Implementation of Dynamic Artificial Intelligence in Game Development

Rohit Nair.S, Manav Varma, Yamini.R

Abstract: ***The aim of this paper is to create an adaptive Artificial Intelligence implemented within our own video game which learns from players and uniquely adapts its playstyle in order to counter the play style of the current player in real time. This is done by creating a basic AI using the AI packages that come with Unreal Engine, which is what is being used to implement this technique. This AI is then trained with many different possible moves in the game represented within a tree. The end goal is to have the AI learn the moves that the user is using most frequently or most effectively and begin countering it more and more effectively as the level rises. This is done by training the AI to various movesets and giving it a sizeable sample space in order to understand and predict as required***

I. INTRODUCTION

Video games nowadays are not interactive enough. The aim is to try and solve that. The gaming industry is growing extremely fast today and the quality of games has risen with that-from graphically stunning environments to sentimental stories. However, with large companies looking to make huge profits from these games, it's becoming more about who can sell these games faster and less about developing the game for more immersion. Progressive AI that can react, adapt and make decisions based on player actions, environmental cues and even other AI characters themselves.

Our literature consists of published papers and reference material consisting of algorithms and concepts related to the project whose features may be incorporated within the course of this project for optimization and general implementation.

It implements a behaviour transformation[1] system, to dynamically generate more reactive and believable NPCs. The paper focus on a behaviour allocation agent, that assigns a character its properties. This is a good strategy, but will fall short in covering the entire search space due to the limitless interaction possibilities.

We learn about present day computer games [2] as often as possible highlighting refined and reasonable conditions, the requirement for keen and exhaustive agents that comprehend the different parts of complex situations is paramount.

Since video game AI is often built specifically for each game, computer game AI tools currently focus on allowing developers of video games to create efficient AI quickly . One concern with this strategy is that it does not productively exploit the numerous similarities that exist between video games not only of the same genre, but also of different genres, making it difficult for each video game to deal independently with the many aspects of a complex environment. Roused by the human capacity to distinguish analogies among amusements and apply comparable conduct on a theoretical dimension, this paper proposes a methodology dependent on the utilization of a brought together calculated structure to empower the improvement of applied AI which depends on theoretical perspectives and activities to characterize essential yet sensible and sensible conduct.

This paper studies how modern games are [3] implementing Artificial Intelligence, and discusses various strategies, possibilities and shortcomings.In this paper, the game takes game objects and the world into consideration when modelling it's actions.It also maintains the importance of user experience while modelling the AI, but does not provide any implementation.

It illustrates the surge in importance of good [4] AI in video games and discusses a novel approach to adaptive AI using automatic domain knowledge gathering and utilisation by the AI without trials and resource intensive learning. It further goes into detail about difficulty scaling and details of evaluation and implementation of this in a test game environment in multiple phases to ascertain accuracy.

It is based around evolving reactive NPCs [5] for real time simulation games. It uses a slightly different co-evolutionary approach however, to ascertain the NPC neural based reactions where the AI evolves with the player over time. Co-trans-formative methodologies have been researched in later, which successfully use the appearance of competition in games. In this paper, the co evolutionary is adopted using the genetic algorithm to produce behavior systems that are obliged to a few conditions. Since (N2-N)/2 rivalries are essential for a solitary species population populace of N people when using a simple competition pattern, two pairwise rivalry designs are received to adequately calculate the fitness of an individual.

It is a systematic review of current concepts [6] of player-game interactions. It states that 'game interactions' are not clearly defined and that several concepts qualify under this banner. It further derives that factors like "engagement" and "enjoyability" of the player, as well as "input/output" and "multiplayer" aspects of the game are relevant to global approaches to playability and game creation.

It illustrates the importance of heuristic [7] evaluation and

Revised Manuscript Received on September 22, 2019.

