Controlling Computer-Generated Life-Forms Using a Fuzzy State Machine

Hung Q Tran, Nguyen K Tran
CAE USA Inc.
Tampa, FL
hung.tran@caemilusa.com, nguyen.tran@caemilusa.com

ABSTRACT

Simulated real-world environments represent a powerful tool for learning and training. Intelligent Virtual Agents (IVAs) can interact with people in these environments, either to facilitate training tasks or to satisfy learning requirements that would normally require additional human participants. An IVA is a computer-generated "life-form" that is not only visually similar to a human but also behaviorally like a human. An IVA can be thought of as a simulated system situated in a virtual environment and capable of autonomous actions to meet its design objectives.

A suitable modeling and simulation approach to simulate the behavior of an IVA is a Finite State Machine (FSM). A FSM is a well-defined method of representing a range of behaviors within a logical framework. However, using a FSM to model human behavior has a couple of drawbacks:

- A FSM can become very complex, especially when the IVA is required to perform its role autonomously and respond to the surrounding environment in a timely fashion.
- A simpler FSM to model the IVA's behavior can appear predictable and repetitive, if not to say unrealistic.

This paper proposes using a Fuzzy State Machine (FuSM) simulation approach to reduce the predictability of IVA behavior while enhancing autonomy with artificial intelligence (AI) during training. First, the fuzzy logic technique and its basic rules are introduced. Then, we will explain the development process and how it can be applied to efficiently control IVA behavior. The effectiveness of the FuSM modeling approach is assessed through the implementation of a typical Navy mission training application. Finally, this paper presents preliminary simulation results, which indicate that it is possible to easily control the behavior of computer-generated life-forms using the FuSM modeling technique.

ABOUT THE AUTHORS

Hung Q. Tran is a Tactical Systems, Aircraft Sensors, and Interoperability Group Leader at CAE USA. He joined CAE USA more than 20 years ago and has worked on the modeling and simulation of several Electronic Warfare (EW) systems. Hung is the lead designer of the Computer-Generated Forces (CGF) currently used by all United States Air Force C-130J Weapons System Trainers. Hung holds a Bachelor of Science in Electrical Engineering from Polytechnique Montréal at University de Montréal in Quebec, Canada.

Nguyen K. Tran is a Flight Dynamic Software Engineer at CAE USA. Khoi has worked on the design of several autonomous drone systems. Khoi holds a Bachelor of Electrical Engineering and a Master of Mechanical Engineering from the McGill University, Montreal, Canada.

Controlling Computer-Generated Life-Forms Using a Fuzzy State Machine

Hung Tran, Nguyen K Tran
CAE USA Inc.
Tampa, FL
hung.tran@caemilusa.com, nguyen.tran@caemilusa.com

INTRODUCTION

Computer Generated Forces (CGF) training scenarios consist of a set of predefined events that occur during training (i.e., scripted scenario). From a modeling perspective, it involves setting the parameters of computer-controlled entities. When a desired state of simulation occurs, the computer-controlled entity executes a series of actions, as it was preprogrammed to do. Occasionally, the instructor may intervene and perform additional actions to achieve the desired state of the training task. CGF-generated "life-forms" represent a special type of computer-generated entity. If the simulated life-forms represent Intelligent Virtual Agents (IVAs), they must be capable of autonomous actions to meet its design objectives. An intelligent and autonomous virtual agent must possess the following characteristics:

- Reactive – A virtual agent should perceive the surrounding environment and respond to the interpretation of the environment in a timely fashion. This characteristic is directly related to the perceptual models used by IVAs.
- Proactive – A virtual agent should not simply act in response to its environment; it should be goal-oriented. Therefore, it must take initiative, where appropriate, to reach the predefined simulation objectives.
- Social – A virtual agent should be able to interact with other virtual agents or with humans collocated in the same environment.

Artificial intelligence (AI) is used in a multitude of applications, such as electronics, medicine, machine industry, and military. The overall research goal of AI is to be able to use computing technology to operate in an intelligent manner, similar to how humans think. Just as in applications, there are many AI techniques, such as artificial neural networks, genetic algorithms, cellular automata, multi-agent systems, and swarm intelligence (Russell & Norvig, 1995). Fuzzy logic is one such AI technique that could be used to simulate human reasoning. As opposed to the usual "true" or "false" Boolean logic (1 or 0), fuzzy logic represents an approach to computing based on the "degree of truth." This type of logic manipulates values with more than just a simple true or false. Furthermore, when linguistic variables are used, they can be represented by the degree of truthfulness or falsehood. For this reason, fuzzy logic can handle complex control problems at low computational cost and still preserve the subtle aspect of human thinking. In fact, fuzzy logic control is not just another control method—it represents a quality behavior generator. Compared to other intelligent techniques, fuzzy rule-based systems are easy to design and implement because they require only the usage of a rule-based set extracted from a knowledge database (Magdalena, 2015). A fuzzy rule-based expert system contains fuzzy rules defined by its knowledge database. It derives conclusions or decisions from user inputs and the fuzzy reasoning process. The rule-based expert system uses very descriptive linguistic variable terms (e.g., Low, High, Fast) to deal with input and output data like a human operator would. Moreover, the knowledge database can grow incrementally, so the performance of the system improves while it is being developed. CGF-generated human-like entities present a perfect application for using the fuzzy rule-based expert system to model their behavior. This modeling technique provides the constructive entities with human-like behavior and decision-making, instead of predictive machine-controlled behavior.

