Analytical Framework for Facial Expression on Game Experience Test

SEUNGJIN KWON¹, JAEHYUN AHN¹, HYUKGEUN CHOI¹, JIHO JEON¹, DOYOUNG KIM¹, HOYEON KIM¹, AND SHINJIN KANG²

¹Intelligence Laboratories, Nexon Korea, Seongnam 13487, Republic of Korea
²School of Games, Hongik University, Sejong Campus, Sejong 30016, Republic of Korea

Corresponding author: Shinjin Kang (directx@hongik.ac.kr)

This work was supported by the Culture, Sports and Tourism Research and Development Program through the Korea Creative Content Agency funded by the Ministry of Culture, Sports and Tourism through the Project of Development of Artificial Intelligence-Based Game Simulation Technology to Support Online Game Content Production, in 2022, under Project R2022020070.

ABSTRACT Game experience testing, an essential process for developing and servicing high-quality games, aims to evaluate and improve the play experience from the user perspective. Previous game experience tests are usually held offline with recruited test participants. Participants play the game in a controlled environment, and their experiences are recorded and analyzed by domain experts. However, this process not only requires substantial resources in terms of time and cost but also misses momentary data that can infer the user’s experience directly and quantitatively, such as facial expressions or body gestures of the participants while playing the game. In this paper, we propose a framework that can automatically collect and analyze video data of participants in a remote environment. We designed various experiments to verify that the proposed framework can automatically collect and analyze facial data in a remote environment. The experimental results show that our framework can be applied in gaming experience tests for popular commercial games currently in service. This paper presents a pioneering work that applies the face analysis method to the commercial game development process, and its purpose is beyond that of a pure research work.

INDEX TERMS Game user research, game experience test framework, face detection, facial expression recognition, head pose recognition.

I. INTRODUCTION

The modern game development process is very complex, and it takes a substantial amount of time and cost to develop games. Hence, game development companies perform verified tests based on data and only release games with a high level of completeness to the market [1]. User research, which is one of these verification tests, systematically studies the needs and pain points of the target users. In user research, the game experience test is one of the essential tests for game development. It is conducted to evaluate the game from the user’s perspective and improve the game playing experience [2].

In the process of game development, the role of game experience testing is to enable game designers to evaluate whether the game is designed as intended and to improve the game design. For evaluation and description of the user experience during game play, data based on player–game interactions are used [3]. One of the data to be monitored for evaluation is the expressions exhibited by test participants (tester) during game play.

If it is possible to identify whether testers make positive or negative expressions while playing the game through analysis of expressions, this could be an important feedback in the game development process. The main purpose of this analysis is to provide the game development team with information about how testers perceived the intended design [4]. Based on this information, the game development team would be able to provide a better game experience by repeating the feedback loop, for example, by further polishing the positive feedback section or identifying and removing potential negative factors. Various attempts have been made to find out the response of users while playing a game.
The most common approach is to collect questionnaires and interviews after the tester have played the game. This approach is useful for collecting and judging the tester's emotional response directly; however, it is performed after the testers have finished playing the game. As a result, these methods may not completely reflect the user experiences while playing the game and data distortion may occur while quantifying data [5]. Another method is for the researcher to observe the tester and record the observation as data [6]. This method has the advantage of documenting a vivid record of the playing experience of testers. However, there is variation among different researchers, which can lead to a bias in the observation results. In addition, there is a drawback in that it requires a large amount of staff to work with a large number of testers [7]. Finally, there is a method that records changes in the biosignals automatically by attaching sensors to the testers [4], [8], [9], [10], [11], [12]. This type of biosignal measurement technique collects data automatically; hence, it can collect large-scale data. However, it requires expensive equipment, and testers may feel upright or tense because of the sensor. In addition, if the sensor malfunctions, it could cause a collection error, and the collected data may include a large amount of noise [1].

However, considering the recent advancement in the game development process, it is necessary to introduce more scientific and quantitative analytics based on standardized and large-scale data. For this reason, an automated process is required in the data collection and analysis for swift feedback and interaction with users. We designed a framework that automatically captures, collects, and analyzes real-time video data of testers.

During the game experience test, the analyzer performs automatic collection and analysis of facial expressions while the testers play the game, thus providing a support to the research of user experience (ux). Game play video data, face-cam video data, and the results of face-cam analytics are sent to the ux researcher, and the analytics results include face location, facial expression, and engagement. Timestamp-synchronized game data and the analysis of facial expression data allow identification of positive and negative factors for the quality of game play. This framework is based on real-time analysis. There is no data distortion of the questionnaire because the proposed framework collects and analyzes the changes in users’ facial expressions in real time for every frame. In addition, as this framework is quantitative and automated, large-scale data collection is possible; and if the users are equipped with a physically simple camera and sufficient network specifications, the framework allows the creation of a remote test environment in any place. We detail the network specifications in Section III-B. We designed various experiments to verify that our framework is suitable for the “KartRider” game. In the first experiment, video data were collected from testers. Testers were recruited for a game in real service, and the video data of them playing the game were recorded. The recorded data were directly examined by a game ux expert, the facial expressions were classified by sections, and these data were used as ground truth. In the second experiment, based on the collected data, three classification problems were designed, and a test was performed by the game testers using our framework. In the process, facial analytics was performed with the streamed data and the performance was measured. The following factors were tested: the face detection module accurately identifies the location of the face in the captured image, the head pose estimation module makes an inference from the angle of the head to produce an accurate classification result on the status of tester’s engagement, and the facial expression recognition module performs accurate classification of the present facial expression out of seven emotions. Finally, the proposed framework was tested to examine and verify if it showed a sufficient level of performance and accuracy when applied to a real test environment. To set the expressions exhibited by the users suitable for actual game testing, the data was processed using the sliding window method, and the valence mapping technique was applied to verify whether the framework showed good performance in facial expression recognition. Considering a real game test environment, we evaluated how well the framework actually performed classification of emotions and whether it showed sufficient level of performance for remote operation. The remainder of this paper is summarized as follows. Section II presents related works. In the following Section III, we introduce the framework architecture. Section IV describes the experiments conducted to verify the applicability of the framework. Finally, Section V concludes the paper.

