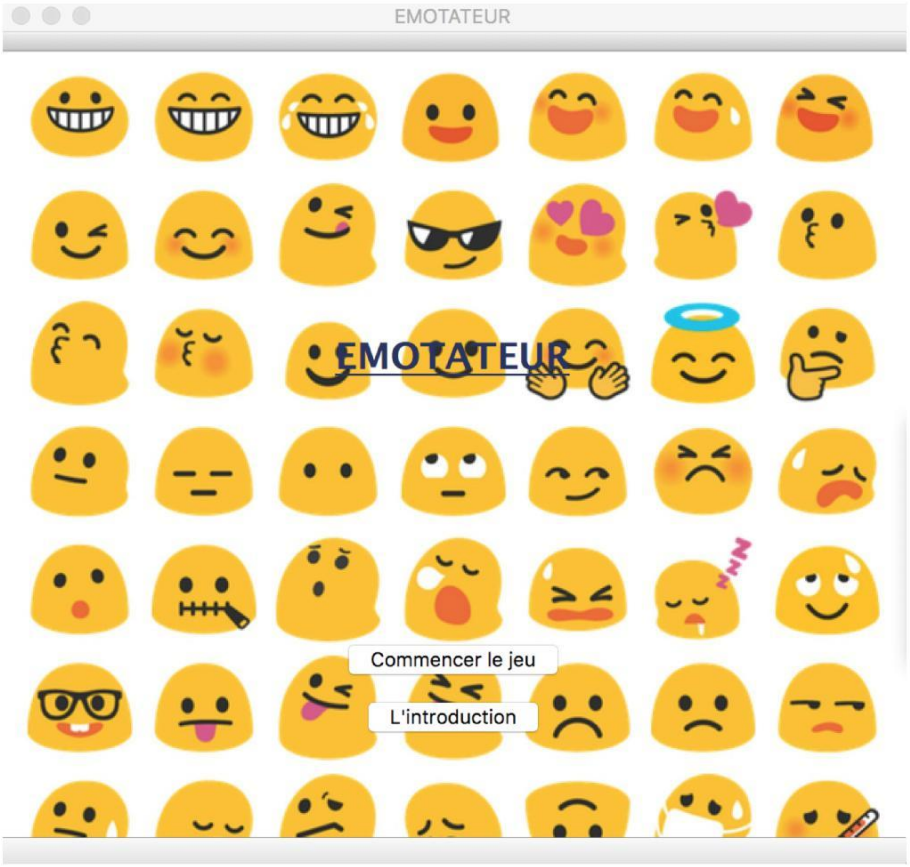
Chef d'œuvre couleur

Projet 2016-2017 《Emotateur》



ZHANG Heng
YAN Yutong
SUN Yunyun
XU Sixiang

Professeur :
Legrand Anne-Claire

SOMMAIRE :

OBJECTIFS

Dans ce projet, on voudrait créer une application amusante, dans laquelle les utilisateurs peuvent imiter les expressions des visages du modèle et obtiendraient une note, qui permet de nous indiquer la similarité entres les visages modèles et leurs visages capturés en temps réel.

De manière spécifique, les objectifs détaillés sont comme suivants :

