Real-time emotion recognition in virtual reality

Jeremy Di Dio and Sruti Bhattacharjee

Bachelor Project Report
École Polytechnique Fédérale de Lausanne
EPFL-IC-IIG

Project advisor: Prof. Dr. sc. Ronan Boulic
Project supervisor: Nana Tian

Abstract

This project has two objectives. The first one is to build a facial expression dataset using a virtual reality head-mounted display (HMD) providing specific facial features. The second one aims to implement an emotion recognition machine learning system using supervised learning from the dataset created, to enable real-time emotion prediction in virtual reality.

Introduction

Emotions play a central role in rational decision-making, perception, learning and a multitude of other processes influencing social interactions. Understanding other people's feelings is necessary to anticipate people's reactions. A multitude of research studies has been conducted on the subject to better define what emotions are and understand how they are expressed.

In 1972, Dr Ekman identified the 6 basic emotions that are common and innate amongst all human cultures [1]. These six emotions are namely *anger*, *disgust*, *fear*, *happiness*, *sadness*, and *surprise*.

Although recognizing emotions is trivial for humans, it remains a difficult task for computers. It is therefore not surprising to see that emotion recognition has recently been an active area of study in the field of computer vision and we have seen a lot of progress in the recognition of the 6 basic emotions.

In addition, with the emergence of virtual reality technologies in various fields such as gaming, health, or education, emotion detection using VR compatible devices is becoming more and more necessary.

This project addresses the problem of recognizing the 6 basic emotions in a virtual reality environment. The first contribution of our works is the creation of a database of posed expression, which is described in section 3. Using this newly created database, we trained two different machine learning models to provide real-time emotion recognition as described in section 4. Finally, section 5 presents the evaluation of our models.

I. Background

To provide a better understanding of the technical background of this report, we clarify here some dependencies used in our project.

Virtual reality devices

For this project, we used the virtual reality Head-Mounted Display (HMD) Vive Pro Eye from HTC as well as their new, at the time, Facial Tracker. These devices enabled us to collect specific facial features that we called "blendshapes". These blendshapes are a collection of facial expressions

represented as a percentage of activation and evaluated at a frequency of 60Hz. Figure 1 shows two examples of such blendshape.

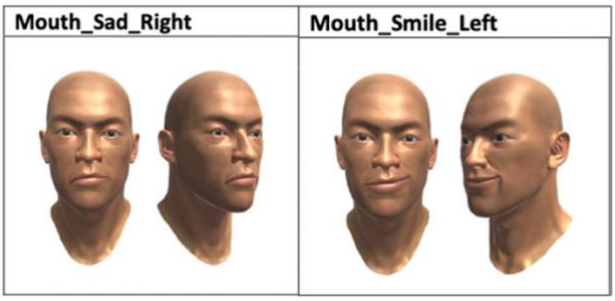

Figure 1: Sample of blendshapes provided by the facial tracker

In total, the devices provided us 52 different blendshapes, 14 for the eyes' expressions and 38 for the lips' expressions.

Facial Action Coding System

The Facial Action Coding System (FACS) [4] is an anatomical based system used to describe facial movements. It breaks down the facial expression into 44 single components of muscle movement, called Action Units (AUs). This coding system is often used for emotion recognition as one of the most important and simplest way to define emotions is through facial expression.

CK+ database

The extended Cohn-Kanade (CK+) database [5] is one of the most used posed facial expression databases for emotion recognition systems [6]. As the name implies, it is an extension of the primary Cohn-Kanade database (CK). This second version includes data from 123 participants in the form of 593 video sequences from 10 to 60 frames each depicting a transition from the neutral expression to the peak expression. Among these videos, some are labelled with the seven basic expressions known as *anger*, *disgust*, *fear*, *happiness*, *sadness*, and *surprise* in accordance with the Facial Action Coding System described above.

II. Feasibility

Before starting to collect any data, we studied the feasibility of the project. We ensured that this new kind of data we were provided, the blendshapes, could indeed be indicators of emotions. To do this, we took profit of the similarity between these blendshapes and the Facial Action Coding System. Indeed, we observed that the data at our disposal were similar to the AUs used in FACS. Starting from this observation, we established a mapping between these two types of data to indicate that, in the same way as AUs, blendshapes could be indicators of emotions.

