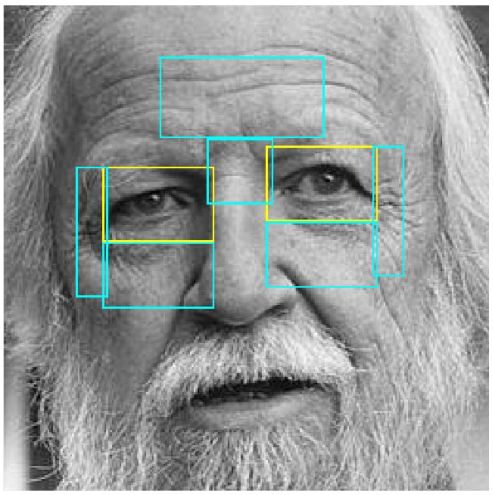
Chargée de TP : Cindy CAPPELLE

Automne 2018



Estimation de l'âge à l'aide des rides du visage

PROJET IN52 & IN54

Auteurs :

Ercin KADIR

Florian RIFFLART

Hessou PANASSIM

TABLE DES MATIERES

INTRODUCTION

Dans le cadre du projet commun des UV IN52 et IN54, nous avons décidé d'essayer d'estimer l'âge d'une personne à partir de ces rides. Dans un premier temps, nous avons essayé d'estimer l'âge par tranche de 10 ans à l'aide de la méthode KPPV. Ensuite, nous avons utilisé l'algorithme Fuzzy c-mean (FCM) pour estimer l'âge exact.

Nous avons utilisé une base d'image contenant des images de personnes allant de 30 à 79 ans réparti dans 5 dossiers (un par tranche de 10 ans).

Dans ce rapport, nous décrirons les différentes étapes permettant de créer notre base d'apprentissage, ainsi que les deux méthodes de détection utilisées pour l'estimation de l'âge avant de présenter les résultats de notre projet.

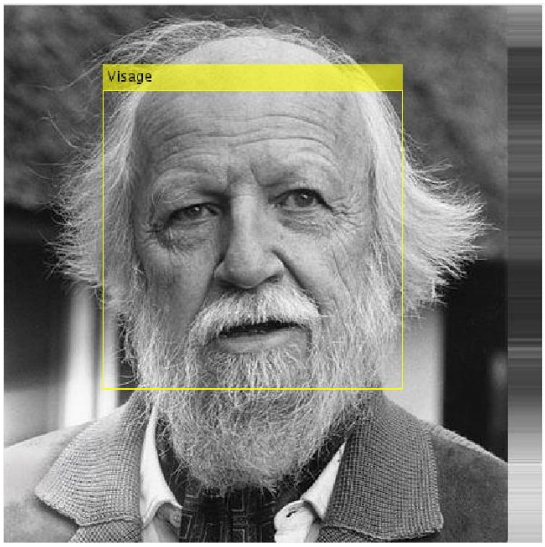
I. DECOUPAGE

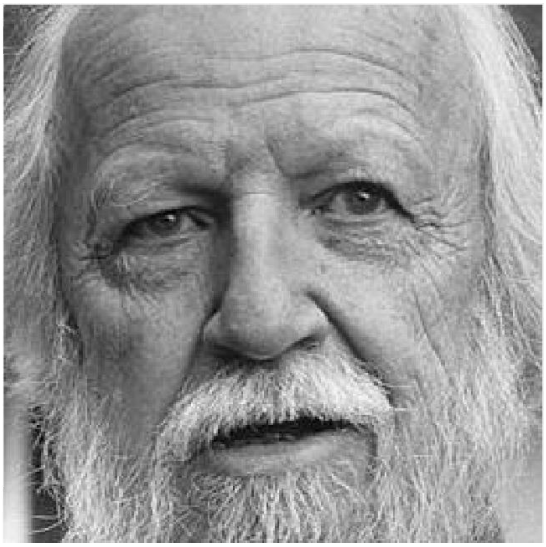
La première étape consiste à découper l'image afin de ne garder que le visage de la personne. Pour cela nous avons utilisé un outil de Matlab permettant de détecter les visages à l'aide de l'algorithme de Viola-Jones.

