Introduction to SIDS Neural Network in Game

Kavita Sheoran
Department of Computer Science and Engineering
Maharaja Surajmal Institute of Technology
New Delhi, India
kavita.sheoran@msit.in

Shivank Awasthi, Ishtdeep Singh Malhi, Devvrat Tyagi, Shubham Goel
Department of Computer Science and Engineering
Maharaja Surajmal Institute of Technology
New Delhi, India
shivankawasthi.cse.msit@gmail.com,
ishtdeep1996@gmail.com, tyagidevvrat1995@gmail.com,
shubhamgoel415@gmail.com

***Abstract*— Game AI is hard coded and it cannot take decision on its own. So, there was a need to design human like intelligence for the opponent. This can be achieved through neutral network. This paper developed a new approach called SIDS neural network for games that is used to provide player a realistic gaming experience.**

Keywords*— *Neural Network; Artificial Intelligence; Speech Recognition; Gesture Recognition

I. INTRODUCTION

In this epoch of computers, it has always been expected that computer become as intelligent as humans. There were several attempts to increase computer's intelligence. Most of these did not succeed. Later, inspiration for making computers intelligent was taken from nature and from here the concept of Neural Network (NN) evolved. NN is an approach to replicate the human thought process and was first proposed by Warren McCulloch [1]. It allows computer to act like human however it is so naive that it cannot perform all functions but still can be used to provide basic human intelligence. Unlike conventional programming, Genetic algorithms can improve itself overtime. These algorithms work on the nature's principle of natural selection. It chooses the best values based on the algorithm and discards the irrelevant ones. It then performs mutations on those values and creates a new generation. This process continues and it keeps on improving.

Authors have proposed and implemented a novel approach known as SIDS NN. This approach is used in game to strengthen opponent's intelligence and provides the ability to analyze player's moves and make strategies to match player's skill set. SIDS also assists player to control movement in game through Gesture Recognition and Speech Recognition.

II. LITERATURE REVIEW

Authors have proposed a novel approach in [2] to design energy efficient processing element for setting up convolutional neural network. Authors have used a combination of fixed point and floating-point representation thus achieving a tradeoff in energy efficiency and expressing wide range of values. Proposed approach has shown 47% efficiency in terms of power consumption and has occupied 16% less area. The approach resulted in 54% less interaction between DRAM and SRAM with only 3% loss in accuracy.

Authors in [3] have proposed an approach called Fuzzy Rough Cognitive Network. Work claims that most of the NNs work like a black box while the ones working at granular level, with practically zero performance loss. It also has proper elucidation about selection of neurons at each level. Results have verified the performance of proposed work.

This paper [4] discusses about the structuring of NN, made by clustering of neurons in different layers. These neurons possess different functionality such as taking inputs, processing these inputs in hidden layers and some are used for giving outputs. Author also explains the methods to train the neural network such as Supervised training and Unsupervised training.

A new approach of gesture recognition is proposed in [5] based on multi-layer NN. This approach mainly follows four main steps namely 1. Gesture modeling 2. Segmentation 3. Feature Extraction 4. Classification. Gesture modeling is used to create vocabulary in accordance with the appropriate gesture. Segmentation is used to extract the hand shape from the frames. After that Feature Extraction is used to recognize the gestures and features.

In this paper [6] author has proposed a new approach to control urban traffic using combination of NN and collect traffic data. The author used improved Elman NN which are trained using back propagation algorithm, result obtained from previous stage is mixed work has a great potential to handle and solve problems related to urban traffic congestion, for an experimental data of 100 points the maximum error found was 2 and average error is 0.77.

Authors have implemented an artificial NN in [7] to predict performance of a sophomore enrolled in engineering. Certain factors were decided on the basis of experience and were considered to predict the performance. Feed forward back propagation algorithm is used for training this NN. The proposed approach gives an accuracy of 84.6%.

In this paper [8] author proposed new method for facial recognition. The algorithm used integrates principle component analysis, back propagation NN and discrete cosine

978-1-5386-1719-9/18/$31.00 ©2018 IEEE

transformation. This approach helped to improve the performance of face recognition, the normalization technique helps to reduce redundancy and PCA reduces dimension. The propose method has mean square error of the order of 10^{-4}after 26^{th} training iteration.

