Fuzzy User Satisfaction in Games

Jeff Craighead
Institute for Safety, Security and Rescue Technology
University of South Florida
4202 E. Fowler Ave.
Tampa, FL, USA
craighea@cse.usf.edu

***Abstract*— This paper presents a fuzzy logic based method to track user satisfaction without the need for devices to monitor users physiological conditions. User satisfaction is the key to any product's acceptance; computer applications and video games provide a unique opportunity to provide a tailored environment for each user to better suit their needs. We have implemented a non-adaptive fuzzy logic model of emotion, based on the emotional component of the Fuzzy Logic Adaptive Model of Emotion (FLAME) proposed by El-Nasr, to estimate player emotion in UnrealTournament 2004. In this paper we describe the implementation of this system and present the results of one of several play tests. Our research contradicts the current literature that suggests physiological measurements are needed. We show that it is possible to use a software only method to estimate user emotion.**

***Index Terms*— Fuzzy Logic, User Satisfaction, Adaptation**

I Introduction

User satisfaction is the key to any product's acceptance; computer applications and video games provide a unique opportunity to provide a tailored environment for each user to better suit their needs. The question is how to judge user satisfaction in an non-intrusive manner.

Typical methods, such as those proposed by Rani[1] use external devices that can monitor heart rate, stress level, brain activity, and other physiological conditions. The devices required to measure physiological conditions are intrusive, requiring the user to be attached to a machine with several wires, wearing the device, or by using a special controller. This approach is cumbersome to set up and would detract from the user experience in a commercial application.

Online, affective tuning of an application using fuzzy logic to approximate a user's emotional state will allow the application to provide the user with a better experience, without the need for cumbersome wires or special controllers. This paper presents a proof of concept fuzzy logic based method for tracking user satisfaction via estimated emotional state. Our test case measures user satisfaction of players of the UnrealTournament 2004 game. Section II presents a review of recent work in the area of user satisfaction and fuzzy logic based affective user models. Section III explains the approach taken for developing the fuzzy satisfaction tracker. Section IV details the implementation of this system. Section V presents the results from one test run of the application. Finally in Section VI we conclude with a discussion of fuzzy logic as a viable tool for judging user satisfaction.

II Related Work

This section contains a review of recent work in the area of determining user satisfaction. Popular methods in the literature are to attach to the user a device that measures some physiological condition such as heart rate, brain activity, or skin conductivity. Several of the works reviewed use fuzzy logic to fuse physiological measurements in order to gauge a user's stress level. While these methods can successfully measure user satisfaction, they require the use of an external sensor. This makes them impractical for use in commercial products.

Rani[1], [2] uses a fuzzy logic based measurement of a user's heart rate in order to facilitate human robot interaction. Rani states that a device measuring a user's physiological state could be built into a small wearable device, however this is not demonstrated in the paper. Instead the experiment was conducted using a MATLAB based system on a desktop PC. The system is able to successfully capture the user's anxiety level and send this to a robot using an architecture based on Brooks' subsumption architecture[3]. The user is not directly controlling the robot, but issuing commands to a semi-autonomous system; when the user is under stress the robot activates a certain behavior. However, in the case of a typical computer application, the need for yet another input device is cumbersome. Rani's work shows that the idea of fuzzy logic based affect measurement is sound, but a nonintrusive implementation is needed.

Other work such as that presented by Conati[4] and Zhou[5] attempt to ascertain the emotional state of a student playing an educational game. Conati examined several physiological conditions, deciding on EMG and skin-conduction as the means of determining the players emotional state. One thing noted is that these metrics use a series of thresholds to determine the user's state, and these thresholds change with each user. This requires a calibration routine to be performed for each user. Zhou attempts to assess user goals based on personality traits and game interaction style using the OCC cognitive theory of emotions[6]. Personality traits were identified in a written test given before the study. In order to evaluate the success of the real-time assessment, the students were given a survey after playing the game with questions about their goals. Zhou shows that it is possible to assess user emotion based on game interaction and non-physiological metrics such as personality traits. Our method attempts to improve on these works by assessing user emotion using only interaction and performance data obtained during game play.

