Note On Authorship

Although there are multiple coauthors listed, I was the main person who wrote the text of this paper. Zhang collaborated with me on the research, and Schiffer served as our mentor and guide to the project.

The paper was later adapted to one that was published in the Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications – HUCAPP, but this version was not published.

-Max Levine

Neural Networks for Meta-Emotions

Samantha Zhang*, Maximilian Levine†, Sheldon Schiffer‡
Department of Computer Science
Cornell University*, University of North Carolina at Asheville†, Georgia State University‡
Email: sjz46@cornell.edu*, mlevine@unca.edu†, schiffer@gsu.edu‡

***Abstract*—To increase Non-Player Characters' (NPCs) realism, we trained recurrent neural networks modeling human emotion patterns. We generate training data by putting acting footage through Facial Expression Recognition (FER) software. We used 4 different actors reacting to each other in a simulated police investigation setting. Using Blender, we created 3D models of the actors' heads and animations of Paul Ekman's 6 basic emotions. In Unity, two head models face each other, with cameras coming from their eyes to view the other. We allow the heads to receive an emotion vector and animate them changing emotions. The neural network can eventually be attached to the Unity scene, to make the faces react to each other intelligently.**

I. Introduction

Non-Player Characters (NPCs) are characters in a video game that populate the game world, interact with players, and contribute to immersion. Current NPCs, typically controlled with finite state machines, are predictable. Unrealistic interactions break the player's immersion. The quality of NPCs is essential, as people are good at identifying uncanny humans. The face is the most expressive part of the human body. We aim to design a system standardizing more thoughtful NPCs through the complexity of facial expressions. By making an emotion model that supports a high degree of facial complexity, we hope to increase NPC believability. Making NPCs more intelligent and realistic is a fundamental, challenging problem. And by giving actors the creative control to frame NPC behavior, we can have more realistic and expressive NPCs.

II. Related Work

The problems with current NPCs, their limitedness and predictability, and their importance with regards to immersion in games, are discussed by Paige [8]. The standard way of controlling NPC's behavior is through finite state machines, directed graphs which define actions on nodes and causes on edges. For example, a guard will change from standing to attacking when seeing the player. But as the name suggests, any implementation of this method will be limited and finite. An improvement on the finite state machine is the fuzzy finite state machine. This allows for multiple possible transitions from one state, depending on other variables. For example, whether the guard attacks is determined by its relationship to the player. Clearly, these prior methods are very predictable.

Kozasa, in a pioneering 2006 work, created an artificial neural network using current NPC emotion parameters and user's interaction, physical action, and input expression. The NPC has a certain personality and mood. The system took as input dialogue, which can be speech or text, as well as the player's facial expression. This is put through a neural network, and then the system outputs one of five possible emotions to be displayed on a 3D face, as well as a text or spoken response. The emotion may influence future decisions. To train the network, Kozasa and their team manually wrote their own training data. This was a process in which the team recorded many different levels of states of emotions, and then tried to predict what the response would be like, which limited the amount and quality of the training data. However, we agree that a neural network is a great way to process emotions, with its similarity to the human brain and built in fuzziness of response. [2]

Zhou created an NPC model simulation using the Pleasurability, Arousal, Dominance (PAD) emotional state model and vectors to determine emotions, set in a mining rescue scenario [11]. PAD, and other attempts at systematizing emotions, are shown in Fig. 1. It is best to stick to an emotional model's elements and avoid abstraction.

"Facial Animation Using Emotional Model" by Kozasa	The Big Five	PAD Emotional Model by Russel & Mahrabian	Circumplex Model of Affect by Russell & Mahrabian	Paul Ekman's Basic Emotions
Emotional parameters as extroversion, agreeableness, valance, arousal, likeability, and intimacy, as well as possible input actions and the input expressions of neutral, smile, anger, sadness, and surprise [2].	Extroversion, agreeableness, conscientiousness, neuroticism, and openness, each on a scale from 0 to 1. Widely accepted in marketing and game design.	Core Affect is determined by pleasure/displeasure, arousal/non-arousal, dominance/submissiveness. Any discrete emotion label can be represented as an imperfect blob-cone coming from the origin out of the sphere.	Represented on a 2d circular space with degrees of active/inactive (arousal) and pleasant/unpleasant (valance)	Happy, sad, angry, surprised, scared, disgusted, and neutral We use Ekman's basic emotions as an emotional model for our research as it is widely accepted in the emotional intelligence industry and many resources, like FACS coding, are founded on this model [5].