Rohit Nair, B Tech. CSE, SSRMIST, Chennai rohitnair_sa@srmuniv.edu.in
Manav Varma, B Tech. CSE, SRMIST, Chennai. me.manavvarma@gmail.com
Dr.Yamini.R, Assistant Professor , Dept. Of CSE, SRMIST, Kattankulathur, yamini.r@ktr.srmuniv.ac.in

its application to today's games and entertainment spectrum in general. It expands upon the field of Human-Computer Interface(HCI) and gives a breakdown of game anthropology.

It illustrates the growth of Artificial [8] Intelligence over the course of the century and the weight it puts on operating non-player characters. It also describes the role of conflict resolution in the virtual world between AI and players which is directly correlated to player interactivity. This also briefs us about the difference in AI application across different genres of games like Action, Simulation, Strategy, Role-Playing etc.

II. EXISTING SYSTEM

The system currently in place uses a basic Rule-Based AI system which makes its decisions based on predefined looped clauses and not based on player actions or any other dynamic factors. This leads to a lack of realism as well as a repetitive gaming experience over time.

The System would consist of the world designed using unreal or any other engine, and two characters, one player controlled and an Non Playable Character.

The character animations and movement are advanced interactions such as damage and reactions can be created using blueprints, with unreal that let's program our character dynamically. A blueprint exists for the playable character (shown in Figure 3) where the control of the character is given to the user, and the second blueprint which is the NPC blueprint consisting of identical moves to the user and a basic system is implemented using unreal's behavior trees to create a basic AI controlled NPC.

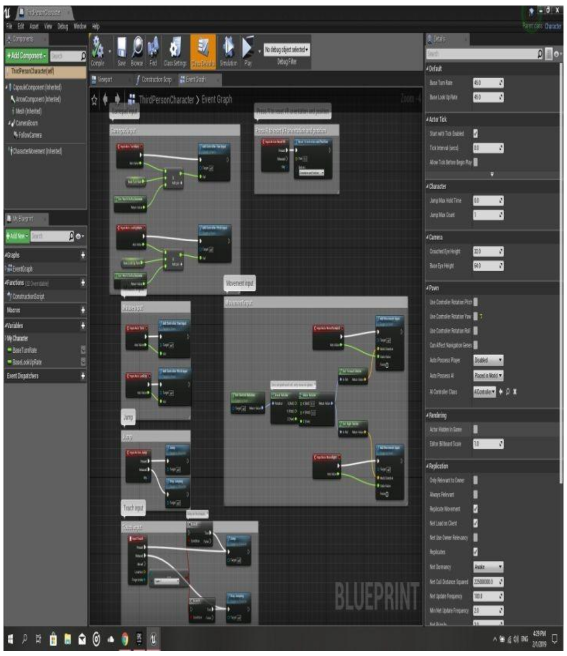
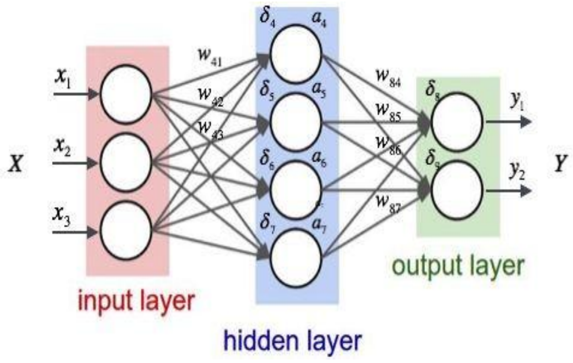


Figure 3

NEURAL NETWORKS

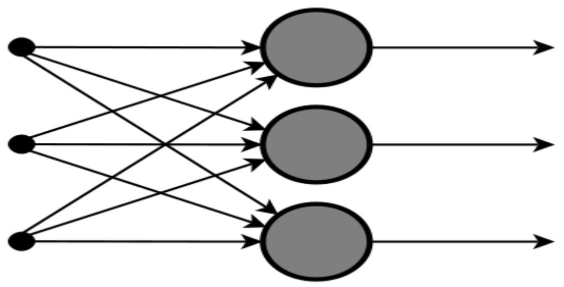
Biological neurological systems are the motivation behind those created artificially. Artificial Neural Networks (ANN) computing systems themselves are not an algorithm, but rather a structure to work together and process complex data inputs for some, different machine learning algorithms (shown in Figure 4.1). Such frameworks "learn" to carry out undertakings by considering precedents, usually without any programming.



