Including Affect-Driven Adaptation to the Pac-Man Video Game

Ahbiya Harris
Arizona State University
University Drive and Mill Avenue
Tempe, AZ 85281, USA
ahharris@asu.edu

Andrew Hoch
Arizona State University
University Drive and Mill Avenue
Tempe, AZ 85281, USA
ahoch@asu.edu

Ryan Kral
Arizona State University
University Drive and Mill Avenue
Tempe, AZ 85281, USA
rdkral@asu.edu

Michael Teposte
Arizona State University
University Drive and Mill Avenue
Tempe, AZ 85281, USA
mteposte@asu.edu

Alyza Villa
Arizona State University
University Drive and Mill Avenue
Tempe, AZ 85281, USA
agvilla3@asu.edu

Maria Elena Chavez-Echeagaray
Arizona State University
University Drive and Mill Avenue
Tempe, AZ 85281, USA
helenchavez@asu.edu

Javier Gonzalez-Sanchez
Arizona State University
University Drive and Mill Avenue
Tempe, AZ 85281, USA
javiergs@asu.edu

Robert K. Atkinson
Arizona State University
University Drive and Mill Avenue
Tempe, AZ 85281, USA
Robert.atkinson@asu.edu

ISWC'14 Adjunct, September 13 – 17, 2014, Seattle, WA, USA
ACM 978-1-4503-3048-0/14/09.
http://dx.doi.org/10.1145/2641248.2641360

Abstract
Building affect-driven adaptive environments is a task geared toward creating environments able to change based on the affective state of a target user. In our project, the environment is the well-known game, Pac-Man. To provide affect-driven adaptive capabilities, diverse sensors were utilized to gather a user's physiological data and an emotion recognition framework was used to fuse the sensed data and infer affective states. The game changes driven by those affective states aim to improve the user experience by keeping or increasing player's engagement.

Author Keywords
Affective states, affect recognition, affect-driven adaptation, video games

ACM Classification Keywords
H.5.2 [Information interfaces and presentation]: User Interfaces --- interaction styles, input devices and strategies.

Introduction
Building affect-driven adaptive environments comprises reading a user's physiological information through sensors, inferring the user's affect, and then using this information to create a feedback loop: the software change driven by the affect and its changes aims to alter the user's affect. Our project is geared toward

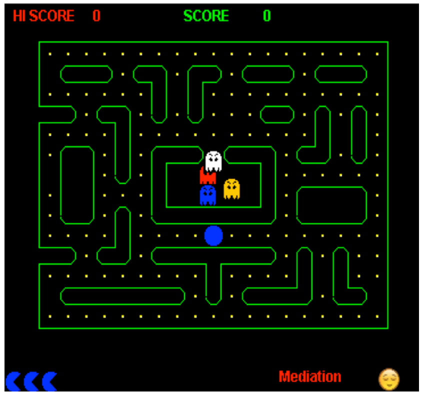

Figure 1. When the player is meditating, the speed of ghosts and the music tempo are increased; the special features (e.g., power pellets) are disabled; and when the game ends, a slightly more difficult map is loaded.

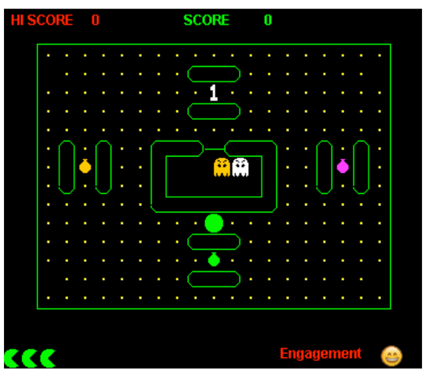

Figure 2. While the player is engaged, the settings of the game stay the same.

altering the well-known game, Pac-Man [1], in order to elicit affective responses from users. A user is able to play Pac-Man while wearing various sensors that communicate with a server application, which then sends information to the Pac-Man client application. The server application synchronizes and fuses the data collected by the sensors and infers the affective state [2]. The affective state is represented as a pleasure, arousal, and dominance (PAD) vector. The Pac-Man client utilizes the PAD vector at regular intervals to alter the game and ultimately attempts to push the user into an engaged state.