Fig. 1: Ways of representing personality and emotion.

Another method is the Facial Action Coding System (FACS), which is regarded as the standard objective measure of facial expressions. FACS describes every unit of discernible muscle movement on the face. FACS consists of 51 Action Units (AUs), which are the smallest units of visible muscle movement. In order to get results using FACS, a trained, human coder has to go frame by frame, measuring the facial expressions on each frame, which is cost and time intensive [5]. However, Facial Emotion Recognition (FER) software to automate this process, such as FaceReader and Affectiva, is becoming an alternative.

To process a face, FaceReader first finds where the face is; it does so with the Viola-Jones algorithm. Then, it creates a 3D mesh of this face that includes all of the key points with the Active Appearance Method. Finally, a neural network classifies the facial expression. If steps one or two fail (due to

the face being only partially visible), then the fallback, called Deep Face, simply uses a neural network without the pre-processing. All of these neural networks are trained both by databases of actors making the expressions, as well as images coded by FACS coders. FaceReader uses the AUs to determine the 6 basic emotions. [4]

Psychologists Ortony, Clore and Collins were among the first to propose an emotional model to be simulated on a computer. Psychological and computational models of emotion were developed to have some systematic description of emotion that could be translated into software [7]. For this, Ortony et. al developed appraisal theory. Appraisal consists of assessing the environment, which impacts internal beliefs. These theories both jive well with the performance theory that entertainment actors follow: actors are best able to imitate an emotion by addressing their affective memory, i.e. actually reliving the emotion based on experience. Under this model, emotions are determined through a process of appraisal of the environment. The environment is built of inanimate objects with a certain likability, animate agents with a certain praiseworthiness, and events with a given desirability. Appraising the environment of these aspects results in affirming or dis-affirming beliefs, fulfilling desires, and enabling or dis-enabling intentions. Finally, there is a process of reappraisal, which adjusts the beliefs, desires, and intentions to the situation. This system represented emotions with the circumplex model of affect, or Core Affect (see Fig. 1). In 2014, Dias et. al implemented a computational cognitive psychology system in the form of FAtiMA that implemented a version of the affect model [10].

Schiffer did an experiment to test a computational emotion system. The first step was to create the environmental situation that the agents would be performing in. This consisted of a graph structure that represented varying states, attributes such as goals and obstacles, possible actions with corresponding facial animation, resulting emotions, and many possible paths through the graph. Video data was collected of an actor acting out all paths through the graph. A neural network was trained to handle video input of a player playing the game and interacting with the NPC through dialogue choices as well as facial expression [10].

III. Methodology & Implementation

A. Emotion Model

We based our NPCs emotion model on Paul Ekman's 6 basic emotions (see Fig. 2). Rather than modeling single emotions, our model combines different intensities of each emotion. We chose Ekman's model due to the extensive amount of resources supporting it.

B. Neural Network

Using a system of several recurrent neural networks trained to an actor's character, we can predict the character's facial expressions. The idea of training a neural network has similarities to training actors. Looking at this more closely provides some justification for using neural networks to make

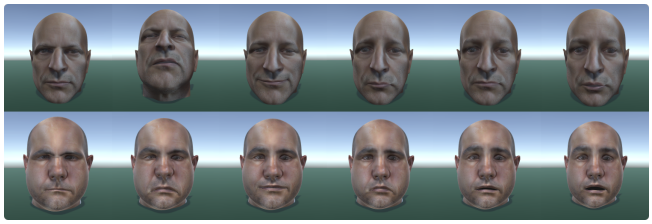

Fig. 2: Ekman's 6 basic emotions on two actors. From left to right: anger, disgust, happy, sad, scared, surprised.

intelligent NPCs. The Meisner technique trains actors through repetition to build a character using affective memory [6]. Actors call on their memory of similar emotions to determine their character's feelings. So, by repeating action exercises, an actor can train their brain through memory towards creating a character. This technique is similar to how neural networks are trained. We can enter repeated scene footage data to train the neural networks on how an NPC character should act.