II. RELATED WORK

A. EXPERIENCE TEST

An experience test is the most widely used method of collecting user feedback in the field of game development. Dumas et al. and Medlock introduced the most common experience test techniques [2], [13].

B. QUESTIONNAIRE ON GAME EXPERIENCE TEST

In this method, testers fill out a questionnaire in the middle and at the end of the game experience test. Testers can evaluate their impression of the game and provide their thoughts on the test through this process. Depending on the questions, the research of the questionnaire can be classified as either quantitative or qualitative. However, because the questionnaire is filled out after playing the game, it may be incomplete, and the user may not remember the experience completely [5].

C. OBSERVING ON GAME EXPERIENCE TEST

In this technique, the testers are observed in real time while the test is in progress. In addition, the game screen and the user’s face are recorded and stored in a recording medium after obtaining the tester’s consent, so they can be used for analysis after the test has been completed. Tasks that are performed include analyzing which section the testers mostly responded to and whether the evaluation was positive or
negative, as well as recording whether testers are concentrating on the game [14]. This method has the advantage of observing the game test precisely, but the results could be biased depending on the researcher. A Kappa test is generally used to solve this problem. However, it requires a sufficient number of researchers to observe the testers for the total test duration; hence, it is difficult to perform this test on a large scale. According to Mandryk and Atkins et al. [15], the appropriate ratio of observation analysis time to data sequence time ranges from 5:1 to 100:1.

D. PHYSIOLOGICAL SIGNAL DETECTION ON GAME EXPERIENCE TEST

In this method, physiological sensors are attached to the testers and are observed during the game test. Biosignals such as electrocardiograms (ECGs), galvanic skin response (GSR) [9], electrodermal activity (EDA) [4], and electromyography (EMG) [10] are mainly used. Sekhavat et al. introduced an experiment that tracks the game testers’ interests using Eye Tracker [11]. Jai et al. introduced an experiment to analyze comparatively the user’s heart rate (HR) and rating of perceived exertion (RPE) when playing the game using the PlayStation move controller both while sitting and standing [12]. Besides these, various techniques for measuring biosignals have been proposed in the field of game development [16]. Collecting bodily data in this way has the advantage of acquiring quantitative emotional state data in testers measured by a machine. This method also has the advantage of collecting data automatically. However, expensive equipment is required for each test session; hence, it is difficult to implement this method on a large scale. Moreover, testers may feel upright or tense because devices are attached to them, and there could be a collection error owing to interference from other biosignals.

E. FACE DETECTION

When analyzing a tester’s face, the first task is to detect the face on the screen. Recently, CNN-based methods have received increasing attention. Many recent studies that pursue high accuracy in face detection while achieving real-time speed have made advancements in this field. Deng et al. proposed a RetinaFace model that can be processed by the CPU in real-time speed [17], and He et al. proposed a light and fast face detector (LFFD) model for edge devices [18].

F. HEAD POSE ESTIMATION

We applied the head pose estimation technique to improve the facial expression recognition accuracy. Head pose estimation (HPE) estimates the orientation of the head in an image or video. Recently, Zhou and Gregson et al. tried to overcome this problem by proposing WHENet, which estimates the HPE from all views, not just the frontal view, using a dataset captured with 3D Capture [19].

G. FACE EXPRESSION RECOGNITION

In this study, we attempted to detect the user’s facial expressions in real time and reflect the analysis result in the game development. Facial expression recognition (FER) studies have been actively conducted over the past several decades. Ekman and Friesen et al. proposed a method for using facial muscle states to associate facial expression with six basic emotions: happiness, sadness, anger, disgust, surprise, and fear [20]. Recently, deep learning technology has been used, such as that in Savchenko et al. [21].

H. WebRTC

This study aims to develop a web-based framework for exchanging key information with commercial games in real time. Web real-time communication (WebRTC) is a technology standard [22] that enables real-time communication between web browsers and is provided as open source [23]. It enables video and audio communication within a web page without the need to install additional plug-ins or apps by establishing a peer-to-peer connection between browsers. Moreover, WebRTC is widely used in video telephony. Jansen et al. demonstrated that video conferencing could be carried out smoothly between continents without disconnections [24]. Such connection stability shows that the web platform using this technology can be applied in various gaming environments.