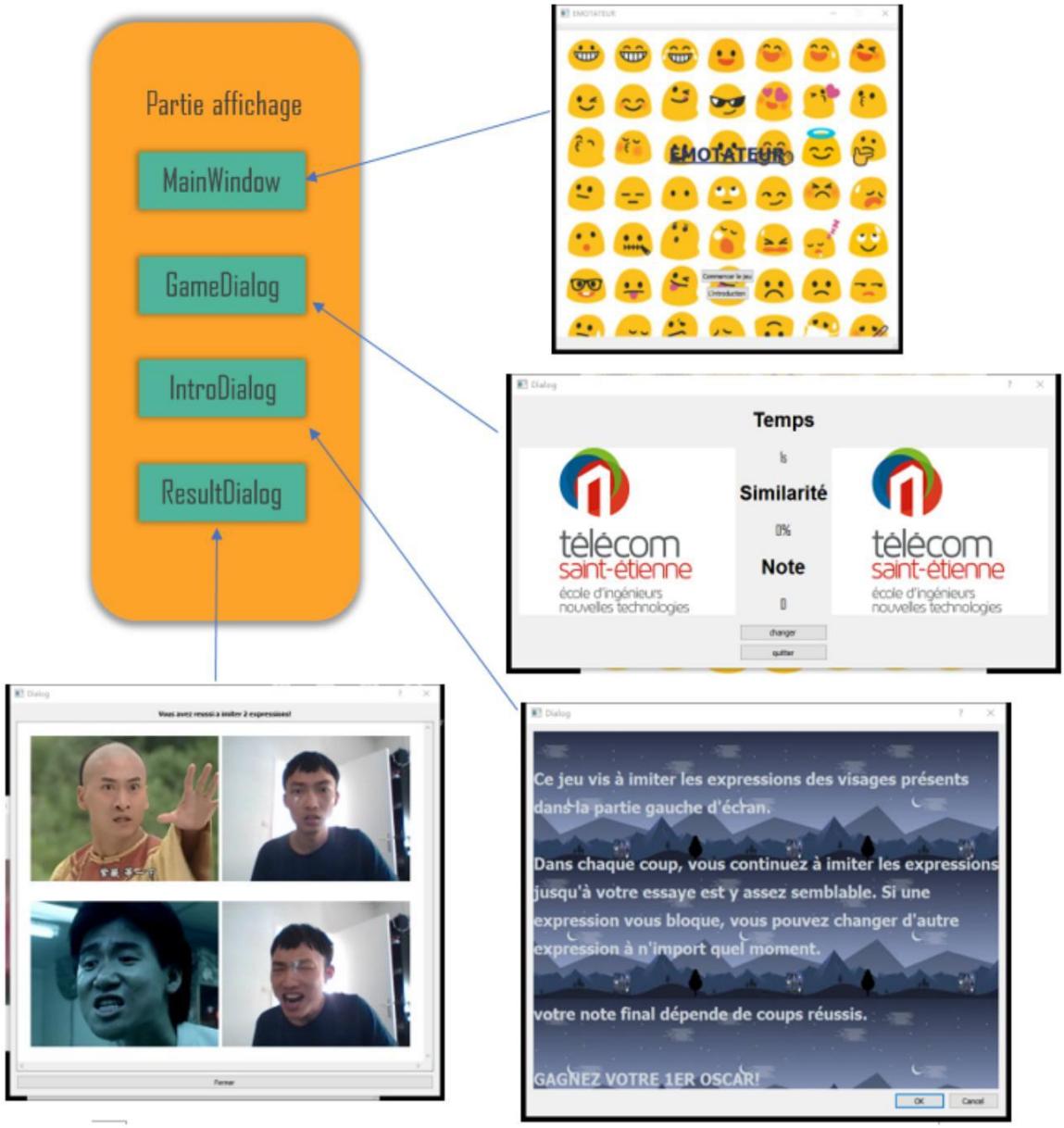
- Les visages capturés en temps réel peuvent s'afficher dans l'écran de l'application couramment.
- Il faut que Les calculs de la note de similarité prennent peu de temps, dans l'intérêt de renouvellement en temps réel. Autrement dit, il faut qu'on utilise des algorithmes prompts.
- La clé de comparaison se situe dans l'extraction des points d'intérêt. C'est ce qui définit les régions d'intérêt
- Pour obtenir une note de similarité adéquate, il faut qu'on penne une signature pertinente. Si les signatures ne sont plus assez propres, on aura besoin de les corriger et les mettre sur une forme logique.
- En plus, pour une bonne expérience en jouant ce jeu, il faut qu'on conçoive une interface pratique. Dans lequel on pourrait changer le modèle si celui-ci bloquait les utilisateurs. Et il y aura un chronomètre, car ce jeu se passe dans une période limite et les utilisateurs peuvent saisir le temps reste facilement.