Technology Background

The technologies, both hardware and software, utilized for this project are described in [3] and summarized in the follow paragraphs.

Posture sensor

The project utilizes a chair posture sensor. This device produces raw data values based on how the user is positioned on a chair. Values are related with the user's interest level: if the user is engaged, he/she would be more likely to be sitting forward rather than relaxed and leaning back against the chair; for an engaged user, the values in the sensor would measure low or lack of pressure on the back of the chair and high pressure on the front of the seat.

Pressure sensor

This device is based on the mouse pressure sensor described in [3]; however, since Pac-Man uses the directional keys on the keyboard, the mouse pressure sensor was modified to put the sensor pads of the mouse into the fingertips of a glove so that the pressure sensitivity the user has on the keys can be measured. A frustrated user would press on the directional keys in an aggressive manner rather than in a gentle or calm manner.

EEG sensor

Electroencephalographically (EEG) sensors use brainwaves as an information source; they measure the electrical activity along the scalp, produced by the firing of neurons within the brain over a period of time, which is gathered from multiple electrodes placed on the scalp. They are able to infer diverse affective constructs such as engagement, excitement, boredom, meditation, and frustration.

The affect recognition module

The server application is an affect recognition framework that we use off-the-shelf [2]. It is a server application that receives sensor data, synchronizes and fuses the data, determines an affective state, expresses the affective state as a PAD vector, and communicates it with other applications, named client applications.

We feed the server application with the raw values that posture, pressure, and EEG sensors gather. Then, our client application (the game) uses the PAD vector, provided by the server, as an input and adjusts itself according to the values on the PAD vector.

The Game

The game is a highly modified version of an open source version of the video game Pac-Man [1]. We chose this game due to its simplicity, which allows a considerable amount of modifications. The goal of the game of Pac-Man is to eat all the small pellets in a certain maze to win the game by avoiding being eaten by the ghosts. In our game, affective components,

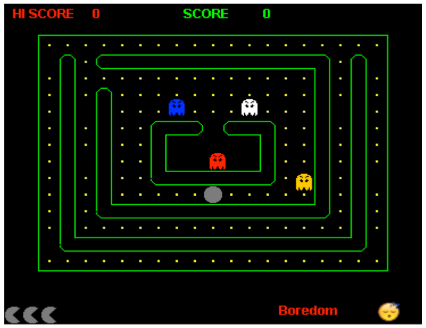

Figure 3. When the player is bored, the speed of ghosts and the music tempo are increased; the special features (e.g., power pellets) are disabled; and when the game ends, a very difficult map is loaded.

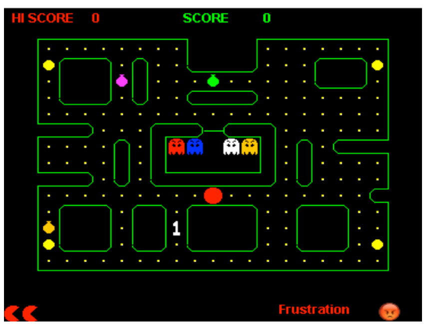

Figure 4. When the player is frustrated, the speed of ghosts and the music tempo are decreased; the special features (e.g., power pellets) are enabled; and when the game ends, an easier map will be loaded.

expressed in PAD vectors, are taken as inputs to allow the game to change as discussed further below.

Affective state changes

Four affective states were focused on for this game: meditation, engagement, boredom, and frustration. As the goal of the affect-driven adaptive version of the game is to keep or bring the player into the engaged state, the settings are therefore maintained once the player reaches this state. Table 1 shows the changes that the game will undergo when the user reaches each of the affective states. The features that are adapted accordingly with the affective state include: the color of Pac-Man (blue, green, gray, and red); the speed of Pac-Man, which can increase (+) or decrease (-); the number of ghosts, which can increase (+) or decrease (-); the speed of the ghost, which can increase (+) or decrease (-); the music tempo, which can be faster or upbeat (+) or slower or ballad (-); the special features (such as fruits, power pellets, and 1-up component), which can be enabled (E) or disabled (DE); and the difficulty level of the next maze, which can increase (+) or decrease (-).