III. PROPOSED WORK

SIDS NN stands for Self-Adaptive Intelligent Dynamic System. Authors have developed a new approach of NN as shown in Fig. 1. based on Convolutional and Recurrent NN.

- Recurrent NN is a type of unidirectional and cyclic NN whose weights are adaptive. In this type of NN, the output is dependent on previous computations. They find applications in tasks like Natural Language Processing.
- Convolutional NN is a type of Feed-forward NN. The inspiration behind this type is the visual cortex of animals in nature. This NN is capable of handling overlapping visual fields. Visual field refers to restricted region on which neuron responds.

SIDS NN has following components: -

A. Opponent Intelligence

B. Player Control

1) Gesture Recognition

2) Speech Recognition

A. Opponent Intelligence

It may be defined as ability of a program to decide the best possible output on the basis of player's input moves. Authors have designed opponent intelligence module which helps enemy to understand player's moves and then its own moves accordingly. It is a recursive neural network with 3 input neurons with 2 hidden layers and 1 output neuron and each hidden layer has 4 neurons. This specific neuron scheme was used for this particular game. However, there is an option to redesign the neural network according to the game in which this module is used.

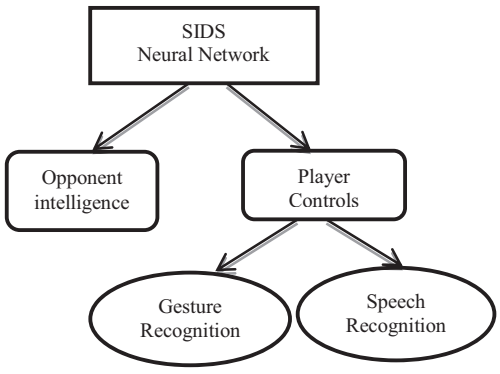

Fig. 1. Components of SIDS Neural Network.

B. Player controls

It is a new method provided to the player to control the main character in the game. Conventionally input in games is provided through keyboard and joystick. Authors have implemented player control module and have further categorized it into two sub modules namely: - Gesture Recognition and Speech Recognition.

1) *Gesture Recognition:* It is a component of the Player Control Module that allows the player to control the direction of movement of the main character using the input taken from the webcam, the player moves the object and character in game responds according to the movement of object in front of the camera.

2) *Speech Recognition:* This is another component of Player Control Module that helps the player to enable voice control feature and gesture control component which work together and provide the complete control of the main character, voice command handles the interactions of the main character in the game.

IV. IMPLEMENTATION

Authors have implemented SIDS NN in an open world interactive horror game using Unity Game Engine ™ and C# as the implementation language. Further implementation details of each module are as follows:

A. Implentation of Opponent Intelligence Module

This component is the brain of Opponent in game. It helps opponent to make decision on the basis of input provided. NN learns to counter player's moves and tries to match player's skill level.

Firstly, the number of layers, neurons per layer and weights have to be specified on the basis of the input provided. NN is generated by this module. Then this generated NN is used by opponent to defeat player. NN takes player's move as input and tries to adjust the weights to give suitable output for maximizing Fitness Level. Fitness level is a measure used to decide how effective NN is in defeating or countering player moves. This module is further categorized into 9 sub modules.

1) Layer Initializer – It initialize the layers, neurons and the weights of the NN. The tasks performed by this sub module are:

- Initialize layers by allocating memory to store layer data.
- Initialize Neuron Initializer and then Weight Initializer.

2) Deep Copy Creator – It creates a copy of NN and all its values before performing mutation operations. Functions performed by this sub module are:

- Create a new array to store a copy of layer data and assign layer details to copy array.

- Initialize Neuron Initializer and then CopyWeight Function.

3) CopyWeight Function – It copies the weight values of all the weighted connection in the NN.

4) Neuron Initializer – It runs through each layer and adds neuron in each layer, initializing them with values.

5) Weight Initializer – Functions performed by this sub module are:

- Allocate memory for weight list and then iterate over neurons that have a weighted connection.

  i. Create weight list for layer which is being scanned.

  ii. Store the no of neurons in previous layer in a count variable and iterate over all neurons in current layer.

- Create neuron weight array that has to be assigned using count variable as array length.

- Iterate over neurons in previous layer and assign weights randomly in the range -1 to 1.

