ECAI 2014
T. Schaub et al. (Eds.)

doi:10.3233/978-1-61499-419-0-1103

Surrogate-Agent Modeling for Improved Training

Ales Tavcar[1] and **Bostjan Kaluza**[1] and **Marcel Kvassay**[2] and **Bernhard Schneider**[3] and **Matjaz Gams**[1]

Abstract. Computer-aided practice can help improve personnel training for demanding scenarios in terms of time and quality. In this paper, we concentrate on asymmetrical conflicts, such as a unit that deals with hostile crowds robbing a store, with the aim of preventing further criminal activity and at the same time minimizing physical and emotional damage. We propose a surrogate-agent modeling approach based on execution of the following loop: (i) observe a human (the unit leader) playing a set of scenarios in a simulated environment and induce strategic patterns of human play; (ii) use patterns to construct a surrogate agent (digital clone); (iii) test the surrogate under all possible circumstances through data farming; and (iv) evaluate the performance and highlight deficiencies in the agent's responses, thereby enabling human improvements in new attempts. Experiments on two domains indicate that the proposed approach could significantly improve the training procedure and help trainees to properly perceive the cognitive properties of the crowds.

1 INTRODUCTION

Exhaustive training in complex and demanding domains is practically impossible due to time, danger to humans, and damage to devices and equipment. In such domains, personnel training often involves simulated environments in order to increase efficiency in a variety of initial conditions, scenarios, and simulation-parameter settings. Previous work on this subject can be roughly categorized into one of two approaches. The first approach includes fixed simulation policies that mimic how trainees should perform in certain scenarios [1]. This approach can identify weaknesses in general policies, but lacks the ability to personalize the training. In operative simulations and war gaming, the commercial off-the-shelf computer game characters are able to adapt the difficulty level [2], but this has been shown to be insufficient for enhanced decision-making and detailed analysis.

The present paper proposes a novel approach using three key ideas. First, the Cognitive Multi-Agent Strategy-Discovering Algorithm (CMASDA) extracts graphical and symbolic descriptions of strategic behavior patterns from the simulation logs of the trainee. Note that the learning is "from scratch," which means that prior knowledge is elemental. Second, the identified behavior patterns are transferred into a surrogate agent (SA) in similar manner to [3], which makes it possible to reproduce the behavior of the trainee in the simulator. Third, a data-farming procedure validates the cloned behavior of the trainee throughout the entire simulation parameter space. This process identifies the worst-case situations for the SA at the same time as generating the improved version of the SA, both of which are then presented to the human during the next round of training. Based on these ideas, the Surrogate-Agent Modeling for Improved Training algorithm (SAMIT) was designed.

The proposed approach was evaluated on two domains that had the goal of training a unit leader. In the Riot Domain (RD), a physical conflict develops between security forces and a crowd trying to enter a building. In the Looting Domain (LD), a crowd is already looting a store as the unit approaches.

2 SURROGATE AGENT AND DATA FARMING

The SAMIT algorithm is presented as follows:

1. **repeat**
2. *let human play simulation scenarios until satisfied*
3. *learn physical and cognitive patterns from human performance*
4. *design surrogate agent from the learned patterns*
5. **repeat** // datafarming
6. *create scenario with new parameters*
7. *apply surrogate agent on the datafarming scenario*
8. *estimate damage*
9. *store worst played scenarios into a list*
10. *perform one step of learning*
11. **until** all parameters tested
12. **until** satisfactory overall performance achieved

The SAMIT algorithm is applied iteratively, where experience from the previous run is added to the next one. First, the input data to the human includes worst-played scenarios, behavior patterns constituting the SA, and the improved SA from the previous run. Second, the set of newly learned patterns is added to the set of already learned patterns to construct a new SA. These steps are explained in greater detail below.

Line 1-3: Observe human play and induce strategies. A trainee (a human unit leader) receives instructions and policies to play serious games in the simulator. Once satisfactory performance has been achieved, 10 simulations (due to the stochastic nature of the scenarios) are stored for learning with CMASDA, which extracts physical and cognitive strategies in the form of behavioral patterns.

Line 4: Design a surrogate agent. The obtained strategies are reconstructed to design a behavioral clone in the form of SA that is capable of taking the same actions in the simulator as the human trainee did. As soon as one or several patterns fire under the conditions in a simulator, the SA chooses the most successful chained strategy sequence and starts executing it until new patterns fire due to changed circumstances.

Lines 5-11: Apply SA in data farming. The SA is used in data farming in all possible circumstances; for example, some members of the gang use hidden weapons, someone shouts a warning, etc.

[1] Department of Intelligent Systems, Jozef Stefan Institute, Slovenia, email: {ales.tavcar, bostjan.kaluza, matjaz.gams}@ijs.si
[2] Institute of Informatics, Slovak Acad. of Sci., Slovakia,marcel.kvassay@savba.sk
[3] EADS Deutschland GmbH, Germany, email: bernhard.schneider@cassidian.com

Line 10: Improve the SA through learning. The SA is a decision module consisting of behavioral patterns. During data farming, evaluation provides feedback that enables learning, i.e., improves behavioral patterns. The final result is an improved SA.

Line 12: Evaluate and retrain. Evaluation of SA performance shows the trainee results indicating the worst-played scenarios, overall performance of the original and improved SA, and the set of strategic patterns.

3 EXPERIMENTS

We used the MASON (Multi-Agent Simulator Of Networks [4]) platform for interactions between security and adversary agents. The behavior of agents was modeled using the PECS reference model, which can characterize the humans with several personality factors and internal processes, thereby generating situation-dependent behavior driven by the humans' motivational and emotional states.