C. Collecting Data

FER software can be used on acting footage to generate emotional training data for a character's neural network. Actors were recorded while acting scenes of their characters in response to stimuli, a simulation of a police investigation scene. One actor serves as an investigator while the other serves as a suspect. 288 total video recordings were compiled for each actor, with each scene being shot 9 times. By running this footage through FaceReader, we generated emotion data for each actor's character. Performing 3 emotion analyses each second, FaceReader measured the intensity for each of the emotions.

D. Modeling Emotional Data

Fig. 3: Constructing meshes of actors using FaceBuilder.

KeenTools's FaceBuilder plugin for Blender allows for the creation of 3D models of faces. Using photos of an actor's face at orthographic angles, we reconstructed their face and generated textures for the 3D polygon face meshes (see Fig. 3). FaceBuilder also produces shape keys in Blender for Ekman's 51 AUs on the mesh, which allow users to deform objects into new shapes for animation. We determined the AUs to use and combine by referring to the iMotions emotional structure on AUs for the basic emotions and matching an image of the actor at the highest respective emotional intensity [3]. Sometimes, additional AUs were added somewhat subjectively by looking

at the reference photos; for example, the emotion disgust usually does not include squinted eyes, but one of our subjects expressed squinted eyes for the emotion, so that AU was added to their model. Degree of AU activation was determined by matching the apparent activation in the reference photos. The photos were 6 stills chosen from the acting footage data. We combined multiple AUs into discrete emotion labels as new shape keys, by using the New Shape Key From Mix option in Blender to combine AUs. We then exported the face object from Blender and its respective shape keys as an FBX file for facial animation in Unity.

E. Animating Emotions in Unity

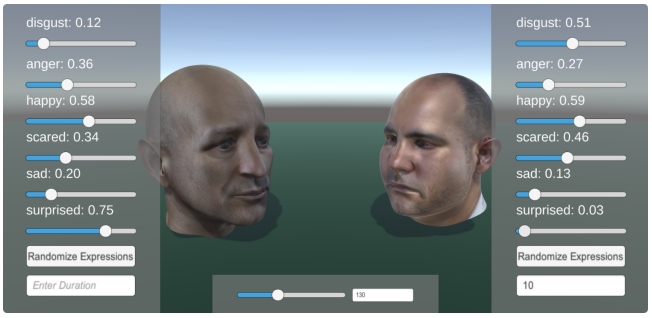

Fig. 4: Model A (left) and Model B (right) facing each other in Unity, with controls for emotional expression.

Blender shape keys are translated into Unity blendshapes. The blendshapes for the 6 emotions can be adjusted and also combined, which allows us to express meta-emotions, creating more realistic results. We set up two heads in Unity with GUI sliders to control the blendshapes and each of their emotions. We allowed the heads to rotate to face each other. Cameras were aligned with their eyes so that, in the future, they can receive input from the other's expression. We implemented a random expression generator to send data to the models, for when the neural network is applied to our system. A button press generates a random 6D emotion vector. Then the current blendshape values are linearly interpolated incrementally to the vector over a given duration. Thus, we gradually animate the original facial expression to a target emotional expression. (See Fig. 4)

F. Technologies

The following technologies were used to complete the project: FaceReader, Affectiva (Beta), FaceBuilder, Blender, Python, Unity, C#, Google Sheets.