We first built a mapping assuming which blendshape would be activated for each AUs. Then, based on this preliminary mapping, we collected real data and improved our mapping. Without surprise, we observed that our preliminary mapping was correct most of the time. Table 1 shows some examples of this mapping.

AUs	Blendshapes
AU 1	Eye_Left_Up Eye_Right_Up
AU 5	Eye_Left_Wide Eye_Right_Wide
AU 9	Cheek_Puff_Right Cheek_Puff_Left
AU 15	Mouth_Sad_Right Mouth_Sad_Left

Table 1: Sample of mapping between AUs and Blendshapes

Finally, since each AUs were mapping to a unique set of blendshapes, we concluded that they could be directly related to emotions and thus ensured us the feasibility of the project.

III. Database creation

Once we made sure that emotion recognition based on blendshapes data was possible, we worked on the creation of a posed expression database using virtual reality devices.

Design

To build our database, we chose to align our strategy of recording on the CK+ model. This means that we have tried to replicate, with some personal modification, the recording flow used in the creation of the CK+ database. We defined the recording protocol which worked as follows: participants had to perform a requested expression starting from the neutral state to the apex point of the expression during a timeframe of 5 seconds. We chose 5 seconds as, after multiple attempts, this was the most suitable timeframe to use since it was long enough to let the participant to reach the apex point.

Once the recording flow was defined, we established a particular set of expressions we wanted to record. At first, we selected a subset of AUs that were relevant for emotion recognition [8][9]. These specific AUs are of the number of 16 and are the following: AU 1, 2, 4, 5, 6, 7, 9, 10, 12, 14, 15, 16, 17, 20, 23, and 26.

Their correlation to each emotion is defined in Table 2.

Basic emotions	Corresponding AUs
Happiness	AU 6, 12
Sadness	AU 1, 4, 5
Fear	AU 1, 2, 4, 5, 7, 20, 26
Surprise	AU 1, 2, 5, 26
Disgust	AU 2, 4, 9, 15, 17
Anger	AU 4, 5, 7, 23

Table 2: Relevant AUs to basic emotions

We also decided to record the 6 basic emotions independently. To summarize, each participant had to express 23 different expressions, 16 AUs, 6 basic emotions, and the neutral state.

This recording procedure was thus repeated 22 times for 22 different expressions, the neutral state being expressed at the beginning of each recording. Figure 2 illustrate this recording process.

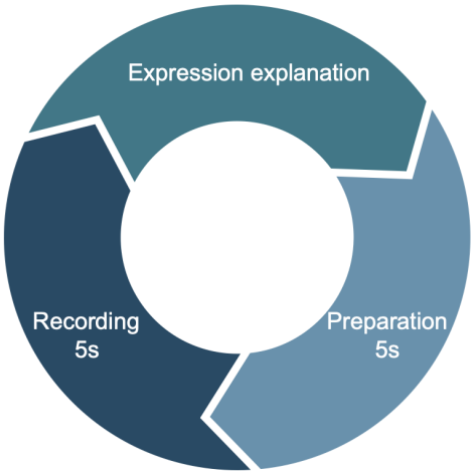

Figure 2: Illustration of the recording process

Implementation

To respect this recording protocol, we needed to create a virtual reality application to first, collect the data and second, to allow a visual representation to monitor and facilitate the recording sessions.

To do so, we used a SDK provided directly by HTC Vive. This SDK contained a Unity project sample enabling the use of both devices, the Vive Pro Eye and the Facial Tracker, in an open world with simply two objects: an avatar and a mirror reflecting the avatar's face, which itself reflects the player's expressions. To make this application suitable for our project, we had to proceed to multiple modifications.

First, we found a way to collect the data provided by the devices, the blendshapes, by changing the script allowing the avatar to render the user's expression. We have been able to write these data into two different *.csv* files, one for the lips and another for the eyes. We also proceeded to record the screen directly to have a visual interpretation of the recorded data.

Secondly, to have a more controlled environment, we have added a timer, the raw video of the lips and the actual requested expression

directly on the mirror for participants to better monitor their expression. The timer started a countdown of 5 seconds, allowing the participant to prepare himself where a sound at zero would let him know that the recording has begun for another 5-second timeframe. This timer was therefore controlling the entire recording process and was initiated at the beginning of each recording by the experimenter pressing the Enter key. Regarding the expression number visible in Figure 4, it helped us to track which expression was currently recorded.