- Add neuron weight of current layer to layer weight list.

- Convert weight list array to 2D and then to 3D.

6) Feed Forward NN – Functions performed by this sub module are:

- Store inputs in the neural matrix and iterate over all neurons by calculating sum of weights of neuron in this layer and previous layer then assign that value as new weight.

- Use S(x) (Sigmoid function) or tanh(value) (tanh function) to squash values between -1 and +1

$$S(x)=1/(1+e^{-x}) \quad (1)$$

$$tanh(value) = (e^{value} – e^{-value})/ (e^{value} + e^{-value}) \quad (2)$$

7) Mutator Function – Functions performed by this sub module are:

- Iterate over all weights.

  A random number is generated between 0 and 10 and stored in the variable RN.

  If (RN <= 2): weight = -1x weight

  If (RN <=4): weight = randomly assigning any number between -1 and 1

  If (RN <= 6): Random factor between 0 and 1

  weight = weight + (weightx(random value between 0 and 1))

  If (RN<=8):

  weight = weight – (weightx(random value between 0 and 1))

- Assign mutated weight value.

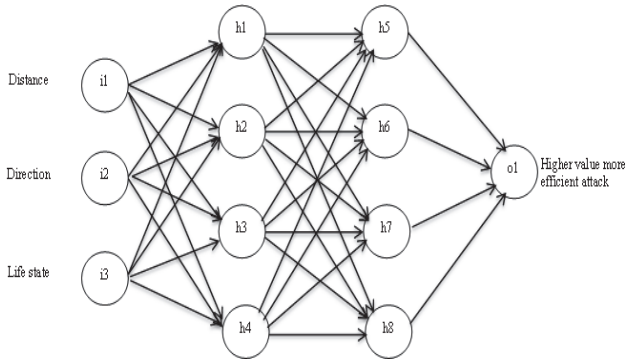

Fig. 2. Implemented Opponent Intelligrnce NN

8) Fitness Adjuster – This sub module adjusts the fitness level of a NN on the basis of its efficiency. If efficiency is greater, then

fitness=fitness+fitfactor

Else

fitness=fitness-fitfactor

where fitfactor value can be set to any positive number. It also has two methods which are Setfitness and Getfitness. Setfitness is used to set fitness level of NN. Getfiness returns fitness value of NN.

9) Decision Maker – It returns an integer which is used to decide whether NN is fit enough to replace the previous NN.

If there is no other NN to compare then do nothing.

If (fitness > other fitness): return 1 as NN is better.

If (fitness < other fitness): return -1 as NN is worse.

Else return 0 as both NN perform similarly.

Authors have developed a NN and implemented it in horror game. Sub modules of opponent intelligence module have assisted in creating this NN. Fig. 2. is the neural scheme which is implemented in game. There are 3 input neurons, 1 output neuron and 2 hidden layers with 4 neurons in each layer.

B. Implementation of Gesture Recognition Module

Gesture Recognition Module is shown in Fig. 3. Webcam specifications of the setup were 640x480 pixel resolution and frame capture rate is 30 frames per second. Therefore, setup's height is 480 pixels and its width is 640 pixels. So, there is a need to allocate memory to store data of 640x480 pixels which results in 307200 pixels of data.

For each pixel 32 bits are required to store data

- 8 bits – red component
- 8 bits – green component
- 8 bits – blue component
- 8 bits – alpha component

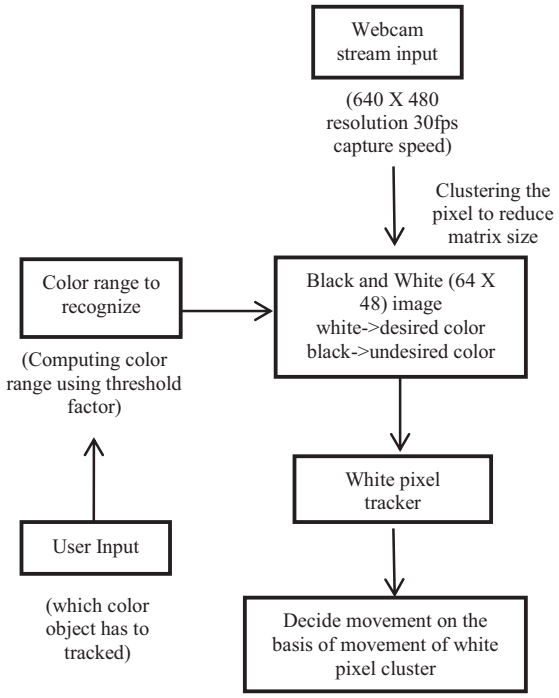

Fig. 3. Working of Gesture Recognition module

Hence, Color-component Size (CS) is 32 bits. Memory to be allocated (MA) is expressed as:

$$MA = CS \; x \; height \; x \; width \qquad (3)$$

Now the data of 640x480 pixels has to be compressed down to 64x48 pixels data to limit computational resource usage. Method followed to crunch the pixel data:

The compression factor (CF) for height and width is 10. So, data of 100 pixels (height to compressed x width to compressed) will be replaced by 1 pixel in compressed data form.

$$Compressed\ width\ size\ (CWS) = width/CF = 64 \qquad (4)$$

$$Compressed\ height\ size\ (CHS) = height/CF = 48 \qquad (5)$$

$$Compressed\ data\ size\ (CDS) = CWS \; x \; CHS \qquad (6)$$

To compute value of 100 pixels a lot of computational resources are needed. So, edge value and mid value out of 100 pixels chunk as shown in Fig. 4. is used to create one pixel of compressed data size.

Formulas used to compute the values from uncompressed pixel array are discussed below. These values are then stored in the LL, LR, MP, UR and UL arrays. The size of these arrays is also equal to size of compressed data size array. It stores the UR, UL, MP, LR and LL values corresponding to each pixel in compressed data size array. These values will be used to compute which values will be stored in compressed data size array.

$$LL[IV]=MA[0+((IV/(width/CF))xwidthxCF)+(mod(IV,width/CF))x\ CF)] \qquad (7)$$

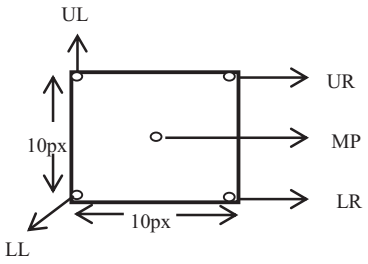

Fig. 4. Representation of 100 pixel cluster and its edge values

$$LR[IV]=MA[0+((IV/(width/CF))xwidthxCF)+(mod(IV,(width/CF))x\ CF)+(CF-1)] \qquad (8)$$

$$UL[IV]=MA[0+((IV/(width/CF))xwidthxCF)+(mod(IV,(width/CF))x\ CF)+(heightx(CF-1)) \qquad (9)$$

$$UR[IV]=MA[0+((IV/(width/CF))xwidthxCF)+(mod(IV,(width/CF))x\ CF)+heightx(CF-1)+(CF-1)] \qquad (10)$$

Adjustment factor is taken into account for calculating middle pixel value as middle value cannot be found directly from CF. Therefore, instead of taking average, it is taken alternatively to either 5th pixel or 6th pixel. This keeps alternating

$$Adjusting\ factor\ (AF) = \begin{cases} (CF/2)-1 \\ (CF/2) \end{cases} \text{ Used alternately} \qquad (11)$$

$$MP[IV]=MA[0+((IV/(width/CF))xwidthxCF)+(mod(IV,(width/CF))xCF)+(widthx(CF-1))+AF] \qquad (12)$$

Now all the pixel color values have to be checked for the color that is provided as user input. This will be later used in gesture control. User input value is captured from a view port of 200x200px in the center of webcam display, an average of that value is taken and a threshold is applied in order to get minimum and maximum of the range of colors to be scanned.

$$avgpixel = avg \sum_{i=0}^{sizeof(CDS)-1} CDS[i] \qquad (13)$$

In this test case suitable value for Threshold factor (TF) is 0.2. All the color values within the range of min value and max value will be accepted. These values are computed as follows:

$$min\ value = (1-TF) \; x \; avgpixel \qquad (14)$$

$$max\ value = (1+TF) \; x \; avgpixel \qquad (15)$$

Now removal of unwanted pixel has to be done. It will be represented by black color and acceptable pixel color will be represented as white in compressed data size. If the following condition holds

min value <= LL <= max value|| min value<= LR<= max value|| min value<= MP<= max value|| min value<= UL<= max value|| min value<= UR<= max value

CDS[IV] =white color

Else CDS[IV] = black color

Now tracking of white pixel cluster has to be done in order to find movement of object and that movement will be used to

compute player motion. For example: If white pixel cluster center moves up then it will move player head up or if it moves left then it will move player left. Same procedure will happen for each case. Here the voting system is used to decide the direction of movement of player if the white pixel cluster moves left then the left movement region gets more votes as compared to movement regions and same will happen for each case.