Voici la fonction permettant de découper le visage de l'image :

```
function imgCropped = decoupage(img)

faceDetector = vision.CascadeObjectDetector();
box = step(faceDetector, img);
imgCropped = imcrop(img, box(1, :));

end
```

Exemple de découpage du visage :

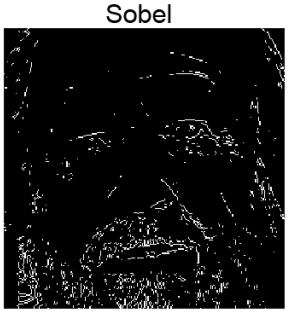




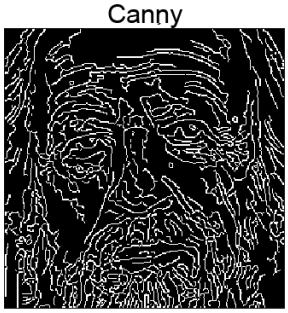
II. CONTOURS

Il faut ensuite traiter l'image afin de faire ressortir les rides. Nous avons donc utilisé la fonction *edge* de Matlab avec différents filtres de détection de contours (Sobel, Canny, Roberts, Prewitt et log).

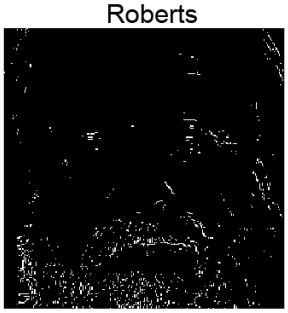
Sobel



Canny

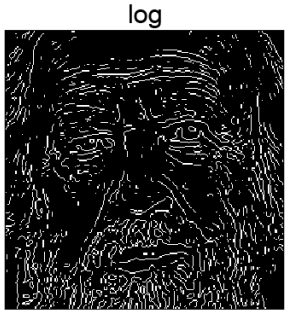


Roberts



Prewitt

log

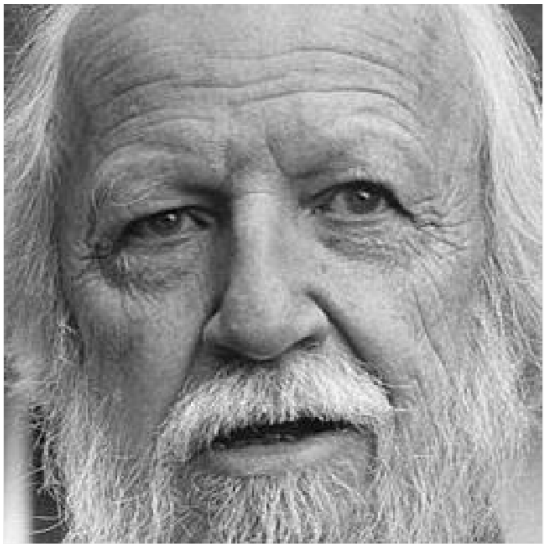


Finalement, la détection de contours avec le filtre de Canny était la plus adaptée pour mettre en valeur les rides.

Voici la fonction permettant de récupérer les contours de l'image :

```
function imgContours = contours(img)

[m n c] = size(img);

if c==3
	imgGray = rgb2gray(img);
else
	imgGray = img;
end

imgContours = edge(imgGray, 'Canny');

end
```

Exemple de récupération des contours :



III. ZONES RIDEES DU VISAGE

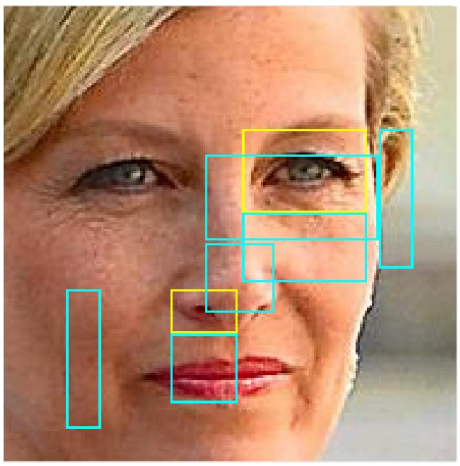
A. DELIMITATION DES ZONES

Enfin, nous avons sélectionné les zones du visage où les rides sont présentes en grande quantité :

