Data-Driven Behavior Modeling for Computer Generated Forces

Dr Linus J. Luotsinen
FOI - Swedish Defence Research Agency
SE-164 90 Stockholm
SWEDEN
linus.luotsinen@foi.se

Dr Rikke Amilde Løvlid
FFI - Norwegian Defence Research Establishment
NO-2007 Kjeller
NORWAY
rikke-amilde.lovlid@ffi.no

ABSTRACT

Computer generated forces (CGFs) are autonomous or semi-autonomous actors within military, simulation based, training and decision support applications. The CGF is often used to replace human role-players in military exercises to, ultimately, improve training efficiency. The modeling and development of CGFs is a complex, time-consuming and expensive endeavor where military domain expertise and doctrinal knowledge are interpreted and programmed into the CGF by hand. Furthermore, CGFs often represent human actors and behaviors (pilots, soldiers, manned systems, etc.) making it an even more challenging task.

In recent years the Artificial Intelligence (AI) research community achieved some remarkable results where Intelligent Agents (IA) successfully defeated human champions in games such as chess and Jeopardy. AI researchers have demonstrated that Machine Learning (ML) algorithms can be used to learn IA behaviors from recorded observations such as log-files, GPS coordinate traces and, more recently, pixels from images and video.

The ability of the machine learning approach to learn the ”behavioral rules” of the CGFs, which we from now on will refer to as Data-Driven Behavior Modeling (DDBM), has many potential advantages compared to the traditional CGF modeling approach where the ”behavioral rules” are manually hand-crafted using subject matter experts and doctrines. Using DDBM the modeling efficiency with respect to cost and time may improve, in particular, when modeling complex CGFs designed to mimic human actors and behaviors within complex environments. The DDBM approach may also improve behavior realism and objectiveness resulting in better and more realistic training and decision support tools.

In this work we introduce the concept of DDBM including its main components in the context of CGF behavior modeling. We also provide preliminary results of experiments where our DDBM-prototype is used to generate behaviors using both observational and experiential learning strategies.

1 INTRODUCTION

Modern simulation platforms use AI-techniques such as behavior trees, hierarchical finite state machines, rule-based systems, etc. to represent and embed behaviors in well defined structures/models that are executable, maintainable, reusable and scalable [16]. These techniques are employed to design and implement behaviors for Computer Generated Forces (CGF) that are more advanced and realistic compared to CGFs developed using traditional, ad-hoc, scripting techniques. However, creating and populating these structures with content, which ultimately defines the ”behavioral rules” and decision making skills of the CGF, remains a painstaking, time-consuming and expensive endeavour.

In this work we introduce the concept of Data-Driven Behavior Modeling (DDBM) where Machine Learning (ML) techniques are applied to automatically generate the ”behavioral rules” of CGFs using observations and recorded data. The idea is to improve modeling efficiency by transferring much of the manual modeling work (i.e. the handcrafting of ”behavioral rules” extracted from doctrines or domain expertise) to instead acquiring, editing and labelling datasets that can be fed into and automatically processed by machine learning algorithms. Besides improving modeling efficiency the DDBM-approach may also, potentially, be used to enhance behavior realism by exploiting the ability of ML algorithms to automatically identify patterns in large datasets.

In an attempt to empirically evaluate the DDBM-approach we have developed a prototype that implements and integrates the main components of DDBM to form an intuitive and easy-to-use behavior modeling tool. In this work we will describe the prototype, including its main components, and provide experimental results where toy-problems are used to demonstrate the prototype's capability to imitate and optimize CGF behaviors using observational- and experiential learning strategies respectively.

Although we are only using toy-problems in this work, the long term goal of our research is to apply the DDBM-prototype in real-world military applications to further investigate and gain insight into the following research questions:

- Is the DDBM approach more efficient, with respect to cost and time, compared to the traditional modeling approach? That is, what can be gained by shifting the modeling work from manually handcrafting behavioral rules to acquiring, creating, editing, labelling or pre-processing datasets?

- Can DDBM be used to create behaviors that are too complex to model using the traditional modeling approach? Are ML algorithms able to identify rules, relations or behavioral patterns in recorded data or observations that could not be identified by hand?

- Can DDBM be used to create objective behavior models that imitates the behavior of its real-world counterpart? If so:

  - Is it possible to analyze the imitated behavior using what-if-simulation to, for instance, identify its weaknesses and strengths?
  - Is it possible to use models of imitated behavior to improve military training? That is, can we analyze a trainee's behavior more efficiently by automatically identifying deviations using models of imitated experts?

This paper is organized as follows. In Section 2, we present related works. In Section 3, we introduce DDBM and its main components. In Section 4, we describe the implementation of our DDBM-prototype. In Section 5, we present experimental results using the DDBM-prototype. Finally, conclusions and future works are presented in Section 6.