To compare the similarity among the simulation runs, we introduced a similarity measure based on dynamic time warping [5] that compares measures of effectiveness consisting of 14 parameters [6] logged during the simulation; these include anger, fear level, escalation, and the number of people injured. Results above 0.5 indicate quite similar behavior, while those below 0.25 indicate very different behavior.

The results in Figure 1 present the similarity measure between different behaviors. RD and LD denote the rioting and the looting domain; F – fixed-policy agent (simulation); FS –fixed-policy agent surrogate; T – trainee (human); and TS – trainee surrogate. The first bar (F-F) in Figure 1 shows the similarity between the different simulations of the fixed-policy agent, while the second bar (F-FS) shows the similarity between the fixed-policy agent and its SA in RD. The four left-most bars show that surrogate modeling reproduces behavior that is very similar to the fixed-policy agent in the RD and LD domains. The next four columns represent the similarity of behavior of the trainees and their surrogates in both domains. For sanity check, the last four bars compare the behavior of the fixed-policy agent to that of the human trainees and the trainee surrogates in both domains.

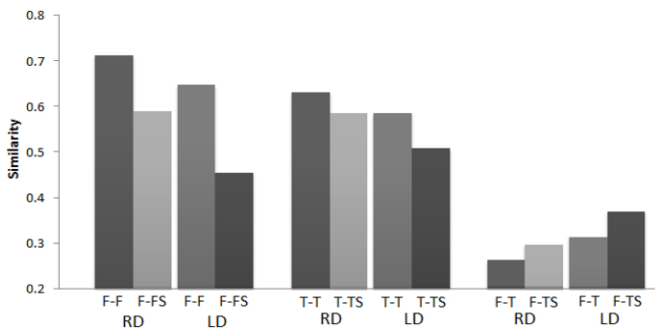

Figure 1. Comparison of behavior of the trainee (T), the fixed policy (F) and their surrogate agents (TS, FS).

Figure 2 represents the 10 simulations that had the highest damage values in the first simulation run. The left-paired (dark) bars show damage in the first run, while the right-paired (light) bars show damage in the second run. In the first simulation, the average damage is 1252, while 5 percent of all simulations resulted in a damage higher than 1000. In the second run, none of the simulations scored damage higher than 1000 and the average damage value for the 10 worst simulations was 337. Overall, the average damage over the complete data-farming simulation was reduced by over 31 percent, from 251 to 171. The reduction is statistically significant with a paired t-test with the 0.05 significance level. The main insight in terms of the semantic difference in behavior was that when a crowd was agitated in a specific way, the unit leader should immediately resort to more aggressive defensive actions. A description of the system is given in [6].

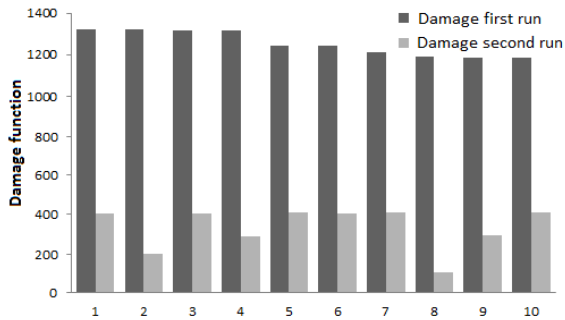

Figure 2. Comparison of the 10 worst simulations in the first data-farming run with their modifications in the second run.

4 CONCLUSIONS

The SAMIT algorithm first aims to learn the specific strategic behavior of a trainee in conflict situations dealing with cooperating and opposing agents. It then transforms the learnt physical and cognitive human behavior patterns into a surrogate cognitive agent. The agent is tested and improved through data farming to obtain performance evaluation under all circumstances, thereby improving the next round of training. The motivation is that humans can test only a limited amount of all possible scenarios, while the SA can estimate the overall performance of the trainee and pinpoint weak points in the learned behavior in specific situations.

SAMIT reliably reproduced the observed behavior of either fixed-policy agents or the complex behavior of human trainees in two domains. A significant similarity between the original simulation logs and the logs of surrogate behavior was observed. The SAMIT algorithm in one domain identified key weaknesses in the behavior of trainees, which enabled faster and more complete training.

REFERENCES

[1] D. Kallfass and T. Schlaak, NATO MSG-088 case study results to demonstrate the benefit of using data farming for military decision support, *Proceedings of the 2012 Winter Simulation Conference*, article no. 221, 2012.

[2] O. Missura and T. Gärtner. Player Modeling for Intelligent Difficulty Adjustment. *Proceedings of the 12th International Conference on Discovery Science*, 197–211, 2009.

[3] D.J. Stracuzzi, A. Fern, K. Ali, R. Hess, J. Pinto, N. Li, T. Konik and D. Shapiro, An application of transfer to American football: From observation of raw video to control in a simulated environment. *AI Magazine*, 32 (2), 107–125, 2011.

[4] http://cs.gmu.edu/~eclab/projects/mason/, 2014.

[5] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria and E. Keogh, Data Mining a Trillion Time Series Subsequences Under Dynamic Time Warping. *IJCAI-2013 Proceedings*, 3047–3051, 2013.

[6] M. Kvassay, L. Hluchý, S. Dlugolinsky, M. Laclavik, B. Schneider, H. Bracker, A. Tavcar, M. Gams, D. Król, M. Wrzeszcz and J. Kitowski, An integrated approach to mission analysis and mission rehearsal. Winter Simulation Conference 2012: 362