Proceedings of the 2024 Winter Simulation Conference
H. Lam, E. Azar, D. Batur, S. Gao, W. Xie, S. R. Hunter, and M. D. Rossetti, eds.

ARTIFICIAL INTELLIGENCE AND SIMULATION FOR ENHANCED PILOT TRAINING

Larry Lowe[1], Luis Rabelo[1], Marwen Elkamel[1], Mitchell Hunsucker[1], Katalina Arias-Marin[1], Nathalia Davila[1], Omar Allaz[1], Mario Marin[1], Gene Lee,[1]
[1]Industrial Engineering & Management Systems, University of Central Florida, Orlando, FL, USA

ABSTRACT

This paper discusses the integration of Virtual Constructive (VC) simulation and Convolutional Neural Networks (CNNs) into an Agent-Based Model (ABM) to study pilot performance. By leveraging the strengths of VC for immersive training scenarios and CNN for advanced image recognition and decision-making processes, the research aims to provide a comprehensive understanding of how AI and machine learning can help pilot training programs. The paper's series of experiments within the ABM demonstrate the potential of this integrated approach to improve decision accuracy and response times under simulated operational conditions. The findings underscore the effectiveness of integrating VC and CNN into an ABM for training simulations, with implications for pilot training and developing environments that capture operator behavior.

1 INTRODUCTION

The aerospace sector faces pressing demands to adapt and innovate in an evolving global landscape. As new technologies redefine aircraft capabilities and operational landscapes, pilots must adapt their skills and strategies, enforcing the importance of advanced pilot training. Simulations can play a pivotal role by offering pilots a platform to learn and perfect their skills to navigate these scenarios safely. Life simulations and games constitute a subset of simulation in which players control virtual characters (Adams and Rollings 2006). These concepts have been proven instrumental in training, particularly in strengthening weapons combat effectiveness (Kirby et al., 2011). Nonetheless, their standalone deployment poses challenges from resource constraints, reproducibility issues, and associated costs (Kirby et al. 2011, West et al. 2013). Moreover, live simulations may impede training retention from simulated environments to real-world applications (Summers 2012). Virtual simulations provide solutions but also come with limitations. One approach to address these limitations is achieving a 'Live/Synthetic Balance,' which combines live training with synthetic environments to optimize training outcomes and effectively capture operator behavior (Kirby et al., 2011). Leveraging performance data extracted from live and synthetic environments can provide comprehensive insights (Tannenbaum et al., 1993). Evaluating the performance data and implementing machine learning models create the potential for human strategy refinement (Muller 2002).

Multiple papers have explored standalone Live Virtual Constructive (LVC) systems and combat effectiveness analysis, employing diverse methodologies, including discrete event simulation (Armo 2000) and agent-based models (ABMs) (Connors 2015). In Armo (2000), discrete event simulation (DES) was utilized to analyze the effectiveness of a light antisubmarine warfare torpedo against a submarine with a countermeasure system. The ABM framework in Connors (2015) modeled pilot behavior and was employed to analyze weapon effectiveness, focusing on weapons' range, speed, and accuracy. Other simulation models were used to model the combat effectiveness of other military vehicles in complex scenarios, such as attack helicopters (Jung and Lee 2010) and Unmanned Ground Vehicles (Lee et al. 2015). Additionally, data-driven approaches have been instrumental in extracting valuable insights from logs concerning reaction time, strategy, and design limitations. These methodologies, which often integrate machine learning techniques (Oztekin et al., 2013), offer a nuanced understanding of system usability and

979-8-3315-3420-2/24/$31.00 ©2024

its implications for weapon combat effectiveness (WCE). Moreover, combining data-driven and judgment-driven approaches has emerged as a promising avenue for comprehensive analysis (Hassenzahl 2000). Various judgment-driven methods, such as questionnaires, Subject Matter Experts (SMEs) heuristics, user interviews, and focus group data analysis, have been proposed in diverse contexts ranging from Navy recruit training (Tannenbaum et al., 1993) to pilot performance evaluation (Proctor et al., 2007). However, these methods prioritize training satisfaction and lack direct connections to performance metrics.

The effectiveness of simulations in enhancing pilot performance significantly depends on the ability to represent scenarios accurately. This is where integrating AI and Machine Learning (ML) technologies can aid in mitigating these concerns, allowing for improved decision-making. In Smith et al. (2000), a genetic algorithm was coupled with a constructive simulation to build effective aircraft tactics for one versus one scenario, and a similar approach was utilized for large-scale teams in air combat (Mulgund et al., 1998). Neural networks were used to evaluate ship-borne weapons (Gu et al., 2007) and for improved weapon effectiveness evaluation (Huang et al., 2005).

Based on an investigation of the literature, a gap exists that suggests integrating AI/machine learning with LVC models to address existing gaps and promote system improvement. Nonetheless, analyzing multiple VC simulations poses unique challenges, including resource constraints, safety concerns, and reproducibility issues. Therefore, achieving a "Live/Synthetic Balance" emerges as an optimal solution, enabling the development of effective methods for weapon effectiveness, operator training, package design, and sortie strategy through a comprehensive analysis of multiple simulations.

