On Developing Facial Stress Analysis and Expression Recognition Platform

Fabio Cacciatori  
Illogic S.r.l  
Turin, Italy

cacciatori@illogic.xyz

Sergei Nikolaev  
Cyberphysics LLC  
Moscow, Russia

s.nikolaev@cyberphysics.xyz

Dmitrii Grigorev  
Illogic S.r.l  
Turin, Italy

dm.srg.gr@gmail.com

Anastasiia Archangelskaya  
Skolkovo Institute of Science and Technology  
Moscow, Russia

anastasiia.archangelskaya@skoltech.ru

Abstract—This work represents development and experimental process of facial expression recognition and stress analysis platform for immersive digital learning platform. The video stream received from the user from the webcam evaluates the condition using neural network algorithms to score and improve learning process. Also presented some implementations of algorithms for facial expression recognition and stress analysis in real-time mode, which are necessary to overcome the limitations associated with the minimum hardware requirements to run the ANN and improve accuracy of facial expressions recognition as well increase response speed. Experimental results showed that using the developed algorithms allow to detect the heart rate with better rate in comparison with social equipment.

Index Terms—digital learning, facial expression recognition, facial stress analysis, artificial neural network, vision analysis, real-time media identification, computer vision, synthetic training, virtual being, human digital twin.

I. INTRODUCTION

Avatar Platform (AVATAR) [1] is a software for the creation of "Serious Games"; informally defined as "video games with a purpose that is not exclusively playful", for example for educational purposes. The platform allows users to immerse in a virtual training experience. Thanks to the use of innovative technologies such as artificial intelligence, computer vision algorithms, and virtual assets (people digital twins or avatar, and scenographies), the users are able to experience a situational or face-to-face training simulation (i.e. synthetic training), according to their needs or indications from their employers. AVATAR is composed of a back-Office platform designed for the management, creation, and modification of training contents, and the Player, which allows the user to carry out the courses.

Stress negatively affects the learning process. When a student receives a new task, he becomes anxious. If the task is too complex and large, fatigue sets in. The brain can no longer resist stress, and the student ceases to be productive. For this reason, the process of monitoring student stress and the emotional state plays a huge role in the learning process.

One of the main objectives of the application is to bring innovative training experience to users and make the training process more engaging and therefore more efficient in terms of storing shared contents and learning. The platform software uses artificial intelligence algorithms based on neural networks to implement four features: speech recognition, facial expression classification, stress level analysis, and attention level measurement. This paper considers two of these features: facial stress analysis (FSA) and facial expression recognition (FER) algorithms.

The FSA algorithm is able to analyze - applying a specific color filter - the flush of blood after every heartbeat and estimate the number of heartbeats of the user, thus also allowing an estimate of the level of stress of the learner who is taking the course.

The facial expression recognition algorithm implemented in the AVATAR platform allows the recognition of up to 7 different facial expressions and, if the trainer wishes, to influence the learner’s training path based on these results. The facial stress analysis (FSA) algorithm is able to analyze - applying a specific color filter - the flush of blood after every heartbeat and estimate the number of heartbeats of the user, thus also allowing an estimate of the level of stress of the learner who is taking the course.

II. RELATED WORKS

The development and application of facial expression recognition and stress analysis platforms have been the subject of numerous studies in recent years. These studies have explored various aspects of facial analysis, from diagnosing diseases to real-time emotion recognition, and have employed a range of techniques, including traditional machine learning and advanced deep learning algorithms.

Jin et al. [4] explored the use of facial expression recognition to diagnose Parkinson's disease. They used relative coordinates and positional jitter to extract facial expression...
features and applied both traditional machine learning and advanced deep learning algorithms for diagnosis. Their work demonstrates the potential of facial expression recognition in the field of digital diagnosis.

In another study, Koujan et al. [5] proposed a novel method for human emotion recognition from a single RGB image. They constructed a large-scale dataset of facial videos, rich in facial dynamics, identities, expressions, appearance, and 3D pose variations. Their proposed framework was capable of robustly estimating parameters of 3D expression variation and accurately recognizing facial expressions from in-the-wild images.

Bhatti et al. [6] proposed an instructor’s facial expression recognition approach within a classroom using a feedforward learning model. They used deep features extracted using multiple convolution neural networks along with parameter tuning which were then fed to a classifier. Their proposed method showed significant performance gain on parameters such as accuracy, F1-score, and recall.

