Rapport de projet individuel
Les yeux nous trahissent

Projet de second semestre de Master Informatique

Janvier à Mai 2015

Auteurs :

— Elliot VANEGUE
— Gaëtan DEFLANDRE

Encadrants :
— Marius BILASCO
— José MENNESSON
— Benjamin ALLAERT

Remerciement

Résumé

Il existe de plus en plus d'application qui requiert la détection et le traitement d'information présentent dans le visage. Dans le cadre de nos étude, nous participons à un projet visant à amélioré la normalistion d'un visage extrait d'une image. Pous cela, nous travaillons sur une application réalisé pas l'équipe FOX qui détecte un visage à partir de l'algorithme de Viola et Jones, et qui effectue des traitement afin d'obtenir une image stable du visage quelque soit l'orientation ou la distance de la tête de l'utilisateur.

Pour améliorer cette normalisation, nous effectuons des recherches sur les solutions possible pour détecter les contours d'une image. Plusieurs piste sont envisageable comme l'algorithme de Canny qui détecte les contours en analysant les variations de fréquences présentent dans l'image. Notre solution final utilise une image extrainte d'un canal du modèle colorimétrique YCbCr, qui permet de s'abtraire de la luminosité de la vidéo et de la couleur de peau de l'utilisateur.

Après de nombreux traitements, sur la première chrominance du modèle YCbCr, nous obtenons une forme correspondant à l'oeil, ce qui nous permet de calculer le centre de l'oeil avec le barycentre de cette forme.

Table des matières

Table des figures

1 Introduction

Pour nos études en master informatique, nous participons à un projet proposé par une équipe de recherche afin de découvrir ce milieu. Étant intéressés par le domaine de l'image et de la vision, nous avons sélectionné ce projet. En effet, nous pensons nous orienter vers le master IVI [1] de Lille 1. Participer à ce projet nous permet de confirmer ce choix et nous apporte des connaissances utiles pour la suite de notre parcours. La réalisation de ce projet, nous donne une bonne expérience pour notre avenir professionnel dans cette voie.

1.1 Présentation

Pour ce projet, nous sommes amenés à améliorer la détection des yeux à l'intérieur d'un visage à partir d'un flux vidéo. Avec l'évolution du matériels (processeurs, caméras ...), ce procédé se démocratise et prend sa place dans certaines applications temps réel. Cette fonctionnalité peut être intégré dans les domaines :

— de la sécurité, ce qui permettrait de détecter des comportements inhabituels chez un individu.
— du loisir, pour les jeux vidéo ou encore pour une interaction plus intuitive avec un ordinateur.
— de la publicité, pour que celle-ci corresponde au besoin des individus.

Ce type d'application utilise différentes technologies, mais certaines sont souvent trop intrusives et demandent à l'utilisateur d'utiliser du matériel assez spécifique. Or le but des algorithmes actuels est de faire en sorte que l'utilisateur ne se rende même pas compte qu'il utilise ce type de technologie. Les applications plus modernes effectuent des algorithmes de reconnaissance de formes, tels que l'algorithme de Viola et Jones, dans le but de détecter un visage à partir de simples caméras. Cela permet de mieux intégrer ces applications afin qu'il n'y est aucune contrainte pour l'utilisateur.

Dans le cadre de ce projet, nos travaux sont utilisés afin d'effectuer de la reconnaissance d'émotions lors d'applications du type e-learning [2]. Ce procédé permettrait de détecter si un cours intéresse ou non les élèves afin de pouvoir l'améliorer. Si on voit sur une partie du cour que de nombreux élèves ont présentés des signes de fatigue ou qu'ils n'écoutaient plus, il sera alors possible aux enseignants de modifier cette partie.