This paper utilizes a collaborative environment that incorporates virtual and constructive simulations. This environment facilitates the generation of scenarios by utilizing both manual (virtual simulation) and automated (constructive simulation) processes. These scenarios yield datasets suitable for performance-driven and judgment-driven analysis. This environment has been developed in our previous work (Lowe 2020), and a Convolutional Neural Network (CNN) was formulated to process and interpret visual data from the VC simulation, allowing for informed decision-making, stimulating pilot response, and enhancing training scenarios. In this paper, an ABM is formulated to simulate the complex interactions between pilots and their environment in various training scenarios, allowing for a comprehensive analysis. The integration of CNN within our ABM enables us to assess AI-enhanced decision-making processes, particularly in threat identification and responses, and their impact on pilot performance. This integrated approach aims to illustrate the potential of improving pilot training programs by enhancing realism and decision-making accuracy.

The subsequent sections of this paper are structured as follows. Section 2 outlines the establishment of the selected case study utilizing VC simulation. Section 3 details the methodology employed in this study, beginning with introducing the developed CNN model integrated into the ABM. It then delves into the formulation of the ABM and discusses the decision-making processes of each agent within the model. Section 4 presents the findings of the developed CNN network and the ABM. Finally, the paper summarizes the completed work and contributions and proposes avenues for future work.

2 CASE STUDY

This research employs a case study to address critical lower-level questions essential to the overall analysis. The developed case study (Lowe 2020) concentrates on capturing and investigating jet fighter pilot behavior. A data-driven approach is employed, and information is extracted from a VC simulation, utilizing the combat simulator SIMBox, widely adopted by various air forces worldwide, boasting 6 degrees of freedom capabilities and high-resolution geographical information systems. This extraction provides timestamps, attributes, locations, and operator actions. Challenges include the classification of unstructured data and accurately capturing operator performance. Model validation entails reaching a consensus among subject matter experts (SMEs), ensuring the precision of assumptions. The developed case study endeavors to acquire qualitative and quantitative data through judgment-driven and data-driven methodologies, thereby addressing limitations identified in the literature review.

A flight simulator served as the platform for VC experiments, with an F-16 pilot navigating scenarios involving SA-8 TELARs. The objectives encompassed capturing pilot performance and decision-making within a neural network, categorizing measure of effectiveness (MOE)-related satellite imagery, and utilizing virtual simulation data to predict participant outcomes regarding the MOE. The acquired behavior was integrated into an agent-based environment for scenario testing. Despite excluding survey data, this approach still facilitates tailored training by focusing on single-user scenarios, software environments, and a reduced set of attackers and defenders. Initial experiments involved a single F-16 targeting a control tower behind an SA-8 TELAR, adjusting various Measures of Performance (MOP). Subsequent tests varied the numbers of F-16s and SA-8 TELARs while modifying MOP values. Each scenario featured a human F-16 pilot, complemented by support through constructive simulation, and both F-16 and SA-8 TELAR survival were tracked.

The MOEs are amalgamated into a single value to train the deep learning model. A *pass* requires the survival of any F-16 and the destruction of the control tower. For an *escape*, any F-16 must survive alongside the intact survival of the control tower. Conversely, the *failed* condition necessitates the destruction of all F-16s. MOPs were assigned values instead of exhibiting a range, selected from options available within VC simulation, and meticulously reviewed by our team of experts. The following subsections discuss the development of the VC simulation environment, and then the extracted data is utilized to develop the ABM framework. A summary of the MOPs used in SIMbox for the case study and the ABM is presented in Table 1, detailing the corresponding values and descriptions.

Table 1: Parameters Used for the Case Study and ABM with Corresponding Values and Descriptions.

Parameter	Values	Description
X1	{1, 3, 5}	Quantity of released bombs by aircraft
X2	{10, 20, 30}	Chaffs carried by aircraft
X3	{2, 5, 7, 10}	Radar system control fire range (miles)
X4	{5, 10, 15, 20}	Radar system control track range (miles)
X5	{15, 30, 45, 60}	Radar system, time between missile fires (seconds)
X6	{1, 2, 5, 6}	Radar system, number of missiles
X7	{1, 2, 3}	Aircraft attack method
X8	{1, 2, 3}	Total aircraft count
X9	{1, 2, 3}	Total radar system count
X10	{Easy, Difficult}	Scenario complexity

2.1 Fighter Mission Logic

The *initial mission* setting provides five modes: none, patrol, intercept, computed continuous impact point (CCIP), and continually computed release point (CCRP). Following simulation experiments, the optimal mission setting, CCIP, was determined based on MOE criteria. This setting consistently yielded the highest success rates across multiple constructive simulations. CCIP calculates impact points by considering environmental factors, facilitating safe payload release during steep climbs, and minimizing exposure to enemy threats. Conversely, CCRP, reliant on pre-determined target coordinates, proved ineffective due to unsafe flyovers. Employing CCIP logic in the case study ensured consistent results in each simulation experiment. With this logic, computer-controlled F-16s safely destroyed enemy targets, achieving the desired MOE.

2.1.1 F-16 Release Bombs (X1)

The F-16 has 11 bays for mounting mission-specific equipment and weapon systems. SIMbox permits missiles placed in 9 bays, with air-to-air missiles limited to 4 bays. The remaining 5 bays can accommodate either air-to-air or air-to-surface missiles. Given the mission's emphasis on ground assault, all 5 bays are configured for air-to-surface missiles.