III. FRAMEWORK

In this section, we introduce a framework system that supports game experience testing in a remote environment, automatically collects and analyzes video data, and visualizes the analytics. We designed the framework to achieve two main goals. First, we focus on providing a comfortable gaming experience test for both the tester and observer in a remote environment. The framework provides testers with an environment where they can join a test with minimal settings and focus on gameplay. The system should allow the observer to track the gameplay and face-cam of the tester in real time. Second, it automates the data collection and analysis process for accurate and contextual analytics. Through our framework, it is possible to collect and analyze the required data automatically and through the collected data, to quantify and visualize the change in engagement and the response of facial expressions, and transmit them to the test observer in real time. Consequentially, it is possible to analyze the facial expressions of a tester accurately in each situation, and the observer can lead the test more proactively based on the analytics provided in real time. Furthermore, by enabling such analysis, it is possible to collect large-scale data from a large number of testers for the same content and derive results of a quantitative analysis. In addition, as our framework can be operated independently of the data collection and analysis methods used in the existing test, combining the quantified results of our framework with the previous qualitative results allows much deeper understanding of the user experience.

A. FRAMEWORK ARCHITECTURE

An overview of the framework is shown in Figure 1. First, the service server creates a new session for the game experience
As depicted in Figure 2, which helps to understand the status database but also transmitted to the observer and visualized, models. The analytics obtained are not only stored in the database. For estimating the engagement and facial expression of the tester using three deep learning face-cam of the tester and sends it to the analytic server, formed at regular intervals. The tester’s client captures the face-cam video. The analysis proceeds in three steps, as shown in Figure 3. First, the face location is found in a captured image of the face-cam video. The analysis result is sent to the observer. Through this process, the observer can view the results of the real-time analysis of the tester’s face on the screen.

After the session creation is completed and the tester is ready to play the game, the test begins and the game play video, face-cam video, and audio of the tester are transmitted and stored in the database. For estimating the engagement and facial expression, the following tasks are repeatedly performed at regular intervals. The tester’s client captures the face-cam of the tester and sends it to the analytic server, where this captured image is used to estimate the engagement and facial expression of the tester using three deep learning models. The analytics obtained are not only stored in the database but also transmitted to the observer and visualized, as depicted in Figure 2, which helps to understand the status of the tester in real time.

**B. REMOTE TEST ENVIRONMENT**

Most of the previous game experience tests are conducted offline, and thus there are many limitations in terms of time, place, and cost [25]. To overcome these shortcomings, there is a tendency to collect as many and diverse data as possible by recruiting many testers in a single test. However, as in our case, if the game experience testing is conducted remotely, it becomes less affected by these constraints. Many small unit tests can be conducted at all times, and it becomes possible to diversify and subdivide the tests. In addition, the tester can participate in the test in a comfortable environment, free from an unfamiliar and burdensome situation watched by the observer. This can elicit natural reactions of the tester, and based on this, more accurate analysis is possible. Moreover, it is possible to recruit more testers for the same cost than with the existing test method while relaxing the constraint of a test location. For these reasons, we used WebRTC to design the framework as a real-time data transmission environment and created a remote test environment based on this design. Many researchers have attempted to implement remote testing of the usability of applications. Andreasen *et al.* introduced an experiment in which the results of the face-to-face usability test and the remote usability test were almost identical for the application test [26]. Nevertheless, there are disadvantages that occur while testing in a remote environment. First, some physical equipment is required for the testers; a camera is needed because the analysis is performed using the images captured from the face-cam video. Furthermore, the tester requires a PC with high-end hardware for testing modern games such as the latest 3D FPS. Given that UX tests are conducted by recruiting participants, specifications of the test computer and network bandwidth must be prepared. Second, because real-time transmission of face-cam video is required, the internet bandwidths must also be considered. Chen *et al.* [27] reported that classic online games require between 7 and 40 kbps of bandwidth. As for the state-of-the-art cloud game platforms with heavy network traffic, experimental results showed that the required bandwidth was approximately 27–28 Mbps. The bandwidth of the 720 p 30 fps WebRTC used by our framework is of 1.0–2.0 Mbps. According to a report by the web service speedtest.net [28], the worldwide average internet speed is over 30 Mbps, and thus our test framework is a universal framework that can be applied for testing both classic online games and the latest cloud games in most of the tester’s clients. Furthermore, we checked the network bandwidths in advance and started the test only when there was no problem with the bandwidths. More technological advances are needed to research the interactions among testers, such as those in group testing.

**C. ANALYTICS SERVER**

The analytics server measures the engagement and facial expression of a tester based on the captured image of the face-cam video. The analysis proceeds in three steps, as shown in Figure 3. First, the face location is found in a captured image using the face detector. As our analysis is based on the face, we find the location of the face in the image, crop it, and use only this part for the subsequent analysis. Second, the angle of the face of the tester is measured using head pose estimation. Based on this, the observer can detect whether the tester is paying attention to the monitor. Third, only for images in which the tester is gazing at the monitor, the expression of the face is estimated through the facial expression recognition interface.
FIGURE 2. Display of the observer system. 1) session hash id, 2) face-cam, 3) game play screen, 4) face detected and engagement estimated based on head pose estimation, 5) timeline, 6) estimated facial expression probabilities, and 7) control panel.

FIGURE 3. Flowchart of face analysis. The server receives the face-cam image and sends it to the face analysis module. In the analysis module, the face detector identifies the face and crops it. Then, the head pose estimator determines whether the face is looking straight ahead. If the tester’s face is looking straight ahead, the cropped facial image is sent to the facial expression analyzer, which generates the analysis results. The analyzed data is output in the form of detected face coordinates, estimated head pose, and estimated facial expression probabilities.