1.2 Contexte

Pour la réalisation de ce projet, nous travaillons avec l'équipe FOX qui étudie l'analyse du mouvement à partir de vidéos. Leurs recherches portent sur l'extraction du comportement humain depuis les flux vidéo. Leurs travaux sont divisés en quatre grands domaines : le regard, qui est la partie sur laquelle nous travaillons, l'événement, l'émotion et la reconnaissance de personnes. La grande majorité de leurs travaux sont des applications temps réel, ce qui permet d'avoir un niveau de réactivité très élevé.

Le projet sur lequel nous travaillons est basé sur les travaux des membres de l'équipe qui

1. Image Vision Interaction
2. Formation en ligne

se sont concentrés sur la détection de visages et des yeux. Cette détection permet ensuite de normaliser le visage, c'est-à-dire d'avoir une image statique contenant ce visage afin de réaliser des traitements sur celle-ci. C'est traitement peuvent par exemple être la détection de la fatigue ou de la concentration d'un individu.

Puisque la normalisation du visage repose sur la détection du visage et des yeux, il est nécessaire que cette détection soit correct et précise, afin d'avoir une normalisation du visage cohérente.

1.3 Problèmatique de l'existant

Les algorithmes utilisés dans l'application sont limités et ne permettent pas de faire un suivi correct dans toutes les situations d'une application temps réel. De nombreux cas ne sont pas traités dans ce type d'algorithme comme par exemple l'orientation du visage. Lorsque l'utilisateur effectue une rotation de la tête, les algorithmes utilisés ne sont pas capables de suivre ce mouvement. Il faut donc, grâce à différents axes présents dans le visage, détecter ce mouvement afin de garder une région d'intérêt correcte pour les traitements suivants.

De plus, les points représentant le centre des yeux ne sont pas parfaitement stables. Ce décalage a de nombreuses conséquences sur les traitements effectués par l'application, car l'application se base sur les ombres provoquées par le mouvement de certains muscles du visage. Il est donc primordial que la position de ces muscles soit stable, pour ne pas les confondre lors des traitements. Et pour cela il faut que l'image normalisé du visage soit stable, afin que l'image du visage soit invariant au changement de repère.

Actuellement l'application n'arrive pas à localiser le centre des yeux lorsque ceux-ci sont fermé. Cela implique que la normalisation du visage ne se fait pas correctement et que les points du visage servant à faire des traitement ne sont pas parfaitement stable.

La qualité de la vidéo et sa taille peuvent également être problèmatique dans les traitements. Afin de pouvoir valider nos travaux, nous avons accès à une base de vidéo possèdant les véritables points du centre de l'oeil. Malheureusement, ces vidéos sont relativement ancienne, la qualité n'est pas très bonne et elles sont plus petite que les vidéos sur lesquels l'application a été testé. Les calcules actuel du centre de l'oeil ne fonctionne pas très bien sur ces vidéos.

La majorité des applications de l'équipe FOX reposent sur la détection du visage et des yeux. Il est donc important de remédier à ces différentes problèmatiques.

1.4 Objectif du projet

Le première objectif est donc de stabiliser la position du centre de l'oeil, afin d'avoir une normalisation de l'image totalement stable. Pour cela nous utilisons les calcules réalisé précédement par l'équipe afin de récupérer une ROI de la zone péri-oculaire. Puis nous affinons la localisation du centre de l'oeil avec de nouveau traitement.

Le second objectif est de calculer le centre de l'oeil lorsque celui-ci est fermé. Cette étape est plus complexe, car contrairement à un oeil ouvert, l'oeil fermé à la même couleur que la peau. Il faut donc réussir à différencier ces deux situations afin d'appliquer le bon traitement.

Pour répondre aux problématiques du projet, nous allons d'abord voir la structure actuelle du projet, ainsi que les algorithmes utilisés. Puis, nous détaillerons nos démarches pour résoudre les problématiques et nous expliquerons la solution finale que nous avons implémentée dans l'application.

2 Application existante

2.1 Présentation de l'application

L'application sur laquelle nous travaillons est écrite en C++ avec l'utilisation de QtCreator. Le programme a été développé par l'équipe Fox et par des étudiants. Cette application il a fait l'objet de divers améliorations auparavant.