2 RELATED WORKS

In recent years the Artificial Intelligence (AI) research community has achieved remarkable results where Intelligent Agents (IA) successfully defeated human champions in games such as chess [4] and Jeopardy [6]. AI researchers have demonstrated that machine learning algorithms [17] can be used to learn and recognize behaviors from recorded observations such as log-files [8, 22], GPS coordinate traces [12] and, more recently, pixels from images and video [18].

The literature indicates the existence of diverse approaches to learning processes of behavior modeling. Several authors have investigated *observational learning* in different domains, using a variety of techniques [5, 9, 19, 10]. For instance, Johnson and Gonzalez [9, 10] present a prototype with the focus on learning team behavior from observations. The approach is semi-autonomous and observations are manually processed to identify domain specific contexts representing different states of the observed behavior. The use of contexts restrains the amount of the observational training data to those only relevant within the context. In the robotic research a slightly different approach has been developed and used. In these methods, a human intentionally demonstrates and teaches the robot how to perform a given task (learning by demonstration) [2, 3].

A quite different approach is *experiential learning* in which the agent (with no human supervision or involvement) explores the environment on its own and learns by attempting to optimize some performance metrics defined by the modeler [22]. There is a substantial body of research using this approach, although they do not explicitly use the term experiential learning (e.g. see [1, 14, 25]). Aihe and Gonzalez [1], propose using *Reinforcement learning (RL)* to compensate for situations where the domain expert has a limited knowledge on the subject being modeled. Merrick and Maher [14] present motivated reinforcement learning agents to create non-player game agents that explore their environment and learn new behaviors, in response to interesting experiences. Teng et al. [25] use a self-organizing neural network that learns incrementally through real-time interactions with the environment and improves air combat maneuvering strategies of CGFs.

Several authors have used a *hybrid approach*, combining observational learning and experiential learning methods [3, 22]. The hybrid approach is similar to experiential learning, with the main difference that the agent, instead of random initial solutions, improves solutions that are obtained by observational learning. In the work of Bentivegna and Atkeson [3] a robot playing air-hockey, first observes and learns the behavior of an expert. Then a reinforcement learning process is used to improve the learned behavior. Stein and Gonzalez [22] use a hybrid method, in which agents learn tactical skills by observation as well as by experiments (in different domains). The authors suggest that the agents using the hybrid approach are both human-like and perform better than the original human; learning by observation makes the agents behave human-like and during the experimental learning phase, the performance of the agent is optimized.

3 DATA-DRIVEN BEHAVIOR MODELING

In the following, we introduce the DDBM concept, its main components and how they relate to each other emphasizing the work-flow within the main learning strategies (observational, experiential and hybrid) identified in Section 2. Figure 1 provides an overview of DDBM where the left part illustrates the work-flow of the three learning strategies and the right part illustrates behavior model application. As mentioned above, CGFs are typically represented in modular structures such as behavior trees or finite state machines to ensure reuse, scalability, maintainability and execution capabilities. As a result, in Figure 1, the *behavior model* may represent the entire CGF or, perhaps more commonly, a behavioral sub-component or module of the CGF. In the figure we assume that the simulator used during learning and application has an interface allowing the CGF to register its perceived *state* over time and that actions can be injected or invoked by the CGF to affect or alter the state of the simulation.