1) FACE DETECTOR
To analyze facial expressions and engagement from the received face image, it is necessary to crop only the face region through face detection. Deep learning is the state-of-the-art technique in the detection field and pre-trained models have shown excellent detection performance in many fields. In our test environment, the camera installed on the monitor was used to capture the face cam of testers located indoors. Thus, we effectively controlled the test environment by adjusting the location of testers to sit at a distance from the camera installed on their monitor and the camera’s ability to accurately detect their faces in face images. This result indicates that the detection speed for real-time analysis is considered a more important factor than robustness. The output of the face detector system are the estimated location of the face (up, left, down, right) and the accuracy. Our framework used the architecture of the Ultra Light model [29] and pre-trained models were used. The architecture of the model is as follows.

2) HEAD POSE ESTIMATOR INTERFACE
In head pose estimation, a two-dimensional image is taken as an input and the angles of yaw, pitch, and roll for the head are estimated. It is determined whether the tester is looking at the screen. If the yaw estimate of the head pose of the tester is outside a certain range, it is considered that the tester is not looking at the screen. The tester may look away from the screen owing to a specific event experienced during the gameplay. If head pose estimation determines that the tester is not staring at the screen, facial expression recognition is not performed. Furthermore, the estimated head pose value itself can be an auxiliary indicator for changes in the action...
FIGURE 4. Model architecture proposed by [29]. Conv3 refers to a convolution layer with a 3 x 3 filter. DWConv3 refers to a depth-wise convolution layer with a 3 x 3 filter. SConv3 refers to a separable layer with a 3 x 3 filter. This architecture uses a lightweight backbone with sophisticated placement of receptive fields for fast and high accuracy [30].

FIGURE 5. Head pose estimation estimator interface.

of tester. In addition, this layer is designed as an interface, making it easy to change the model. Figure 5 shows the architecture of the head pose estimation interface in our framework.

We select the WHENet architecture proposed by Zhou et al. [19]. This is an end-to-end model and is the only model that infers the head pose estimation even when the face is not detected on the screen (e.g., back of the head). In addition, this head pose estimation model achieves a real-time processing speed. Figure 6 shows the architecture, and the output of the head pose estimation model is yaw, pitch, and roll. This information is stored in the database, and it is displayed on the screen, as depicted in part 2) of Figure 2, when the yaw value exceeds the set threshold value.

3) FACIAL EXPRESSION ANALYZER INTERFACE

The facial expression analyzer interface is designed to estimate the facial expression based on the facial image of the testers playing the game. With the recent advances in deep learning, high accuracy has been achieved in facial expression recognition. Researchers have recently achieved high accuracies and performances in facial expression recognition [21]. Thus, we used a deep learning-based analysis technique to obtain a high accuracy. The interface architecture is shown in Figure 7.

We adopt the model architecture proposed by Savchenko et al. [21]. This model increases the accuracy by fine-tuning the face detection phase during training and by training with facial images without the margins. The reasons for selection of this model include considerations of speed and model performance and the facts that a pre-trained model is provided and the model is easy to handle. This model has suitable characteristics for real-time processing because it uses EfficientNet-B0 as the backbone model. According to the author of the model, it showed a state-of-the-art performance with a score of 65.74 for 7 classes in AffectNet [31]. Moreover, concerning the proposed framework, it is shown in IV. Evaluation that the model produced good results. We used the pre-trained model. The model architecture is shown in Figure 8. The facial expression recognition module is abstracted at the implementation level. Here, the records of analysis are shown in part 6) of Figure 2, and if the largest probability is not the neutral probability, the case is considered as generation of an expression, and the corresponding expression is indicated in part 5). Although it currently provides sufficiently meaningful results, studies are ongoing in the facial expression-related field, and there have been rapid improvements in accuracy. Hence, the interface is designed for this layer so that models can be easily replaced. Currently, this model operates once every 0.3s in a real-time observation environment considering speed and accuracy.

D. RECORDED ANALYSIS REPLAY

Figure 9 shows a view of the results after completion of the analysis. The game data and the analyzed data with synchronized timestamp allow identification of positive and
negative factors for the quality of gameplay, and a replay function for viewing of the analysis results at a glance is also provided. The observer can quantitatively acquire the responses of the tester to the game test by looking at these screens, and can replay the analysis process until meaningful insights are gained. The usefulness of such data collection and visualization in the framework is an essential element when introducing effective computing to work tasks in the industry.

IV. EVALUATION

When the face of testers is observed in real-time using the framework, it automatically detects and records the facial expression and head pose while the tester is testing the game. An experiment was conducted to check whether this framework works properly. The analysis interfaces used in this framework are interchangeable, and studies have already been conducted on these models. Therefore, we focused on determining the suitability of the model performance for the video captured in the actual game test. Finally, we focused on verifying whether data for facial expression recognition and head pose estimation are properly collected from the video of the game tester. For face detection, our experiment focused on determining whether the presence of a face is well detected rather than whether the area of the face is accurately identified. In our test set, the images are labeled true if the eyes, nose, and mouth elements of the face are clearly displayed on the screen; if not, the images are labeled false. Hence, we treated this experiment as binary classification. In addition, some studies have produced high-accuracy results using the WIDER Face dataset [32]. Furthermore, because this study treats face detection as binary classification, very high accuracy was expected. For head pose estimation, it has been reported that AFLW2000 [33] achieved an MAE of 4.83. Because this study treated head pose estimation as binary classification, very high accuracy was expected.