2.2 Architecture

L'application fournis est orientée objet, celle-ci est découpé en Business Objects [3]. Ces Business Objects sont hiérarchisé dans une pyramide, les objets métiers en haut de cette pyramide sont les plus abstrés et utilise les resultats de l'étages infèrieur. Les objets métiers en bas de la pyramide sont les plus proche du flux vidéo brut.

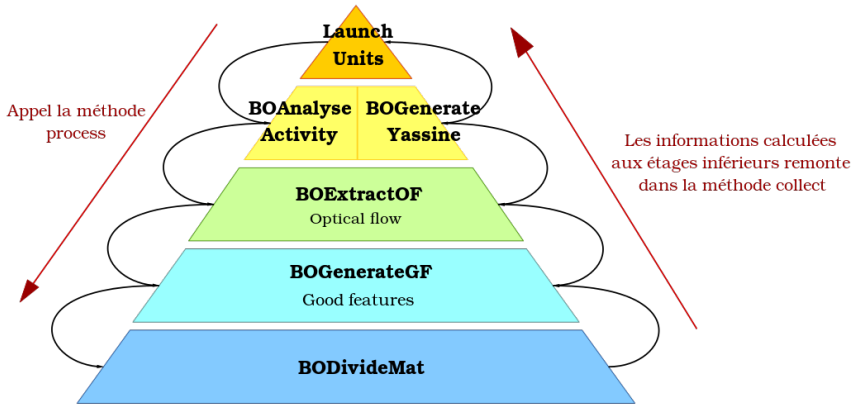


FIGURE 1 – Processus d'extraction des informations pyramidal sur le flux vidéo

2.3 Reconnaissance du visage : Viola et Jones

L'application est divisée en deux parties. La première recherche le visage grâce à l'algorithme de Viola et Jones et la seconde recherche les yeux dans la région délimitée précédemment.

L'algorithme de Viola et Jones[2] est une méthode qui a été créée pour la reconnaissance de visage dans une image. Cette méthode s'est par la suite généralisée à toutes sortes d'objets. L'algorithme nécessite une base de connaissances composée des caractèristiques de l'objet recherché. Cette base de connaissances est utilisé dans un apprentissage supervisé, c'est à dire que l'algorithme a besoin de données représentant l'objet à détecter pour classifier les caractèristiques de celui-ci.

3. les objets métier s'occuppent d'une tâche précise

Cette algorithme est basé sur des caractèristiques pseudo-Haar qui crée des masques rectangulaires et adjacentes dans différente zone de l'image. Chaque masque calcule l'intensité des pixels qu'il contient, puis l'algorithme fait la différence entre les masque blanc et les masque noir. Cette méthode va permettre de détecter des contours ou des changements de texture.

FIGURE 2 – Exemple de caractèristiques pseudo-Haar utilisé pour l'algorithme Viola et Jones

Pour améliorer les perfomance de leur algorithme, Viola et Jones utilise la méthode Adaboost. Son principe est de séléctionner les caractèristiques les plus performante pour la détection de l'objet grâce à un calcule de probabilité utilisant l'entropie [4] des données.

2.4 Suivi des yeux

4. valeur mesurant l'incertitude d'une données

3 Recherche de solution

3.1 L'algorithme de Canny

La première solution que nous testons est le filtre de Canny. Nous avons pu voir ce filtre durant nos cours et nous trouvons que c'est une piste intéressante, car cette algorithme détecte les contours présents dans une image.

3.1.1 Version de base

Le filtre de Canny a été créé en 1986 dans le but d'améliorer les résultats du filtre de Sobel. Le principe du filtre est d'utiliser deux seuils, un seuil haut et un seuil bas. L'algorithme commence par sélectionner les pixels supérieurs au seuil haut, puis recherche à partir de chaque pixel en dessous du seuil haut les pixels qui sont au-dessus du seuil bas dans son voisinage. Chaque voisin qui est entre les deux seuils appartient à un contour, on passe donc sa valeur à 255 pour le prendre en compte dans celui-ci.