MODELS

FEEDFORWARD NEURAL NETWORK

This neural system is one of ANN's least challenging type, where information or information goes one way. The information goes through the hubs of the information and exits at the hubs of the output (shown in Figure 4.2). Quite possibly, this neural system has the hidden layers. In plain words, it usually uses a classification activation function to have a front propagated wave and no back propagation.



RADIAL BASIS FUNCTION NEURAL NETWORK

Radial baseline functions consider the distance of a point with respect to the center. RBF capacities have two layers, first when the highlights are attached to the Radial Baseline function in the inward layer and then the yield of these highlights is mulled over while recording a similar yield in each step that is essentially a memory.

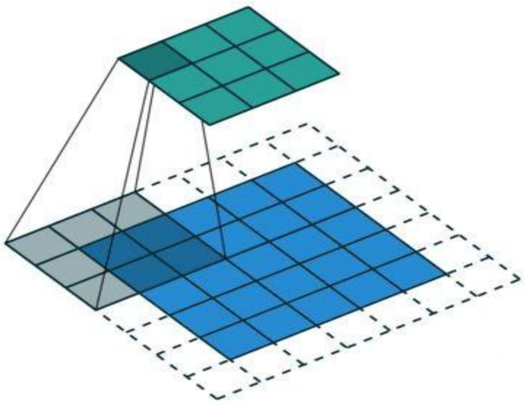
RECURRENT NEURAL NETWORK-LSTM

The Recurrent Neural Network works on saving a layer's output and feeding it back to the input to help predict the layer's outcome. Here, with the result of total loads and highlights, the primary layer is framed as the feed forward neural system. The intermittent process of the neural system begins once this is figured, implying that each neuron will recollect some data it had in the past time - step from one time - step to the next. In carrying out calculations, this makes every neuron demonstration like a memory cell. In this procedure, we must allow the neural system to handle the front spread and to recollect what data it requires for subsequent use. Here, if the expectation is not correct, we use the learning rate or error redress to roll out little improvements with the goal of progressively moving in the direction of making the correct forecast in the midst of the back propagation.

III. CONVOLUTIONAL NEURAL NETWORK

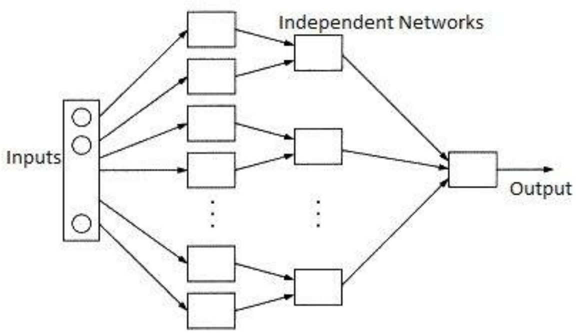
Convolutional neural systems are like feed - forward neural systems, where neurons have loads and predispositions capable of learning. Its application has been in the processing of signals and images that take over OpenCV in computer vision.

ConvNet are connected in strategies like signal handling and image classification techniques. Computer vision strategies are ruled by convolutional neural systems in view of their accuracy in image classification. The picture examination and acknowledgement method, where farming and weather characteristics are extracted from open source satellites such as LSAT to anticipate the future development and yield of a specific land, is being updated.