RÉALISATIONS

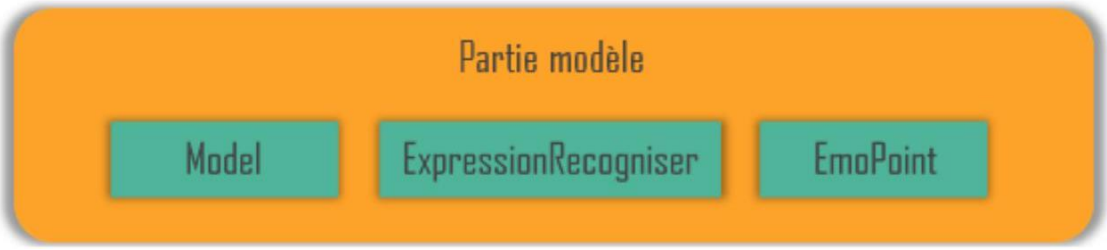
Introduction Globale

Dans notre programme, il y a 2 parties principales :
La partie d'**affichage** et la partie **modèle**.
La partie affichage contient les classes d'interface graphique.
La partie modèle s'occupe l'organisation du jeu.

La partie affichage :



La partie modèle :



La classe **Model** permet de :

1. Charger d'image modèle et d'image en temps réel
2. Récupérer la distance des deux images à l'aide de la classe ExpressionRecogniser.
3. Déterminer le résultat d'imitation selon la distance calculée.
4. Itération de ces promesses jusqu'à la fin de jeu.

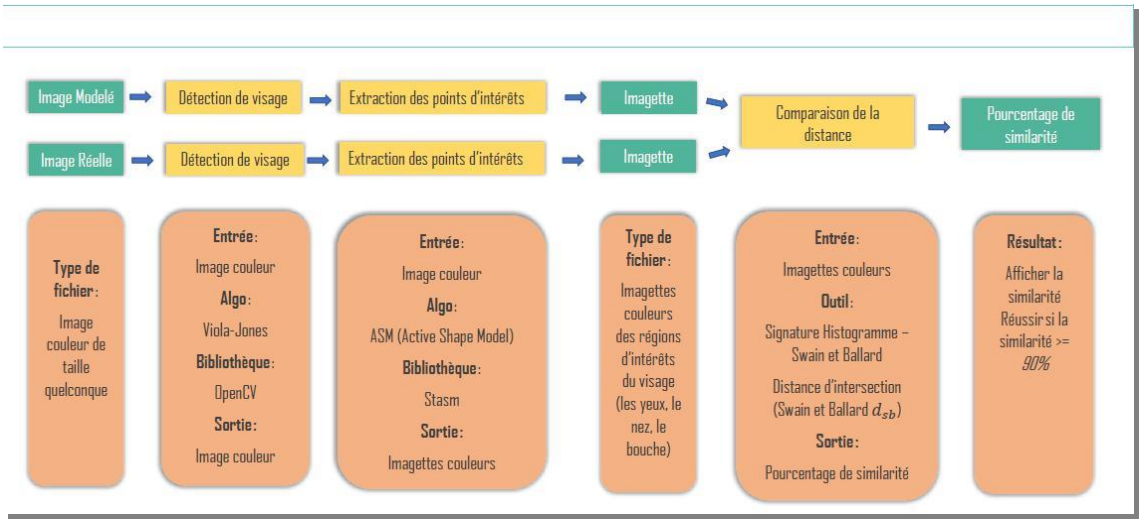
La classe **ExpressionRecogniser** permet de :

1. Détecter de visage sur les images.
2. Extraire des points d'intérêts de visage détecté.
3. Aligner des traits faciaux sur l'image en temps réel (pour éliminer l'influence de la taille et de la rotation de visage).
4. Générer deux groupes d'imagettes des traits faciaux pour les deux images.
5. Quantifier sur couleur d'imagettes (diminuer le nombre de couleur jusqu'à 64 possibilités).
6. Obtenir les histogrammes pour les deux groupes d'imagette.
7. Calculer la somme des distances d'intersection des deux groupes d'imagette.