The process of developing a classical expert system is illustrated in Figure 1. Engineers establish a dialogue or interview with a subject matter expert (SME). The goal is to gather experts' knowledge and then encoded it to construct the rule-based database. The SME reviews and evaluates the rule-based database to provide any necessary adjustments. This process goes through many iterations until the knowledge database is completed.

The purpose of this paper is to present and discuss the modeling of CGF-controlled human-like entities using an AI technique—the Fuzzy State Machine (FuSM). The entities generated by this novel method will behave differently according to changing situations. Their behavior and decision-making are governed by a set of fuzzy logic rules. First, the fuzzy logic technique and its basic rules are introduced. Then we will explain the development process and how it

can be applied to efficiently design training scenarios to achieve more realistic mission training. Finally, we will present and discuss an example implementing this technique to simulate the behavior of virtual pirate entities used for an anti-piracy mission training.

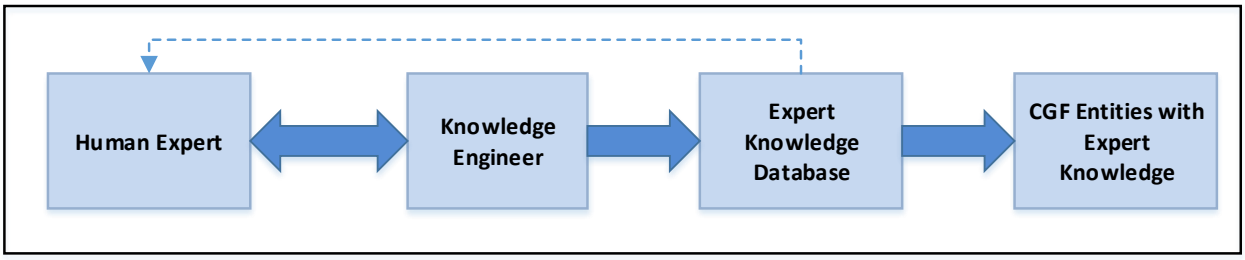

Figure 1. Development of an Expert System

FINITE STATE MACHINE AND FUZZY STATE MACHINE

A suitable modeling and simulation approach to simulate the behavior of an IVA is a Finite State Machine (FSM). An FSM is a well-defined method of representing a range of behaviors within a logical framework. However, it is well known that an FSM will explode in complexity when more states are added, especially when the IVA is required to perform its role autonomously and respond to the surrounding environment in a timely fashion. Another drawback when using an FSM to model IVA action is the sudden change from one state to another state, which results in an unnatural abrupt switch from one decision to another decision during the simulation. State oscillation is another issue when using an FSM to model IVA behavior. The state oscillation condition occurs when the trigger conditions for an IVA to move from state to another state are represented by fixed values. For instance, if hovering at 500 ft. represents a simulation condition where a helicopter changed from one state to another state, it will oscillate from one state to another state, creating an undesired jittering condition.

A gradual change from one state to another state can be achieved with fuzzy logic. Fuzzy logic processes input into the system by using rule-based sets, which are derived from experts of the domain (i.e., SMEs), without having to compute any complex mathematic model. For this specific reason, fuzzy logic is often used to model complex behavior with low computation cost. FuSM represents an extension of the FSM by introducing the fuzzy logic. There are two main approaches to transform an FSM into a FuSM. The first method is to introduce a fuzzy rule-based set during the transition from one state to another state. This method is straightforward to implement with a net benefit of providing a gradual change during the state changes, which is very effective to model IVA behavior. The second approach employs fuzzy states; an IVA can belong to different states at the same time with a different degree of membership. In this paper, we will consider the first approach to model the transition of an IVA from one state to another state.

Another advantage of using an FuSM is related to the linguistic nature of the fuzzy logic. Application-domain SMEs formulate the rule-based expert system, which can then be used to emulate the reasoning similar to the experts. Finally, a fuzzy logic approach has certain advantages over other approaches, such as its ability to use perceptual information and human experience and knowledge, to emulate human thought processes.