MODULAR NEURAL NETWORKS

Modular Neural Networks have an independent collection of different networks that contribute to the output (shown in Figure 4.4). Each neural network has a set of unique inputs compared to other networks that create and perform sub - tasks. In carrying out the tasks, these networks do not interact or signal each other. The advantage of a modular neural network is that it breaks down the complexity of a large computational process into smaller components. This breakdown will help to decrease the number of connections and negate the interaction between these networks, which in turn will increase the speed of the computation. The processing time, however, will depend on the number of neurons involved in the calculation of the results.



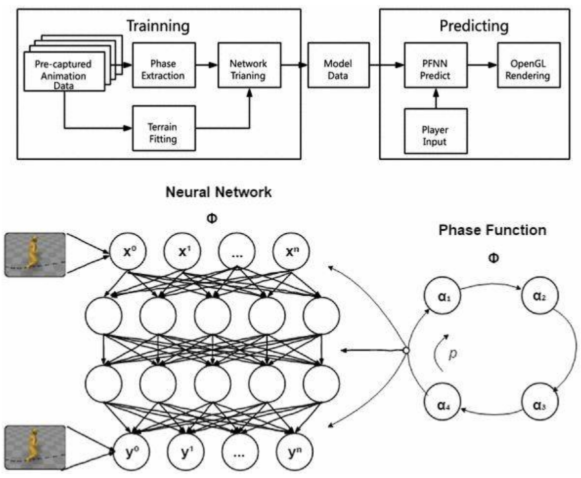
IV. PROPOSED SYSTEM

The problem with the current implementation of Artificial Intelligence is that it just provides predefined set of moves to the non-playable characters or NPCs.

The system proposed here, aims to use neural networks (shown in Figure 5.1) to train a non-playable character to learn and adapt to the different moves of the player. This would make the NPCs more realistic and would make the game feel more immersive.

The System would consist of a genetic neural network attached to the blueprint of the NPC character. The neural network would then need to be trained before it can be tested with either data in a dataset or real time scenario simulations.

- **ARCHITECTURE DIAGRAM**



- **SYSTEM FRAMEWORK**

THE WORLD

The world (shown in Figure 5.2) is the main component of the game without which making the game would be pointless. The world consists of the environment in which the characters are placed and where the game simulation takes place.

The building of the world would consist of designing a suitable environment for the game and adding appropriate textures and various terrains depending on the requirements of the game. Another major component of the world is the lighting. The lighting is responsible for simulating an artificial light source in the game and thus affecting the environment textures and shadows. Modifications can be done to the world to make the world, character interactable and better textures can be used to increase the visual aesthetics of the world.

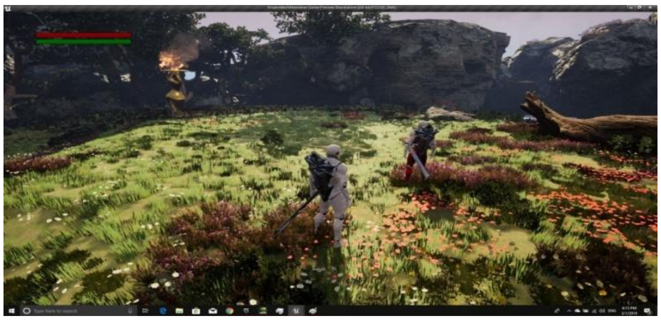


Figure 5.2

THE PLAYABLE CHARACTER

The playable character refers to the character that is controlled by the player using the player controller that is attached to the character blueprint.

The player controller is responsible for mapping the various inputs to the functions of the character and executing them when the user presses a valid button.

The character also consists of the character blueprint which basically defines the actions and functions that the character can perform. It is also responsible for monitoring the various statistics of the character such as health, armor etc. The blueprint is the most important part of the character without which the character would be a dummy.

Another important part of the character is the animation class that has been applied to the character model mesh. The animation class defines the various animations for the character such as running, walking, turning etc to be used when the character is in motion while being controlled by the character.

THE NEURAL NETWORK CHARACTER

The Neural network character will be similar to the playable character in terms of appearance but with the differance the the character is controlled by a neural network controller rather than a player controller.