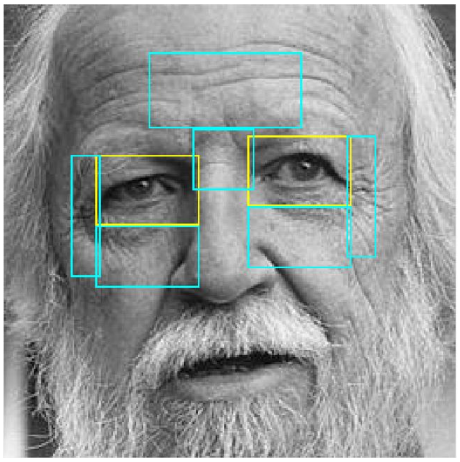
- Zone des rides du front,
- Zone des rides du lion (entre les yeux),
- Zone des rides des pattes d'oie (côté des yeux),
- Zones des rides des joues.

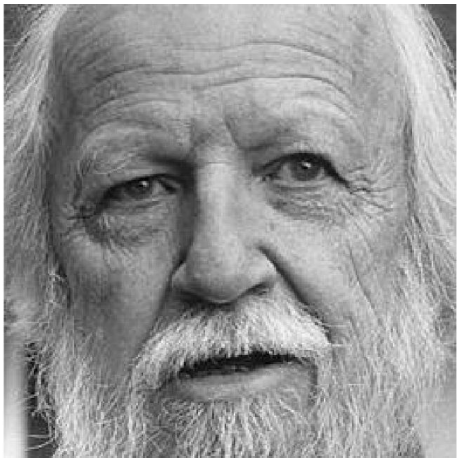
Pour calculer ces différentes zones, nous nous sommes basés sur la détection des yeux de Matlab (à nouveau grâce à l'algorithme de Viola-Jones). Cependant, la détection des yeux n'est pas parfaite et il arrive que les yeux soient détectés au mauvais endroit ce qui entraîne un placement erroné des zones ridées.

Exemple de mauvaise détection des yeux :

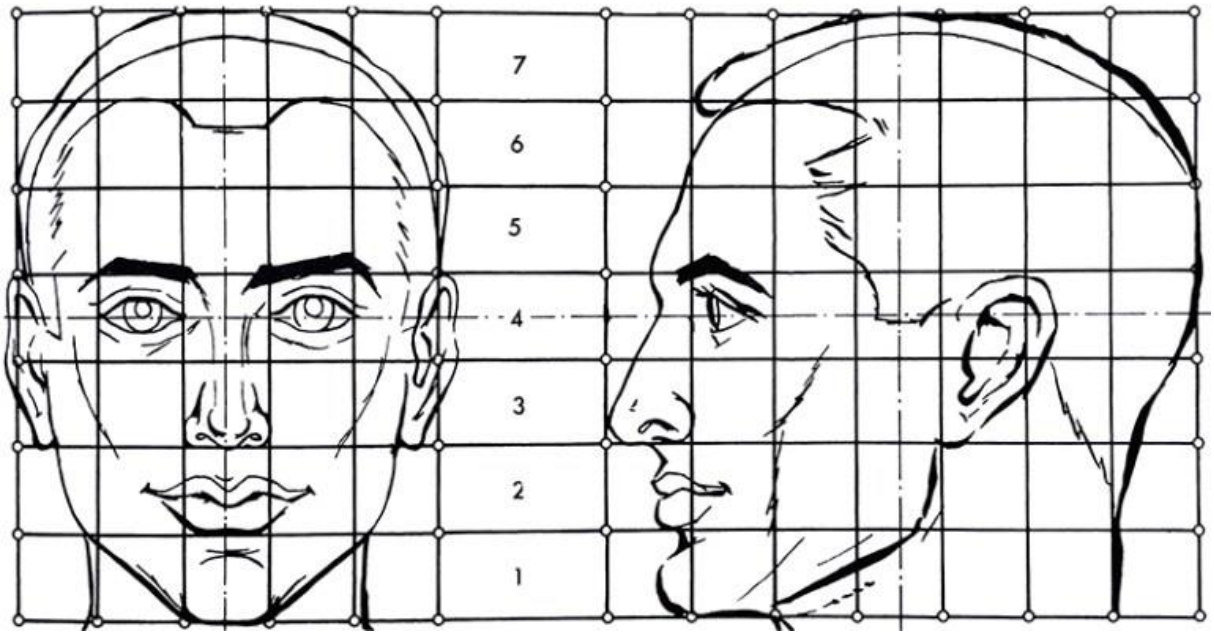


Exemple de bonne détection résultant en une bonne sélection des zones ridées du visage :





Pour déterminer chaque zone d'intérêt nous nous sommes basés sur les proportions d'un visage humain.