FUZZY INFERENCE SYSTEM OVERVIEW

Fuzzy inference (reasoning) is the process of mapping from a given input to an output using fuzzy logic. The basic structure of an inference fuzzy system consists of three main conceptual elements:

1. A rule-based set, which contains a selection of fuzzy rules
2. The definition of the membership functions used in the fuzzy rules
3. A reasoning mechanism, which performs the inference procedure to derive a reasonable conclusion and decision

The concept behind fuzzy reasoning can be represented by fuzzy logic control (FLC), which was derived from control theory and is based on mathematical models of the open-loop control process. FLC has been successfully applied to

several practical applications, as simple as room temperature control and washing machine cycles or as complex as power systems or nuclear reactors. The components of FLC are illustrated in Figure 2.

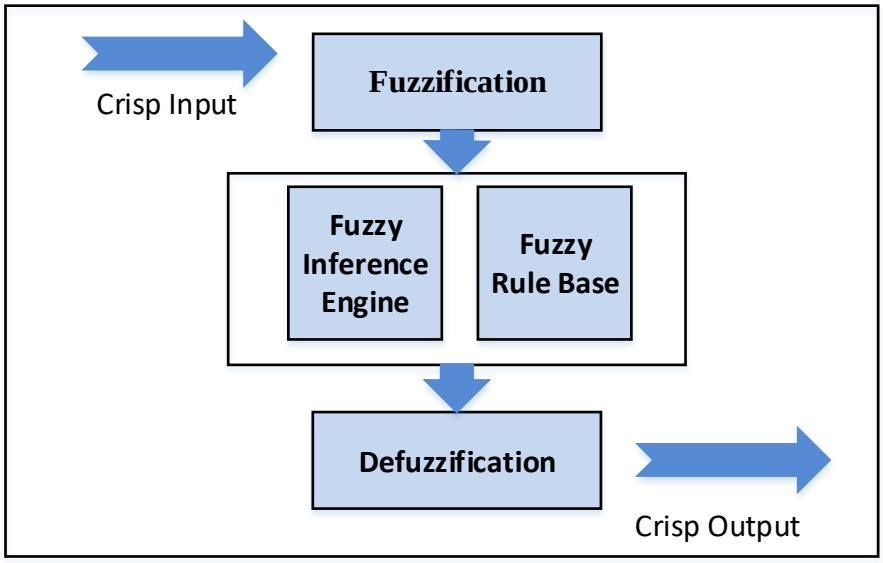

Figure 2. Fuzzy Logic Controller

The three components of an FLC are:

1. Fuzzification of crisp input parameters is a process that consists of converting crisp input parameters into suitable linguistic values represented by a set of fuzzy logic rules. The fuzzy rules contained in the fuzzy logic set were constructed from the knowledge database. The database provides definitions used to define linguistic control rules and data. The rule characterizes the control logic of the domain experts.
2. The Fuzzy Inference Engine represents the FLC processor and simulates human decision-making based on an established Fuzzy Rule Base.
3. Defuzzification is the process of reconverting fuzzy linguistic parameters back to crisp parameters required for the control process.

There are three main types of FLC systems: Mamdani (Mamdani & Assilian, 1975), Sugeno (Sugeno & Kang, 1988), and Tsukamoto (Tsukamoto, 1979). In this work, we use the Mamdani fuzzy inference system to implement the AI rules for CGF-controlled life-forms. The Mamdani method extends the fuzzy controller for use with intervals or linguistic values as inputs. For example, instead of expressing a simulation input like "Entity altitude is equal to 100 ft.," an equivalent fuzzy logic rule would be "Entity altitude is low." The latter expression corresponds better to human reasoning. The Mamdani FLC system refers to a fuzzy controller with N inputs X1, X2, …Xn and one single Y output, with a fuzzy logic rule like "If X1 is A1 or X2 is B2 Then Y is C1." A real-life example of this fuzzy logic rule for a soldier combatant can be similar to "**If** enemy is Far **and** enemy movement is Fast **then** probability to shoot and hit the target is Low."

To illustrate how a Mamdani FLC is implemented, we examined a typical problem with two inputs (X = Target range, Y = Target speed), one output (Z = probability of kill), and a set of three rules. In this example, we used membership functions that were either trapezoidal or triangular shape. Many other shapes, such as a Gaussian shape, have been used for different applications (Klir & Yuan, 1995). Normally, a specific shape is chosen based on the application and availability of data. The following set of rules was considered:

If **target range is high** or **target speed is high**, then **the probability to shoot and hit the target is low**
If **target range is moderate** and **target speed is low**, then **the probability to shoot and hit the target is moderate**
If **target range is low**, then **the probability to shoot and hit the target is high**

In this example, **A** is the fuzzy set of target range, **B** is the fuzzy set of target speed, and **Z** is the fuzzy set of the probability to hit the target if the soldier takes a shot at the enemy. If the current distance of the target is 60 ft and if the target moves at a speed of 70 ft/min, the process of fuzzification and defuzzification can be illustrated as follows:

Step 1: Fuzzification

Fuzzification consists of projecting the crisp input value (**X**) to the corresponding fuzzy sets, as illustrated in Figure 5. The projection of a crisp input "**X**" to a fuzzy variable "**A**" is mathematically expressed by $\mu_{(X = A)}$. Thus, the input target range **X** = 60 projected to the fuzzy variables of "Low" with a value of 0.5, "Moderate" with a value of 0.4, and "High" with a value of 0.0. The values of 0.5, 0.4, and 0.0 represent the degree of membership that **X** belongs to the fuzzy variables of "Low," "Moderate," and "High," respectively. Similarly, the values 0.1 and 0.7 represent the degree of membership that target speed **Y** = 70 ft/min belongs to the fuzzy variables of "Low" and "High," respectively.