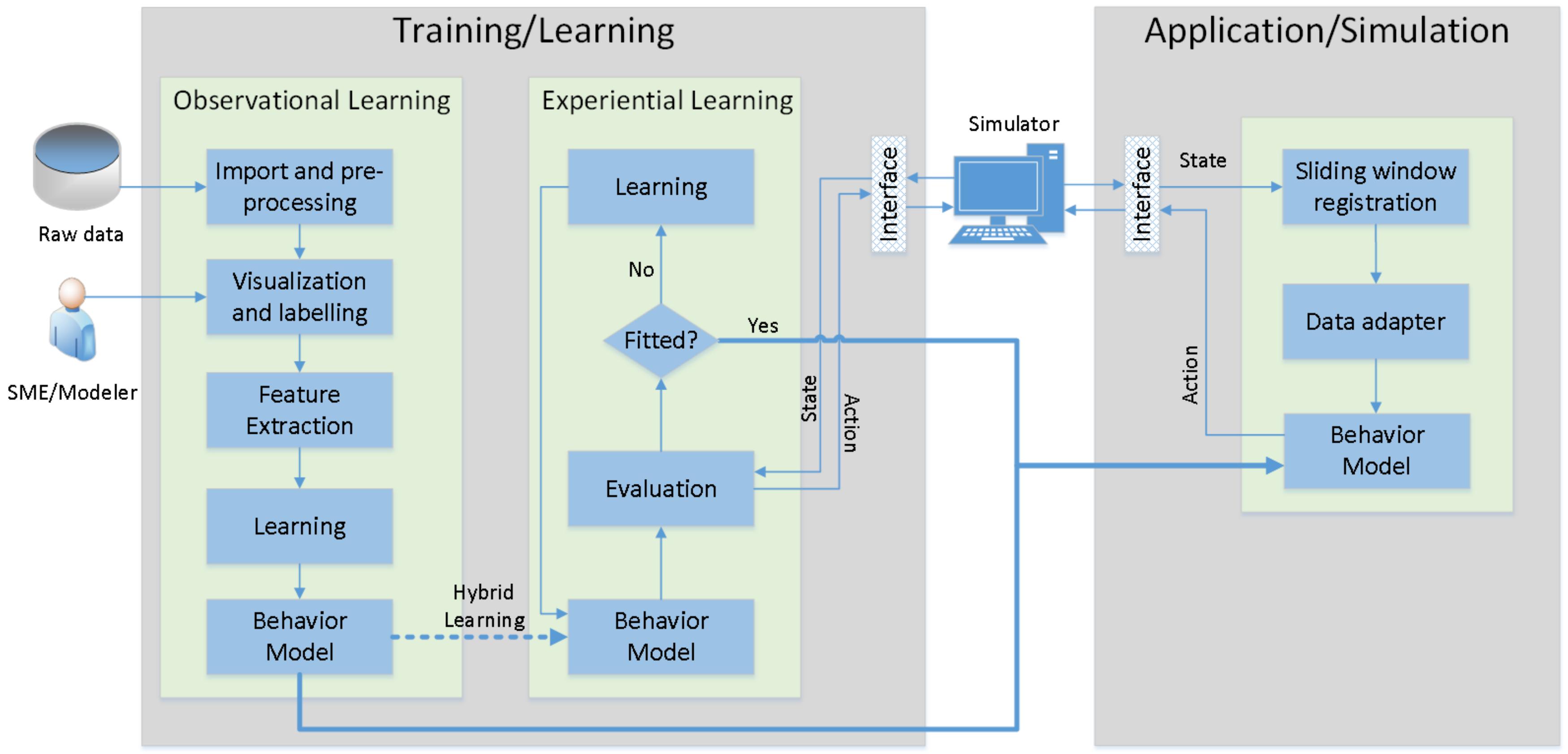


Figure 1: An overview of the main components of DDBM focusing on the work-flow of the three main learning strategies: observational learning, experiential learning and hybrid learning. In observational learning pre-recorded datasets are used to learn the desired behavior. In experiential learning an iterative trial-and-error approach is used to learn the desired behavior. Hybrid learning combines observational learning with experiential learning.

3.1 Observational learning

In observational learning, the goal is to develop the behavior model of the CGF by *observing* the behavior of the agent whose behavior should be learned (so called original agent). The data collected from the original agent performing an activity, in a simulation or in the real world, is used to train the CGF to act similarly when attempting to perform the same activity under similar conditions [23].

The input, as illustrated in Figure 1, to the observational learning strategy is raw data recorded from real-world exercises, simulations, etc. The dataset is then processed by SMEs or modelers to identify and label behaviors of interest. Next, the dataset is further processed using feature extraction functions capable of identifying key features in the data that, ultimately, reduces the complexity of the learning task. After feature extraction the dataset is ready to be used as input to the learning component where ML algorithms are employed to automatically generate the behavior model.

Learning by observation is essentially similar to supervised learning in the sense that it learns from the observed data. However, there are some principal differences between these two. In observational learning, the input data are trace of human performance with different length, and it is not clear where one example starts and ends. Furthermore, the labels are implicit rather than explicit. In traditional supervised learning the input data are explicitly defined with features and labels [23].

3.2 Experiential learning

In experiential learning, the main idea is that the CGF can learn and optimize its behavior using a trial-and-error approach within the target simulator. The desired behavior is defined by the modeler in an evaluation-, fitness- or reward function which is used to measure CGF performance over time. There are several well known ML techniques that can be used to implement experiential learning. Perhaps the most commonly used techniques are genetic algorithms (GA), genetic programming (GP) [11] and reinforcement learning (RL) [24].

Using GP, for instance, the CGF or behavior model is represented by a computer program that, during learning, iteratively evolves and optimizes its behavior over time. Typically, GP is initialized with a randomly generated population of CGFs. The entire population is then allowed to execute in the simulator and the fitness function evaluates each individual CGF separately. In the next iteration, a new population or generation of CGFs is generated based on the ”survival of the fittest” principle. That is, CGFs with high fitness values are more likely to be selected and included in the next generation of CGFs. Diversity is introduced in the population through the use of mutation and cross-over operators. The mutation operator is capable of modifying parameters or variables within the program structure of selected CGFs. The cross-over operator is, by combining the program structures of two parent CGFs, capable of creating offspring with alternative program structures. Mutation and cross-over operators increase the learning algorithm’s ability to find an optimal behavior model as opposed to converging towards a sub-optimal behavior model. The iterative process continuous until a CGF with the desired behavior has been found (as depicted by the fitness function) or until a predetermined number of iterations has been evaluated.