C. Implementation of Speech Recognition Module

This module is used to process the audio signal which are taken as input from microphone by the Human- Computer interaction. The bitrate of the microphone used in this setup is 48000Hz. The diagrammatic representation of working of module is shown in Fig. 5. Following steps are used to process the audio signals.

- Prepare a list of microphones and if the number of microphones is one then set it as default. But in case of multiple microphones ask user to choose the microphone according to their preference.
- Start microphone and capture audio stream provided that the bit rate of that audio must be either 44100hz or 48000hz.
- Analyze bit stream and find possible phoneme matches using phoneme recognizer which in turn uses Phoneme dictionary for comparison purposes

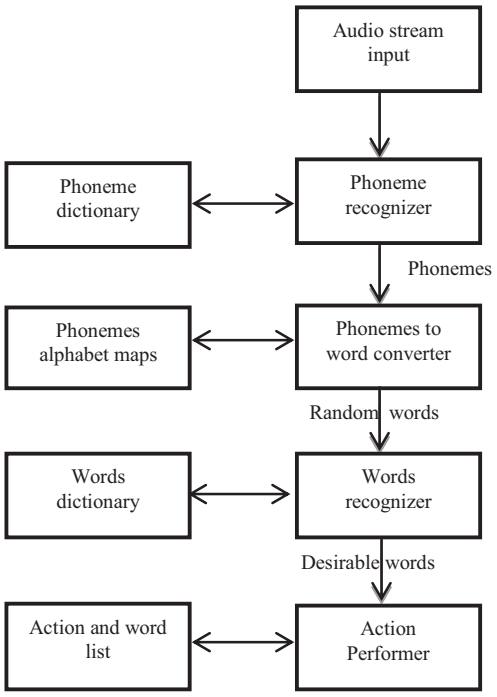

Fig. 5. Working of Speech Recognition module

- From phoneme random words are created using Phonemes to word converter with the help of Phoneme alphabet maps
- Now Desirable words are separated out using Word Recognizer which checks for the list of words from Words Dictionary
- Now actions are performed on the basis of the detected word.
- Use command list dictionary to compare words with words inside the list and perform action accordingly. Various commands are stored in Command list some of them are walk, run, walk back, run back, turn back, stop, turn on, turn off, capture. There is a unique functionality associated with each of these commands in the command list. For example: If command is walk then player starts walking forward or if command is turn on then activate voice command control and gesture recognition (standard control mode off).

V. RESULT

The proposed approach of SIDS NN was implemented in an open world interactive horror game and some snapshots of working modules are analyzed in this section.

The Opponent Intelligence module analyzed player's moves and tried to boost the fitness levels to match player's skill a sudden drop in fitness value was noticed when player tried to defeat AI (value = 98.84454) but after that the AI managed well to boost up its fitness level which is shown in Fig. 6. The opponent Intelligence's improvement in terms of fitness level is by a minimum of 11.73% and the maximum improvement is 36.48%, whenever the player tried to defeat it. How the value of weight of connection between input neuron i1 and hidden layer neuron h1 changes such that it can adapt itself according to the changes is shown in Fig. 7.