La classe **EmoPoint** permet de :

1. Enregistrer des coordonnées des points d'intérêt extraits.
2. Réaliser de la translation et de la rotation d'image pour l'alignement d'image.

Architecture



Détection de visage

Plus précisément, nous avons utilisé l'algorithme Viola-Jones pour la détection de visage. Concrètement, cet algorithme est déjà intégré dans la bibliothèque OpenCV. Nous utilisons le classifieur de visage fourni par OpenCV, et ça va nous renvoyer les coordonnées de visage détectées.

L'extraction des points d'intérêts

En plus, nous avons utilisé l'algorithme ASM (Active Shape Model) pour l'extraction des points d'intérêts. Le principe de cet algorithme est que les modèles de forme active (ASM) sont des modèles statistiques de la forme des objets qui se déforment de manière itérative pour s'adapter à un exemple de l'objet dans une nouvelle image. Dans notre cas, les modèles sont les points d'intérêts des traits faciaux. Nous avons utilisé la bibliothèque **stasm** pour le réaliser. La bibliothèque **stasm** nous fournit les fonctions en c++ pour la localisation des points d'intérêts d'un visage.

Quantification de couleur

Nous avons quantifié les images pour simplifier l'histogramme et faciliter les traitements en suit. Nous allons travailler dans l'espace RGB. Pour chaque imagette, nous divisions chaque composante de couleur par 4 donc le nombre de couleur est délimite (pas plus de 64).

Calcul la distance entre deux histogrammes

Après la quantification, nous allons créer **l'histogramme de Swain Ballard** individuellement pour les deux imagettes. Ensuite, nous allons effectuer l'intersection d'histogramme pour mesurer la similarité.

RÉSULTATS

Dans les résultats, notre application se commence par l'affichage de la fenêtre principale (*Figure 1*).