3.3 Hybrid learning

The hybrid approach is similar to experiential learning in the sense that the method optimizes the solution to improve performance using a trial-and-error approach. However, the hybrid learning strategy does not use randomly generated CGFs to represent its initial population, instead the population of CGFs is generated using existing datasets following the observational learning strategy. In Figure 1 the hybrid learning strategy is represented by a dashed arrow connecting the observational and experiential learning strategies.

4 IMPLEMENTATION

Given the above conceptual description of DDBM we have implemented a prototype that stitches all components (e.g. data-acquisition and visualization, feature extraction, learning strategies and simulation/application) together to form an intuitive and easy to use data-driven behavior modeling tool that can be adapted for use in a variety of applications and simulation tools. In this section we briefly introduce the front-end application or authoring tool that we have developed to support behavior modeling using the DDBM approach. We also discuss the feature extraction and learning components implemented in the prototype.

4.1 Authoring tool

To the best of our knowledge, there is a lack of software tools targeting the requirements of the DDBM approach with respect to data-acquisition, visualization and pre-processing. Researchers and modelers typically rely on a mix of general purpose and ad-hoc tools when modeling agents and CGFs using DDBM. To address this problem, we have developed an authoring and behavior modeling tool that can be used to visualize, import, edit, create and export datasets.

Figure 2 provides a screen-capture of the authoring tool. The view in the top-left lists all data associated with the agents over time. Each agent is represented by one or more timestamps and each timestamp consists of multiple data items that represents the agent's state with respect to position, heart-rate, role, etc. In the sequence-view the modeler can tag or label sequences of behavior observed in the dataset. A behavior sequence is defined by a time-interval and by the agents performing the behavior. The map view is used to visualize the dataset by superimposing agent data such as positions, movement traces on a geographical map. The map view can also be used to create and edit the positioning of the agents in the dataset.

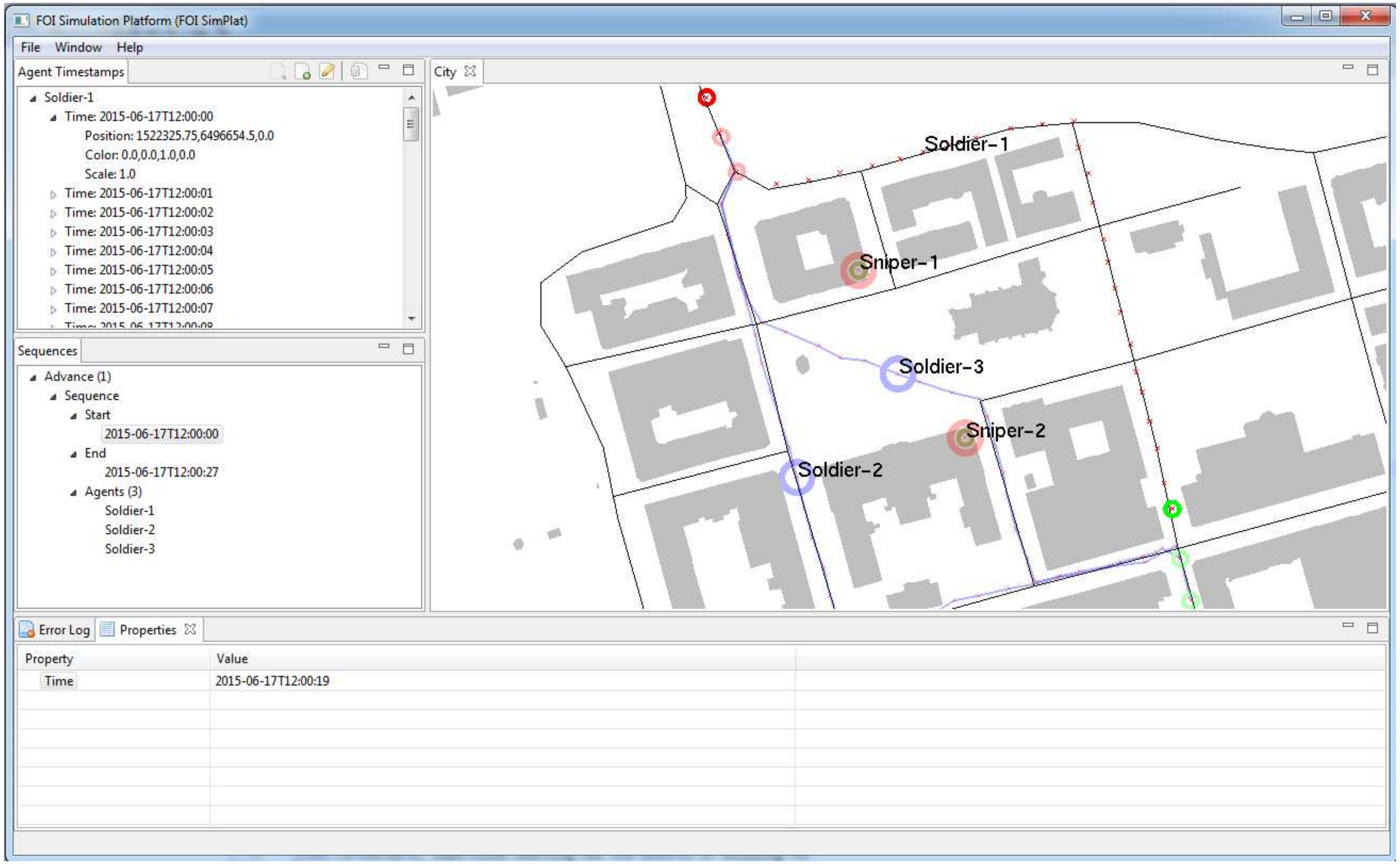
In addition to the core functionality described above, the authoring tool can be used to pre-view datasets processed by the feature extraction component (see Section 4.2). This allows the modeler or SME to gain more insight into the behavior and decision making of the recorded agent even prior to learning the behavior.