III APPROACH

The work presented in this paper uses a scaled down version of the Fuzzy Logic Adaptive Model of Emotion (FLAME) developed by El-Nasr, et al.[7]. El-Nasr states that FLAME is a "computational model of emotions that can be incorporated into intelligent agents and other complex, interactive programs". FLAME allows an agent to learn about its environment via emotional attachment to objects and interactions with other agents and users. In this article we use a similar but less complex, non-adaptive fuzzy model of emotions to gain some insight into a human player's emotional state.

FLAME consists of three components that allow an agent to learn and make decisions based on emotions: a learning component, an emotional component, and a decision making component. We focus on the emotional component. It should be noted that unlike this work which adapts the emotional portion of FLAME to a commercial video game, El-Nasr constructed an environment in which an agent using FLAME could interact with and learn about every facet of its environment. FLAME's emotional component is based on an agents goals and events that impact those goals. Events occur as the agent interacts with the environment and the emotional component assesses those events' desirability with respect to the agents goals. In our case we assume the agent, a player in an FPS deathmatch game, has two goals:

1) Stay alive (prevent others from scoring)
2) Kill others (score as much as possible)

These goals are affected by a set of six game play statistics. Each statistic represents the number of times a particular event has occurred over a given period of time. This is necessary as our implementation does not have direct access to the game engine's event system. Each statistic is evaluated similarly to events in FLAME. Fuzzy logic is used to determine the magnitude of each statistic[1]. The set of statistics used is as follows: Kills, Kills Per Minute, Kills Last Minute, Deaths, Deaths Per Minute, and Deaths Last Minute. After the statistics are fuzzified they have a value that ranges from VeryLow to VeryHigh. These values are then passed to the fuzzy emotion evaluation system, which determines the player's emotional state.

The player's final emotion is based on two component emotions, Fear and Joy. Fear and Joy are evaluated by a set of fuzzy rules to determine which emotional state the player is in. The six emotional states are: Gloating, Excited, Complacent, Anxious, Angry, and Frustrated. The emotion Fear is affected positively by statistics involving Deaths. The emotion Joy is positively affected by statistics involving Kills. Just as FLAME uses a decay rate for emotions, the statistics in our system decay over time. Our statistics are based on kills or deaths per unit of time, thus the emotional system naturally decays if the player's performance worsens.

Once the emotional state of the player has been determined it is recorded in a log for analysis, we also display it on screen for testing purposes (see Figure 1). While this information could be used to change game rules or bot behavior, that is outside the scope of this paper. Section VI discusses future work in which a full implementation of FLAME may be used.

IV IMPLEMENTATION

Our system builds on the emotional component of El-Nasr's FLAME. The emotional model was implemented and tested in UnrealTournament 2004[8]. The system was implemented as a game mutator so that it could be used in any game type[2]. Development and testing took place using deathmatch type games.

The fuzzy emotional system is based on the set of statistics listed in Section III. In addition to those listed, Average Health and Most Kills Before Death were also considered as potential mood altering statistics. While these two extra data items could potentially increase the

[1]The magnitude of a statistic is relative to some game parameter.

[2]An UnrealTournament game type defines the set of rules for the game. Default game types include deathmatch, capture-the-flag, bombing run, etc.

Fig. 1. Screen shot showing the emotional state of the player.

accuracy of the fuzzy system, the additional complexity of the rule set is undesirable. As we will show in Section V the system performs adequately without them.

Each of the statistics are fuzzified using a similar function. The result of the function is a fuzzy value representing the desirability of the statistic. The statistics are given one of five values: VeryHigh, High, Medium, Low, and VeryLow. A VeryHigh fuzzy value is most desirable for Kill statistics and least desirable for Death statistics. Conversely, VeryLow is most desirable for Deaths statistics and least desirable for Kills statistics.