Nous utilisons cet algorithme dans les régions autour des yeux afin de délimiter le contour de chaque oeil. Nous testons cette algorithme avec différent seuil afin de voir si celui-ci peut être intéressant dans notre situation. On voit sur les résultat que si le seuil est trop bas l'oeil n'est pas détecter, et si le seuil est trop haut les rides de la personne dans la vidéo apparaissent. Afin d'avoir un seuil adaptatif, nous prenons la valeur moyenne des pixels de l'image, ce qui nous donne un résultat où le sourcil apparait et ou une partie de l'oeil est détecté.

FIGURE 3 – Image de test et résultat de l'algorithme de Canny avec une moyenne des pixels

On voit que cette méthode, n'est pas optimale et supprime trop d'informations. De plus, l'algorithme prend en compte les ombres présentes dans l'image. Nous devons adapté cette méthode afin de mieux détecter les contours dans les zones que nous souhaitons.

3.1.2 Avec une moyenne de pixels sur des parties d'image

Plutôt, que de prendre la moyenne des pixels sur l'ensemble de l'image de la zone péri-oculaire, nous segmentons l'image en plusieurs zones dans lesquels nous calculons la moyenne. Puis nous effectuons l'algorithme de Canny sur chacune de ces zones. Le résultat nous fournit bien une image dans laquelle l'oeil est détecté.

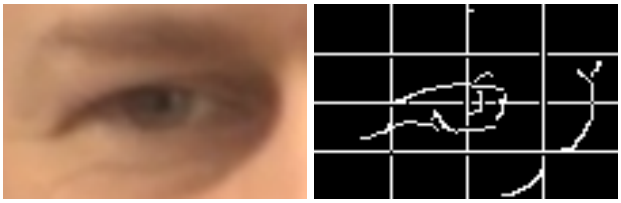


FIGURE 4 – Image de test et résultat de l'algorithme de Canny avec division de l'image en 16

Avec cette méthode, nous obtenons une image bruité avec les bords de chaque rectangle utilisés pour le calcul précédent, car l'algorithme de Canny ne prend pas en compte les bords de l'image qu'il traite. Avec ce bruit nous ne pouvons pas effectuer une recherche de l'oeil dans l'image. Il nous faut donc trouver une solution pour supprimer les traits sans supprimer les contours présents dans l'image résultante.

3.2 Ouverture des sous-images

Pour résoudre le problème précédent, nous utilisons deux opération morphogique qui sont la dilatation et l'érosion d'une image.

La dilatation d'une image est une opération morphologique [5] qui utilise un élément structurant afin d'effectuer une convolution de l'image. Si l'un des voisins du pixel traité est blanc alors celui-ci sera blanc, sinon il sera noir. Cette opération a, en général, pour effet d'élargir une forme et de combler les trous qui peuvent être présent dans celle-ci. Nous avons également l'opération inverse qui est l'érosion, qui va diminuer la surface de la forme. Lorsqu'une dilatation est effectuée sur une image en niveau de gris, le pixel traité prend la valeur maximum parmi ses voisins, et le minimum pour l'érosion.

L'ouverture est la combinaison des deux opérations morphologiques vues précédemment, en commençant par l'érosion puis la dilatation. Cela a pour effet de fusionner des formes si elles sont proches (dans ce cas il y a des chances que ce soit la même forme) et de diminuer la largeur de la forme pour s'approcher de la forme réel.

Afin de supprimer le bruit présent dans l'image de l'oeil après utilisation du filtre de Canny, nous effectuons une ouverture de chaque sous image.