IV. Results & Conclusions

The error of our six basic emotions is negligible, the primary difference differing on average by 0.3%. We determined the accuracy of our system by running the images of the 6 optimal stills from the acting footage of two actors and 6 images of each emotion blendshape at max intensity for the Unity heads of their respective actors through Affectiva (Beta), a FER software. Then, we calculated the mean difference for each emotion between the actor image and the corresponding animated face image, treating the actor image data as our truth values (see Fig. 5).

The primary difference indicates only the emotion that was being tested for, e.g. the happiness value when comparing the actor still and Unity model for happiness. The secondary difference averages the traces of emotions detected by Affectiva that were not being tested for, but which still may maintain consistency across the actor still and Unity model.

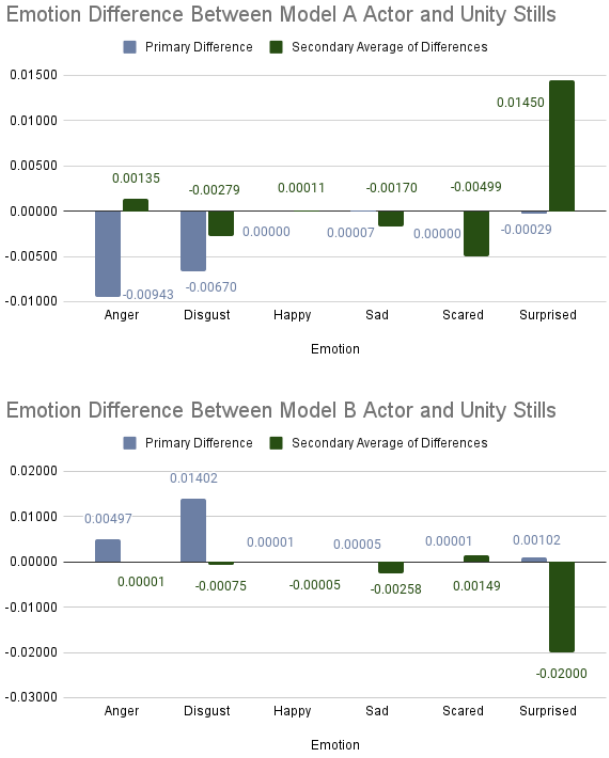

Fig. 5: Mean difference between actor stills and Unity models of each emotion when put through Affectiva.

Looking only at the primary difference, we obtained the following average percentage of error by emotion: 0.00% happy, 0.01% sad, 0.72% angry, 0.07% surprised, 0.00% scared, and 1.04% disgusted, for a total average of 0.30%. Looking at the secondary differences, we obtained the following average percentage of error by emotion: 0.01% happy, 0.21% sad, 0.01% angry, 1.73% surprised, 0.32% scared, and 0.18% disgusted, for a total average of 0.42%.

Affectiva is typically used for processing video and not still images. It utilizes convolutional and recurrent neural networks [1], and its assessment of an input, even a still image, changes over time. So we calculated the variance over time for each emotion, stills off which were put through Affectiva for 1 minute each (see Fig. 6).

We are able to conclude that our system for converting an actor's emotion into a Unity model was accurate within less than 1% of error. And the error that appeared in our analysis made sense: variances and differences occurred less

for happy, sad, and scared, and more for anger, disgust, and surprise. The reason for this is likely that the former emotions are more represented in databases of training data for neural networks, whereas the latter emotions are less represented and thus harder for FER software to recognize [9].