2.1.2 Chaffs and Flares (X2)

Chaffs and flares serve as vital countermeasures, diverting and confusing adversarial missiles. Chaffs disrupt radar tracking systems, while flares divert heat-seeking missiles. SIMbox permits adjustable levels for both countermeasures, up to a maximum value of 30. F-16s are equipped with chaff and flares, which cannot be interchanged. MOP values for chaff and flares are aligned and set at 10, 20, and 30. In this case study, TELAR systems employing radar-guided missiles render flares unnecessary.

2.1.3 SA-8 TELAR Control Fire Range (X3)

The TELAR, or Transporter Erector Launcher and Radar, is a multifunctional vehicle responsible for transporting, erecting, and launching missiles while featuring an integrated radar system for target tracking in surface-to-air engagements. In SIMbox, measured in miles, the control fire range attribute determines the range at which the TELAR can engage primary targets. According to (Puttre 2004), the SA-8 system's maximum range is 15 km, beyond which the radar's tracking capabilities diminish. For this case study, distances of 2, 5, 7, and a maximum of 10 miles were chosen.

2.1.4 SA-8 TELAR Control Track Range (X4)

In SIMbox, the control track range, measured in miles, dictates the distance at which the primary target begins to be tracked. With a TELAR acquisition range of 30 km (Puttre 2004), selections of 5, 10, 15, and 20 miles were made to illustrate the effects of varying distances in this study. The tracking range is consistently adjusted to precede the control fire range, ensuring effective engagement. For instance, a common scenario might entail X3 = 10 and X4 = 20, preventing situations where the firing range surpasses or equals the tracking range.

2.1.5 SA-8 TELAR Time Between Fires (X5)

In SIMbox, the time between fires indicates the firing rate, measured in seconds, determining the interval between missile launches. We selected 15, 30, 45, and 60 intervals for this study to analyze varying rates. Documentation suggests that each TELAR could engage only one target at a time (Puttre 2004), implying a disadvantage against multiple F-16s.

2.1.6 SA-8 TELAR Number of Missiles (X6)

This module exclusively supports SA-8 missiles and accommodates up to six launch compartments. According to (Puttre 2004), the system can simultaneously track "two missiles on different frequencies" when targeting. This study employs missile counts of 1, 2, 5, and a maximum of 6.

2.1.7 F-16 Attack Method (X7)

Different attack methods are explored from various directions: west, north, and east. In Attack Method 1 (X7 = 1), F-16 assaults originate from the west, with either one or multiple F-16s attacking simultaneously from the same direction. Attack Method 2 (X7 = 2) involves assaults from both the west and north, with a single F-16 attacking from the north, or if multiple F-16s are deployed, they are evenly distributed between both directions. In Attack Method 3 (X7 = 3), assaults occur from the west and east, with a single F-16 attacking from the east or multiple F-16s evenly distributed to both sides.

2.1.8 F-16 Total Count (X8)

The attacking forces will consist of 1 to 3 F-16s. Our team of experts concluded that a single F-16 can defend itself against a SAM assault. The case study by (Puttre 2004) involving seven F-16s was unnecessary. The objective is to illustrate how smaller and direct skirmishes influence MOE values.

2.1.9 SA-8 TELAR Total Count (X9)

We opted to test against 1 and 2 TELARs for the case study. Our expert panel deemed this attribute crucial in assessing mission success. The MOE focuses on detecting overall mission outcomes for the entire squadron of jets, implying that the F-16 is typically expected to prevail unless there's an equal number of SA-TELARs. Thus, our case study aims to address this by testing with more F-16s, SA-8 TELARS, and evenly matched forces.

2.1.10 Scenario Complexity (X10)

Scenario complexity was assessed using image processing of satellite imagery via a CNN. Our experts concluded that a one-on-one scenario, where an F-16 faces a TELAR, is generally deemed easy. However, mismatched scenarios, like two TELARs against a single F-16, are typically labeled difficult or complex.

2.2 Data Preparation

The MOPs and final MOE values were documented and prepared for input into an ABM, and map images were exported and analyzed as unstructured data. When interpreting an image, a computer perceives it as an array of numbers representing pixels. For instance, an image could be 100 x 100 x 3 in size, where the number 3 corresponds to RGB values, each ranging from 0 to 255. These values indicate pixel intensity at each point and serve as the sole inputs for the computer. The goal is to utilize computer capabilities to automatically categorize images, evaluating battlefield imagery for operator risk classification in low- and high-risk scenarios. The imagery is assessed against the MOE, denoting pilot survival for low-risk scenarios and mission failure for high-risk scenarios (Figure 1).

Figure 1: Example of a High-Risk scenario for F-16 pilot.

3 METHODOLOGY

3.1 Convolutional Neural Network (CNN) Model

A CNN is a deep neural network optimized for image recognition, excelling in classification tasks (Elhassouny and Smarandache 2019). Its architecture comprises a convolutional, pooling, flattening, and fully connected layer that autonomously learns to recognize and classify images. It is ideal for identifying high-risk and low-risk scenarios in combat simulations. The scenarios derived from VC simulations are processed through a CNN architecture, incorporating multiple customized parameters for image classification. An expert classified battlefield images, creating a seven-layer CNN architecture refined through iterative testing and optimized for battlefield imagery. The entire workflow was implemented in KNIME, and includes data acquisition, pre-processing, and post-processing stages and the CNN architecture.