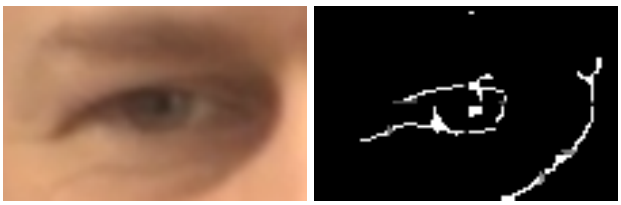


FIGURE 5 – Image de test et résultat de l'algorithme de Canny avec dilatation puis érosion des sous-images

Grâce à ce procédé, nous obtenons une image sans la grille que nous avions précédemment. Cependant, le bruit du aux ombres et aux rides ne permet pas encore d'effectuer des traitement

5. filtre non linéaire

pour localiser le centre de l'oeil. Nous avons un second problème qui est la couleur de peau de l'utilisateur. Nous avons appliqué notre méthode sur des personnes ayant des couleurs de peau différentes et les résultats n'étaient pas exploitable (voir annexe p18).

Afin de supprimer les bruits du aux ombres présentent dans l'image, nous décidons de travailler sur d'autre modèle colorimétrique comme le HSV [6] dont la composante « saturation » peut nous permettre de ne plus prendre en compte les changements importants de luminance.

3.3 Les modèles Colorimètrique

Un modèle colorimètrique est une façon de représenter numériquement les couleurs. En général on sépare ces valeurs en trois groupe, les canaux. Le modèle colorimètrique le plus connus est le RGB, où on retrouve les canaux rouge, vert et bleu. Les canaux peuvent porter d'autre information comme nous allons le voir dans la suite.

3.3.1 Le modèle HSV

Le modèle colorimètrique HSV est composé de la teinte [7], la saturation [8] et la valeur [9]. La *teinte* correspond à la forme la plus pur des couleurs. La *saturation* est l'intensité de la couleur, plus cette valeur est faible et plus la couleur semble fade. Et enfin la *valeur* correspond à la luminosité de la couleur, plus cette valeur est petite et plus la couleur est sombre.

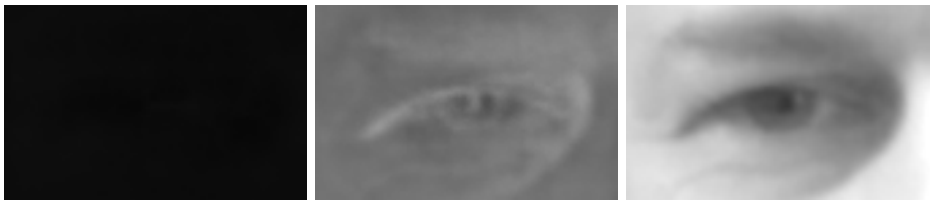


FIGURE 6 – Décomposition du modèle HSV

Etant donné que notre problème actuel est l'ombre présente dans l'image, nous séparons les trois canaux du modèle HSV. Cela nous permet de garder la plus approprié pour nos traitement. Nous ne prendrons pas le troisième canal puisqu'il représente la luminosité et que notre problème actuel est du aux ombres présentent dans l'image. Le canal le plus approprié serait la *saturation*. Malheureusement, lorsque nous appliquons le filtre de Canny sur ce canal, le résultat ne nous permet pas de détecter l'oeil.

FIGURE 7 – Résultat du filtre de Canny sur le canal saturation

6. Hue Saturation Value
7. Hue
8. Saturation
9. Value

Ce modèle colorimétrique ne nous permet donc pas de détecter l'oeil, car aucun des problèmes que nous avons rencontré précédemment n'a été résolu. Cependant, après plusieurs recherche, nous avons trouvé une solution qui pourrait résoudre le problème de la couleur de peau grâce à un autre modèle colorimétrique.

3.3.2 Le modèle YCbCr

Nous avons testé un second modèle colorimètrique suite à l'étude des travaux de Evangelos Skodras et Nikolaos Fakotakis[1]. En effet, leur étude montre que les différentes couleurs de peau de l'Homme ont à peu près la même valeur dans deux des composantes du modèle YCbCr.