Lastly, Achyutha et al. [7] aimed to create a real-time Graphics User Interface based Automated Facial Recognition as well as Mask Detection System. They used Principal Component Analysis (PCA) and the HAAR Cascade Algorithm for the proposed methodology.

These studies provide valuable insights into the current state of research in facial expression recognition and stress analysis. In the following sections, we will discuss how our work builds upon these studies and contributes to the field.

III. FSA ALGORITHM DESCRIPTION

Stress level analysis is derived directly from an estimate of the participant’s heart rate. To make the system easier and cheaper to deploy, the heart rate measurement is not done through wearable devices, but rather through an analysis of facial skin coloration. A computer vision algorithm can notice the sudden change in the coloration of the face (invisible to the human eye) due to the influx of blood as a result of the heartbeat. To date, no avatar-based training solutions are using this technique.

Heart rate (HR) is an important indicator of people’s physiological state. Recent research has shown that these physiological signals can be measured with a simple camera. The human visual system has limited spatial-temporal sensitivity, but many signals that do not fall within this limitation can be informative. For example, human skin color varies slightly with blood circulation. This variation, although invisible to the naked eye, can be used to extract a person’s heart rate.

The idea of monitoring physiological parameters without sensors worn by the person is to be found in the cardiovascular system of the human body. The cardiovascular system allows blood to circulate in the body thanks to the continuous pumping of blood from the heart. Our heart pumps blood through the blood vessels, and with each heartbeat, the circulation of the blood creates a color change in the skin of the face. Therefore, it is possible to extract HR from the change in the skin color of the face. We are nowadays experimenting as well with the possibility to calculate blood pressure.

The approach developed in [2] combines spatial and temporal processing to emphasize subtle temporal changes in a video. First, the video sequence is decomposed into different spatial frequency bands. These bands may be magnified differently because they may have different signal-to-noise ratios, or they may contain spatial frequencies for which the linear approximation used in the motion magnification may not be suitable. In the latter case, the amplification for these bands is reduced to suppress artifacts. The goal of spatial processing is, therefore, simply to increase the temporal signal-to-noise ratio by putting together more pixels, spatially filtering the frames, and applying downsampling for computational efficiency. In the specific case, however, a complete Laplacian pyramid is calculated as in [8].

The second stage involves performing temporal processing on each space band. The time-series corresponding to the value of a pixel in a frequency band, are considered and a band-pass filter is applied to extract the frequency of the band of interest. For example, frequencies within 0.4-4Hz could be selected, which is approximately 24-240 beats per minute. The temporal processing is uniform for all spatial layers and all pixels within each layer. The extracted signal is then multiplied by a magnification factor called $\alpha$. This factor can be specified by the user or can be obtained automatically.

In the next stage, the zoomed signal is added to the original signal and collapsed space pyramid to get the final output. For further details about the mathematics behind this system, please refer directly to the paper [2].

The technique can be performed in real-time to show phenomena occurring at user-selected time frequencies. The developed version has the following changes:

1. Heart rate monitoring via webcam.
2. Face detection algorithm implementation.
3. Tuning of some computation parameters.

As for the first point, the beat calculations were inserted directly from the webcam. In this way, if the debug mode is activated, a window is displayed in which the image captured by the camera is represented and the computed bPM value is reported. The algorithm needs an initial stage equivalent to about 5 seconds for the initialization of the calculations and to keep the output bPM value more stable.

Face detection is performed using the algorithm in [9]. This method is very fast and quite accurate in the calculation. It could also be replaced with a CNN network: this option is under evaluation since this system has been integrated into the platform. Face detection is as well used to identify if there is a face within the area to be inspected. Intersection over union metric is used to decide if a face is present within the area in question. This metric works very well, and we have decided to implement it according to this scheme: our ground truth is represented by the inspection area for the recognition of the beat while the recognized face is our identified object. At this point, if the intersection between the union has a ratio greater than a certain threshold, the beat is computed, otherwise, that
frame is not considered and a new face is expected to be identified. The screen, in debug mode, is shown in figure 1. In green is the area being analyzed, and in blue is the result of the face detector.