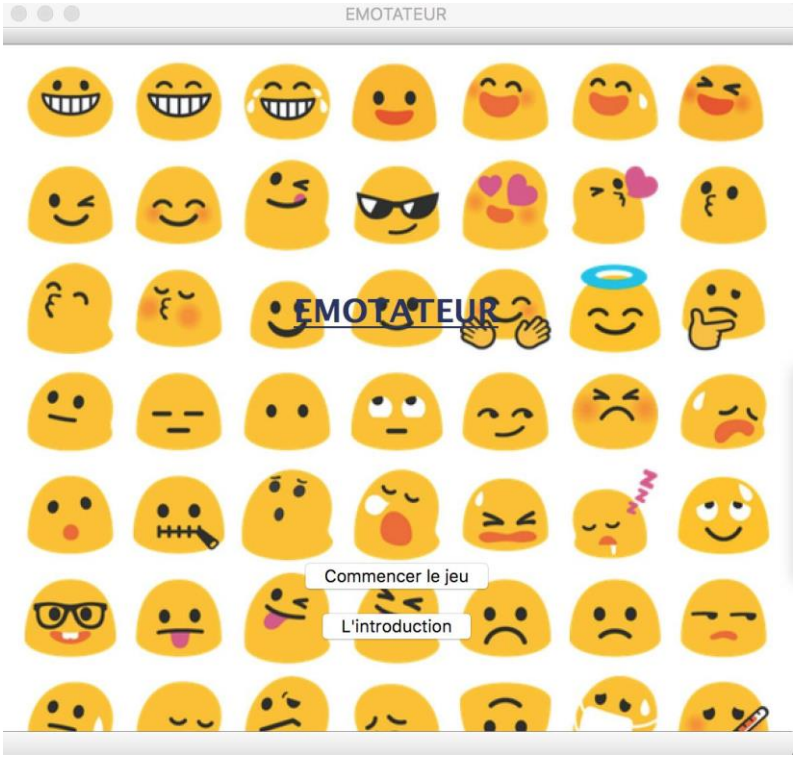


Figure 1

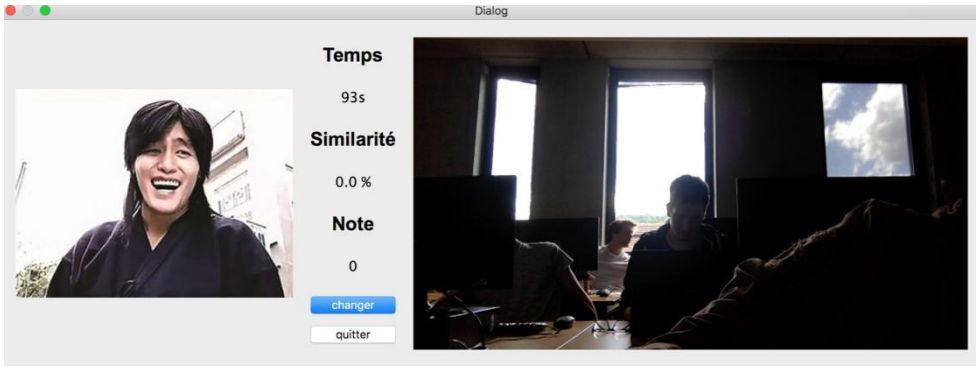
En cliquant les deux boutons, les fenêtres correspondantes vont s'apparaitre automatiquement.

Quand on clique le bouton L’introduction, une fenêtre (*Figure 2*) va s’afficher en nous expliquant les fonctionnements de cette application.

Figure 2

Quand on entre la fenêtre principale de cette application, il y aura un comptage décroissant de 3 secondes, qui permets de se préparer avant commencer réellement. Au cours des 3 secondes, le logo de TELECOM SAINT-ETIENNE va remplacer de l'image modèle et le frame actuel.

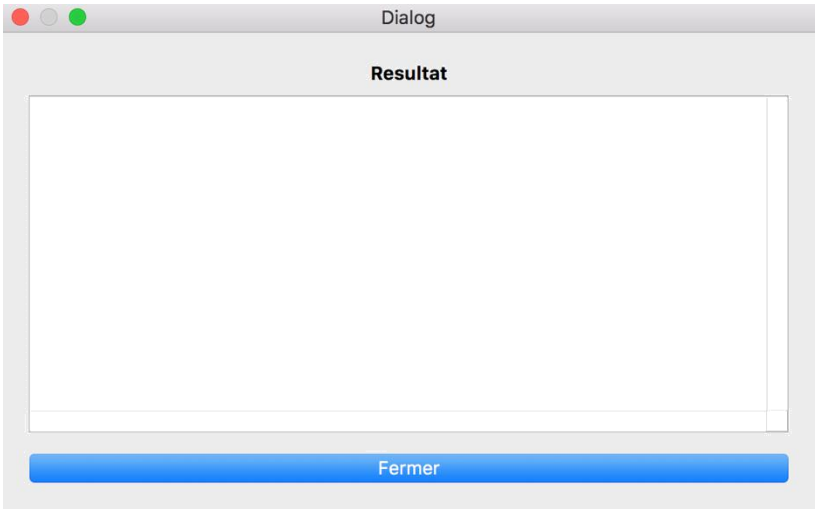
Après ces 3 secondes, le jeu qui se durera 60 secondes va réellement commencer. L'image modèle de notre base de l'image va s'afficher a gauche, et le frame actuel capture va afficher à droite. Au milieu situent les informations essentielles comme le temps restant, la note, le pourcentage de similarité.



Si l'utilisateur trouve que cette image modèle est particulièrement pénible, il peut tout de même changer une image modèle en cliquant le bouton "passer".