On peut voir sur le résultat de la décomposition de l'image que le canal représentant la première chominance met en évidence la pupille de l'oeil.

FIGURE 8 – Décomposition du modèle YCbCr

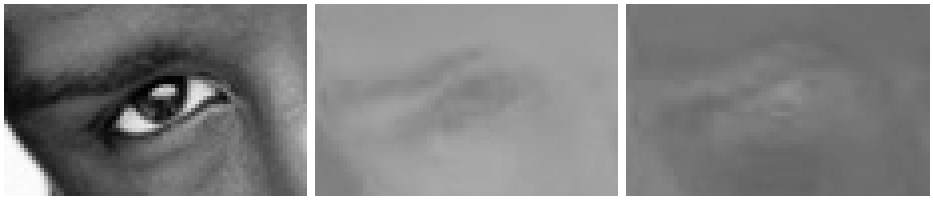


FIGURE 9 – Décomposition du modèle YCbCr avec une couleur de peau plus foncée

Pour obtenir la valeur des différentes composantes de cet espace colorimètrique à partir d'une image RGB il faut effectuer les calculs suivant :

$$Y = 0.299R + 0.587G + 0.114B$$

$$Cb = -0.1687R - 0.3313G + 0.5B + 128$$

$$Cr = 0.5R - 0.4187G - 0.0813B + 128$$

Nous appliquons ces calculs sur la couleur la plus représentatif de la peau des trois modèles que nous avons pris précédemment afin de voir si les résultats confirment le fait que la première chrominance est similaire quelque soit la couleur de peau.

couleur de peau	rouge	verte	bleu	luminance(Y)	chrominance 1(Cb)	chrominance 2(Cr)
blanc	234	171	130	185	97	163
bronzé	175	112	95	129	109	161
asiatique	202	143	111	157	102	160

FIGURE 10 – Tableau des valeurs des couleurs de peau dans les espaces RGB et YCbCr

Nous pouvons voir avec ce tableau que la valeurs des deux chrominances sont très proche quelque soit la couleur de la peau. Pour la suite des traitements, nous décidons de garder la chrominance 1 qui semble varier un peu moins avec le changement de luminosité. En prenant le canal de la première chrominance et en prenant un seuil qui semble correcte de vue pour la binarisation, nous obtenons une image comportant la forme de l'oeil au bon endroit.

FIGURE 11 – Résulat après traitement sur la chominance 1

Il nous faut maintenant trouver les bons paramètres afin que le seuil soit correcte quelque soit la luminosité de la vidéo.

3.4 Seuil pour la binarisation de l'image

3.5 Sélection de composantes convexes avec des blobs

Maintenant que nous obtenons une image binarisé comportant la forme de l'oeil, nous pouvons déterminer ou est positionné son centre. Pour cela nous avons choisi l'utilisation de blob afin de résoudre un second problème que nous avons rencontré. Lorsque nous effectuons les traitements précédent sur la vidéo, nous avons parfois des éléments qui perturbe la détection de contour comme des montures de lunette ou les cheveux de la personne filmé. Ces éléments peuvent faire partie des formes que nous récupérons lors de la binarisation.

Les blobs nous permettent d'étiqueté chaque composantes convexes présentent dans l'image en annotant chaque pixels de celle-ci. Si un pixel n'a aucun voisin d'annoté on lui donne un nouveau numéro et si l'un de ces voisins est annoté alors il prend le numéro de ce voisin. Nous obtenons ainsi des enveloppes convexe sur lesquel nous vérifions qu'elles respectent bien certaines contrainte.

Dans notre cas nous avons posé les conditions suivantes :
- Le blob de l'oeil est censé être convexe, il ne devrait donc pas y avoir de trous dans le blob
- S'il y a plusieurs blobs alors l'oeil devrait être celui qui est le plus proche du centre étant donné que la ROI doit être centré sur l'oeil.