A. EXPERIMENT SETTING

As a test game, we selected KartRider, developed and serviced by Nexon Co., Ltd. KartRider is a casual racing game and one of the classic online games, which has been in service for over 15 years. We had one tester play KartRider on a test PC in a separate, independent space. A webcam was installed in the test PC and focused on the face of the tester. An observation PC was placed in a separate observation room, and the test PC was connected to the observation PC in a peer-to-peer manner through our framework. At the same time, the test PC sent the face screen of the tester to a separate analysis server three times every second. Actual face analysis was performed in the analysis server, an i7-10700 (CPU only). The test was run with a playing time of approximately 3 min per person, and 58 testers participated in the experiment. Because the test PC sent face screen three times every second, a total of 27,384 facial images with a resolution of 640 × 480 were analyzed. The ground truth dataset had to be constructed first in order to evaluate the performance of our framework. A game user experience expert observed 58 videos and manually labeled expressions by observing the face images of testers. We classified the frames of face image sequence into three categories and labeled them. First, we manually labeled frames that did not have faces on the screen. Second, we manually labeled frames in which the faces were not directed toward the screen. Third, we labeled the facial expressions on the faces of testers. The method for labeling facial expressions is as follows. If a change in expression is detected while observing the video, the changed expression is marked with one of the labels, namely, happiness, surprise, sadness, anger, disgust, and fear, at the starting point where the change was detected, and the ending point is marked. Because there is a physical limitation to marking every frame of 30 fps video data, marking is done with reference to sections. We created a ground truth label filled with the corresponding facial expressions at the starting and ending points through post-processing. Because our experiment is performed on the video, and manual labeling is also carried out on the video, there were some errors in units of 0.33 s duration. For example, when a tester shows a smile after not showing any expression, the point at which the tester begins to smile may be different between the ground truth determined by a person and the prediction made by the model, and the difference is in units of 0.33 s.

B. EXPERIMENTAL RESULT

We evaluated the proposed framework based on a total of 27,384 inference results. If face detection was not performed, facial expression recognition and head pose estimation were not performed.

1) FACE DETECTION

Face detection should find the area occupied by a face on a given 640 × 480 screen. However, in our experiment, the focus was on whether the presence of the face was well detected rather than whether the area of the face was accurately identified. Our test set was treated as binary classification, where the image was labeled as true if the eyes, nose, and mouth elements of a face were clearly displayed on the screen, and if not, the image was labeled as false. We also artificially controlled the test environment such that only one person appeared on the camera. Furthermore, during the experiment, we set up an environment where testers played the game under bright lighting in a separate, independent space. The face detection performance was tested in bright light, and the testers played the game and looked at the screen in most cases. As a result, it was an unbalanced classification problem where most of the labels were true except when the testers intentionally moved their faces away from the camera. The Ultra Light [29] model was adopted for face detection. The test results are presented in Table 1.

The accuracy was 99%, and the f1 score was 0.98. Because most of the values were true positive (TP), the inference results for true negative (TN), false negative (FN), and false positive (FP) cases were analyzed. For such cases when only the crown of the head was visible because the tester had
Figure 9. Screen replaying the facecam, gameplay, and facial analysis results. The x-axis shows the variation of the observed expressions flow of one tester over time.

Table 1. Confusion matrix of the face detection experimental results.

| Actual   | Predicted | Positive | Negative |
|----------|-----------|----------|----------|
| Positive | 99.98%    | 0.02%    |          |
| Negative | 2.44%     | 97.56%   |          |

2) Head Pose Estimation
In the case of head pose estimation, our experiment focused on identifying whether the head pose was facing the screen when the face moved by more than a certain angle. Face pose was inferred by inputting 27,313 cropped facial images, which were the result of the face detection process. The expert viewed the video and labeled it as true when the user’s head pose was directed toward the screen, and labeled it as false in other cases. Hence, this was treated as binary classification. Moreover, we artificially controlled the test environment so that only one person appeared in the camera. The model of [19] is a model that produces yaw, pitch, and roll as results. We made the model classify the result as true when yaw was between [45°, 45°] and classify it as false in other cases. Hence, this model also operated as binary classification. Because the experiment was conducted in a test environment where the tester looked at the screen and played the game, most of the labels were true. Hence, this...
was an unbalanced classification problem. The results are represented in the confusion matrix shown in Table 2.

**TABLE 2. Confusion matrix of the head pose estimation results.**

| Actual | Predicted |
|--------|-----------|
|        | Positive  | Negative |
| Positive| 99.31%    | 0.69%    |
| Negative| 29.70%    | 70.30%   |

For the head pose estimation, the accuracy was 99% and the f1 score was 0.84. Because most of the values were TP, the inference results for TN, FN, and FP cases were analyzed. In cases where the tester was not looking at the screen and turned the head to look to the side, the label was recorded as false, and the model also predicted it as false. However, for cases where the tester was looking at the screen but raised a hand near the face, such as when scratching the face or touching the head, the model yielded the yaw, pitch, and roll results with a very large error. This result caused a large number of FNs. In addition, the expert determined the labeling of the head poses not facing the screen; however, the model evaluated these cases by accurately dividing them based on 45°. As a result, FP and FN cases were sometimes found. The number of occurrences where the user did not look at the screen was of 1586 out of a total of 27,313 sets of input data, accounting for approximately 6%. If this methodology was not applied, inaccurate facial data would be delivered to facial expression recognition for approximately 4.8% of the total data, considering the accuracy. This indicates that many noise values could be generated as a result of inaccurate facial data.