![BPM: 76](image)

Fig. 1. Debug mode with display of the output window of the developed code.

IV. FER ALGORITHM DESCRIPTION

Facial Expression Recognition is a feature that allows you to monitor the user’s facial expression. It is based on Computer Vision algorithms able to identify seven different facial expressions (neutral, angry, disgust, fear, happy, sad, and surprised) and indicate the level of certainty. The facial expression recognition system allows to perform an analysis of the emotional state of a user who performs the course during their training session. In particular, the system must capture the webcam stream and recognize the expression of the user’s face. The seven emotional states are: Angry, Disgust, Fear, Happy, Sad, Surprise, Neutral.

The algorithm that detects the user’s expression consists of three main components:

- REST interface to receive the input frame to analyze and return the output;
- Python class that uses OpenCV’s Viola-Jones [9] algorithm to identify the user’s face;
- Neural network implemented in Keras [11] that takes previously identified face and classifies its expression.

The reference dataset is called FER-2013 [3]. The data consists of 48x48 pixel grayscale face images. The faces have been registered automatically so that the face is more or less centered and takes up about the same amount of space in each image.

The training dataset consists of 28,709 examples. The public test dataset used for ranking consists of 3,589 examples. The final test dataset, which was used to determine the contest winner, consists of another 3,589 examples. In [9], Global Average Pooling is used to eliminate the fully connected layer. This was possible by having in the last convolutional layer the same number of feature maps as the number of classes and by applying a SoftMax activation function to each reduced feature map. In this way, their initial architecture is, in fact, a fully convolutional standard neural network consisting of 9 layers of convolution, ReLU, batch normalization, and Global Average Pooling.

Since our initially proposed architecture completely eliminated the last connected layer, we have further reduced the number of parameters by now eliminating them from the convolutional layers. This was done using depth-wise separable convolutions [12]. Depth-wise separable convolutions are composed of two different layers: depth-wise convolutions and point-wise convolutions. The main purpose of these layers is to separate the spatial cross-correlations from the channel cross-correlations [9]. It was achieved by applying a DxD filter on each M input channel and then applying N convolution size 1x1xM filters to combine the M input channels into N output channels. The application of 1x1xM convolutions combines each value in the feature map without considering their relationship space within the channel.

The final neural network architecture contains four residual depth-wise separable convolutions where each convolution is followed by batch normalization and a ReLU activation function. The output layer applies a global average pooling and a soft-max activation function to make a prediction. This architecture has about 60,000 parameters, which corresponds to a 10-fold reduction compared to the originally proposed architecture. Furthermore, everything happens in real-time with a time of about 0.22 seconds (also considering face recognition).

The following is a description of several architectural improvements to the current system that allows better results in emotion detection. The first option is to use one of the following solutions [9], [12]. These solutions implement VGG network architectures [13]. These two solutions achieved a result of 70.382% and 73.112% accuracy on the FER-2013 dataset respectively. Unfortunately, the calculation times of this network are unknown as they are not reported and, therefore, it is a question of testing the network to verify the actual real-time of the system.

Another way involves creating a new network starting, perhaps, from one of those previously seen and then verifying the results that can be obtained by comparing them with other state-of-the-art networks. This solution was chosen for the final implementation. This solution had the most advantages from the point of view of practicality and implementation speed. In addition, it allows one to have full control of the network and not rely on a network developed by a third party. After several attempts to try to optimize the network model to be created. The confusion matrix obtained after training presented in Table 1.

The first integration was done by encapsulating a model proposed by [3] that managed to achieve 66% accuracy with two models. The first model uses about 600,000 parameters, while the second one, inspired by Xception [7], uses only 60,000 parameters. This second model is therefore preferable for performance reasons and succeeds in performing the classification in only 0.22 seconds on a medium-performance PC.
VGG16 and VGG19 architectures allowed accuracy of 70.1%. It is important to note that an accuracy value of 70%, although far from the desired 100%, is actually a very good result. In fact, in a task of discrimination among 7 different membership classes, a purely random selection would result in a starting accuracy of "one-seventh", i.e., about 14.3%.