The NN actor will consist of animation class for the model mesh that is applied to its actor to allow for its various motions and animations to be displayed during the game.The NN character will also consist of a blueprint to define its det of functions and actions and also to monitor is statistics during game play. Since the NN character is not controlled by the player, we do not need to account for inputs from the user while designing the character blueprint.

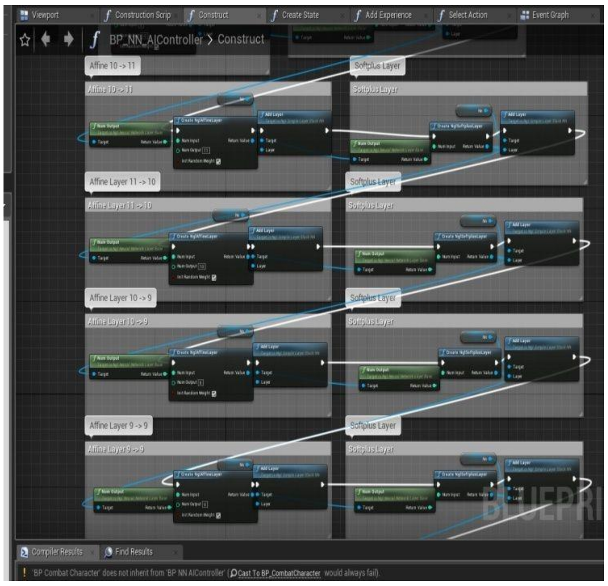
The most important part of the NN character is the Neural Network Controller which will consist of the Neural network and the other controller functions such as linking the controller with the character blueprint and storing the data for the neural network in the form such that it creates a learning experience.

- **THE NEURAL NETWORK**

The neural network is built inside the NN controller (shown in Figure 5.3) and is a subpart of the controller. While implementing the neural network, it is important to choose a network that will best perform with the situation presented.

For our system we have chosen to implement a genetic neural network. A genetic neural network basically passes on the information that a current generation has learned onto a future generation in order to improve future performance.

While designing the neural network it is first important to decide what the inputs and outputs are to be for the neural network.

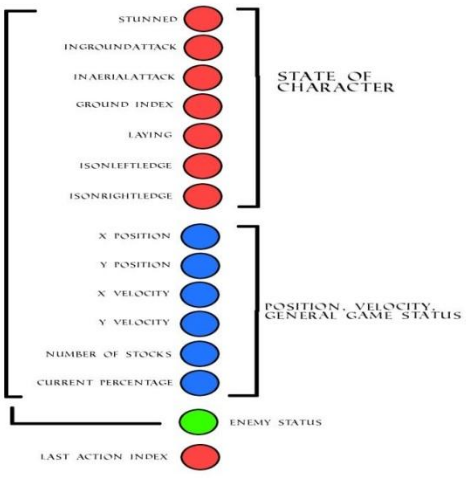


'BP Combat Character' does not inherit from 'BP NN AIController' (Cast To BP_CombatCharacter would always fail).

Figure 5.3

INPUTS

The inputs (shown in Figure 5.4) to the NN will be the NN actor stats such as health, position, action, state, last action index, etc as well as the opponent player statistics .



The green node represent a duplicate of all the blocked data except pertaining to the enemy character Figure 5.4

OUTPUTS

The outputs (shown in Figure 5.5) are the controls being sent to the NN character directly and then the next frame is executed. These are representative of all possible inputs that could be input by a human player on a controller.

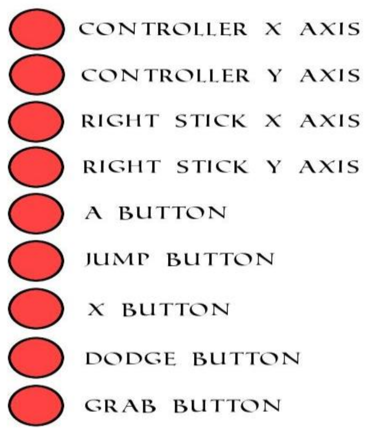


Figure 5.5

EVALUATING FITNESS

Designing a technique that lets us quantify the fitness of the network to our proposed application is important.