Si la similarité dépasse 80%, le coup de cette image modèle est estimé réussit. La note se rajoutera. Ensuite une nouvel image modèle va remplacer l'image modèle ancienne. La note finale sera la somme des images modèle qui sont estimées réussites pendant 60 secondes.

À la fin, les résultats vont s'afficher dans l'écran avec choix propres pour les opérations suivantes comme 'Recommencer', 'Sauver les résultats' etc.



ANALYSES

Notre application est basée sur la technologie de détection de visage et l'analyse d'image couleur.

Avantages et inconvénients

Avantages

1. Calcul est rapide : Tous les algorithmes que l'on a choisis sont efficaces, et ça nous permet de réaliser un système en temps réel.

2. Notre algorithme éliminer les facteurs sans aucune relation avec l'expression faciale, comme le couleur de la peau, le taille et la rotation de visage.

3. Notre système est indépendant de l'appareil de réception des données, du coup ça va marcher sur la caméra de résolution quelconque.

4. Notre système est extensible, vous pouvez rajouter les photos que vous voulez imiter dans le répertoire <imageBase>.

Inconvénients

1. Nous avons utilisé la distance entre deux histogrammes pour déterminer les différences entre deux expressions, c'est peu confiant, du coup il faudra tester d'autres algorithmes afin de trouver l'algorithme le plus convenable.

2. Notre système n'est pas robuste par rapport à l'éclairage.

3. Notre système ne permet pas enregistrer les résultats d'imitation par les utilisateurs dans un fichier.

Amélioration :

Considérons les inconvénients, nous pouvons améliorer notre projet en :

1. Trouvant un algorithme plus convenable
2. Ajoutant la fonction d'enregistrement du résultat d'imitation.

Application

1. Ce système peut être développé dans un jeu de reconnaissance d'expression plus complet, contenant une variété de différentes bibliothèques d'expression, l'utilisateur peuvent choisir différents niveaux de difficulté en choisissant la bibliothèque correspondante, et ils peuvent créer leur propre bibliothèque d'expression. Les résultats d'imitation peuvent être enregistrés, partagé, et commenté.

2. De l'autre côté, nous pouvons développer la technologie de détection d'expression dans notre application et l'appliquer dans le domaine de reconnaissance d'émotion selon l'expression.

CONCLUSION

Organisation

En général, notre équipe est efficace au cours du projet. Il n'y avait pas de problème de langue et nous pouvons discuter des problèmes du projet sans soucis de mal compréhension. En plus, c'était comme d'habitude que nous faisions d'un réunion chaque semaine dans laquelle il n'y avait pas de cours couleur même si c'était en vacances.

Pilotage

A Chaque réunion, nous présentions les réalisations personnelles pendant la semaine dernière. Si la réunion était dans le cours, nous aussi discutions comment intégrer des conseils de l'enseignant dans notre projet. A la fin de cours, chacun remplissait sa page d'avancement personnel du fichier « chef d'œuvre ».

Après avoir combiné la partie Modèle et la partie d'affichage, nous partagions le code sur GitHub afin que chacun de l'équipe puisse améliorer le projet et le code modifié puisse être mis à jour rapidement parmi l'équipe.

Répartition de tâche

En gros, Xu et Sun étions responsables à l'interface d'utilisateur et Zhang et Yan étions responsables à la partie de traitement d'image.

D'un point de vue de gestion de projet, c'est une moyenne efficace car tout le monde a des travaux à faire et la frontière entre l'interface et l'image semble claire. Par exemple, si je suis un membre responsable à la partie d'interface, je n'ai pas besoin qu'une classe définie par les responsables de images. Quand ils ont fini la classe, je l'ai intégré et c'était tout bon.

Malheureusement, d'un point de vue de cours couleurs, les responsables de la partie d'affichage perdent la chance d'apprendre plus en couleurs. Il semble que notre projet s'avançait bien mais c'est le fait que presque à la fin du projet, l'équipe d'interface a commencé à comprendre l'algo de traitement d'image.