V. FSA ALGORITHM TESTS

The FSA algorithm tests were divided into 2 stages. These tests have been conducted on the system to verify its correctness and therefore analyze the results obtained. In the first stage, three three-minute videos were recorded in 3 different stages of the morning (to have slightly different lighting conditions). The first 30 seconds of the videos were used to initialize the two systems. The results were analyzed by comparing the developed algorithm data with the data extracted from an smartwatch device. The heart rate data was extracted every 10 seconds during video recording and subsequently reported in tables to able to compare the graphs.

In the second stage, three three-minute videos were recorded during the afternoon (also in this case to test different lighting conditions). Again, the first 30 seconds of the videos were used to initialize the two systems. The results were analyzed by comparing the data of the developed algorithm with the data extracted from an oximeter that allows for evaluating the accuracy of "one-seventh", i.e., about 14.3%.

Some functional tests of possible integration of the FSA algorithm inside a moveable camera were also included. Compared to the tests seen in stages 1 and 2, these provide for a distance of the face from the camera equal to 1.5 / 2 meters. For this reason, we carried out 2 tests (one at a distance of 1.5 m and one at a distance of 2 m) to evaluate the reliability of the code and discussed the results.

In Table 2 we report the comparison values between the averages calculated on the three tests between the two approaches (smartwatch device and developed algorithm). In particular, the last column shows the average error difference.

The data in Table 2 show an error between the two algorithms of about 3.1 beats per minute. In general, the data estimated by the algorithm are always lower than those measured by the smartwatch device. However, the comparison is encouraging. The average absolute error between the two approaches is 3.1875 beats. For the system that is to be achieved, this gap turns out to be a good value to be able to carry out the analysis. The difference can also be explained as a function of the different brightness, the more or less still positions, and the head movement of the subject on which the measurement is performed. In any case, this also helps us to better understand how the algorithm behaves in different conditions. During the tests, moments were verified in which the algorithm greatly underestimates the value of the beat (e.g. 30-40 bpm) this could be due to some error in the calculation of the beats or to a problem related to the area in which the computation is carried out. Fortunately, the values return to normal after a few seconds. We would like to underline that this behavior occurs very rarely and that, for this reason, the tests were conducted by measuring the heartbeat every 10 seconds.

As in the previous example, in Table 3 we report the comparison values between the averages calculated on the three tests between the two approaches (Pulse oximeter and algorithm developed). In particular, the last column shows the average error difference.

The data in Table 3 show an error between the two algorithms of about 1 beat. The error, in this case, is lower than the one calculated via smartwatch device, probably also due to a different evaluation between the smartwatch device and oximeter or to the different lighting conditions of the subject since the test was conducted in the afternoon. In general, the data are even more encouraging than the previous ones and this is an excellent signal on the goodness of the algorithm developed. The oscillations of the oximeter between

| Test | Method | Average bpm | Offset bpm |
|------|--------|-------------|------------|
| 1    | AW     | 84.8125     | 5.4375     |
|      | AL     | 79.375      |            |
| 2    | AW     | 86.5        | 3.8125     |
|      | AL     | 82.6875     |            |
| 3    | AW     | 80.8125     | 0.3125     |
|      | AL     | 80.5        |            |
| Total Average |     | 3.1875      |            |
the minimum and maximum calculated frequency are lower than those of the Smartwatch device and this represents a better approximation of the calculation by the oximeter itself.

Even the developed algorithm, as highlighted in the previous tests, sometimes has peaks but the values are uniform by calculating the average over longer times. As described in the introduction, this paragraph will analyze the results obtained with our algorithm having a camera-face distance of 1.5 and 2 meters. For this reason, we have carried out two tests to be able to verify the efficiency of this algorithm in these 2 limit situations. The videos are always 3 minutes long and are analyzed starting from minute 2:30 to stabilize the algorithm calculations. The oximeter was used as a comparison approach as it is more accurate than the Smartwatch device. The videos were recorded in the morning with sunlight.

To carry out these new tests, we had to modify some parameters of existing solutions. First, we had to eliminate the face detector. This is because, at distances greater than 1.2 meters, the face detection algorithm is no longer able to find faces (the faces are too small given the resolution of 320x240). The alternative would be to increase the size of the image, but this can lead to a slowdown in the execution of the algorithm itself. Several parameters of the Haar Cascade (face detection algorithm) have been tested but we have not been able to overcome this problem. We also tried using a CNN network (MTCNN) but this too has the same limitations. From an estimate made, the ratio of the face area to the heart rate calculation area is about 10%. As this ratio is very small, we believe that much of the heart rate calculation is related to misreading background information. For this reason, we have added a heart rate measurement carried out on background only, therefore without a human face, to understand if this hypothesis can be confirmed.