Several CNN architectures were built. For example, one of the developed CNN architectures comprises of seven layers (Figure 2). The input layer consists of 10,000 units. The subsequent layer comprises a convolutional layer featuring 32 kernels, a kernel size of 3x3, and a stride of 2x2 in ReLU units. This is followed by a subsampling layer with max pooling, utilizing a kernel size of 2x2 and a stride of 2x2. The fourth layer is an additional convolutional layer with 64 kernels, a kernel size of 3x3, and a stride of 2x2. The fifth layer is a subsampling layer with max pooling, employing a kernel size of 2x2 and a stride of 2x2. Convolutional layers with small kernel sizes capture fine image details while pooling layers reduce dimensionality. The next two layers are dense, with the first layer using ReLU activation and the second layer using sigmoid activation. ReLU activation was employed to prevent vanishing gradients, enhancing learning across deep layers, with sigmoid activation used for outputting probabilities. Finally, the output layer consists of two low- and high-risk neurons. The optimization function utilizes Stochastic Gradient Descent (SGD), with RMS Prop as the updater and Negative Log-Likelihood as the loss function. SGD, paired with RMSprop, handled noisy gradients and dynamically adjusted learning rates, ensuring stable training. The model's configuration was validated against diverse image sets, showing superior performance and robustness. Comparisons with other network setups confirmed the effectiveness of the selected architecture and optimizer, leading to an overall testing accuracy of 89.4%.

The machine learning workflow utilized the RProp Multilayer Perceptron to solve the prediction problem. Using the previously documented MOP values, the platform predicts whether a scenario will result in a pass, failure, or escape outcome. This prediction aids in determining whether approaching the control tower is viable, or fleeing is the best course of action. As only a training workflow was necessary for this aspect of the case study, a configuration with 100 maximum learning iterations, 2 hidden layers, and 5 hidden neurons per layer was chosen. The remaining test set is input into the Multilayer Perceptron Predictor. The workflow achieved an average testing accuracy of 98.5%.

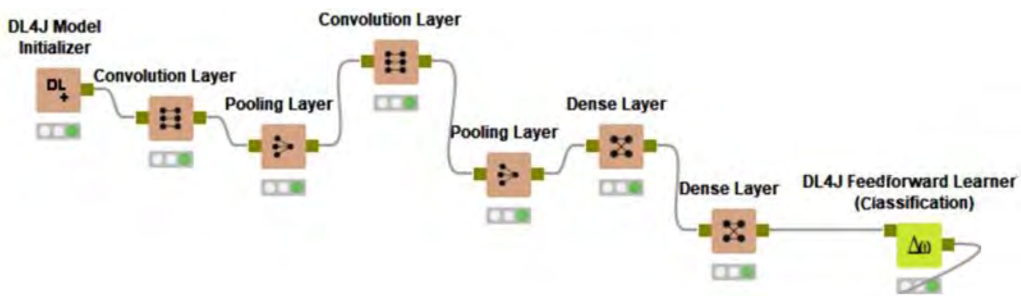

Figure 2: The CNN architecture in the KNIME environment.

3.2 Agent-Based Simulation

ABMs consist of autonomous agents, each characterized by attributes derived from data or distributions and decision-making processes that are either rule-based or based on heuristics. These agents interact with one another and their environment, leading to emergent phenomena that aim to enhance our understanding of individual and collective behaviors. In this study, ABM simulates aircraft agent behaviors with embedded CNNs, providing them with image recognition capabilities. This integration allows for the simulation of various operational conditions and supports developing advanced pilot training programs based on dynamic scenario outcomes. An ABM is developed to simulate an air defense system featuring a radar apparatus designed to safeguard assets against enemy aircraft bombardment. Upon detecting hostile aircraft within its operational range, the radar system initiates missile launches to intercept the incoming threats. The CNN is integrated to enhance model fidelity to represent enemy aircraft behavior. If unable to evade the defensive radar system, these aircraft avoid deploying bombs on the protected assets. Aircraft agents within the model possess adaptive capabilities, continuously refining their efficacy in fulfilling their objectives throughout the simulation. The CNN provides radar images to the aircraft agent, enhancing its decision-making should the parameters within the modeling environment change. The aircraft must adapt and relearn to achieve its objectives.

The model comprises five agent types: aircraft, bombs, radar systems, missiles, and assets (building), each playing a distinct role. It also incorporates two sets of adjustable parameters impacting simulation outcomes. These parameters align with the MOP inputs used in the VC simulation, as previously outlined in section 2, and include values for F-16 and SA-8 TELAR specifications based on Puttre (2004). The case study establishes the number of aircraft, radar system count, and scenario complexity, incorporating input from Subject Matter Experts (SMEs). The first set of adjustable parameters includes the number of enemy aircraft per day, the speed of the aircraft, and their altitude. Regarding the radar system, changeable parameters include the coverage radius, missile firing capacity upon aircraft detection, and missile speed. These parameters are adjustable while the simulation runs and can influence simulation outcomes. The other set of tunable parameters occurs in the initialization phase of the ABM. It facilitates the creation of diverse scenarios, allowing a greater understanding of each parameter's impact on the aircraft and radar system agents' success rates. These parameters are summarized in Table 1, which details each parameter, its values, and a short description. Tuning these parameters allows for a comprehensive exploration of dynamics and identifying critical factors influencing agent performance.