4.2 Feature extraction component

The feature extraction component we have implemented in this work is capable of extracting spatial features such as the agent's position, velocity and orientation relative the environment as well as agent relative features such as the relative positioning between agents with respect to for instance distance and orientation. The feature extraction module can be extended to include more advanced or domain specific feature extraction functions as well. For instance, in ongoing works we are implementing terrain analysis capabilities capable of calculating line-of-sight, area-of-visibility and routes given Geographical Information System (GIS) databases representing buildings, roads, terrain types, etc.

4.3 Learning component

We have implemented learning capabilities using open-source implementations of both observational and experiential learning algorithms. In this work we have used machine learning libraries such as Weka [7], RapidMiner/YALE [15] and JGAP [13]. The main advantages of using these libraries are that they provide, in addition to implementations of a wide range of ML algorithms, tools that can be used to evaluate learning performance (classification accuracy, confusion matrices, etc.), tools designed to improve and tune learning performance using feature selection algorithms, parameter optimization techniques, dimensionality reduction algorithms and so on.



```
Soldier-1
    Time: 2015-06-17T12:00:00
        Position: 1522325.75,6496654.5,0.0
        Color: 0.0,0.0,1.0,0.0
        Scale: 1.0
    Time: 2015-06-17T12:00:01
    Time: 2015-06-17T12:00:02
    Time: 2015-06-17T12:00:03
    Time: 2015-06-17T12:00:04
    Time: 2015-06-17T12:00:05
    Time: 2015-06-17T12:00:06
    Time: 2015-06-17T12:00:07
```

```
Advance (1)
    Sequence
        Start
            2015-06-17T12:00:00
        End
            2015-06-17T12:00:27
        Agents (3)
            Soldier-1
            Soldier-2
            Soldier-3
```

Property	Value
Time	2015-06-17T12:00:19

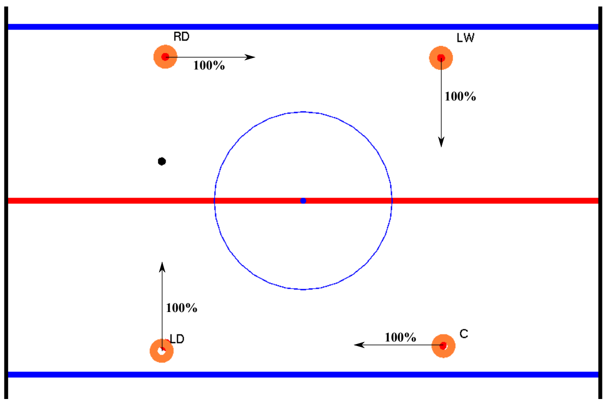
Figure 2: Screen-capture of the DDBM authoring tool developed to visualize, import, edit, create and export datasets. The view in the top-left lists all data associated with the agents over time. Each agent is represented by several timestamps and each timestamp consists of multiple data items such as position, heart-rate, role, etc. In the sequence-view the modeler can tag or label behavior sequences for selected agents within a time-interval. The map view is used to visualize the dataset by superimposing agent data such as positions, movement traces on a geographical map.

5 EXPERIMENTS

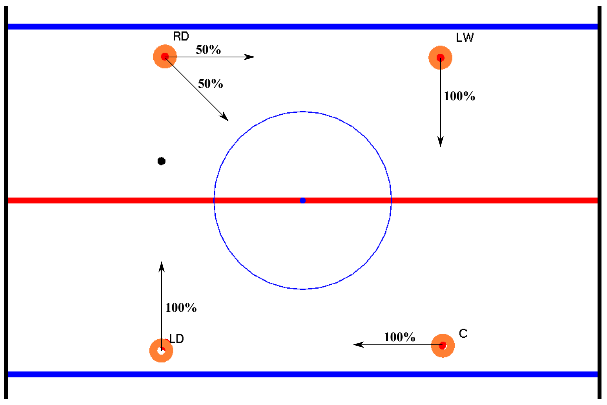
In this section we present results from two experiments. The purpose of the first experiment is to verify that the DDBM prototype is able to imitate collaborative behaviors through observational learning. In the second experiment we explore the prototype's ability to learn and optimize behaviors using the experiential learning strategy.