3) FACIAL EXPRESSION RECOGNITION

Facial expression recognition used 25,727 cropped facial images as the input, which were the result of the face detection process, and inferred which facial expressions by the facial images. The composition of the labels for 25,727 data is presented in Table 3.

**TABLE 3. Number of frames labelled.**

| label  | count |
|--------|-------|
| Neutral| 23281 |
| Happiness| 2056 |
| Surprise| 44   |
| Sadness| 53   |
| Anger  | 293  |
| Disgust| 0    |
| Fear   | 0    |

This framework collects and analyzes the signals that testers express while testing the game. Expressions are very important signals because they reveal the feeling of testers indirectly. First, as a baseline, facial expression recognition predicted one of seven categories. The results are represented in the confusion matrix in Table 4.

The accuracy for facial expression recognition was 80%. Suppose that TP is the number of correctly classified expression categories, FP is the number of incorrectly classified expression categories, and FN is the number of samples belonging to a specific expression category but which have been classified incorrectly. Then, the f1 score calculated using the macro average method is 0.54. The score of the data set for testers was lower than the score of the benchmark dataset because the expressions in the videos were a sort of streaming signal. Hence, it was necessary to group them in sections rather than to process the result of each scene. A study conducted by Chu et al. [34] also verified that processing the results by combining them using the sliding window algorithm is rational and achieves higher accuracy in experimentation. The sliding window algorithm is used based on the following processes. The time series data are grouped into a window unit. The data undergo processes such as voting and averaging, and the results are listed as time series again. We also applied the sliding window algorithm to the results of our experiment. It was used because it made it easy to identify the trend of the signals. Because facial expressions are also streaming signals with continuity, application of the sliding window algorithm allows easier understanding of the trend of the signals. We selected a method of voting for the largest count of labels using the sliding window of the previous n values from the current value. This method had a significant effect. Because our experiment had the analysis results of real-time video streaming, it was rational to apply the sliding window algorithm, and a higher accuracy was achieved. The results are presented in the confusion matrix in Table 5.

When the sliding window was applied, the accuracy was 80% with a window size of 3 or 5. However, the f1 score calculated using the macro average method was 0.58 when the window size was 3, and 0.60 when the window size was 5. Neutral and happiness showed an accuracy of over 79%, and surprise, sadness, and anger showed an accuracy of over 60%. Hence, these expressions reached an accuracy level that can be applied in practice. However, the accuracy for disgust and fear did not reach a reliable level. This problem can be attributed to the limitation of not having label samples marked as disgust or fear in the videos of testers evaluated by the game ux expert. Moreover, this is a problem that causes an imbalance in general facial expression recognition training dataset. The occurrence of disgust or fear is rare, and they tend to have low accuracy. For this reason, we associated the expressions of testers during game testing with higher-dimensional connotations, and conducted an experiment that outputs positive or negative as a result. As reported by [35], this dimensional connotation enables an intuitive comparative evaluation, which compares the results of game tests with each other. We conducted an experiment by mapping the results of the multi-dimensional expression model to valence (i.e., positive vs. negative). Happiness and surprise were classified as positive expressions, and sadness, disgust, and
TABLE 4. Confusion matrix of the facial expression recognition experimental results.

|       | Neutral | Happiness | Surprise | Sadness | Anger | Disgust | Fear |
|-------|---------|-----------|----------|---------|-------|---------|------|
| Actual|         |           |          |         |       |         |      |
| Neutral| 79.68%  | 19.73%    | 0.03%    | 0.24%   | 0.19% | 0.13%   | 0.00%|
| Happiness| 11.35%  | 88.65%    | 0.00%    | 0.00%   | 0.00% | 0.00%   | 0.00%|
| Surprise| 4.76%   | 38.10%    | 57.14%   | 0.00%   | 0.00% | 0.00%   | 0.00%|
| Sadness| 16.39%  | 0.00%     | 0.00%    | 62.30%  | 8.20% | 13.11%  | 0.00%|
| Anger| 22.70%  | 11.66%    | 0.00%    | 6.13%   | 52.15%| 7.36%   | 0.00%|
| Disgust| 0.00%   | 0.00%     | 0.00%    | 0.00%   | 0.00% | 0.00%   | 0.00%|
| Fear| 0.00%   | 0.00%     | 0.00%    | 0.00%   | 0.00% | 0.00%   | 0.00%|

TABLE 5. Confusion matrix when the sliding window is 5.