The first tunable parameter, X1, determines the number of bombs released by the aircraft agent, and in the developed model, this affects the potency of the bomb's impact on targeted assets. Higher values result in accelerated destruction of assets upon bomb deployment. The second parameter, X2, is the number of chaffs an aircraft agent has, which function as countermeasures against missiles launched by the defensive radar system. Chaffs, composed of aluminum-coated glass fiber, serve to evade detection by enemy radar systems. With a minimum value of 10 and a maximum of 30, X2 defines the quantity of chaffs available for aircraft defense. The third adjustable parameter, X3, denotes the firing range of the radar system. This parameter dictates the range within which missiles are launched by the radar system agent. Following this, the subsequent parameter, X4, pertains to the control track range of the radar system, measured in miles, and signifies the distance from which the radar system begins tracking a targeted incoming aircraft. It is essential to ensure that the control track range, represented by X4, always exceeds the firing range parameter, X3, for effective tracking and engagement of incoming threats.

The fifth parameter to be initialized is X5, representing the interval between missile firings by the radar system, measured in seconds. This parameter signifies the duration the radar system must wait before launching another missile. Another adjustable parameter, X6, denotes the quantity of missiles supported and employed by the radar system during aircraft tracking and engagement. The subsequent parameter, X7, governs the attack strategy employed by aircraft within the simulation, offering three distinct methods for selection. The first approach (X7 = 1) involves aircraft initiating attacks solely from the west of the designated assets. In the second method (X7 = 2), aircraft are directed to commence attacks from the west and north of the targeted asset, with a single aircraft attacking from the north if multiple aircraft are present. The third strategy (X7 = 3) entails attacks from the west and east of the targeted assets. The subsequent parameter, X8, determines the quantity of attacking aircraft agents within the simulation. Next, parameter X9 denotes the quantity of radar defense systems in the model. The final adjustable parameter, X10, dictates the selected scenario complexity within the simulation, offering two distinct options: *easy* and *difficult*. These classifications are contingent upon the number of aircraft and radar system agents present. For instance, an easy scenario may involve one or more aircraft agents attacking assets guarded by a single radar system. In contrast, a difficult scenario may entail a single aircraft agent confronting two or three radar systems.

The developed model facilitates a deeper exploration of how variables influence overall performance. For instance, it allows for examining how adjustments in aircraft speed or altitude impact the ability to evade radar-based air defense systems. It enables the assessment of whether specific modifications enhance aircraft success rates, and if so, by what margin. Similarly, the model allows for the study of radar system configurations aimed at creating an impenetrable defense. Decision-makers stand to benefit from this model, as it enables simulations based on diverse parameters, empowering them to optimize decision-making in real-world scenarios. Moreover, such simulations can enhance pilot training by providing realistic scenarios for pilots to navigate and adapt to various challenging conditions.

3.2.1 Bomb Agent

In this model, the bomb agent acts as the intermediary between the aircraft and asset agents, facilitating the aircraft's objective of targeting assets. Upon reaching a targeted asset, the aircraft releases a bomb, initiating the bomb agent's two-phase process: falling and exploding, as depicted in the state chart (Figure 3(a)). During the transition between these states, the bomb agent signals the asset agent to commence destruction, synchronizing with the bomb's explosion. Visualized by an image, the bomb agent's explosion signifies a direct hit on the asset. Parameter X1, which governs bomb releases, can be adjusted during model execution. Increasing X1 enhances bomb strength, expediting asset destruction.

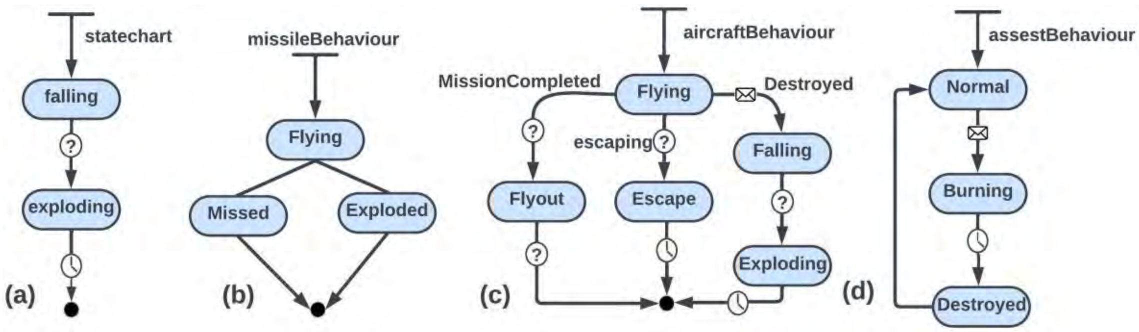

Figure 3 (a): F-16 Bomb Agent State Chart, Figure 3 (b): Missile Agent State Chart, Figure 3 (c): Aircraft Agent State Chart, Figure 3 (d): Asset Agent State Chart.

Figure 4: The ABM platform (AnyLogic) can be integrated with VR-Forces using the HLA.