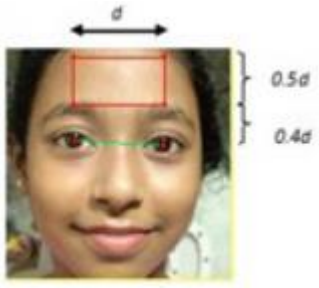
Du front au menton, le visage humain est composé de 3 parties (qui peuvent être chacune divisées en 2). La première partie comprend le front, la deuxième les yeux, le nez, le haut des joues, et la troisième le bas des joues, la bouche et le menton.

Grâce à la fonction de détection des yeux nous récupérons les coordonnées de ces derniers dans l'image.

- Zone du front

Nous déterminons la distance d entre le milieu des yeux. La largeur du front vaut **d** et la hauteur **d * 0.4**.

Le coin supérieur est situé juste au-dessus du milieu de l'œil droit. (Voir figure suivante)



- Zone des rides du Lion

La largeur de cette zone est la distance **d * 0.5**. Néanmoins afin d'éviter de limiter les informations inutiles nous nous contentons de la largeur **d * 0.4**. Cette zone est située juste en dessous du front, nous déduisons donc ces coordonnées.

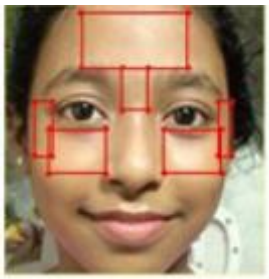
- **Zones des rides des pattes d'oie**

Chacune des zones des commissures des yeux a pour largeur la distance **d * 0.5**. Nous avons fixé comme hauteur utile le double de la hauteur de zone entre les yeux. Ces zones couvrent la moitié au-dessus et la moitié en-dessous des yeux.

- **Zones des rides des joues**

Les joues sont situées en dessous des yeux. La largeur retenue est celle des yeux, et la hauteur vaut la distance séparant les yeux * 0.4 soit **d * 0.4**.

On obtient le zonage suivant :