Moreover, as it was not suitable for participants to remember all the requested expressions, especially the AUs. We decided to add a pane with a sample video [10] of each requested AUs, as shown in Figure 3. These videos were switchable using the arrows on the keyboard and participants were able to see them directly in the HMD between each recording.

Finally, as it was difficult to know what expression was performed by only observing the device's outputted data, we needed to find a way to determine when does the participant has reached his peak expression. To do this, we asked participants to press the space bar each time they felt they reached their expression apex. This action was adding a Boolean value to the recorded data indicating that indeed the apex of the requested expression was reached.

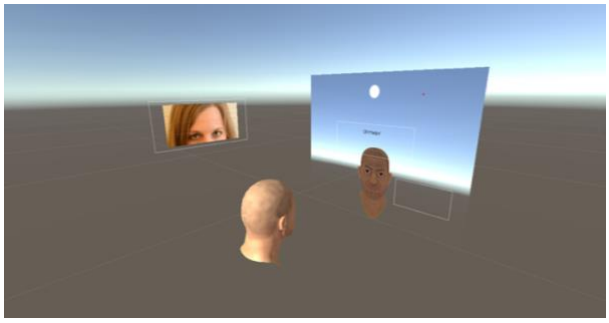

Figure 3: Unity 3D full view of the recording application with an AU sample on the left side

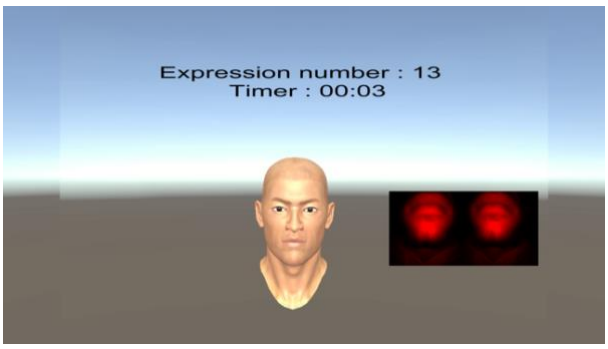

Figure 4: Participant view in the recording application

Participants

Once the recording application has been ready, we requested participants to come to the lab to collect their data. In total, we have been able to record 27 different participants, 19 men and 8 women. Figure 5 shows the facial particularities of these participants, and Figure 6 shows an example of a participant being recorded.

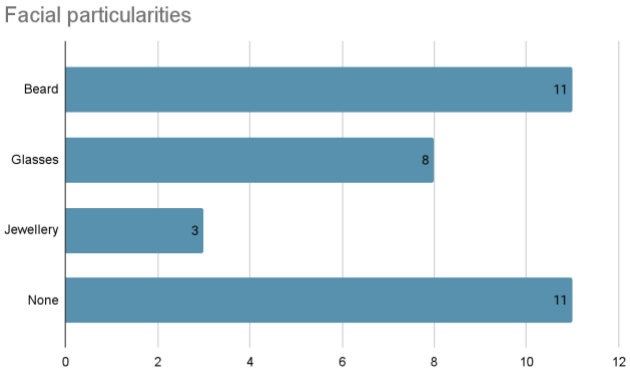

Figure 5: Facial particularities of participants

Unfortunately, we had to remove one participant from the database due to problem during the recording session. The final number of participants integrated into the database was then 26.

To conclude, the number of collected sample was around 5'500 by participants since recordings lasted 5 seconds, at a registered rate of 50Hz which gave us around 250 samples per recording per participant.

Figure 6: Participant, with HMD on and hand on the spacebar, being recorded

IV. Models implementation

The second part of this project was to implement machine learning (ML) models able to recognize the 6 basic emotions using supervised learning with our freshly created database.

Design

To address this classification problem, we selected two different well-known algorithms to be trained on our database. These algorithms were Random Forest and LightGBM [3]. We made this choice due to the small number of samples and the fact that they were labelled.

Random forest classifier

The random forest classifier [2] consists of a collection of decision tree that operates altogether. Every tree of the forest produces a unique class prediction, and the class with the most votes become the prediction of the model. Random forest is a noise-resistant method which can detect non-linear patterns and can also handle numerical data as well as categorical data with ease [7]. One of the advantages of this method is its resistance to overfitting, even with large forest.