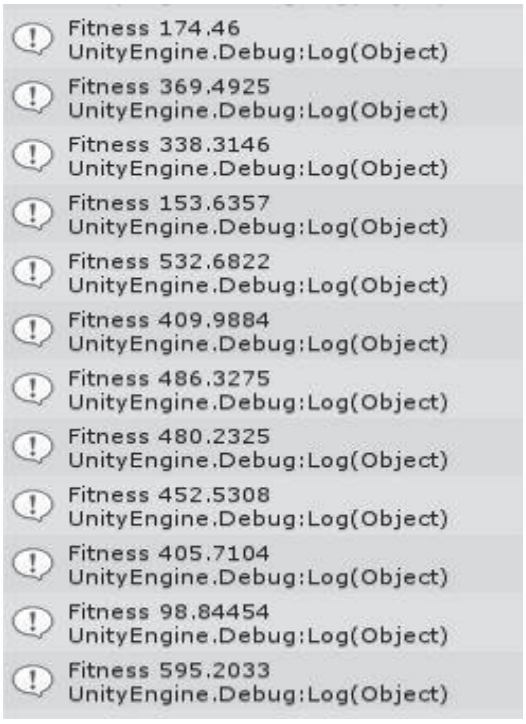

Fig. 6. Changing Fitness values

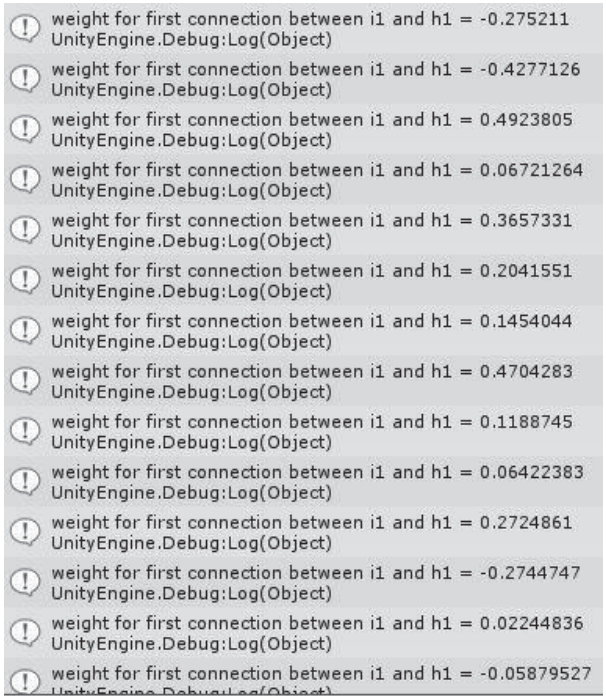

Fig. 7. Weights change as the neural network tries to adpat

The gesture recogniton module detected the color of the object to be tracked which is green in this case and on the basis of the movement of object, the motion of character was controlle. According to the Fig. 8. this character moved to left. Here the threshold value taken was 0.2 that helped in accurately calculating min values and max values. How the gesture recognition affected the CPU latency is howm in Fig. 9. and Fig. 10. In this case it was 5.2ms which means that it doesn't compromise with performance and is CPU friendly.