Ces conditions nous permettent de supprimer les éléments perturbateur qui étaient présent dans l'image binaire. Nous obtenons donc une forme représentant l'oeil, il suffit maintenant de calculer le centre de cette forme pour obtenir la localisation du centre de l'oeil. Pour cela nous utilisons le calcul du barycentre.

4 Implémentation de la solution

Plusieurs des solutions que nous avons testé ne nous ont pas permis d'atteindre l'objectif du projet. Par exempls, nous n'avons pas retenu l'algorithme de Canny, qui malgrès plusieurs traitements sur l'algorithme de base, ne permettait pas de s'abstraire de la luminosité de la vidéo. Nous allons donc voir en détail les solutions qui ont été utilisé pour atteindre nos objectif.

4.1 Solutions utilisées dans l'application

Lorsque nous analysons une vidéo pour en extraire le centre de l'oeil, nous sommes confronté à certaine contrainte. Cependant, la majorité d'entre elles dépendent de la gestion de la couleur de l'image que nous analysons. C'est pourquoi l'algorithme de Canny n'était pas efficace, car celui-ci était appliqué à une image en dégradé de gris et donc subissait les effets de la luminosité de la vidéo.

Nous avons donc choisi d'utiliser un modèle colorimètrique différent afin d'en extraire une des composantes. Notre solution utilise donc le canal de la première chrominance du modèle colorimétrique YCbCr ce qui permet de ne plus prendre en compte la couleur de peau et de diminuer l'effet de la luminosité dans l'image que nous traitons.

Une fois la binarisation effectué, nous devons détecter la forme de l'oeil. Pour cela, nous utilisons les blobs, qui nous permette non seulement de récupérer des enveloppes convexes présent dans l'image, mais aussi de déterminer quel enveloppe est celle de l'oeil. Nous calculons ensuite le barycentre de cette forme afin d'obtenir le centre de l'oeil.

Cette solution ne gère pas complètement le cas ou l'oeil de l'utilisateur est fermé. Dans le cas ou notre algorithme n'arrive pas à calculer le centre de l'oeil, nous récupérons les coordonnées du point qui était calculé par la méthode de l'équipe. Ainsi, dans le cas ou nous ne détectons aucun blob, nous obtenons tout de même un point, sauf si la méthode précédente ne fonctionne pas non plus.

4.2 Comparaison des résultats obtenu avec les anciens résultats

4.3 Améliorations envisageables

Tout les objectif n'ont pas été remplit, car la recherche de solution pour l'optimisation du calcul du centre de l'oeil nous a pris beaucoup de temps à cause des pistes qui n'ont pas aboutit. Notre point obtient de meilleur résultat que celui utilisé précédement, mais il reste toujours un écart avec la vérité terrain. Il est donc encore possible d'améliorer notre solution.

Le second objectif sur la localisation de l'oeil lorsque celui-ci est fermé n'est pas aboutit. Notre solution nous permet détecte parfois un point dans ce cas, mais ce qu'il trouve n'est pas le centre de l'oeil mais le centre des cils qui sont plus visible lorsque les yeux sont fermés.

5 Conclusion

5.1 Bilan sur l'application

5.2 Bilan des compétences

Annexes

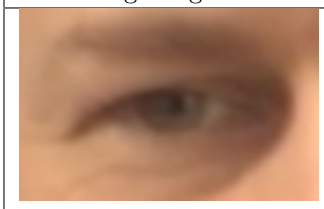
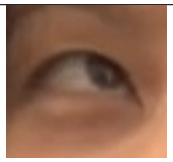
image originale	image après filtre de Canny
	
	
	

FIGURE 12 – Résultats du filtre de Canny sur des personnes de couleur de peau différentes

Références

[1] Evangelos Skodras and Nikolaos Fakotakis. 2012 ieee 24th international conference on tools with artificial intelligence an accurate eye center localization method for low resolution color imagery.

[2] Paul Viola and Michael Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57 :137–154, 2004.