3.2.2 Missile Agent

In the simulation model, the Missile agent embodies real-life missile behaviors, delineated by a state chart (Figure 3(b)). The chart commences with a State Entry Point named *Missile Behavior*, initializing the model and facilitating state definition for the agent. The primary state, *Flying*, signifies the missile's launch toward the aircraft, with outcomes determined by transitions labeled *Missed* or *Exploded*. The *Missed* transition is activated when the radar-to-plane distance exceeds the radar's effective zone, accounting for scenarios where a single F-16 can repel radar-based assaults as informed by exports (Lowe 2020). The *Exploded* transition triggers upon the missile's proximity within 3 meters of the aircraft, resulting in detonation. The state chart concludes with a final state, removing the missile from the model post-explosion or miss, thus rendering it ineligible to be launched again.

3.2.3 Aircraft Agent

Another pivotal agent within the model is the Aircraft agent, governed by a complex state chart depicted in Figure 3(c). Initialization begins with the State Entry Point *Aircraft Behavior*, initiating the subsequent state, *Flying*. This state presents three potential transitions: *Mission completed*, *Escaping*, and *Destroyed*. The *Mission completed* transition activates when the aircraft is near an asset (building) and deploys a bomb, thus triggering the transition to the *FlyOut* state and subsequent removal from the simulation. The transition for *escaping* is also triggered by a condition based on the function containing the neural network. This function informs the aircraft to execute an escape maneuver or to proceed with the mission based on computed values. The last transition is operated via a message. If the plane is hit by a missile, the missiles are programmed to deliver a message instructing the plane that it has been hit. This is captured by a function called *onHit*. Once this message is received, the plane enters the *Falling* state, then transitions to an exploding state that will time out in about 1 second. The escape and the exploding states lead to the final state, where, as mentioned above, the plane is removed from the simulation.

3.2.4 Asset Agents

The asset agent represents valuable properties in the simulation module, such as buildings, factories, homes, hospitals, and military bases. In this exercise, radars defend these assets and their inhabitants. The asset agent initiates in a *Normal* state. Upon bomb impact, as described in the bomb agent section (the asset agent receives a “Hit” message), the asset workflow transitions to the *Burning* state, signifying the onset of fire damage. Once fully consumed by fire, the asset transitions to a *Destroyed* state, rendering it absent from the simulation (Figure 3(d)).

3.2.5 Radar Agent

The radar is another agent in the model that is initialized as soon as the mission begins and detects any incoming aircraft or missiles. Unlike the other agents, the radar does not follow a state chart. Instead, it interacts with missiles (another agent) by gathering data for the following variables: *missileIndex*, *target*, and the function *fireMissile*. The target is the aircraft, the *missileIndex* is an integer that increases by 1 every time a missile is encountered, and *fireMissile* is a function that launches a missile. These points are composed of x,y, and z coordinates, missile speed, the zone, the maximum number of missiles able to be fired, and time. The components mentioned allow the scanner event to occur and, therefore, the radar to function. Once complete, the radar data depicting the missile's location is transferred to the main process.

4 RESULTS AND DISCUSSION

The CNN effectively classified numerous satellite images into low-risk and high-risk categories within the MOE, achieving an accuracy rate of 89.4%. In low-risk scenarios, achieving a *Pass* or *Escape* MOE is expected, while *failure* is anticipated in high-risk environments. This CNN output served as an MOP for the RProp neural network, which accurately predicted participant MOEs with an average accuracy of 98.5%. The trained network was then exported and integrated into an agent-based simulation with the potential to use other visualization schemes using HLA (Figure 4). Within the ABM, each new aircraft agent evaluates the likelihood of failure by polling the trained neural network-weighted MOPs. This evaluation determines whether the aircraft should proceed toward a target or retreat to a safe destination. For example, in one scenario, when outnumbered (One Aircraft vs. 3 Radar Systems), the agent initiated an escape to the north. Conversely, when not outnumbered, the squadron proceeded with the mission. This approach effectively transfers trainee pilot behavior into a new environment for further assessment across various scenarios and software environments.

The ABM simulated two scenarios: one where the aircraft agent had access to the CNN and one without access. Without the CNN, aircraft agents lack image recognition capabilities, leading to impaired threat assessment and higher mission failure. Conversely, with the CNN, aircraft agents benefit from real-time environmental analysis, enhancing decision-making and thus leading to improved mission outcomes. The scenario incorporating CNN has an overall success rate of 92% (both escaping and passing), with only 8% of its predictions failing. Compared to the scenario that did not include the CNN, the bomber succeeded in only 5% of its missions (Figure 5).

Various experiments assessed how changing parameters impact the aircraft agent's success. These experiments, which included simulations with and without the embedded CNN, ranged from altering one parameter at a time to altering a combination of parameters. The parameters are in Table 1, which lists the range of values for each parameter and a description. The experimental design incorporated a 'one factor at a time' (OFAT) approach and a fractional factorial design approach to efficiently explore the impact of various parameters on the aircraft agent's performance. Initially, the OFAT approach was employed, where one parameter is changed at a time while the others are held constant, enabling a greater understanding of the effects of each parameter individually on aircraft performance. This was followed by implementing a fractional factorial design, simultaneously varying combinations of two and three parameters and evaluating their effects. Multiple simulations were conducted for each parameter combination to ensure the robustness and reliability of the results. The results indicated that the most beneficial parameters for an aircraft agent's success were parameters X1 and X8. These are related to the strength of the bombs released by aircraft and the number of aircraft. There was a diverse set of parameter combinations throughout the simulation, and each combination was simulated several times to ensure robustness and reliability in our conclusions regarding the effects of these parameter changes.