|       | Neutral | Happiness | Surprise | Sadness | Anger | Disgust | Fear |
|-------|---------|-----------|----------|---------|-------|---------|------|
| Actual|         |           |          |         |       |         |      |
| Neutral| 79.92%  | 19.75%    | 0.00%    | 0.04%   | 0.16% | 0.13%   | 0.00%|
| Happiness| 9.33%   | 90.67%    | 0.00%    | 0.00%   | 0.00% | 0.00%   | 0.00%|
| Surprise| 5.26%   | 31.58%    | 63.16%   | 0.00%   | 0.00% | 0.00%   | 0.00%|
| Sadness| 13.73%  | 0.00%     | 0.00%    | 68.63%  | 5.88% | 11.76%  | 0.00%|
| Anger| 16.25%  | 10.62%    | 0.00%    | 5.62%   | 61.88%| 5.62%   | 0.00%|
| Disgust| 0.00%   | 0.00%     | 0.00%    | 0.00%   | 0.00% | 0.00%   | 0.00%|
| Fear| 0.00%   | 0.00%     | 0.00%    | 0.00%   | 0.00% | 0.00%   | 0.00%|

TABLE 6. Results of the three-class classification.

|       | Neutral | Positive | Negative |
|-------|---------|----------|----------|
| Actual|         |          |          |
| Neutral| 79.92%  | 19.75%   | 0.34%    |
| Positive| 9.25%   | 90.75%   | 0.00%    |
| Negative| 15.64%  | 8.06%    | 76.30%   |

fear were classified as negative expressions. The results are presented in the confusion matrix in Table 6.

The accuracy was 81%, and the f1 score calculated using the macro average method was 0.73. A relatively high accuracy can be achieved by classifying expressions into three classes. In addition, the game tester’s facial expression regarding the gameplay situation can be compared and analyzed more intuitively using the connoted meanings. Moreover, we were able to verify that a relatively high accuracy can be achieved.

4) SUMMARY
The results obtained from the experiment are summarized in terms of execution speed and performance. The execution speed is presented in Table 7. In the worst case, the processing time exceeded 0.33s. However, because the analysis framework operated asynchronously, this processing time did not affect the analysis processing time of the next frame. The average analysis processing time was 0.15 s, which demonstrates that the real-time system can sufficiently perform one analysis every 0.33 s. The performance is presented in Table 8. The experiment results showed that our proposed framework achieved the f1 score of 0.98 in face detection, 0.84 in head pose estimation, 0.60 in facial expression recognition, and 0.73 in the three classes. Hence, reliable values could be obtained. The introduction of this framework as part of the actual ux test process was considered. When the ux test was used for the experiment, we conducted an interview with the department in charge of the ux test. We received feedback that data values collected through this framework could adequately function as auxiliary indicators, which can be used as evidence in the level design phase. We also received a positive evaluation with respect to the suitability of the proposed framework to capture and record the user responses, such as changes in the game tester’s face and head pose.

V. DISCUSSION
Our work has some limitations. First, the proposed framework can exhibit different performances because of a
difference between the lab environment, where we conducted experiments on this framework, and practical environment, where this framework was tested. Because the proposed framework was used in a remote environment, it can be practically applied in environments different from that of a lab. In our experiments, the proposed framework was tested and verified under the conditions that ensured bright locations and images of testers’ faces, which were captured near the camera installed. However, the setting of a practical remote environment (e.g., a resident place of a tester or a room in the place) can differ from that of the lab environment we used. To solve this problem, the following methods can be applied. First, before conducting experiments on the proposed framework, we clearly inform testers about the where the experiments will be performed. In addition, we receive an agreement from the testers concerning their cooperation to ensure a bright environment and adjust the distance between them and the camera installed as requested. If users of the proposed framework apply these conditions in a practical environment, they can collect videos in a similar environment to that of the lab environment we established and obtain similar results to ours. Second, a class imbalance of facial expressions exists. As testers were asked to play games in our experiments, they tended to show certain facial expressions (e.g., absence of a facial expression and happiness) and did not show certain facial expressions (e.g., disgust and fear). We intended to detect signals of emotion occurrences in game experience tests that were practically conducted instead balancing the classes of facial expressions. Accordingly, we derived results by applying the sliding window algorithm and valence mapping technique to increase the reliability of the signals detected. To overcome this limitation, conditions for using the proposed framework can be slightly customized according to the characteristics of games and the needs of UX researchers.

In the future, we plan to conduct research to enable more precise prediction by collecting additional multi-modal data.

Because our framework was designed to enable the tester and the observer to communicate with each other using audio data as well as face-cam data, we can try to measure additional expressions by analyzing the audio data. Finally, we plan to detect expressions of users more accurately using multi-modal expression recognition by fusing the expression analysis results from two or more methods [36], [37]. Moreover, we obtained consent from the focus groups of our experience test to use the face-cam data to update the classifier model in the future. We plan to use these data to apply various data labeling techniques and upgrade the model. In addition, we are considering the use of semi-supervised learning, which can improve the expression recognition performance by using recent unlabeled data for training [38]. Using this method and the face-cam data, we plan to obtain a model with even higher accuracy.

VI. CONCLUSION

We proposed a framework that utilizes deep learning to collect and analyze the expression and engagement of testers in a remote environment on a game experience test. First, the proposed framework enables video data collection in a remote environment for automated collection of large-scale data. In this way, automatic collection of facial expressions of game testers can be performed. Second, objective and quantitative analysis of the facial expressions of the testers can be performed using a model that infers emotions based on deep learning. Finally, through various tests, sufficient level of accuracy and performance were verified for use in a game currently in service. The proposed framework enables collection of large-scale facial expression data of testers. At the same time, this framework can be combined with questionnaire collection to enhance the completeness of the analysis. Moreover, the real-time observation support of this framework can be run in parallel with the method in which researchers observe the testers. We experimentally demonstrated that this new method of collecting game experience test data has practical accuracy.