Implementation

The project was developed in Java using Eclipse as IDE. The development involved the modification of an open source version of the Pac-Man video game and its connection to the emotion recognition framework, using a client/server approach. Documentation, user-guides, and a configuration file were created. Documentation and user-guides are available for both developers and non-developers to easily maneuver through the code and make changes to the game. The configuration file allows researchers to set up diverse starting variables in the game, such as: the base speed of Pac-Man and the ghosts, the poll time of readings (in seconds), and the affective states to be considered to adjust the game.

Once the user has begun the game, the PAD vector readings are polled at the frequency defined by the researcher in the configuration file and used to determine the appropriate adjustments. The software stores all the information about the affective state and status of all the game features in the log file. The information in the log file is used to determine what happens as the user plays the game and serves as key data for researchers to answer empirical questions about personalizing game environments based on affect.

Feature \State	M	E	B	F
Color	Blue	Green	Gray	Red
Pac-Man speed	=	=	-	+
Ghosts #	=	=	+	-
Ghosts speed	+	=	+	-
Music tempo	+	=	+	-
Fruits	DE	=	DE	E
Power pellets	DE	=	DE	E
1-up	DE	=	DE	E
Next Level Diff	+	=	+	-

Table 1. Changes made in the game when the user reaches each of the affective states: Meditation (M), Engagement (E), Boredom (B) and Frustration (F). Features can increase (+), decrease (-), be enabled (E), be disabled (DE), or stay in the same status (=).

Discussion

An affect-driven adaptive environment was successfully developed (using an open source version of Pac-Man) to work with the sensor suite developed in a prior project of the sponsoring research laboratory. The original open source Pac-Man game allowed us to add, in a short period of time, extra features, which allowed the game to adapt based on the user's affect.

When designing the game adaptation strategy, it was necessary to figure out the affective states on which to focus. After a first brief analysis, we ascertained that changing the speed of Pac Man and the ghost entities were not enough to manipulate the user's affective state, so additional features needed to be manipulated. Additional testing led to the manipulation of the music, colors, and special elements (such as fruits and 1-up).

The strategy for keeping the user engaged by manipulating the described features is defined as follows: (a) we did not want the player to be bored, thus we made the game extremely difficult when the player reached that state by decreasing the speed of Pac-Man and increasing the number and speed of the ghosts and the music tempo; (b) we did not want the user to be frustrated, thus we made the game very easy in this state by increasing the speed of Pac-Man and decreasing the number and speed of the ghosts and the music tempo as well as providing special features such as power pellets and fruits; (c) finally, we did not want the player to be in meditation, thus we made the game slightly more difficult to give the player a push toward the engaged state by increasing the speed of Pac-Man and the music tempo.

Enabling the Pac-Man game with affective-driven adaptive settings resulted in a game that became easier to win, which was more enjoyable to users. Additional research and testing should be done to further hone the types of adaptations that guide users to an optimal state and to explore and test the impacts on other user outcomes including learning and performance management.

Acknowledgments

The project was developed as coursework of CSE423/424 (Systems Capstone) at ASU collaborating with the Advancing New Generation Learning Environments Lab and supported by Office of Naval Research under Grant N000141310438.

References

[1] Open source Pac-Man software. http://sourceforge.net/projects/javaipacman/

[2] Gonzalez-Sanchez, J., Chavez-Echeagaray, M. E., Atkinson, R., and Burleson, W. ABE: an Agent-Based Software Architecture for a Multimodal Emotion Recognition Framework. In *Proc. Software Architecture (WICSA), 2011 Ninth Working IEEE/IFIP Conference on*, IEEE (2011), 187–193.

[3] Gonzalez-Sanchez, J., Christopherson, R. M., Chavez-Echeagaray, M. E., Gibson, D. C., Atkinson, R., and Burleson, W. How to Do Multimodal Detection of Affective States? In *Proc. Advanced Learning Technologies (ICALT), 2011 11th IEEE International Conference on*, IEEE (2011), 654–655.