LightGBM

LightGBM is an open-source Gradient Boosting Decision Tree (GBDT) method developed by a team from Microsoft [3]. In a similar way to random forest, it combines the predictions of multiple decision trees to assess the final prediction. The main advantage of LightGBM is that it is performing extremely well on high dimensional data.

Implementation

Dataset preparation

First, we proceeded to the preparation of the dataset to make it suitable with the chosen algorithms. To do this, we checked every file, one by one, to remove inconsistent data and we ensured correct labelling of the data. The method used was simple, as we only needed basic emotions labelled data. We selected only the emotions records and kept the first 30 samples before the apex point (included), labelled to the corresponding emotion. Regarding the neutral state, we kept only the first 5 samples of each recording and labelled them as the neutral state.

This process reduced our number of usable labelled samples to 210 samples by participants, which gave us a total of around 5’460 samples.

The last modification we made, was to perform a data augmentation on our dataset to synthetically double our number of samples. To achieve this, we took advantage of the fact that lips blendshapes and eyes blendshapes data were divided into two different files. It became therefore easy for us to randomly shuffle the association lips/eyes between participants. For instance, one way to create a synthetic participant was to take the lips file from participant X, the eyes file from participant Y and, combine them to build artificial participant Z, as illustrated in Figure 7.

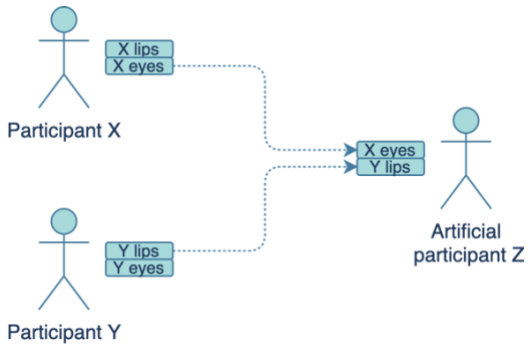

Figure 7: Data augmentation visualization

To prevent overfitting and limit the repetitions of samples, we only proceeded to this augmentation a single time. This meant that for each participant, we created a synthetic one from its data and from another participant's data. With this method, we have been able to double our number of samples to around 10'920 samples.

Finally, after all these changes, we had a well-balanced dataset amongst the 7 different classes, including the neutral state, ready to be trained on.

Random forest implementation

To implement the first algorithm, we made use of the Scikit-learn [11] python library, one of the most used ML library, to help us deal with its complexity. Indeed, using this library, it is extremely simple to build any machine learning model, to train it and to estimate its quality. Thus, the main work executed was to find the best hyperparameters to achieve the best accuracy.

In this first model, we looked at two different hyperparameters which were, the maximum depth and the number of trees. We found the best parameter for each, using cross-validation. Indeed, we built our own "by participant" cross-validation to have a more meaningful overview of the accuracy achieved by the model. This cross-validation was randomly taking 10 participants among the 52 available, setting them as a testing set and the others as a training set. The model was then trained using the training set and evaluated on the test set to obtain a preliminary accuracy. This process was then repeated 10 times and the final accuracy was therefore the mean of all the accuracies achieved during this randomization of the testing/training set.

Finally, we have been able to achieve a mean accuracy of 48% with a standard deviation of 7% among the cross-validation accuracies. Figure 8 below shows the confusion matrix of a single training/testing set split with an accuracy of 54%.

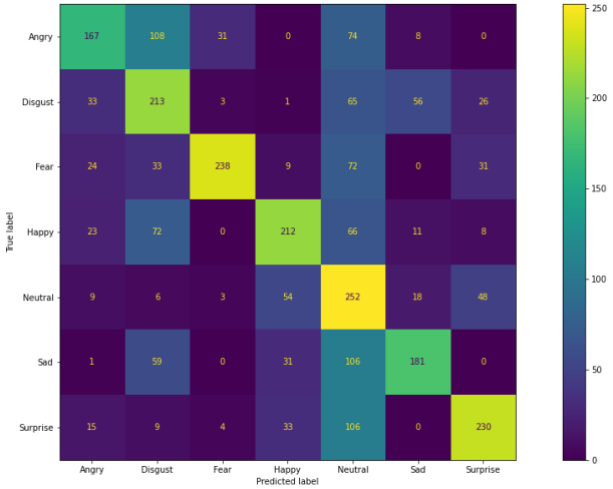

Figure 8: Confusion matrix using random forest classifier

LightGBM implementation

Regarding the second algorithm, we used the LightGBM python API. To build this model, we followed the same methodology as for the first one described above. Thus, the only remaining part was the tuning of the hyperparameters. For this, we choose to tune three different hyperparameters, namely the number of trees, the learning rate, and the number of leaves. As with the random forest classifier, we used our cross-validation to estimate the actual accuracy of our model and find the best hyperparameters.