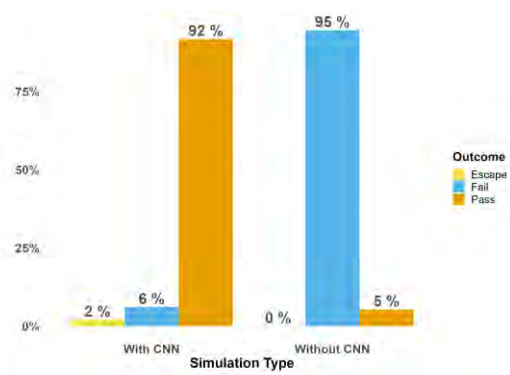

Figure 5: Aircraft Efficiency with and without the CNN.

The aircraft agent failed most of its missions without CNN, but it demonstrated improved success with CNN, as shown by various parameter combinations. The results indicate when altering the initial value of X5 and X9 to 60 and 3, respectively, the aircraft agent succeeded in its mission only 60% of the time, and this is due to the aircraft agent being outnumbered, as well as a longer time between missile launches by the radar agent, signaling that there is a chance of success for the aircraft agent, this is why the escape option was not taken. It's important to note that X10 (scenario complexity) was listed as easy, another reason for the aircraft agent's actions. A similar result was found when altering the initial value of X6 (number of missiles) and X9 (number of radar agents) to 6 and 3, respectively, while X10 was also set to *easy*. The aircraft agent succeeded only 40% of the time with this parameter combination due to being outnumbered 3 to 1 and due to the radar agent launching 6 missiles each time it detected an aircraft agent in its vicinity. The aircraft agent also had difficulties in its mission (60% success) when initial parameters of X3 (the firing range of the radar system), X4 (the control track range of the radar system), and X10 (scenario complexity) were set to 10, 20 and *difficult* respectively. All other parameter combinations led to a *pass* by the aircraft agent, except for the combination where X6 (number of missiles), X9 (number of radar agents), and X10 (scenario complexity) were changed to 6, 3, and *difficult*. For this combination of parameters, the aircraft performed an *escape* 100% of the time. Table 2 summarizes selected parameter combinations explored in this study, detailing the aircraft agent's success rate. Notably, these experiments utilized embedded CNN, and other investigated combinations typically resulted in a 100% success rate. The results highlight the effectiveness of integrating image recognition capabilities into the simulation.

5 CONCLUSION

This paper explores an approach that combines VC environments with CNN and integrates these into an ABM framework to study pilot performance. The VC environment simulates a fighter pilot tasked with neutralizing assets defended by radar systems. The embedded CNN processes visual data from the simulation, categorizing scenarios as Low Risk or High Risk, facilitating informed decision-making within the ABM. Through numerous simulations within the ABM framework, experiments were conducted with and without the integrated CNN. These experiments involved adjusting parameters individually and in combinations, revealing the parameters with the most significant impact on aircraft mission success. By incorporating CNNs into the ABM, we significantly enhance the realism and efficacy of training simulations, endowing agents with advanced decision-making capabilities mirroring real-world conditions more accurately. This underscores the transformative potential of machine learning technologies in revolutionizing traditional training methods, offering new avenues for measuring WCE, mission design, and replicating operator behavior based on training data. This integration could overcome biases inherent in existing distributed simulation methods, paving the way for more precise requirements in designing future systems tailored to individual training needs. Furthermore, it introduces an approach for generating performance metrics based on scenario variation and data analysis, leveraging neural networks, deep learning, and agent-based modeling. The integration of CNNs with ABMs demonstrated in this study can potentially enhance other simulation frameworks requiring image recognition capabilities. This integration facilitates improved decision-making for autonomous agents and enables real-time analysis across diverse scenarios.

Table 2: Selected Parameter Combinations and Their Impact on Aircraft Mission Success when Utilizing the CNN.

Parameter Combination	Description	Scenario Complexity	Success Rate	Typical Outcome
X5=60 X9=3	Time between fires = 60 seconds & Number of radar agents = 3	Easy	60%	Pass
X6=6 X9=3	Number of missiles (Radar) = 6 & Number of radar agents = 3	Easy	40%	Fail
X3=10 X4=20	Firing range of the radar system = 10 miles & Control track range of the radar system = 20 miles.	Difficult	60%	Pass
X6=6 X9=3	Number of missiles (Radar) = 6 & Number of radar agents = 3	Difficult	100%	Escape

REFERENCES

Adams, E. and Rollings, A. 2006. Fundamentals of Game Design, 1st ed. Printice Hall.

Armo, K. R. 2000. "The relationship between a submarine's maximum speed and its evasive capability" (Doctoral dissertation, Monterey, California. Naval Postgraduate School).

Connors, C. D. 2015. "Agent-based Modeling Methodology for Analyzing Weapons Systems". (No. AFIT-ENS-MS-15-M-132). Air Force Institute of Technology.

Elhassouny, A., and Smarandache, F. 2019. "Trends in deep convolutional neural network architectures: a review". In 2019 International Conference of Computer Science and Renewable Energies, ICCSRE. IEEE, pp. 1-8.

Gu, J., Liu, Q., & Li, H. 2007. "Evaluation of Ship-borne Weapons Efficiency by ANN Model". Command Control & Simulation.