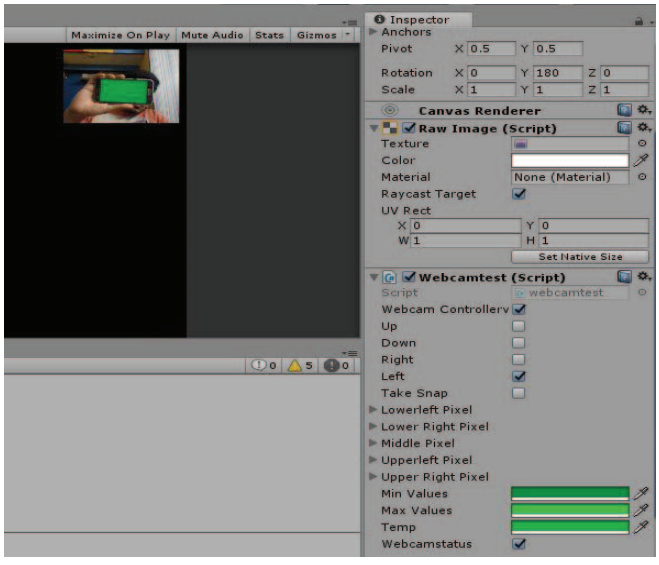

Fig. 8. Gesture recognizer responding to motion

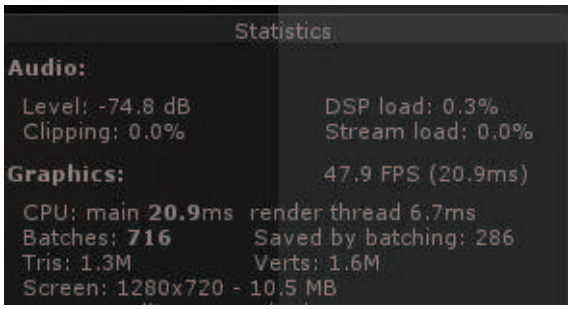

Fig. 9. CPU latency without gesture control

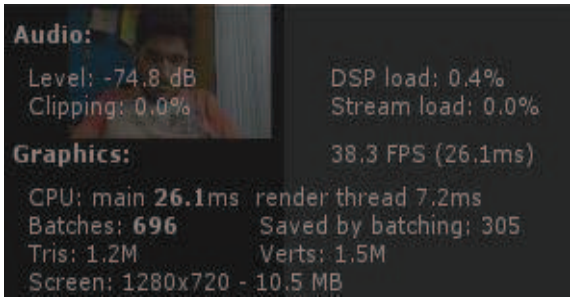

Fig. 10. CPU latency with gesture control

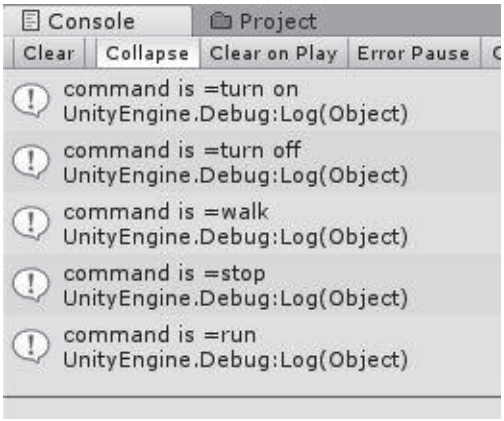

Fig. 11. Speech Recognition module responding to commands

How the Speech recognition works according to given command is featured in Fig. 11.

VI. CONCLUSION AND FUTURE WORK

SIDS can be used in games that have single player mode. It will make the opponent smart enough to tackle moves. It will also provide different gaming expierience by providing new methods for players. This will help to control main character and the Gesture Recognition component is not very resource consuming and will run well on system with average configuration. It is better than other conventional AI techniques because it will provide personalized expierience according to who is playing the game.

Authors propose to make SIDS compatible with Virtual Reality, Augmented Reality, Mixed Reality in the future versions. Authors also aim to improvise the Opponent Intelligence module by following different learning models and collecting the neural network data from various systems running the game and will create a master database in the cloud which help opponent Intelligence perform better.

Limitations of current approach is that the gesture recognition module of SIDS NN so far works only to recognise the movement of a uniformly colored object and track the movement of that object, it is not effective in differentiating patterns.

VII. REFERENCES

[1] http://www.psych.utoronto.ca

[2] Y. Choi et. al., "Energy-efficient design of processing element for convolutional neural network," in IEEE Transactions on Circuits and Systems II: Express Briefs, unpublished.

[3] G. Nápoles et. al., "Fuzzy-Rough Cognitive Networks," in Neural Networks 97 (2018) 19 – 27, http://doi.org/10.10.16/j.neunet.2017.08.007.

[4] S.B. Maind and P. Wankar, "Research Paper on Basic of Artificial Neural Network," in International Journal on Recent and Innovation trends in Computing and Communication, Vol.2, Issue 1, pp. 96-100, ISSN: 2321-8169.

[5] K. Arora, S. Suri, D. Arora and V. Pandey, "Gesture Recognition Using Artificial Neural Network," in Intenational Journal of Computer Science and Engineering, Vol. 2, Issue 4, E-ISSN: 2347-2693.

[6] Z. Zhou and K. Huang, "Research on a Combined Neural Networks Prediction Model for Uran Traffic Volume," in International Seminar on Future Biomedical Information Engineering, DOI: 10.1109/FBIE.2008.15

[7] S. A. Naseer, I. Zaqout, M. A. Ghosh, R. Atallah and E. Alajrami, "Predicting Student Performance Using Artificial Neural Network: in the Faculty of Engineering and Information Technology," in International Journal of Hybrid Information Technology, Vol. 8, No. 2 (2015), pp. 221-228, http://dx.doi.org/10.14257/ijhit.2015.8.2.20.

[8] A.E. Shivdas, "Face Recognition using Artificial Neural Network," in International Journal of Research in Management, Science and Technology, Vol. 2, No. 1, April 2014, E-ISSN: 2321-3264.