While the trend of the oximeter is quite constant, with a difference between the minimum and maximum, for the two experiments, equal to 10 and 13 beats respectively (1.5 meters and 2 meters), the algorithm data show oscillations similar to those of the background only, with differences between minimum and maximum equal to 32 and 38 (1.5 meters and 2 meters), equal to three times the real value read by the oximeter. These values turn out to be much higher than in the previous tests and, combined with the fact that the beats trend is similar to the one read from the background, making us assume that the measurement is strongly conditioned by the latter and that, therefore, the value that we read is not that of the person’s heartbeat.

Even by increasing the resolution, this algorithm cannot be used at such distances because the calculation area is always greater than the face area and therefore the background will increasingly influence the heart rate value.

Alternatively, one could think of calculating the heartbeat only in the face area but due to the structure of the algorithm itself (the Gaussian pyramid), the face should always be resized to a standard size before making the calculations. Furthermore, we would still be bound by the maximum face recognition distance of the face detector.

A series of tests were also carried out on videos extracted directly from the moveable camera. The data processed by the algorithm were compared with those extrapolated from the smartwatch device. The beat values were calculated considering all the 30 seconds of acquisition and not every 10 as occurred in the previous experiments. The results confirmed what was analyzed with the last 2 tests carried out: if the person is at a distance greater than 1.2 meters, the face detector struggles to find the face (since it turns out to be very small) and the pulse value is strongly influenced by the background.

Figure 2 has represented the different facial expressions recognition and heart rate detection.

![Figure 2. Examples of facial expressions recognition (Disgust and Angry).](image)

**TABLE III**

| Test | Method | Average bpm | Offset bpm |
|------|--------|-------------|------------|
| 1    | S      | 84.125      | 1.6875     |
|      | AL     | 79.4375     |            |
| 2    | S      | 78.25       | -0.625     |
|      | AL     | 78.875      |            |
| 3    | S      | 79.3125     | 0.4375     |
|      | AL     | 78.875      |            |
| Total Average |          | 0.9167      |            |

VI. FSA AND FER ALGORITHMS IMPLEMENTATION.

FER and FSA algorithms have quite different implementation logics. While the first requires a single frame extracted from the video to establish the detected emotion, in the FSA algorithm it is necessary, as already highlighted in the previous paragraphs, for a first initial calibration stage, lasting 5 seconds, in which a frame buffer will be populated before feedback can be provided. In addition, enough frames per second must be provided for computation, approximately 15-20.

This substantial difference translates into a longer wait before having feedback on the BPM, as well as unreliability on the returned value when the frames supplied to the algorithm are not enough. The FSA is therefore very dependent on the number of frames passed to the algorithm itself. The greater the number of requests to be performed in a given second, the greater the speed with which the calibration stage will end to provide results as close as possible to the real ones.
Having analyzed the typical work contexts in which the Avatar user will find himself, it has been noticed how with a network that has a download speed of about 6 MBps per second it is possible to perform about 20 requests per minute (1 request every 2-3 seconds), thus making it impossible to have a quick calibration stage and satisfactory results with what has already been implemented. In addition, there are also some limits of Unity (version 2018) that prevent the system from making too many requests without falling into an overflow error and crashing the application itself.

One of the possible solutions to the problem is the implementation of an Apache Kafka architecture, which is an architecture that deals with processing large amounts of data in real-time, allowing the creation of scalable systems with high throughput and low latency. Another solution, which starts from the idea expressed in the first solution, is based on creating a similar structure based on Kafka but present only on the server-side and not on the Unity player. This solution has several advantages. First of all, it does not expose the Kafka orchestrator on the web but confines it only to the server and the server’s architecture. Secondly, it allows keeping the structure based on RESTful API unaltered on the unity side so as not to compromise, possibly, the WebGL version. The basis would always be the Kafka-Python python library. Figure 3 has represented the architectural idea described above.