Hassenzahl, M. 2000. "Prioritizing usability problems: Data-driven and judgement-driven severity estimates". Behaviour & Information Technology, 19(1), 29-42.

Huang, J., Zhang, B., & Lu, Y. 2005. "An Expert System for Weapon Effectiveness Evaluation Based on Improved BP Neural Network". Electronics Optics & Control. 2005-04.

Jung, C. Y., & Lee, J. Y. 2010. “Combat Effectiveness Based Analysis Methodology for Optimal Requirement of Attack Helicopter Using Simulation”. Journal of the Korea Institute of Military Science and Technology, 13(6), 1099-1105.
Kirby, B., Fletcher, G., & Dudfield, H. 2011. “Live Virtual Constructive Training Blend Optimisation Study”. ATO Modelling and Simulation Group Symposium 87. Berne, Switzerland.
KNIME. 2023. Retrieved from: https://www.knime.com/
Lee, G., Kim, J., Marin, M., Lee, K., et al. 2019. “Building Multiple Resolution Modeling Systems Using the High-Level Architecture”. SAE International Journal of Advances and Current Practices in Mobility, 2(2019-01-1917), 838-842.
Lee, J., Shin, S., Kim, J., Bae, S., et al. 2015. “Interrelation Analysis of UGV Operational Capability and Combat Effectiveness using AnyLogic Simulation”. Journal of Applied Reliability, 15(2), 131-138.
Lowe, L. 2020. "Integration of Artificial Intelligence and Virtual and Constructive Simulations to Increase Performance: Two Case Studies with Big Data and Agent-based Simulation". UCF Electronic Theses and Dissertations.
Mulgund, S., Harper, K., Krishnakumar, K., & Zacharias, G. 1998. “Air combat tactics optimization using stochastic genetic algorithms”. In Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference. Vol. 4, pp. 3136-3141.
Müller, M. 2002. “Computer Go”. Artificial Intelligence. 134(1-2):145-179.
Oztekin, A., Delen, D., Turkyilmaz, A., & Zaim, S. 2013. “A machine learning-based usability evaluation method for eLearning systems”. Decision Support Systems, 56, 63-73.
Proctor, M. D., Bauer, M., & Lucario, T. 2007. “Helicopter flight training through serious aviation gaming”. The Journal of Defense Modeling and Simulation, 4(3), 277-294.
Puttre, M. 2004. International electronic countermeasures handbook. Horizon House.
Smith, R. E., Dike, B. A., Mehra, R. K., Ravichandran, B., et al. 2000. “Classifier systems in combat: two-sided learning of maneuvers for advanced fighter aircraft”. Computer Methods in Applied Mechanics and Engineering, 186(2), 421-437.
Summers, J. E. 2012. “Simulation-based military training: an engineering approach to better addressing competing environmental, fiscal, and security concerns”. Journal of the Washington Academy of Sciences, 9-29.
Tannenbaum, S. I., Cannon-Bowers, J. A., Salas, E., & Mathieu, J. E. 1993. “Factors that Influence Training Effectiveness. A Conceptual Model and Longitudinal Analysis” (No. NAVTRASYSCEN-TR-93-011). Naval Training Systems Center, Orlando FL.
West, E. A., Paul, C. A., Webb, D., & Potter, W. H. 2013. “Variation of instructor-student interactions in an introductory interactive physics course”. Physical Review Special Topics-Physics Education Research, 9(1), 010109.

AUTHOR BIOGRAPHIES

LARRY LOWE has a Ph.D. in Industrial Engineering from the University of Central Florida. He has worked extensively in Live Virtual Constructive Simulations. In addition, he has been working in agent-based simulation. His email address is larry.lowe@knights.ucf.edu.

LUIS RABELO is a Department of Industrial Engineering and Management Systems professor at the University of Central Florida and Co-Director of the Simulation Interoperability Lab. He is an expert in management systems, artificial intelligence (AI), and simulation. His email address is luis.rabelo@ucf.edu, and his website is https://iems.ucf.edu/faculty/luis-rabelo/.

MARWEN ELKAMEL has a Ph.D in Industrial Engineering from the University of Central Florida. He is an expert in Agent-based modeling, Machine Learning, and Operations Research. His email address is marwen.elkamel@ucf.edu.

MITCHELL HUNSUCKER has a BS in Industrial Engineering from the University of Central Florida. He is an expert in agent-based modeling. His email is mchunsucker1@knights.ucf.edu.

KATALINA ARIAS MARIN has a BS in Industrial Engineering from the University of Central Florida. Her area of interest is Engineering Management. Her email is k.arias@knights.ucf.edu.

NATHALIA DAVILA has a BS in Industrial Engineering from the University of Central Florida. Her areas of interest are Engineering Management and Agent-Based Modeling. Her email is davilaanathalia@knights.ucf.edu.

OMAR ALLAZ has a BS in Industrial Engineering from the University of Central Florida. He is interested in agent-based modeling and engineering management. His email is oallaz@knights.ucf.edu.

MARIO MARIN has a Ph.D. in Industrial Engineering from the University of Central Florida. He is interested in training, simulation systems, and agent-based modeling. His email is mmarin@ucf.edu.

GENE LEE is an Associate Professor in the Department of Industrial Engineering and Management Systems at the University of Central Florida. He is the Director of the Simulation Interoperability Laboratory. His email is glee@ucf.edu.