At the end, we have been able to reach a mean accuracy of 52% with a standard deviation of 9% among the cross-validation accuracies. Figure 9 shows the confusion matrix of a single training/testing set split with an accuracy of 54% to illustrate the model's performance.

Finally, we concluded that achieving around 50% accuracy on a 7 classes classification problem was acceptable and we then proceeded to the real-time evaluation.

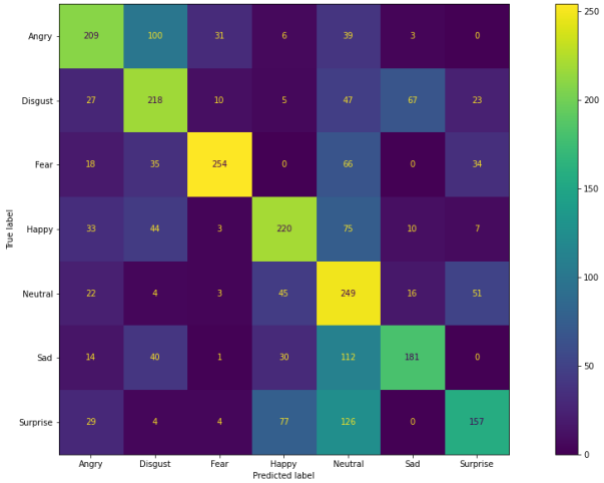

Figure 9: Confusion matrix using LightGBM model

V. Evaluation

The primary idea of the evaluation process was to implement a virtual reality application, like the one used to build the database, capable to evaluate the emotion of the user in real-time and render it directly on the screen. Unfortunately, due to time restrictions and the complexity of the work required to enable communication between our trained python models and Unity 3D, we chose to modify our approach and opt for an offline evaluation using prerecorded data.

Evaluation data

Following our choice of evaluation method, we needed to collect a new set of data. The goal here was to have the more realistic data to simulate real-time evaluation. To address this, similarly to the first database we created, we requested participants to come to the lab to be recorded using our previously built application. However, this time, participants were not requested to express all the expressions one at a time but instead, to perform the 6 basic emotions plus the neutral state sequentially in a single recording of 25 seconds. Participants were again asked to press the spacebar when they reached the apex point of each emotion.

This recording process has then been repeated twice by each participant. The first time expressing the emotions in the order they wanted and the second time in a predefined order, as shown in Table 3.

Evaluation recording predefined order	
Expression 1	Neutral
Expression 2	Happiness
Expression 3	Sadness
Expression 4	Fear
Expression 5	Anger
Expression 6	Surprise
Expression 7	Disgust

Table 3: Predefined expression order used for the evaluation database

Furthermore, we requested participants to always go through the neutral state between each expressed emotion. We also proceeded to record the screen during recordings to have a clearer interpretation of the collected data.

Sadly, due to time constraints and the current sanitary situation, we have been able to record only two participants. However, they still provided us with significant insightse into our models' performances.

Results

Once the data were collected, we proceeded to its review by ensuring consistency and correct labelling. We then constructed a small web application to visualize the performance of our models, accessible at the following link: https://share.streamlit.io/dioday45/emotion-recognition-evaluation-visualization/main.

In this application, the user first needs to select the model, the participant, and the recording to be evaluated. Once these parameters are defined, the user can then launch the simulation. This will create a video containing the emotion prediction in term of probabilities as illustrated in Figure 10.