5.1 Observational learning

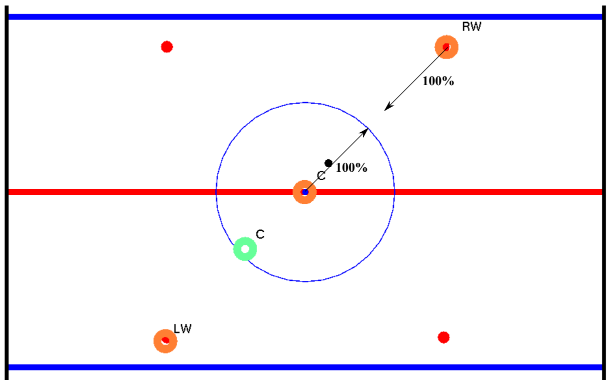
In this experiment the goal is to imitate the behavior of multiple collaborative agents performing tasks of increasing complexity. In this experiment we focus on imitating behaviors representing hockey-player passing exercises as illustrated in Figure 3. In the first exercise, see Figure 3a, the players are passing the puck in a clock-wise manner. In the second exercise one of the players, RD, passes the puck diagonally approximately 50% of the time as illustrated in Figure 3b. In the third exercise the players keeps the puck within the team by avoiding pass options where teammates are covered or intercepted by an opponent player as illustrated in Figure 3c.



(a) Exercise 1: Clock-wise passing.



(b) Exercise 2: Clock-wise and diagonal passing.



(c) Exercise 3: Keep-away passing.

Figure 3: Visual description of the hockey player exercises. In exercise 1 the players are passing the puck to each other in a clock-wise manner. In exercise 2 the RD-player passes the puck diagonally to C approximately 50% of the time. In exercise 3 the players avoid passing the puck to team members that are covered or intercepted by an opponent player.

Given the exercise specifications we created datasets that express the desired behavior for each exercise using our in-house developed authoring tool (see Figure 2). Using the datasets we then applied our DDBM prototype to learn the positioning and passing behavior of each agent using the back-propagation algorithm [21] for standard neural networks and the ID3 algorithm [17] for decision trees respectively.

Preliminary results from our experiments are shown in Table 1. We were able to learn the desired behaviors for all exercises using relatively small datasets. We verified the passing behavior of each agent by visualizing the rules embedded in the generated decision trees. To validate the collaborative behavior of all agents we developed a simple hockey-simulator capable of visualizing the movement of the players and the puck given the previously learned behavior models.

The datasets representing the first and second exercise were created in less than 5 minutes whereas the third dataset representing the keep-away passing exercise took about 10 minutes to author.

Table 1: Observational learning results.

Exercise	Size of dataset (bytes)	Modeling effort (seconds)	Learned correct behavior
Clock-wise passing	2347	273	Yes
Clock-wise and diagonal passing	3400	224	Yes
Keep-away passing	4804	578	Yes

5.2 Experiential learning

In this experiment the goal is to optimize the behavior of a single agent performing a task using the experiential learning strategy. In experiential learning, unlike observational learning, a simulator is integrated within the learning or evaluation phase of DDBM as described in Figure 1.

In this experiment we have used a predator-pray simulation where the predator is represented by a wolf and the pray is represented by a herd of sheep. The sheep's herding behavior was implemented using the flocking algorithm presented in [20]. In the experiment we used the DDBM prototype to generate wolf behavior using genetic programming [11]. The reward function was based on the weighted sum of the number of sheep killed, the wolf's distance to nearest sheep and number of program nodes used to represent the behavior as described in Equation 1:

$$\begin{aligned} f_{reward} = {} & numSheepKilled * w_i \\ & - distance(wolf, nearestSheep) * w_j \\ & + numProgramNodes * w_k \end{aligned} \tag{1}$$

where w_i, w_j and w_k are weights determining the importance of each component in the reward function.

It took approximately 3 days, running a standard desktop computer, to learn a wolf behavior capable of efficiently killing all sheep. The strategy that the wolf eventually learned was to alternate strikes with circular movement patterns as illustrated in the spatial trace plot in Figure 4. Using this strategy the wolf killed all sheep in 9 minutes and 27 seconds which is slightly worse than the strategy used by our scripted, hand-crafted, wolf behavior which completed the task in 7 minutes and 40 seconds.