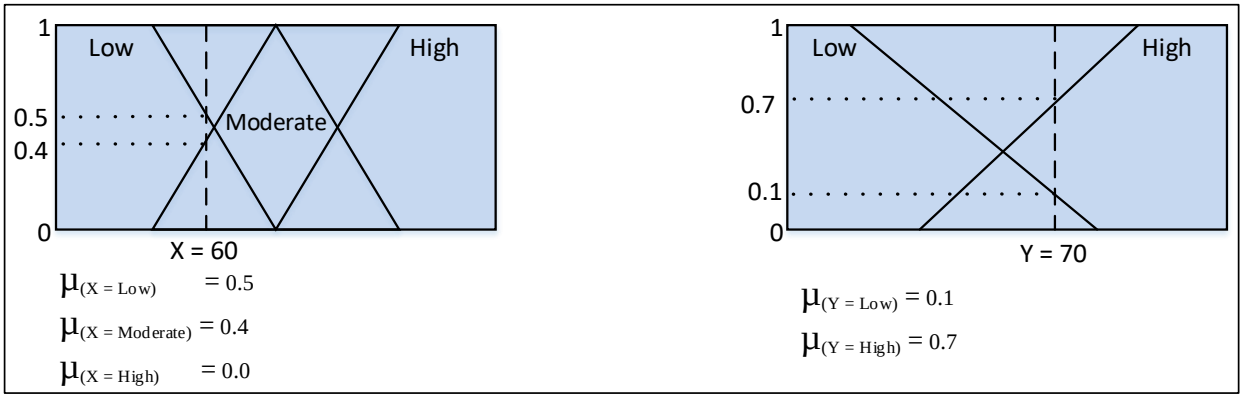

Figure 3. Fuzzification

Step 2: Rules evaluation

In this step, all three rules are evaluated for each fuzzy set of target range, target speed, and probability of hit the target, as illustrated in Figure 4.

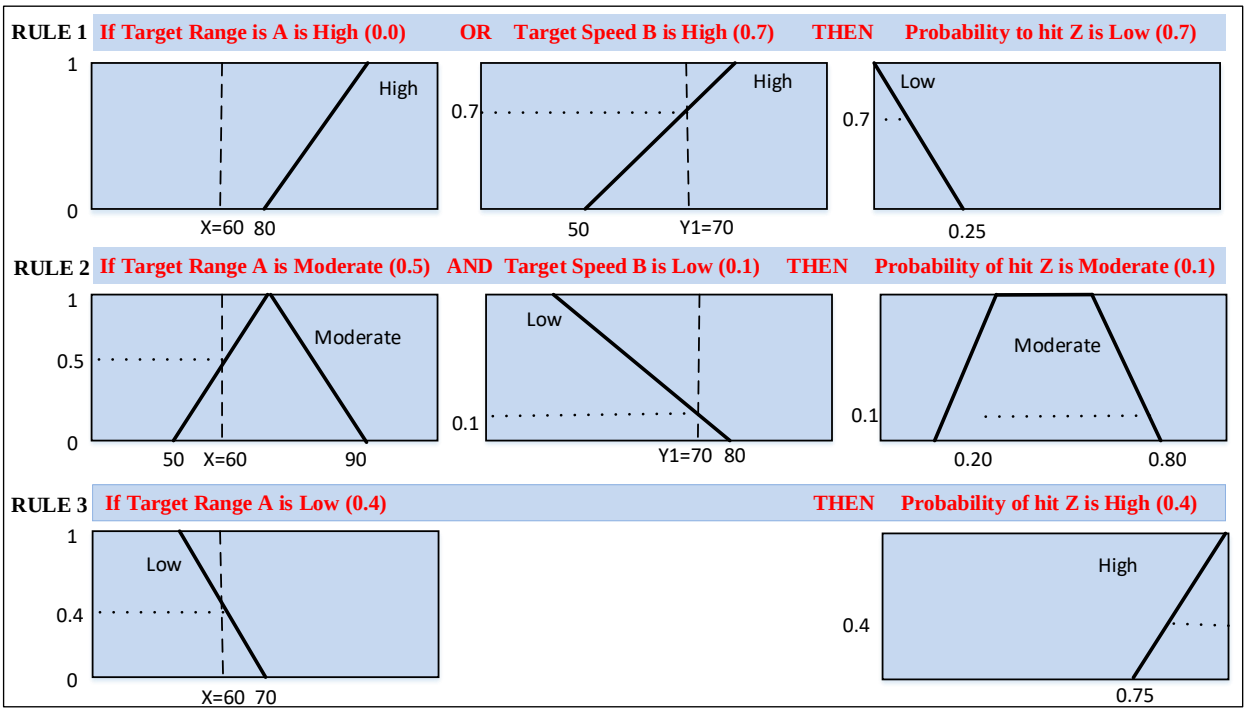

Figure 4. Fuzzy Logic Rules Evaluation

The conditional clauses in the rule-based set were computed as follows:

"X ***OR*** Y = **MAX** (X, Y)" and "X ***AND*** Y = **MIN** (X, Y)".