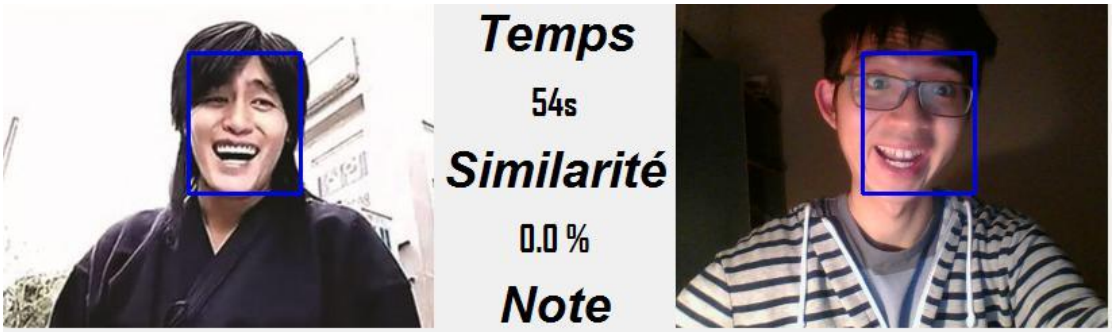
Réussites et difficultés rencontrées

Qu'avons-nous réussi à faire

Dans un premier temps, nous avons conçu l'architecture de logiciel. A la base de cet architecture, nous avons réparti les tâches générales. Ensuite, nous avons cherché un algorithme agréable et son code en c++ qui permet de détecter le visage humain. Après avoir compris le principe de l'algorithme, nous avons mis en place l'architecture de traitement d'image exigée par l'enseignante.

Dans un deuxième temps, nous avons commencé à coder l'interface et implémenté le code de l'algorithme dans notre projet. Rapidement, nous avons fini les deux parties et les avons intégrées. En ce moment-là, la fonction principale a été réalisées.

Enfin, nous étions toujours en trains d'améliorer et ajouter des fonctions. Initialement, l'imitation de l'utilisateur avait été limitée dans un cadre sur la vidéo. Nous avons ajouté une matrice de projection afin que les points d'intérêt de visage d'utilisateur soient projetés sur la visage de modèle. Autrement dit, après l'amélioration, l'utilisateur peut imiter une émotion n'importe où sur la vidéo. La comparaison est équipement correcte même si le visage a une rotation.



Premier version (détection dans le cadre bleu)

Quelles ont été nos difficultés majeures

La difficulté principale fut la réalisation des idées concertantes aux connaissances de couleur. Ce que nous avions fait sur la comparaison avant était sur la partie spatiale. Nous souhaitions une combinaison entre la comparaison spatiale et celle de couleur. En fait, pour la comparaison de couleur, nous avons conçu un algorithme qui tout d'abord quantifié les couleurs limitant par les points d'intérêt et calculer la distance des deux histogrammes de Swain-Ballard. Mais évidemment, cette comparaison n'est pas fiable au cause de, par exemple, l'éclairage.

Perspectives

Du début à la fin du projet, nous étions toujours passionnés à réaliser cette application car nous croyons que l'imitation d'expression est une bonne idée.

Tout d'abord, elle est extensible. Maintenant, beaucoup de personnes deviennent populaire par ses expressions. Et des expressions amusantes apparaissent chaque jour sur l'internet. Ils sont des ressources inépuisables qui peuvent être utilisées comme les émotions des modèles

En outre, nous la considère comme un jeu social. Tout le monde a son unique propre émotion. Il peut produire les émotions de modèle par lui-même. Autrement dit, au futur, nous pourrons ajouter une fonction qui permet aux utilisateurs d'imiter des émotions produise par ses amis ou les inconnus.

En plus, pour l'instant, le résultat de comparaison de couleur n'est pas fiable. Il y a encore des améliorations sur cette partie. Nous pouvons également mettre en œuvre les algorithmes de détection de visage que nous allons apprendre au futur.