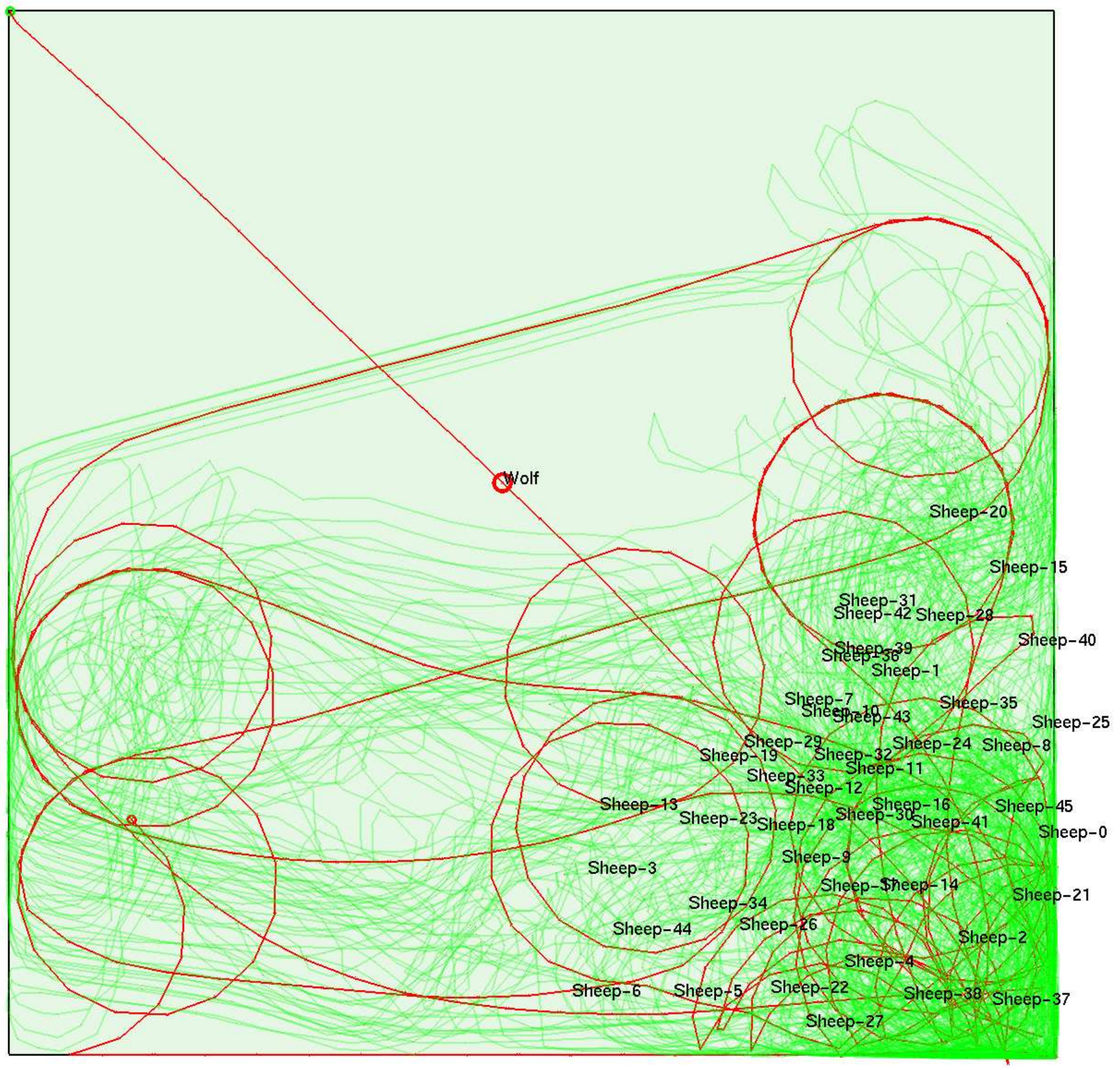


Figure 4: Trace-plot of the wolf and sheep behaviors. The red trace represents the wolf's movement in the simulation and the green traces represent sheep movement. The strategy that the wolf learned was to alternate strikes with circular motion patters.

6 CONCLUSIONS

In this work we have introduced the DDBM concept for CGF behavior modeling in the context of simulation based military training and decision support applications. A prototype was developed and used to verify and evaluate the concept using two different toy-problems exploring observational- and experiential learning respectively. Although the DDBM approach appears promising, when applied to our toy-problems, there are several challenges that need to be addressed when applied in real-world applications:

- Data problems such as insufficient, incomplete and noisy data.
- Verification and validation problems related to black-box representations such as neural networks that are difficult to visualize and analyze by humans experts.
- Real-time simulation problems caused by advanced feature extraction functions (e.g. terrain analysis, route planning).
- The need for intuitive and easy-to-use DDBM authoring tools capable of visualizing, editing and processing datasets acquired synthetically or from military exercises.

In future works we intend to evaluate the DDBM approach in a real-world application targeting for example the Military Operations in Urban Terrain (MOUT) domain. Using the MOUT application we will conduct studies to gain insight into DDBM capabilities with respect to improving modeling efficiency, behavior realism and objectiveness.