The integration of functionalities (recognition of facial expressions and estimation of heart rate) has also required a stage of code adjustment to improve performances and to optimize the method of transmission of webcam images from the client (Browser) to server. The second optimization has been necessary above all to limit the traffic of the net and to allow therefore the computation of the level of the beat with reduced latencies. It was necessary to condense the two applications into a single Unity module that captures the webcam images and that, in parallel, while the recognition of facial expressions is done on the client, sends the images to the server at regular intervals for remote processing.

VII. CONCLUSION

In this work was performed a solution that provides the learner’s training path based on facial expression recognition and facial stress analysis algorithms for the AVATAR platform. by now many application problems of neural networks are solved and available for use, but the actual applicability of these solutions in real contexts requires copious development efforts. During development, facial expression recognition and facial stress analysis algorithms were defined, tested and performance measured. The result of the FSA algorithm work was compared with data from an smartwatch device and oximeter. An error between the proposed algorithms and the smartwatch device measurement result is 3.1 beats per minute and the oximeter measurement result is 1 beat. The measurement results are acceptable for the correct operation of the system. Three approaches to the implementation of algorithms in the current platform are also proposed, and their advantages and disadvantages are given. In general, all the presented solutions can be valid for addressing the problem under consideration. It is more feasible to take the third solution that allows not to overturn the architecture that, to date, has been developed. In this way, development times should also decrease and it will not be necessary to find other solutions for the compatibility part with WebGL since communication always takes place via Rest API.

REFERENCES

[1] Avatar: Situation-based Learning Platform. 2022.
[2] H.-Y. Wu, M. Rubinstein, E. Shih, J. Guttag, F. Durand, and W. Freeman, “Eulerian Video Magnification for Revealing Subtle Changes in the Wild,” ACM Trans. Graph., vol. 31, no. 4, Jul. 2012, doi: 10.1145/2185520.2185561.
[3] Kaggle.com, FER-2013: Learn facial expressions from an image. 2022.
[4] B. Jin, Y. Qu, L. Zhang, and Z. Gao, “Diagnosing Parkinson Disease Through Facial Expression Recognition: Video Analysis,” in JMIR mHealth and uHealth, vol. 8, no. 7, pp. e18697, Jul. 2020. [Online]. Available: https://dx.doi.org/10.2196/18697.
[5] M. R. Koujan, L. Alharbawee, G. Giannakakis, N. Pugeault, and A. Roussos, "Real-time Facial Expression Recognition "In The Wild" by Disentangling 3D Expression from Identity,” in 2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020), Buenos Aires, Argentina, 2020, pp. 1-8. [Online]. Available: https://dx.doi.org/10.1109/FG47880.2020.00008.
[6] Y. K. Bhatti, A. Jamil, N. Nida, M. Yousf, S. Viriri, and S. Velastín, "Facial Expression Recognition of Instructor Using Deep Features and Extreme Learning Machine,” in Computational Intelligence and Neuroscience, vol. 2021, Article ID 5570870, 19 pages, 2021. [Online]. Available: https://dx.doi.org/10.1155/2021/5570870.
[7] P. N. Achyutha, S. Hebbale, and V. Vani, "Real time COVID-19 facemask detection using deep learning," in International Journal of Health Sciences, vol. 6, no. 4, pp. 1-8, Apr. 2022. [Online]. Available: https://dx.doi.org/10.53730/ijhs.v6n4.6231.
[8] D. McDuff, Camera Measurement of Physiological Vital Signs. arXiv, 2021. doi: 10.48550/ARXIV.2111.11547.
[9] P. Viola and M. J. Jones, “Robust Real-Time Face Detection,” International Journal of Computer Vision, vol. 57, no. 2, pp. 137–154, May 2004, doi: 10.1023/b:VISL.0000013087.49260.fh.
[10] G. bradski, "The Opencv Library." Dr. Dobb’s Journal of Software Tools, 2000.
[11] F. Chollet and others, “Keras,” 2015. https://github.com/fchollet/keras
[12] I. J. Goodfellow et al., “Challenges in Representation Learning: A Report on Three Machine Learning Contests,” in Neural Information Processing, 2013, pp. 117–124.
[13] O. Arriaga, M. Valdenegro-Toro, and P. Pfüger, Real-time Convolutional Neural Networks for Emotion and Gender Classification. arXiv, 2017. doi: 10.48550/ARXIV.1710.07557.