Step 3: Evaluation of the rule output
In this step, all the three outputs are aggregated. As illustrated in Figure 5, the aggregation is the process of combining the membership functions of all rules (Low, Moderate, and High) previously clipped or scaled into a single fuzzy set.

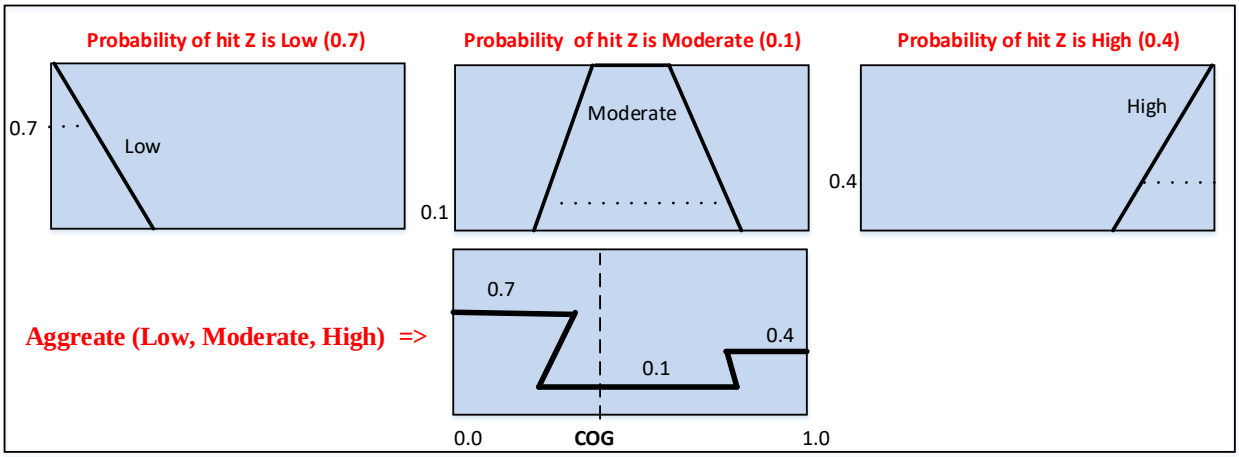

Figure 5. Evaluation of the System Output

Step 4: Defuzzification
Fuzziness was used to evaluate the rule base. To be useful in a modeling system, the final result must be converted back to a crisp output. This process is known as "Defuzzification." There are several defuzzification methods, such as central sums, mean of maxima, and left-right maxima, but the most popular is the Centroid technique. This method determines the point where a vertical line will divide the aggregate set into two equal masses. Mathematically, this point represents the Center of Gravity (COG) and can be expressed as:

$$COG = \frac{\int_a^b \mu\,(x)\; x dx}{\int_a^b \mu(x) dx}$$

As seen in the example above, a Mamdani FCL represents a simple method to evaluate fuzzy logic rules. Although this method is simple, it remains efficient because it uses a rule-based set in linguistic terms, which are equivalent to human thinking and decision-making. The key of successfully modeling CGF entities with Mamdani FCL resides in the ability to acquire and construct the database of knowledge.

PREVIOUS WORKS

Abdellaoui, Taylor, and Parkinson (2005) conducted a comparative study of existing CGF tools. The following seven CGF tools were considered in this study: Joint Semi-Automated Forces (JSAF), One Semi-Automated Forces (OneSAF), XCITE, VR-Forces, STAGE, VBS, and Sonalysts Combat Simulations. The purpose of their study was to identify the AI features missing from each tool, with the end goal to develop an AI module designed to address these gaps. The analysis was based on a requirement wish list that included five main AI categories: autonomous operation, learning, organization, realism, and architectural requirements. Realism represented the subjective characteristics of the entity. In the context of the CGF tool, realism meant that the autonomous entity behaved as if it was controlled by a human instead of a machine. In other words, a constructed entity with adequate realism will behave in ways similar to a human-controlled entity. The results of this study indicated only four of the seven CGF tools that were evaluated met the AI realism standard score (i.e., some degree of realism could be observed from the constructed entities of these four CGF tools). For the aspect of training, autonomous operation referred to the capability of virtual entities to operate without the intervention of the instructors and the ability to adapt and respond adequately to the change of the surrounding environment. Only one CGF tool (VR-Force) demonstrated the capability to construct entities with

autonomous behavior. However, the capability is limited to predefined behaviors, such as "run" or "argue", and the pre-defined behaviors do not react to the change of the surrounding environment.