ACKNOWLEDGMENT

The work presented in this paper is the result of a collaborative effort by the Swedish Defence Research Agency (FOI) and the Norwegian Defence Research Establishment (FFI). The work was supported by the FOI research project “Synthetic Actors”, which is funded by the R&D programme of the Swedish Armed Forces, and by the FFI research project ”LVC simulation for training, exercises and experiments”.

REFERENCES

[1] David Aihe and Avelino Gonzalez. Context-driven reinforcement learning. In *Proceedings of the Second Swedish-American Workshop on Modeling and Simulation*, Cocoa Beach, FL, 2004.

[2] Christopher G. Atkeson and Stefan Schaal. Learning tasks from a single demonstration. In *In Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, pages 1706–1712, 1997.

[3] D.C. Bentivegna and C.G. Atkeson. Learning from observation using primitives. In *IEEE International Conference on Robotics and Automation (ICRA).*, volume 2, pages 1988–1993, 2001.

[4] Murray Campbell, A. Joseph Hoane, Jr., and Feng-hsiung Hsu. Deep blue. *Artif. Intell.*, 134(1-2):57–83, jan 2002.

[5] Hans Karl Gustav Fernlund. *Evolving Models from Observed Human Performance*. PhD thesis, University of Central Florida, 2004.

[6] David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. Building watson: An overview of the DeepQA project. *AI Magazine*, 31(3):59–79, 2010.

[7] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The WEKA data mining software: An update. *SIGKDD Explorations*, 11(1), 2009.

[8] Kwun Han and Manuela Veloso. Automated robot behavior recognition applied to robotic soccer. In John Hollerbach and Dan Koditschek, editors, *Robotics Research: the Ninth International Symposium*, pages 199–204. Springer-Verlag, London, 2000. Also in the Proceedings of IJCAI-99 Workshop on Team Behaviors and Plan Recognition.

[9] Cynthia L. Johnson and Avelino J. Gonzalez. Learning collaborative behavior by observation. In Sorin Draghici, Taghi M. Khoshgoftaar, Vasile Palade, Witold Pedrycz, M. Arif Wani, and Xingquan Zhu, editors, *ICMLA*, pages 99–104. IEEE Computer Society, 2010.

[10] Cynthia L. Johnson and Avelino J. Gonzalez. Learning collaborative team behavior from observation. *Expert Syst. Appl.*, 41(5):2316–2328, 2014.

[11] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, USA, 1992.

[12] Linus J. Luotsinen and Ladislau Bölöni. Role-based teamwork activity recognition in observations of embodied agent actions. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '08, pages 567–574, Richland, SC, 2008. International Foundation for Autonomous Agents and Multiagent Systems.

[13] Klaus Meffert. JGAP - Java Genetic Algorithms and Genetic Programming Package. `http://jgap.sf.net`.

[14] Kathryn Merrick and Mary Lou Maher. Motivated reinforcement learning for non-player characters in persistent computer game worlds. In *Proceedings of the 2006 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*, ACE '06, New York, NY, USA, 2006. ACM.

[15] Ingo Mierswa, Michael Wurst, Ralf Klinkenberg, Martin Scholz, and Timm Euler. Yale: Rapid prototyping for complex data mining tasks. In Lyle Ungar, Mark Craven, Dimitrios Gunopulos, and Tina Eliassi-Rad, editors, *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 935–940, New York, NY, USA, August 2006. ACM.

[16] Ian Millington and John Funge. *Artificial Intelligence for Games*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2nd edition, 2009.

[17] Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997.

[18] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing Atari with deep reinforcement learning. In *NIPS Workshop on Deep Learning*. arXiv.org, 2013.

[19] Santiago Ontanon, Jose L. Montana, and Avelino Gonzalez. Towards a unified framework for learning from observation. In *IJCAI Workshop on Agent Learning Interactively from Human Teachers*, 2011.

[20] Craig W. Reynolds. Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Comput. Graph.*, 21(4):25–34, August 1987.

[21] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Neurocomputing: Foundations of research. chapter Learning Representations by Back-propagating Errors, pages 696–699. MIT Press, Cambridge, MA, USA, 1988.

[22] Gary Stein and Avelino J. Gonzalez. Building high-performing human-like tactical agents through observation and experience. In *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, pages 792–804, 2011.

[23] Gary Stein, Avelino J. Gonzalez, and Clayton Barham. Combining NEAT and PSO for learning tactical human behavior. *Neural Computing and Applications*, pages 1–18, 2014.

[24] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction*. MIT Press, Cambridge, MA, 1998.

[25] Teck-Hou Teng, Ah-Hwee Tan, and Loo-Nin Teow. Adaptive computer-generated forces for simulator-based training. *Expert Systems with Applications*, 40(18):7341–7353, 2013.