ACKNOWLEDGMENT

(Seungjin Kwon, Jaehyun Ahn, and Hyukgeun Choi contributed equally to this work.)

REFERENCES

[1] M. S. El-Nasr, A. Drachen, and A. Canossa, Game Analytics. London, U.K.: Springer, 2016.
[2] J. S. Dumas, J. S. Dumas, and J. Redish, A Practical Guide to Usability Testing. Bristol, U.K.: Intellect Books, 1999.
[3] L. E. Nacke, “Games user research and physiological game evaluation,” in Human Computer Interaction Series. Berlin, Germany: Springer, 2015, pp. 63–86.
[4] L. E. Nacke, “An introduction to physiological player metrics for evaluating games,” in Game Analytics. London, U.K.: Springer, 2013, pp. 585–619.
[5] K. Poels, Y. de Kort, and W. Ijselstein, D3.3 Game Experience Questionnaire: Development of a Self-Report Measure to Assess the Psychological Impact of Digital Games. Eindhoven, The Netherlands: Technische Universität Eindhoven, 2007.
[6] C. Marshall and G. B. Rossman, Designing Qualitative Research. Newbury Park, CA, USA: SAGE, 2014.
[7] C. Fisher and P. Sanderson, “Exploratory sequential data analysis,” *Inter-
actions*, vol. 3, no. 2, pp. 25–34, Mar. 1996, doi: 10.1145/227181.227185.

[8] W. Yang, M. Ritó, C. Marsala, and A. Pinna, “Physiological-based emotion
detection and recognition in a video game context,” in *Proc. Int. Joint Conf.
Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8, doi: 10.1109/IJCNN.2018.8489125.

[9] D. Giakoumis, D. Tzovaras, K. Moustakas, and G. Hassapis, “Automatic
recognition of boredom in video games using novel biosignal moment-
based features,” *IEEE Trans. Affect. Comput.*, vol. 2, no. 3, pp. 119–133,
Sep. 2011, doi: 10.1109/TAFFC.2011.4.

[10] G. Chanel, J. M. Kivikangas, and N. Ravaja, “Physiological com-
pliance for social gaming analysis: Cooperative versus competitive play,”
*Interacting Comput.*, vol. 24, no. 4, pp. 306–316, Jul. 2012, doi:
10.1016/j.tci.2012.04.012.

[11] Y. A. Sekhavat, S. Roohi, H. S. Mohammadi, and G. N. Yannakakis, “Play
with one’s feelings: A study on emotion awareness for player experience,”
*IEEE Trans. Games*, vol. 14, no. 1, pp. 3–12, Mar. 2022.

[12] N. A. M. Jai, M. M. Rosly, and N. A. A. Razak, “Correlating psychophysio-
logical responses of exergaming boxing for predictive heart rate regression
models in young adults,” *IEEE Trans. Games*, vol. 12, no. 4, pp. 398–405,
Dec. 2020.

[13] M. C. Medlock, “An overview of GUR methods,” in *Games User Research.
Oxford, U.K.: Oxford Academic*, 2018, doi: 10.1093/oso/9780198794844.003.0007.

[14] A. Drachen, A. Canessa, and J. R. M. Sørensen, “Gameplay metrics in
game user research: Examples from the trenches,” in *Game Analytics.
London, U.K.: Springer*, 2013, pp. 285–319.

[15] R. L. Mandyck, M. S. Atkins, and K. M. Inkpen, “A continuous and
objective evaluation of emotional experience with interactive play envi-
nornents,” in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, Apr. 2006,
p.1–10, doi: 10.1145/1124772.1124926.

[16] B. Guthier and R. Dörner, “Affective computing in games,” *Entertainment
Comput. Serious Games*, vol. 88, no. 421, pp. 402–441, Mar. 2016, doi:
10.1145/1124772.1124926.

[17] J. Deng, J. Guo, E. Ververas, I. Kotsia, and S. Zafeiriou, “RetinaFace:
Single-shot multi-level face localisation in the wild,” in *Proc. IEEE/CVF
Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 5203–5212,
doi: 10.1109/cvpr42600.2020.00525.

[18] Y. He, D. Xu, L. Wu, M. Jian, S. Xiang, and C. Pan, “LFFD: A light and
fast face detector for edge devices,” 2019, arXiv:1904.10633.

[19] Y. Zhou and J. Gregson, “WhoNet: Real-time fine-grained estimation for
wide range head pose,” 2020, arXiv:2005.10353.

[20] P. Ekman and W. V. Friesen, “Facial Action Coding System,” Consulting
Psychologists Press, Palo Alto, CA, USA, 1978.

[21] A. V. Savchenko, “Facial expression and attributes recognition
based on multi-task learning of lightweight neural networks,” 2021,
arXiv:2103.17107.

[22] webrtc 1.0: Real-Time Communication Between Browsers. Accessed:
Sep. 12, 2022. [Online]. Available: https://www.w3.org/TR/webrtc/

[23] Git Repositories on webrtc. Accessed: Sep. 12, 2022. [Online]. Available:
https://webrtc.googlesource.com