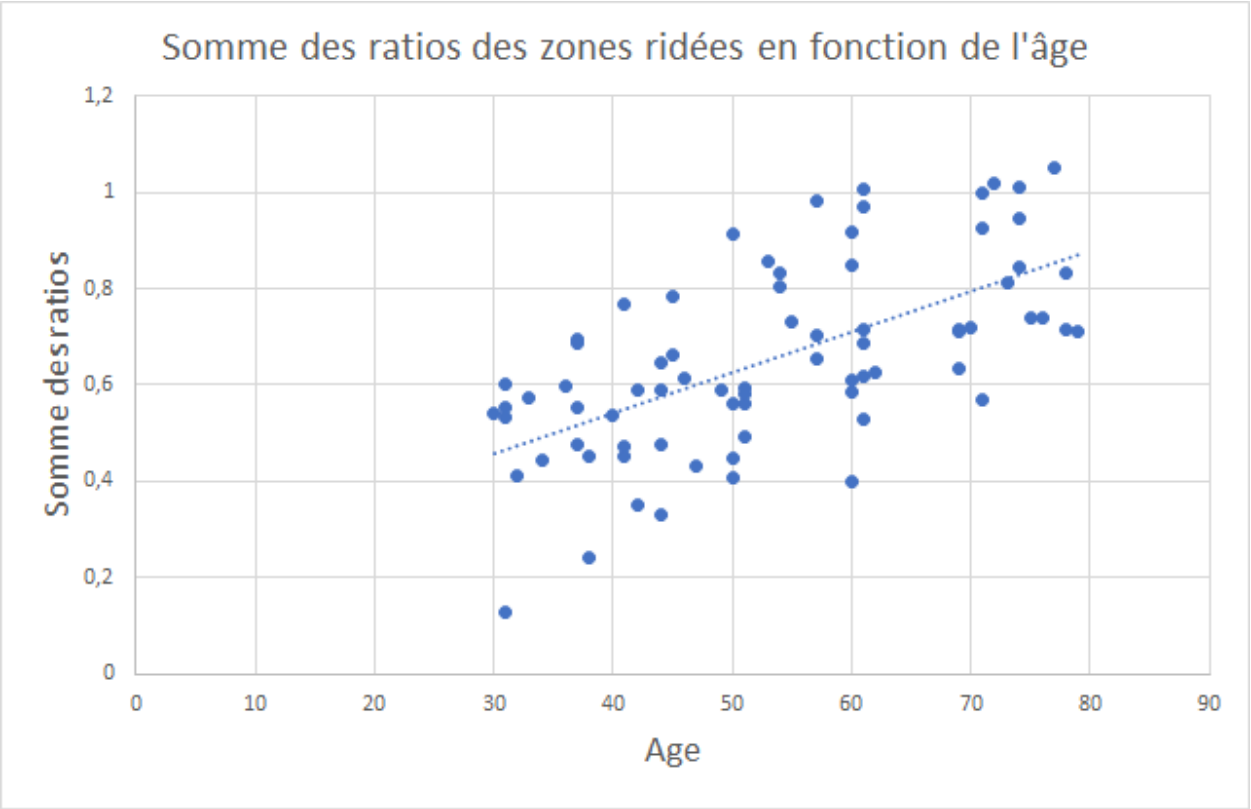
B. SOMMES DES RATIOS DES ZONES

Nous avons créé une fonction servant à calculer le ratio de pixels blancs par rapport au nombre de pixels totaux de chaque zone de rides puis à les additionner. Cette somme des ratios sera ensuite utilisée dans l'apprentissage des 2 méthodes d'estimation de l'âge.

Ci-dessous, la fonction permettant de récupérer la somme des ratios des zones ridées d'une image :

```
function sommeRatios = getSommeRatios(img)
	% Découpe l'image pour ne garder que le visage
	imgCropped = decoupage(img);

	% Fait apparaître les rides sous formes de contours
	imgContours = contours(imgCropped);

	% Récupère les coordonnées des rectangles correspondant aux
	% parties du visage sélectionnées
	partiesVisage = getCoordPartiesVisage(imgCropped);

	sommeRatios = 0;

	for j = 1 : size(partiesVisage, 1)
	    box = partiesVisage(j, :);
	    partieVisage = imcrop(imgContours, box(1, :));

	    % Additionne les pixels blancs
	    whitePixels = sum(sum(partieVisage == 1));
	    % Calcul le nombre total de pixels
	    totalPixels = sum(sum(partieVisage == 0)) + whitePixels;
	    ratio = whitePixels / totalPixels;

	    sommeRatios = sommeRatios + ratio;
	end
end
```

Si on trace le nuage de points de la somme des ratios des 6 zones ridées en fonction de l'âge des différentes images de notre base, on remarque que plus la personne est âgée, plus la somme des ratios a tendance à être élevée.



IV. BASE D'IMAGES

Nous avons récupéré une grande base d'image provenant de Wikipédia, dans laquelle nous avons sélectionné les images qui nous semblaient les plus adaptées : visage bien droit, face à l'objectif, visage dégagé, expression neutre, photo de qualité…etc.

Ensuite, pour chacune de ces images, nous avons vérifié manuellement que notre traitement des visages était correct, c'est à dire, que les différentes zones ridées soient correctement placées sur le visage.

Nous avons constitué une base d'apprentissage de 15 images par tranche de 10 ans (de 30 à 70), soit 75 images. Pour la base de test, celle-ci comporte 22 images de personnes dont l'âge va de 31 à 74 ans.

V. METHODE KPPV

On détermine avec cette méthode la tranche d'âge d'un individu.

A. APPRENTISSAGE

L'apprentissage consiste ici à enregistrer la somme des ratios de chaque image dans 5 fichiers (un par tranche d'âge).