The above CGF tools also use different AI approaches, such as Hierarchical Finite State Machine, Hierarchical Goals Structured Ruled-base Systems, Constraint-Satisfaction Systems, and Hierarchical Planning Systems (Laird, 2000). While these techniques are fundamentally different from each other, they all share the common goal of encoding AI behavior about strategy, tactic, and doctrine. A fuzzy logic-based system represents another AI technique that can generally be used to deal with uncertain and/or imprecise information. Additionally, fuzzy logic uses linguistic variables in the same way as human reasoning, such as "low," "high," "slow", or "fast." For this specific reason, we believe this approach is more suitable for CGF applications where precise input values are unavailable or are difficult to estimate (e.g., classified values).

A USE CASE OF NAVAL MISSION TRAINING

A typical naval mission is Anti-Piracy. The primary goal of the anti-piracy mission is to disrupt piracy and armed robbery at sea in order to protect global maritime commerce and secure freedom of navigation.

A virtual training environment designed for anti-piracy mission training requires the capability to generate virtual pirate entities. To be effective, computer-generated pirate entities must capable of autonomous actions and their behavior should be driven by "intelligent" decision-making principle similar to human.

A reasonable set of characteristics defining a virtual pirate entity is:

- Health: health of the pirate
- Vision distance: distance the pirate can see
- Distance to enemy: true distance between the pirate and the enemy
- Ammo: remaining ammo
- Running velocity: speed the pirate can move

Typical action states of a pirate are:

- Idle / Patrol
- Move to the combat position
- Fire at the target
- Reloading
- Run to Hide

Table 1 provides the description of the pirate simulation use case.

Table 1. Pirate Simulation Use Case

Use Case	**Description**	
Simulation of Pirate behavior	Navy helicopter flies to the boat occupied by the pirates and attacks the pirates. Pirates fight back.	
Description	**State**	**Behavior**
No target in sight	Idle / Patrol	Pirate entity initialized at "Idle / Patrol" state on a boat
Target sighted – within vision range	Combat position	Navy helicopter is within vision range and position for attack
Target within firing range	Fire at the attack helicopter	Navy helicopter is within firing range, then fire at the target
Pirate is low on ammo	Reload weapon	If pirate is low on ammo, then reload weapon
Re-engage the target after weapon reloaded	Fire at the attack helicopter	Navy helicopter is within firing range, then fire at the target
Target out of firing range	Combat Position	Navy helicopter is out of firing range, but within vision range
Target out of sight	Idle / Patrol	Navy helicopter fly out of vision range
Pirate is injured	Run to hide	Pirate injured, stop firing

Figure 6 presents the state machine diagram of the pirate simulation use case.