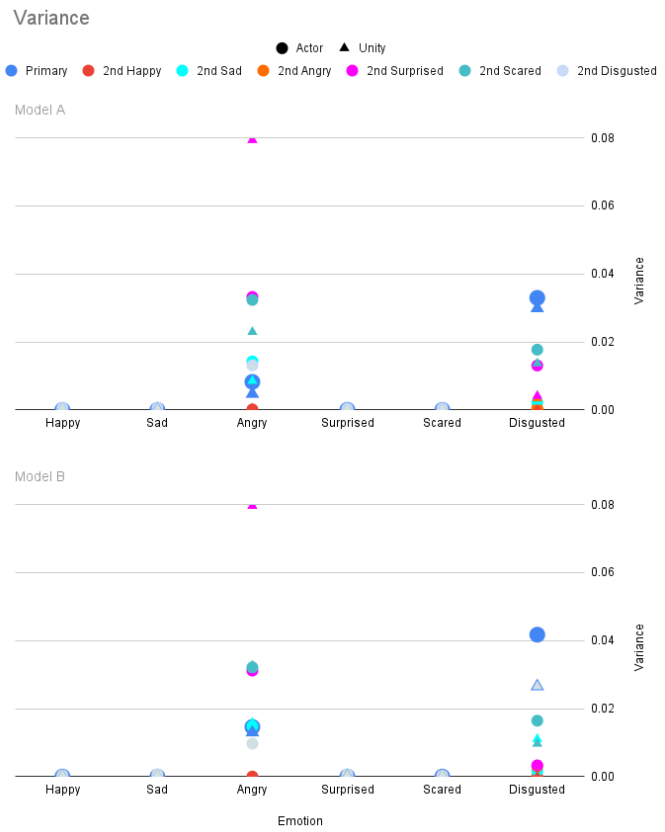

Fig. 6: Variance of Affectiva output over 1 minute analysis of actor and Unity model stills.

V. Future Work

In the future, we can fully integrate the recurrent neural network system into our Unity facial animator program. We can then further improve our calculations on the accuracy of this system by comparing emotional data generated from the reaction of an actor to a stimulus to the reaction of the facial animation controlled by the neural network.

We would also like to converge an accurate error coefficient for FER programs. FER is imperfect and produces different results for a real actor and their model. Through iteration, we can determine a coefficient to get the results for the actor and model to be the same. This would effectively make the variance seen in Fig. 6 zero.

Another avenue of research would be to find crowd sourcing methods to produce better training data for neural networks for emotions. To do this, we could gather various correlations between emotions and response in order to create a function that would be able to reasonably predict possible emotional response. We can use crowd sourcing to gather this data, by asking participants to identify correlations, or answer yes or no questions about randomly chosen correlations. Data could also be collected remotely from actors using FaceReader Online.

Acknowledgments

This material is based upon work supported by the National Science Foundation. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

We would like to thank Georgia State University and CSCI Faculty, our fellow REU interns, and the National Science Foundation.

References

[1] Affectiva, Science Deep Dive Series: How Affectiva Brings Emotional Intelligence to Technology with Algorithms. Accessed 28 July 2021.

[2] H. F. Chihiro Kozasa. Facial animation using emotional model. *International Conference on Computer Graphics, Imaging and Visualisation (CGIV'06)*, 2006.

[3] iMotions. Facial action coding system (facs) – a visual guidebook. Accessed 19 July 2021.

[4] O. K. Leanne Loijens. Facereader methodology note. *Noldus Information Technology*, 2019.

[5] P. Lewinski, T. den Uyl, and C. Butler. Automated facial coding: validation of basic emotions and FACS AUs in FaceReader. *Journal of Neuroscience, Psychology, and Economics*, 7(4):227–236, 2014.

[6] S. Meisner. *Sanford Meisner on Acting*. Vintage Random House, 1987.

[7] A. Ortony, G. L. Clore, and A. Collins. The cognitive structure of emotions. *Cambridge University Press*, pages 34–58, 1990.

[8] N. Paige. How to create smarter npcs in games. *Medium*, 2020.

[9] R. Panda, H. L. J. Zhang, J. Lee, L. X, and A. Roy-Chowdhury. Contemplating Visual Emotions: Understanding and Overcoming Dataset Bias. *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 579–595, 2018.

[10] S. Schiffer. *Bridging the Gap Between AI, Cognitive Science, and Narratology With Narrative Generation*. IGI Global, 2021.

[11] C. e. a. Zhou. Affective computation based npc behaviors modeling. *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology Workshops*, 2006.