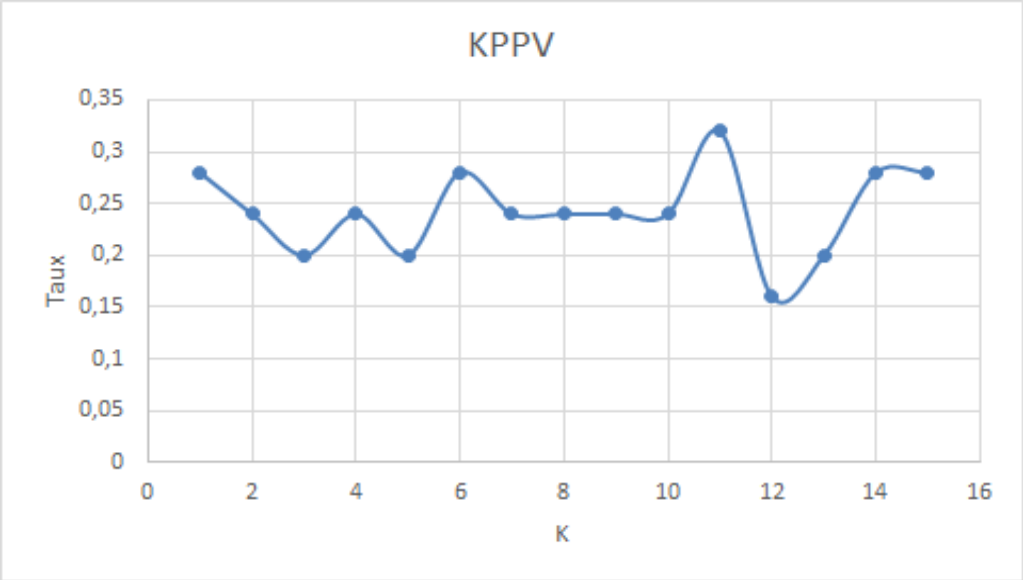
B. DECISION

La décision se fait via la méthode des K plus proches voisins :

1. On calcule d'abord la somme des ratios de l'image dont on veut déterminer la tranche d'âge.
2. Ensuite, les K plus proches voisins dans la base d'apprentissage par rapport à ces ratios sont retenus.
3. La tranche d'âge de l'image est déterminée en choisissant, parmi les K plus proches voisins, la tranche d'âge la plus représentée.

C. RESULTATS

Evolution du taux de reconnaissance de la tranche d'âge en fonction de k :



On obtient une reconnaissance maximale de 0.325 pour K = 11. Cette méthode nous donne donc toujours de mauvais résultats, malgré des K différents. Ceci est peut-être dû à la qualité et à la taille de notre base d'images.

VI. CLUSTERING FLOU (FUZZY C MEAN)

Le principe de cet algorithme est d'agglomérer des données dans plusieurs clusters comme le ferait le partitionnement en k moyennes. Contrairement aux k moyennes, les données n'appartiennent pas exclusivement à un seul cluster, mais possèdent un degré d'appartenance pour chacun des clusters.

L'objectif de cet algorithme est de minimiser la distance entre les données et le cluster, pondéré par le degré d'appartenance des données aux clusters.

Fonction objectif :

En entrée : X_i : la valeur de la donnée i.

En sortie :

- W_{ij} : le degré d'appartenance de la donnée i au cluster j,
- C_j : le centre du cluster j.

Paramètres influant sur la fonction objectif :

- Réel m : Coefficient de flou (Fuzziness)
- Entier maxIter : Nombre d'itérations maximal pour la convergence.
- Réel epsilon : Critère de convergence de la fonction objectif à minimiser.