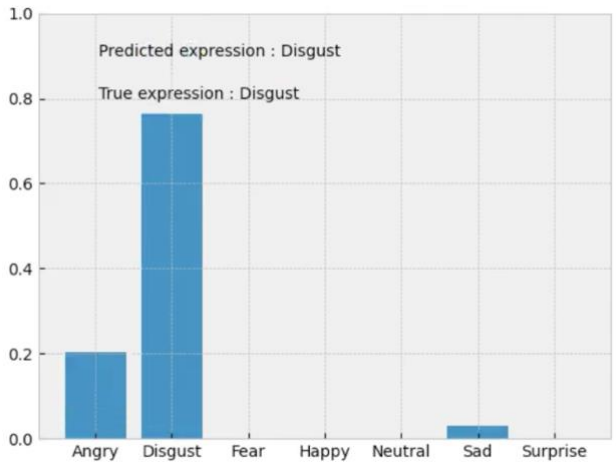

Figure 10: Illustration of the probability prediction used in our web application

The video is built by running the model iteratively on each sample and evaluating the probability prediction. Since the recording was at a rate of 50Hz, we made the evaluation corresponding to the exact same rate to simulate real-time prediction in accordance with participant real video.

Finally, we proceeded to benchmark our models and the best accuracy we could achieve was around 45% using the LightGBM model. Figure 11 and Figure 12 show the confusion matrix of our two models on our best recording.

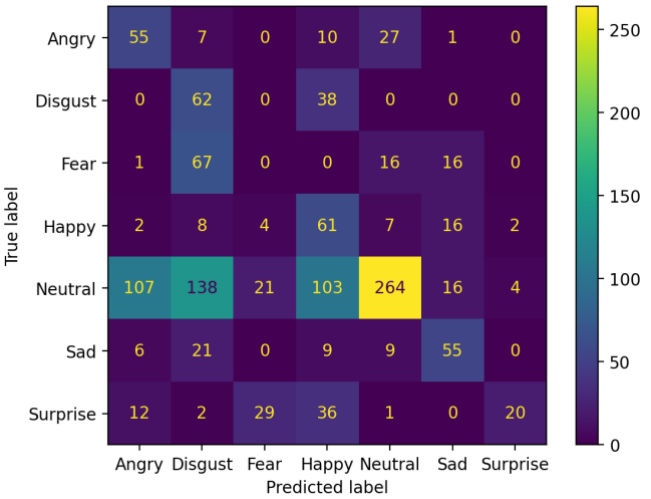

Figure 11: Confusion matrix of our random forest model achieving 41% accuracy

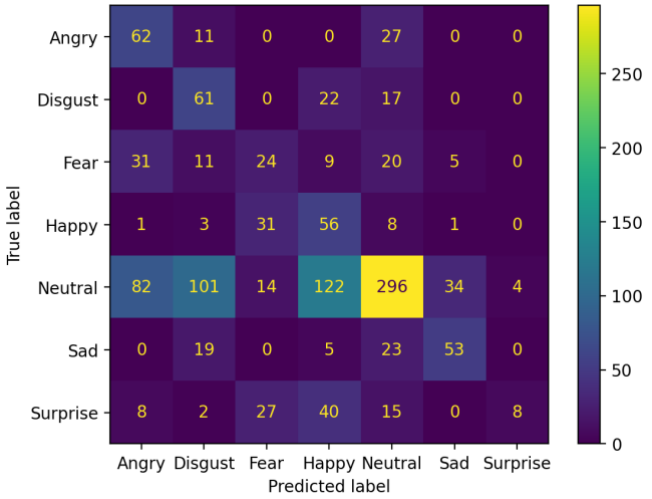

Figure 12: Confusion matrix of our LightGBM model achieving 45% accuracy

We can see from the results above that our models were less performant here than on our first database. This can be due to two different reasons. First, it has been difficult to precisely label the data and second, due to our restricted number of participants, it is hard to conclude that our models were performing poorer on this evaluation set than on our training database. Moreover, we can also note that some emotions are predicted well, and some others are not. This difference is shown in Table 4.

Emotions	Random Forest	LightGBM
Anger	55%	62%
Disgust	62%	61%
Fear	0%	24%
Happiness	61%	56%
Sadness	55%	53%
Surprise	20%	8%
Neutral	40%	45%
Total	41%	45%

Table 4: Accuracy by emotion for both models on a single recording sample

We observed that *anger*, *disgust*, *happiness*, and *sadness* are all quite well recognized in contrast to *fear* and *surpris*. One explanation to this difference comes from the fact that *surprise* and *fear* can be visually similar, depending on the participant idea of the expression. Furthermore, *surprise* is a

positive emotion and thus implied for most of the participants to perform a smile, which then made it hard for models to clearly differentiate it from *happiness*. This explains why *surprise* is often categorized as *happiness*. Regarding *fear*, we observed that it is often misclassified as *anger* or *disgust*. This misclassification comes from the fact that being a negative emotion, some participants expressed it by frowning, typical to *anger*, and lowering lips corner, typical to *disgust*, which could explain the misclassification.