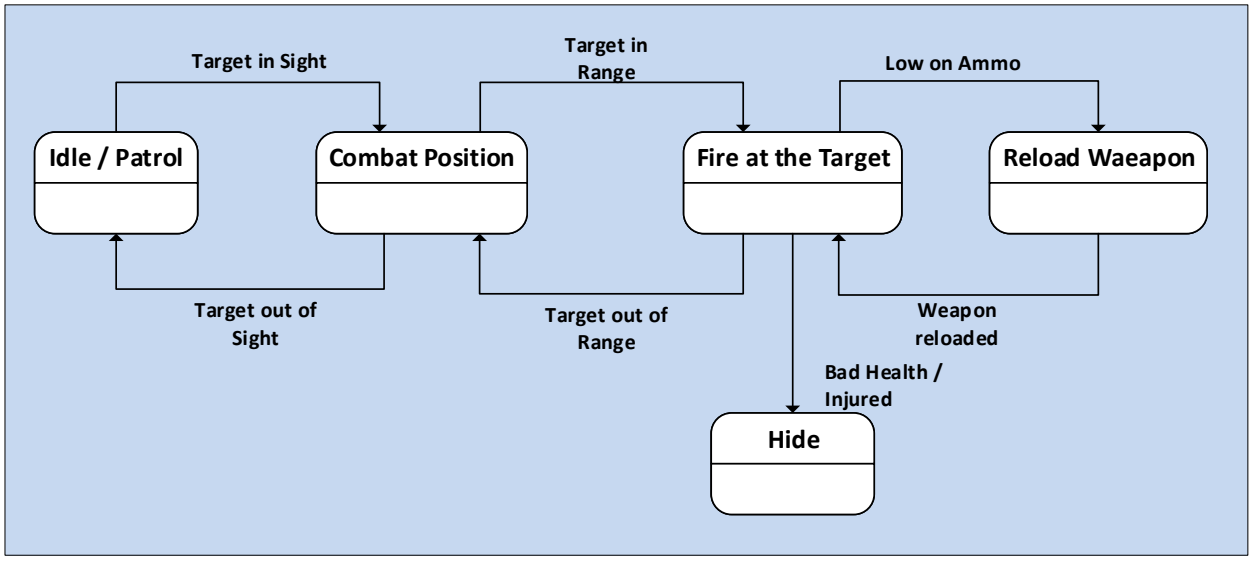

Figure 6. Pirate Entity Action States

IMPLEMENTATION AND SIMULATION RESULTS

The implementation of the pirate use case as illustrated in Figure 6 is straightforward and can be accomplished with a classical FSM algorithm as illustrated in Figure 7.

```
Initialize State to “Idle / Patrol”
Switch State
   Case Idle / Patrol
     If Target is in sight then
        Set State to “Combat Position”
   Case Combat Position
      If Target in Range
         Set State to “Fire at the target”
      If Target is “out of sight”
         Set State to “Idle / Patrol”
   Case Fire at the Target
      If “Low on ammo” then
         Set state to “Reload weapon”
      If “Bad heath / Injured” then
         Set state to “Hide”
      If Target out of range then
          Set State to “Combat Position”
   Case Reload Weapon
      If “weapon reloaded”
          Set State to “Fire at the target”
     End case
End Switch state
```

Figure 7. FSM of the Pirate Entity simulation

For the FSM algorithm, all transition logic is encoded in a set of conditional logics. For instance, the pirate will switch from “Idle / Patrol” to “Combat position” when the Navy helicopter is “in sight.” This condition is normally

implemented like this: "If distance of helicopter is < 5 nmi." As previously mentioned, a FSM is suitable to implement such use cases. However, the simulation using such FSM logic will look very repetitive and unrealistic. For instance, 5 nmi is the cut-off value, therefore the pirate will repeatedly switch to the state "Combat Position" every single time the Navy helicopter reached exactly 5 nmi of distance. Additionally, the simulation will be very sensitive to oscillation conditions, which will make the pirate simulation oscillate from one state to another state when the distance of the Navy helicopter is around the cut-off value.

To address the issue of an FSM, a fuzzy logic set was introduced to model the transition between simulation states. Therefore, instead of using a single condition to determine which state the pirate is currently in, a fuzzy logic rule-based set was designed. A fuzzy logic rule-based set for 4 inputs variables (Health, Vison distance, Distance to enemy and Ammo) and three set members for each fuzzy set ("Low", "Moderate" and "High") will contain a total of 43 fuzzy logic rules. A sample of these rules is illustrated in Figure 8.

If target range is low and ammo is high then fire at the target
If target range is moderate and ammo is high fire at the target
If target range is high or ammo is low then reload the weapon

If ammo is high and health is high then fire at the target
If ammo is moderate and health is high the fire at the target
It ammo is low or health is low then run to hide

Figure 8. Sample of fuzzy logic rules used for Pirate Entity simulation

We observed that by using a fuzzy logic rule-based set, which normally derived from an expert, the pirate would behave in a more intelligent manner. Furthermore, a fuzzy logic set, which used the linguistic terms (e.g., "low", "moderate" or "high"), produces a decision-making process similar to what a SME would produce.

Both FSM and FuSM approaches were implemented in an existing synthetic environment that has the capability to enable human-like entities. The simulation results are summarized in the following sections:

Finite State Machine
As expected, during the simulation the pirate switched from one state to another state. At any time, the pirate always belonged to a specific state. The transition occurred when the trigger conditions are met, and consequently, the current state is replaced by a new state. Additionally, because the trigger conditions in the FSM are set to fixed values, the pirate switched immediately from the current state to the new state when input values satisfied the trigger conditions (Boolean logic). While the FSM technique is simple to implement, the simulation looks predictable and repetitive. For this reason, once the trainees have completed a scenario, they will likely know how it will behave during the next training session, thereby reducing or removing the reuse value of the scenario.

Fuzzy State Machine
FuSM was introduced to enhance the realism of the pirate during the execution of the scenario. Therefore, the pirate behaves in a more intelligent fashion. During the simulation, we observed that while the pirate also switched from one state to a new state similar to the FSM, his movements appear to be smoother because the decision of switching states is now computed using a fuzzy logic set. The fuzzy logic set used a combination of two or more input variables (e.g. "if ammo is low and health is low then run to hide"), instead of using a single fixed condition (e.g. "if health < constant then run to hide"). Another important observation is the execution of the scenario is less predictable because the fuzzy logic set is using linguistic terms (e.g. "low" or "high") instead of classical crisp values.

When comparing the simulation performance in terms of "autonomous operation" and "realism", the utilization of FuSM in this type of training scenario provides some advantages over the FSM:

- A smooth transition between states prevent undesired effects, such as oscillation or a sudden switch to a new state, that could happen during the execution of the training scenario.

- By using linguistic terms to determine the transition between states, FuSM introduce a mode of reasoning similar to human. This approach improves the IVA autonomous operation, and consequently the IVA behavior looks more realistic.