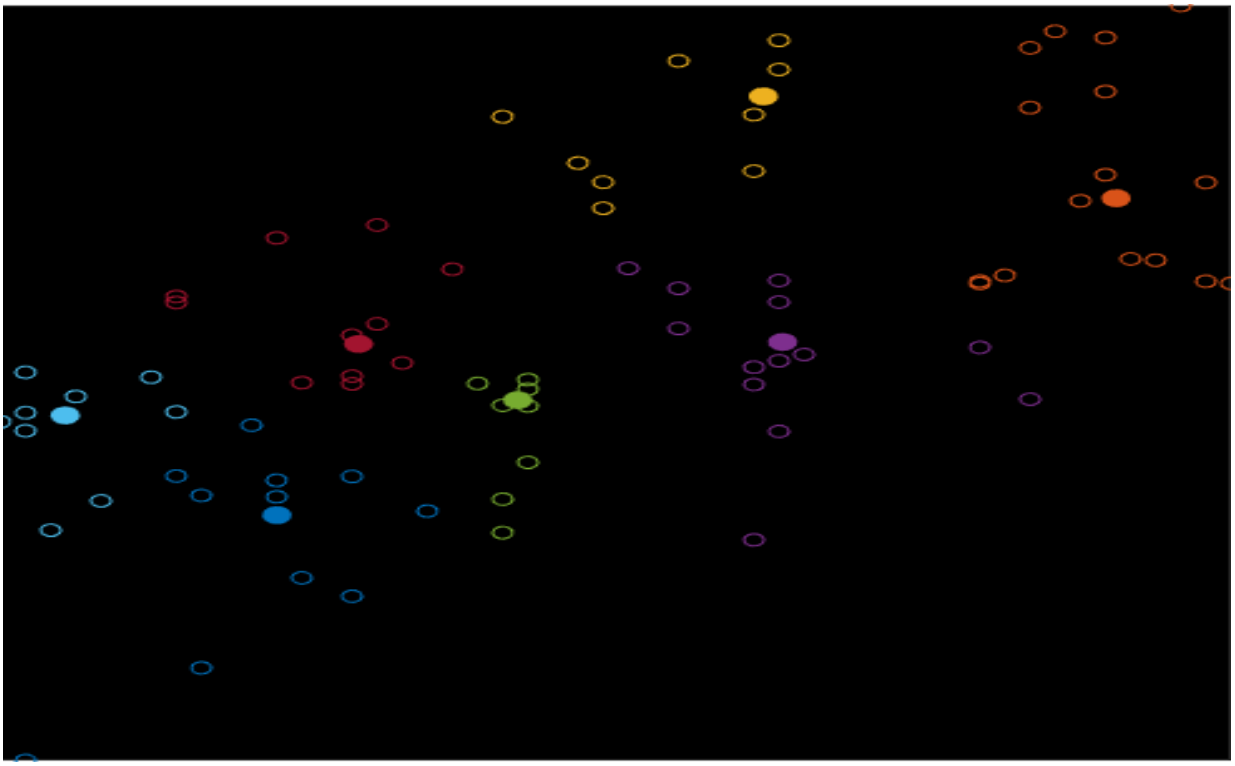
Ces paramètres sont à tester et calibrer pour obtenir les meilleurs résultats possibles.

A. APPRENTISSAGE

L'algorithme prend en entrée (en plus des paramètre) une matrice, c'est à dire les données que l'on veut *clustériser*. Dans notre cas nous avons fourni en donnée, la matrice des couples sommes des ratios + âge de chaque image de notre base d'apprentissage.

On reçoit en sortie la matrice W (degrés d'appartenances aux clusters) ainsi que la matrice C représentant les centres des clusters.

Dans notre cas d'utilisation, la matrice C est composée du couple âge et ratio des centres des clusters. C'est cette matrice qui nous sera utile, et que l'on sauvegardera dans un fichier, pour l'utiliser lors de l'étape de décision.



Exemple d'un apprentissage, avec 7 clusters (représenté chacun par une couleur)

B. DECISION

Pour l'estimation de l'âge d'une image, on récupère dans un premier temps sa donnée (sommes des ratios), ce qui nous permet de calculer, à partir de la matrice C des centres, les degrés d'appartenance de cette données aux différents clusters.

Le degré d'appartenance W_{ij} de l'image i au cluster j utilisé par l'algorithme est le suivant :

$$w_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{\|\mathbf{x}_i - \mathbf{c}_j\|}{\|\mathbf{x}_i - \mathbf{c}_k\|} \right)^{\frac{2}{m-1}}}.$$

Ce coefficient s'interprète comme étant la similitude d'une donnée par rapport à un centre ainsi que sa disparité par rapport aux autres. Ce coefficient à un centre donnée est élevé, lorsque la donnée est proche par rapport à ce centre, et éloigné par rapport aux autres centres.

Notre méthode consiste à faire ensuite, par rapport à cette donnée, la somme du produit de la moyenne d'âge du centre de chaque cluster par les degrés d'appartenance de l'image test à chaque cluster.

Ce résultat nous donne directement l'âge estimé de l'image en question.