Kills - Kills represents the total number of kills the player has, this is the score within the game. Kills is directly accessible to the mutator as an integer value indicating the players score. Kills is fuzzified based on either the goal score for the game or the game time limit if no goal score is defined. To fuzzify Kills, we first normalize the value with 0 being the minimum and the maximum based on goal score or time limit. The maximum for normalization is either 80% of the goal score, 3 times the time limit[3], or 15 kills. This value is the baseline for the emotion Joy and is used as such because it is the indicator of player performance over the entire game. Kills is combined with Kills Last Minute and Kills Per Minute in a fuzzy rule set to determine Joy.

Kills Last Minute - Kills Last Minute represents the number of kills made within the last 60 seconds of game play. It is a measure of instantaneous performance identifying periods of play where a player is performing extremely well. Kills Last Minute is not provided by the engine, it is derived by monitoring the number of kills each second and comparing that to the previous value to get the number of kills in the last second. These are saved in a 60 entry array and summed to get Kills Last Minute. Kills Last Minute has the most effect on Joy which causes the amount of Joy to quickly change to match the players current state. Kills Last Minute is fuzzified using either 20% of the goal score or 5 as the maximum.

Kills Per Minute - Kills Per Minute is the average number of kills per minute the player has made during the game. This serves as a short term memory of past kills. Kills Per Minute has a slower decay rate than Kills Last Minute. As the game progresses, the player must keep killing other players for Kills Per Minute to remain high. If the player is unable to maintain a certain kill rate, Kills Per Minute begins to decay. Conversely, if a player begins to kill more (has a high Kills Last Minute) Kills Per Minute will increase. Kills Per Minute affects Fear less than Kills Last Minute, but more than Kills. Kills Per Minute is fuzzified using either 20% of the goal score or 5 as the maximum.

Deaths - As Kills is to Joy, Deaths is to Fear. Deaths is the total number of time a player has died during a game. Deaths sets the baseline value for Fear and has the least effect over the course of a game. The maximum for normalization is 80% of the maximum number of lives a player can have if this value is present. If max lives is not set, 5 times the time limit is used allowing a player to die up to 5 times per minute before reaching VeryHigh Deaths. If neither a maximum lives or time limit is set, we set 25 deaths to be our maximum.

Deaths Last Minute - Deaths Last Minute is tracked exactly like Kills Last Minute. The number of Deaths are compared each second, and any difference is entered into a 60 entry array. The values in the array are summed giving the total number of player deaths occurring in the last 60 seconds. Fear is most influenced by Deaths Last Minute as it is a good indicator of brief periods of poor performance. If a maximum number of lives is specified in the game rules, we use 10% of this value as a VeryHigh Deaths Last Minute. Otherwise, we choose 5 deaths as a VeryHigh number of Deaths Last Minute.

Deaths Per Minute - Deaths Per Minute is the average number of times the player has dies over the course of a game. Deaths Per Minute has less of an effect on Fear than Deaths Last Minute, but more of an effect than Deaths. Deaths Per Minute is a good indicator of past periods of high Fear levels. If the player has a period of high Deaths Last Minute, Deaths Per Minute will increase. In a period of fewer deaths, Deaths Per Minute will decay. A Medium level of Deaths Per Minute

[3]The time limit is in minutes.

indicates that the player has at several points in the game had a high death rate, or has had a constant rate of deaths throughout the game. If a maximum number of lives is specified in the game rules, we use 10% of this value as a VeryHigh Deaths Per Minute. Otherwise, we choose 5 deaths as a VeryHigh number of Deaths Per Minute.