DISCUSSION

Military training requires more complex and dynamic training scenarios within a synthetic environment. As illustrated by the implementation of a typical Navy mission training use case, a FuSM represents a suitable approach to introduce complex concepts without the need for intricate formulas and massive amount of computations. Additionally, a fuzzy logic-based system can produce results similar to a complex system controller, but requires only the usage of a small set of fuzzy rules and can dynamically service a large range of input values. Another advantage of a FuSM is the computation of the solution does not have to be centralized like the FSM approach (i.e., all the logical computations between states are encapsulated by a "switch" statement). For instance, the fuzzy set determining the "probability of shoot and hit the target" as presented previously in this paper was used to generate the behavior of a Navy soldier that does the fighting with the pirate. Using this fuzzy set, the "health" of the pirate can be simulated.

The topic of this study is to propose the usage of an artificial intelligent technique (e.g. FuSM) to generate IVA behavior for military training. In this paper, we choose a typical Naval Mission training scenario to illustrate how a FuSM can be utilized to simulate IVA behaviors. Nonetheless, this approach can be applied to a wide range of military training applications requiring the usage of IVAs. Immediate training applications of FuSM for other military branches are:

- US Air Force: Simulation of an expert human pilot flying tactical fixed-wing aircraft for: dog fighting, Air-to-Air refueling (AAR) mission, Station Keeping Equipment (SKE) for formation fly, etc.
- US Army: Virtual Warfare Scenario (VWS) where IVAs can represent either team members or enemy combatants, Integrated Air Defense System (IADS) operators, etc.
- US Coast Guard: Simulation of rescuers or survivors in Search-and- Rescue (SAR) missions. For instance, SAR mission is applicable to all others US military branch.
- US Marines: Urban Warfare training.

CONCLUSION

The majority of current CGFs use a FSM approach to simulate behavior and decision-making. In this paper, we described an AI approach that can be used to drive the behavior of human-like entities generated from a CGF. Instead of using a classical FSM approach alone to model the pirate behavior, this paper proposed to enhance the FSM approach by introducing the fuzzy logic set to model the transition between states. Furthermore, using variable linguistic terms can emulate human-like decision-making processes and is suitable to emulate many aspects of human-like decision-making for entities generated by CGFs.

Several topics related to the approach presented in this paper to simulate IVAs behavior can be addressed in future research. One of the topics is to extend the role of the instructors. We all agree that human experts play an important role in military training as they are usually employed as instructors. Besides the instructor role, their expertise can be expanded into the creation of expert rule-based behaviors, which will be used to simulate the behavior and decision-making of virtual constructive entities (e.g. IVA). The modeling and simulation approach as presented in the paper provide this required framework, so the human experts can contribute beyond their usual instructor role.

REFERENCES

Abdellaoui, N., Taylor, A., Parkinson, G. (2009). Comparative Analysis of Computer Generated Forces Artificial Intelligence. Proceedings of the NATO RTO Modelling and Simulation Group Symposium. 2. 1-2.12.

Ball, R. E. (2003). The Fundamentals of Aircraft Survivability Analysis and Design, 2nd Edition. American Institute of Aeronautics & Astronautics.

Birkmire, B. (2011). Weapon Engagement Zone Maximum Launch Range Approximation Using Multilayer Perceptron. Wright State University.

Klir, G. J., Yuan, B. (1995). Fuzzy Set and Fuzzy Logic. Prentice Hall PTR.

Laird, J. E. (2000). An Exploration into Computer Games and Computer Generated Forces. Eighth Conference on Computer Generated Forces and Behavior Representation. Orlando, FL.

Macfadzean, R. H. M. (1992). Surface-Based Air Defense System Analysis. Artech House Edition.

Magdalena, L. (2015). Fuzzy Ruled-Base Systems. Springer Handbook of Computational Intelligence. Chapter 13. 203-218.

Mamdani, E. H., Assilian, S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller, Int. J. Man-machine Studies 7: 1–13.

Robert E. B (2003). The Fundamentals of Aircraft Survivability Analysis and Design, 2nd Edition. American Institute of Aeronautics & Art.

Robert H. M.M (1992). Surface-Based Air Defense System Analysis. Artech House Edition.

Russell, S. T., Norvig, P. (1995). Artificial Intelligent: A Modern Approach. Prentice Hall, Inc.

Sugeno, M., Kang, G. T. (1988). Structure identification of fuzzy models. Fuzzy Sets and Systems 28: 15-33.

Tsukamoto, Y. (1979). An approach to fuzzy reasoning method. In Madan M. Gupta, Rammohan K. Ragade, and Ronald R. Yager, editors, Advances in Fuzzy Set Theory and Applications, North-Holland, Amsterdam: 1979, pp.137-149.

Zadeh L. A. (1965). Fuzzy Sets. Intl J. Information and Control. Vol 8. Issue 3: 338-353.