Fonction permettant d'estimer l'âge (calcul coefficient + âge) :

```
function age = decisionFCM(classifieur, img, m)
    sommeRatios = getSommeRatios(img);
    age = 0;
    exp = 1; % exp pour faire plus ou moins influer les coefficient des centres les plus proches
    for j = 1 : size(classifieur,1)
        if(isnan(classifieur(j,1)) == false)
            dist = sommeRatios - classifieur(j,2);
             somme = 0;
             for k = 1 : size(classifieur,1)
                 somme = somme + (dist / (sommeRatios - classifieur(k,2)))^(2 / (m-1));
             end
             % degré d'appartenance Pi de l'image au cluster j
             Pi = 1 / somme^exp;
             age = age + Pi * classifieur(j,1);
        end
    end
end
```

C. RESULTATS

En faisant varier les différents paramètres de l'algorithme FCM, on obtient sur notre base de test un delta d'âge moyen |détecté - réel| de 8.2 années avec les paramètres suivants :

- m = 2.2
- Nombre de cluster = 7
- Epsilon = 1e-7
- Max itération = 100

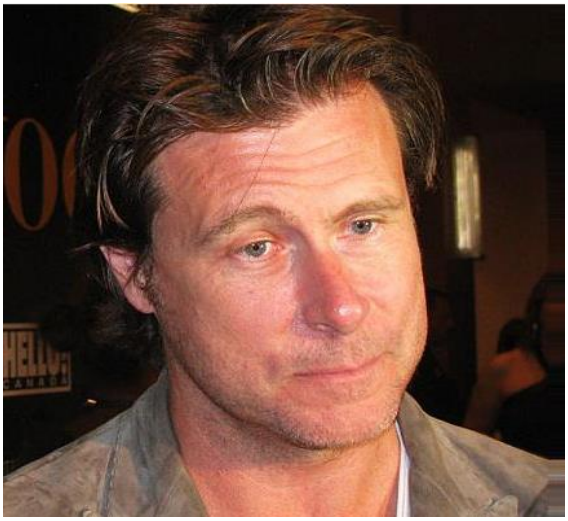
Le delta est mauvais, nous n'avons pas réussi à obtenir de meilleurs résultats que celui-ci, en variant les paramètres de l'algorithme.

Nous nous sommes dit que dans l'estimation, nous n'accordions pas assez "d'importance" aux clusters les plus proches, qui pourraient peut-être permettre de mieux estimer l'âge. On a donc essayé de rajouter un facteur exponentiel dans la détection de l'âge, et de le faire varier, pour les degrés d'appartenance, mais les résultats ne se sont pas améliorés.

Estimation de qualité moyenne :

34 ans estimé / 31 ans en réalité

40 ans estimé / 33 ans en réalité



51 ans estimé / 45 ans en réalité

60 ans estimé / 51 ans en réalité

Estimation de mauvaise qualité :

36 ans estimé / 51 ans en réalité

La qualité de cette image laisse à désirer, les contours des rides du front ne sont pas visibles, l'algorithme « pense » donc que c'est une personne plus jeune qu'elle ne l'est.

48 ans estimé / 32 ans en réalité

Ici la personnes prend une expression faciale faisant apparaître des rides sur son front, ce qui le rend plus vieux dans notre cas.

VII. CONCLUSION

Les résultats obtenus au niveau du taux de reconnaissance sont assez mauvais pour chacune des méthodes de d'estimation de l'âge proposées. Cela est en partie dû à la constitution de notre base de d'images. En effet nous avons utilisé une base d'image faite à partir de Wikipédia, comportant des photos qui n'étaient pas destinée à l'apprentissage et à la détection des contours. Il fallait donc parmi cette base, extraire les meilleurs images possibles. Cela a été assez difficile et nous avons parfois dû baisser certains critères de sélection concernant la qualité de l'image, car nous ne trouvions pas d'images convenables pour certaines tranche d'âge. De plus, même parmi les images qui satisfaisaient nos critères de qualité, notre traitement d'image ne récupérait pas toujours correctement les zones de rides du visages.

Nous pensons donc que l'élément le plus important pour obtenir de meilleurs résultats est de posséder une base d'image de meilleure qualité, mieux triée, et également plus conséquente. De plus, nous utilisons la détection des yeux via Matlab à partir desquels nous calculons les zones ridées qui présente une composante aléatoire pouvant poser problème. En cherchant et utilisant d'autres méthodes de détections plus précises, nous aurions pu constituer une base d'image plus facilement.

Jouer sur les paramètres de l'algorithmes d'apprentissage de la méthode FCM et améliorer la récupération des contours des rides nous permettrait également d'obtenir de meilleurs résultats.

Une autre possibilité est d'utiliser des réseaux de neurones, tout en conservant notre étape de traitement avec la somme des ratios, qui est selon nous un bon indicateur de l'âge d'une personne. Cependant, les résultats seraient tout aussi dépendant de la qualité de notre base d'image.