Eyeglasses impact

Despite the restricted number of participants, we have been able to observe the impact of eyeglasses on the performances of our models. Indeed, by chance, one of our two participant was wearing eyeglasses. Thus, we have been able to compare the accuracy of our models on participants, with/without eyeglasses.

Table 5 below shows the accuracy achieved by the LightGBM model on both participants.

Emotions	With glasses	Without glasses
Anger	47%	62%
Disgust	100%	61%
Fear	13%	24%
Happiness	62%	56%
Sadness	2%	53%
Surprise	6%	8%
Neutral	42%	45%
Total	40%	45%

Table 5: Participant's accuracies comparison using LightGBM model

We observed that the model is less accurate when participants wore eyeglasses. This can be explained by the fact that eyeglasses induced the HMD to incorrectly detect the upper facial part expression, thus leading to misclassification. However, it is hard to conclude that eyeglasses impact accuracies due to the restricted number of samples collected.

Conclusion and future work

This project proposes a way to predict emotion in virtual reality using a new type of data that we have called "blendshapes". We introduce a new database of posed expressions recorded with virtual reality devices. The data were annotated according to the expression performed by participants among a list of 23 different expressions composed of 16 Action Units, 6 basic emotions and the neutral state. We also present a random forest model and a LightGBM model trained on this new database, allowing the recognition of basic emotion in real-time. Finally, we expose an offline evaluation of our trained models.

Despite the overall good performance of the presented models, further work could still be conducted to complete this project. First, increasing the size of the database would be more than beneficial to improve the robustness and accuracy of our models. Second, since we did not use our AUs labelled data, it might make sense to implement a new system for AUs recognition instead of directly recognizing emotions. This model could then predict emotion using predefined mapping between AUs and emotions. Finally, to improve the evaluation of our models, it would be useful to implement a real-time application in Unity 3D, using Open Sound Control for example, allowing the user to visualize his emotion instantaneously on the screen.

References

[1] P. Ekman, "An argument for basic emotions," *Cognition and Emotion*, vol. 6, no. 3–4, pp. 169–200, May 1992, doi: 10.1080/02699939208411068.

[2] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[3] G. Ke *et al.*, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," in *Advances in Neural Information Processing Systems*, 2017, vol. 30. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf

[4] P. Ekman and W. V. Friesen, “Facial Action Coding System.” American Psychological Association, Jan. 14, 2019. doi: 10.1037/t27734-000.

[5] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews, “The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops*, San Francisco, CA, USA, Jun. 2010, pp. 94–101. doi: 10.1109/CVPRW.2010.5543262.

[6] S. Li and W. Deng, “Deep Facial Expression Recognition: A Survey,” *IEEE Trans. Affective Comput.*, pp. 1–1, 2020, doi: 10.1109/TAFFC.2020.2981446.

[7] A. Chaudhary, S. Kolhe, and R. Kamal, “An improved random forest classifier for multi-class classification,” *Information Processing in Agriculture*, vol. 3, no. 4, pp. 215–222, Dec. 2016, doi: 10.1016/j.inpa.2016.08.002.

[8] L. Yao, Y. Wan, H. Ni, and B. Xu, “Action unit classification for facial expression recognition using active learning and SVM,” *Multimed Tools Appl*, vol. 80, no. 16, pp. 24287–24301, Jul. 2021, doi: 10.1007/s11042-021-10836-w.

[9] M. Ghayoumi and A. Bansal, “Unifying Geometric Features and Facial Action Units for Improved Performance of Facial Expression Analysis,” Jun. 2016.

[10] B. Farnsworth, “Facial Action Coding System (FACS) – A Visual Guidebook.” https://imotions.com/blog/facial-action-coding-system/

[11] L. Buitinck *et al.*, “API design for machine learning software: experiences from the scikit-learn project,” *arXiv:1309.0238 [cs]*, Sep. 2013, Accessed: Dec. 31, 2021. [Online]. Available: http://arxiv.org/abs/1309.0238