The fitness of the model can be determined by the simple statistics of NN character health, opponent character health, and time.

These factors can be influenced in order to train different types of adaptive AI.

The purpose of the NN actor is to kill it's opponent without dying while maximizing its interaction time with the player.

We can provide a negative reinforcement to the learning process every time the NN actor is killed and the value of the bias can modified depending on the amount of opponent player health remaining.

The character will be set to have achieved good fitness when it kills its opponent by minimizing its damage taken and maximizing the interaction time. By doing this, we will be preventing the NN character from becoming overpowered or too powerful for the player to defeat. It will promote the NN character to use a series of defensive options to counteract the opponent actions while simultaneously choosing from a variety of attack interactions as well.

GENETIC NEURAL NETWORK

The Neural Network is connected to the character via the NN controller. We can design the neural network in such a way that at each point when the character dies, a new generation of the character is spawned that has all the learned experience from its previous generations. So each time the character is killed , a new generation of the NN is implemented which improves on the previous generation by better trying to fit the model.

After rigorous training, we should arrive at a generation that will suit the problem according to our requirements.

We can also create various species per so that we can test and encounter a greater variety of possibilities.

An important part of the genetic implementations is the memory of a species, that has to be saved, updated and passed on whenever a new generation is implemented.

V. COMPARISON OF VARIOUS ALGORITHMS

GRADIENT DESCENT

Gradient descent, best known as steepest descent, is a simple and basic training algorithm. It is a first order method and moreover it requires information from the gradient vector.

This training algorithm has the flaw that it requires many iterations for functions which have long, narrow valley structures.

NEWTON'S METHOD

The Newton method is an algorithm of second order because it uses the Hessian matrix. The goal or purpose of this method is to use the second derivatives of the loss function to find better training directions.

CONJUGATE GRADIENT

The method of the conjugate gradient can be considered as an intermediate method between the gradient descent and the of Newton's method. The goal of accelerating the typically slow convergence associated with gradient descent is inspired by it. This technique also tries to avoid the information criteria associated with the Hessian matrix evaluation, storage, and inversion, as required by the Newton's method. The search is performed in the combined gradient training algorithm, which generally results in faster convergence than gradient descent.

QUASI-NEWTON METHOD

The use of the Newton technique is computationally expensive, as it requires numerous activities to evaluate the Hessian lattice and record its inverse. To tackle this downside, elective methodologies are produced, known as semi - Newton or variable metric techniques. These techniques develop a guess to the the backward Hessian at every cycle of calculation, rather than figuring the Hessian straightforwardly and then evaluating his converse. This estimate is recorded using only data on the loss function's first derivatives.

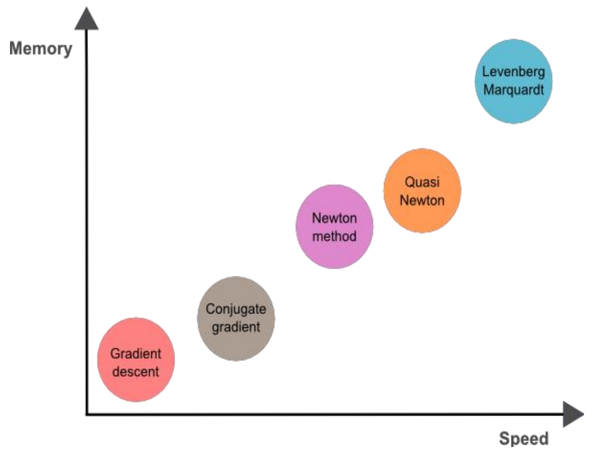
LEVENBERG-MARQUARDT ALGORITHM

The Levenberg-Marquardt algorithm was designed to work specifically with loss functions that take the form of a sum of squared errors, also known as the damped least-squares method. It works without the exact Hessian matrix being calculated. Instead, it works with the Jacobian matrix and the gradient vector.