To fuzzify the game statistics it was necessary to implement a function within the UnrealTournament mutator that would normalize an integer value and return the fuzzy set to which the value corresponds. A function $fuzzify()$ was created for this purpose. $fuzzify()$ takes three arguments: *min*, *max*, and *value*. It normalizes *value* based on *min* and *max*, then uses the normalized value in a set of if…then statements and a maximization function to determine fuzzy set membership. Note that the numbers chosen for normalization limits are subjective. They were chosen empirically based on testing with several players. This is one parameter of the fuzzy system that could be adjusted based on player skill or game rules. Figure 2 shows the membership functions used in $fuzzify()$. These membership functions were chosen arbitrarily to provide some overlap between sets. Tuning the membership functions could provide a more accurate estimation of player emotion.

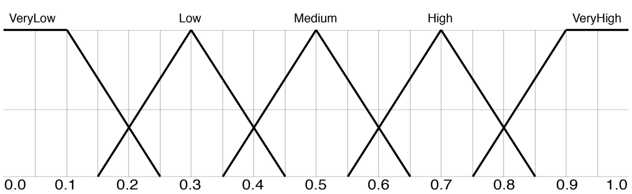

Fig. 2. Fuzzy Membership Functions for Event Fuzzification

V Results

This section will show that it is possible to design a fuzzy system to estimate a user's emotional state using in-game statistics. We present the results of a five minute deathmatch game played with our fuzzy emotion estimator active. The game was played with ten players, we show results from one of these players.

The player's statistics are shown using data from the game log file to present the transitions of Joy, Fear, and Emotion over the course of the game. This data is recorded for all players, however this player was chosen because they reach every emotional state possible. Figure 3 shows a graph of Joy, Figure 4 shows a graph of Fear, and Figure 5 shows a graph of Emotion. The transition points in the graphs have been numbered so that changes in Joy and Fear can be identified with corresponding changes in Emotion.

In Figure 5 we can see that the player begins in the Complacent state. At roughly 15 seconds into the game Transition 1. occurs; the player becomes Angry. Transition 1. is caused by an increase in Fear as seen in Figure 4. Transition 2. occurs around 30 seconds into the game. Fear transitions from VeryLow to Medium, most likely because they have just been killed. Joy and Fear are at equal levels and the player becomes Complacent. Figure 3 shows Transition 2. Transition 3. occurs at 45 seconds. The player becomes angry again because its Fear has reached VeryHigh. Transitions 4. and 5. demonstrate the natural decay of emotions over time. The player becomes Frustrated as Joy transitions from Medium to Low and then to VeryLow. This is because the player has not scored recently. In addition, Fear remains at VeryHigh indicating that the player is repeatedly dying. Transitions 6. and 7. show a short spike in emotion from Frustrated to Angry as the player's Joy increases to Medium because of one or more kills in a short period of time. This Joy quickly wears off and the player becomes Frustrated again. Transitions 8., 9., and 10. show a particularly high point in the game for the player. Joy quickly rises to VeryHigh because of many successive kills and Fear decays naturally. Transitions 11. through 18. show that while Joy is VeryHigh emotion is still dependent on Fear. When Fear spikes because the player dies, the player's overall emotion dips until the Fear of death decays.

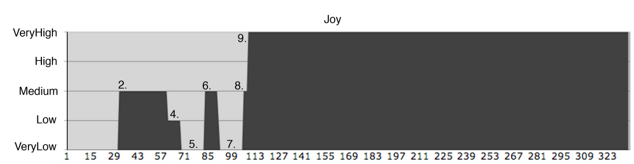

Fig. 3. Transitions of Joy during game play.

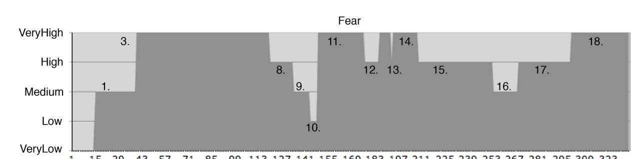

Fig. 4. Transitions of Fear during game play.

