGE SCORES AGAINST T

τ	ROUND 1	ROUND 2	ROUND 3	GAME
1001	554	548	551	551
550	520	513	511	515
500	474	469	474	472

TABLE II. AVERAGE SCORES AGAINST SEJONGAI

τ	ROUND 1	ROUND 2	ROUND 3	GAME
1001	508	538	519	522
550	465	515	534	505
500	456	485	490	477

TABLE III. AVERAGE SCORES AGAINST KAIJU

τ	ROUND 1	ROUND 2	ROUND 3	GAME
1001	557	675	664	632
550	491	572	554	539
500	467	514	511	497

VI. CONCLUSIONS AND FUTURE WORK

We proposed a method for performing online adjustment of the AI's strength. The proposed method is based on the k-nearest neighbor algorithm for predicting the opponent's next action and on a game simulator for determining the AI's next action that would lead the AI's scores slightly above those of the opponent. Our results show that the proposed method is effective, but care must be taken in choosing the value of τ. We plan to investigate in future the role of τ in detail.

REFERENCES

[1] Jenova Chen: "Flow in Game," Communications of the ACM, Vol. 50, No. 4, pp. 31-34, 2007.

[2] B.H. Cho, S.H. Jung, Y.R. Seong, and H.R. Oh, "Exploiting Intelligence in Fighting Action Games using Neural Networks,"IEICE Transactions on Information and Systems, vol. E89-D, no. 3, pp. 1249–1256, 2006.

[3] S, Lueangrueangroj and V. Kotrajaras, "Real-time Imitation based Learning for Commercial Fighting Games," Proc. of Computer Games, Multimedia and Allied Technology 2009, pp. 1–3, 2009.