Consider a loss function that can be expressed as a sum of squared errors of the form

f = ∑ ei2, i=0,...,m

Here m is the number of instances in the data set.



VI . IMPLEMENTATION

The focus of this system is to train a non-playable character by using a neural network. The neural net would train the NPC against all the possible play styles that the user could choose to use. The neural net can also be implemented to allow the NPC to choose from smart range of response actions to counter the user. Moreover, our system also aims to give the NPC a unique personality that will make him or her unique from other interactable characters in the game.

This implementation lets the user face new challenges and opportunities with different characters, thus making the game more immersive.

TRAINING PHASE

The training phase comprises of training the neural network or the AI System being implemented by feeding it mass data regarding different game scenarios or acquiring the training data in realtime by pitting it against a player or another AI, where it will encounter different scenarios repeatedly and learn from past scenarios to do better in future encounters. The extent of training of the Neural Network can be set during the development phase.

Another possibility would be to train the Neural network on data stored on a dataset, but the data for a game would be too large to be stored on a conventional dataset for training purposes, thus making real-time simulation/training preferable.

The main AI neural network can be trained by it manually playing against a human, but a human can not train an AI completely, thus we will prefer to train the AI with another AI upto a certain state, and further training be done based on the individual user that plays the game, so as to deliver a custom experience.

TESTING PHASE

The testing phase will consist of the trained Neural Network controlling the NPC played against the player, where the AI should be able to adapt to the moves of the player in order to offer a more engaging and challenging performance.
The AI can be trained to a certain threshold and then be used in the testing phase. In the beginning of the testing the phase the AI can be trained further by a small amount by placing it against the player in order to offer a customized experience based on the user's playstyle.

SCOPE

The scope of this project is to have maximum immersion and "realistic" behaviour from AI in video games. This could lead to a revolution in an already thriving entertainment industry. Having AI that reacts to players and environments in real time and make the right decisions could further have uses outside of this sector, as AI has a lot of unexplored progress scope.

TRADE OFF BETWEEN PROPOSED AND PRESENT SYSTEMS

The proposed system will redefine the way we play video games. It replaces the old and boring traditional AI Non-Playable Character System with a new, smart and intuitive system that adds realism to the game and better engages the player.
The system will however require research and training and tuning with respect to different kind of games which may increase the time and cost of the development. Moreover the games made by the proposed system may only be able to run effectively and efficiently on the latest systems thus minimizing the marketing opportunities.

VII. RESULT AND CONCLUSION

As seen in the above paper, as well as the surveys and existing publications, we can ascertain the value of such AI in today's market and the changes it brings to existing systems so that gaming in general is made more enjoyable.

REFERENCES

1. Artificial Intelligence for Adaptive Computer Games - Georgia Tech College of Computing https://www.cc.gatech.edu › papers
2. Artificial Intelligence in Video Games: Towards a Unified Framework - Hindawi https://www.hindawi.com › journals › ijcgt
3. IEEE Xplore: IEEE Transactions on Computational Intelligence and AI in Games ... https://ieeexplore.ieee.org › xpl › topAcc
4. Expressive AI: Games and Artificial Intelligence - Semantic Scholar PDFhttps://pdfs.semanticscholar.org
5. Rapid adaptation of video game AI. - https://www.researchgate.net › publication
6. Applications of Artificial Intelligence to Game Design | CustomWritings.com Blog https://www.customwritings.com
7. Human Computer Interaction in Game Design - Semantic Scholar PDFhttps://pdfs.semanticscholar.org
8. Player–video game interaction: A systematic review of current concepts - HAL-Inria PDFhttps://hal.inria.fr › document
9. Evolving Reactive NPCs for the Real-Time Simulation Game - Semantic Scholar PDFhttps://pdfs.semanticscholar.org