VI Conclusion

We have shown that it is possible to estimate player emotional state without monitoring physiological conditions. We assume that a user in a positive emotional state is satisfied with their gaming experience. As with

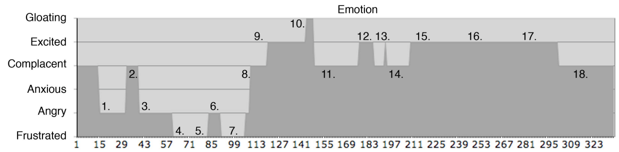

Fig. 5. Transitions of Emotion during game play.

all fuzzy logic applications, the fuzzy conversion is subjective based on the opinion of a subject expert. In this case the subject expert was an experience player of video games. Our experiment used a video game, UnrealTournament 2004, to test the fuzzy system proposed. Video games present the user with a limited number of goals, thus making it easier to define a fuzzy system to estimate the user's emotional state based on interaction and performance.

The system proposed uses a simplified version of the Fuzzy Logic Adaptive Model of Emotion (FLAME) demonstrated by El-Nasr. The current implementation of our system uses a non-adaptive fuzzy logic model of emotions. Due to the limited amount of data available from the game engine, our model consists of only 2 motivational states, Fear and Joy, which combine to determine one of five emotional states the game player is in. Those emotional states are Frustrated, Angry, Complacent, Excited, and Gloating. The player begins each game in the Complacent state, and then based on interaction with other players, reaches different levels of Fear and Joy, thus varying emotional states.

We believe a full implementation of FLAME within a video game or other application would eliminate some of the subjectiveness surrounding the definition of goals and motivations. The version of FLAME presented by El-Nasr is able to learn about the environment to determine which interactions benefit or hurt an agent as well as determine the goals of an agent by monitoring its actions. While this type of system could be implemented in any application, video games present the best case because of the limited number of goals and interaction types a user may have.

The benefits of a system such as the one presented in this article are clear. The literature shows that an application that can dynamically adapt to users goals and interaction style is desired. However, existing work suggests that physiological measurements are needed to determine a users emotional state. Our research contradicts this notion by estimating user emotion in software only.

Jeff Craighead Jeff Craighead is a Doctoral Candidate at the University of South Florida. He received a B.S.C.S. from the University of South Florida in 2004. Since that time he has worked with Dr. Robin Murphy's Center for Robot Assisted Search and Rescue and Institute for Safety Security Rescue Technologies. His research interests include robotic systems, human robot interaction, machine learning, and serious games. Current research involves the use of serious games for training robot operators to successfully conduct urban search and rescue operations.

References

[1] P. Rani, J. Sims, R. Brackin, and N. Sarkar, "Online stress detection using psychophysiological signals for implicit human-robot cooperation," *Robotica*, vol. 20, pp. 673–685, 2002.

[2] P. Rani, N. Sarkar, C. A. Smith, and L. D. Kirby, "Anxiety detecting robotic system – towards implicit human-robot collaboration," *Robotica*, vol. 22, pp. 85–95, 2004.

[3] R. A. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal of Robotics And Automation*, vol. RA-2, no. 1, March 1986.

[4] C. Conati, R. Chabbal, and H. Maclaren, "A study on using biometric sensors for detecting user emotions in educational games," in *Proceedings of the Workshop "Assessing and Adapting to User Attitude and Affects: Why, When and How? ". In conjunction with the 9th International Conference on User Modeling*, 2003.

[5] X. Zhou and C. Conati, "Inferring user goals from personality and behavior in a causal model of user affect," in *Proceedings of the International Conference on Intelligent User Interfaces*, 2003, pp. 211–218.

[6] A. Ortony, G. L. Clore, and A. Collins, *The Cognitive Structure of Emotions*. Cambridge University Press, 1988.

[7] M. S. El-Nasr, J. Yen, and T. R. Ioerger, "Flame-fuzzy logic adaptive model of emotions," *Autonomous Agents and Multi-Agent Systems*, vol. 3, no. 3, pp. 219–257, September 2000.

[8] "Unrealtournament 2004." [Online]. Available: http://www.unrealtournament.com/