

Traitement d'images

Introduction et objectif

Dans ce TP, l'objectif est d'implémenter un programme de détection de formes simples par la transformée de Hough. Nous appliquons deux algorithmes différents ; premièrement, la transformée de Hough afin d'extraire les principales droites directrices de l'image. Deuxièmement l'algorithme de Evough qui reprend la base de de Hough en ajoutant un algorithme génétique.

Avant de rentrer dans le détail nous faisons remarquer que nous parlerons de « valeur » d'un pixel pour désigner sa couleur sur une échelle de gris (allant de 0 à 255). Cependant cette valeur peut être aisément adaptée à des images RGB ou autres.

L'objectif de ce TP sera donc d'analyser et comparer ces 2 algorithmes afin d'en comprendre le fonctionnement et le rôle joué par les différents paramètres des fichiers de configuration.

Méthode

1. Algorithme de Hough

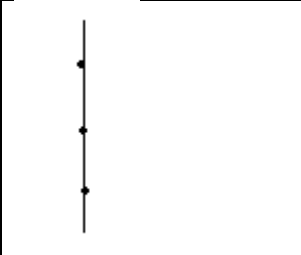
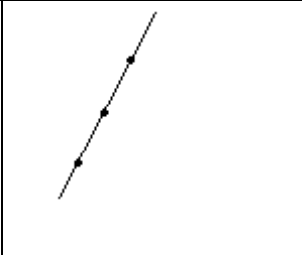
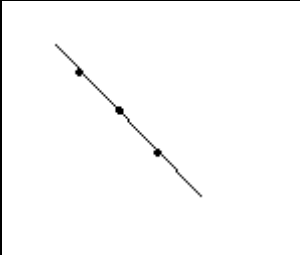
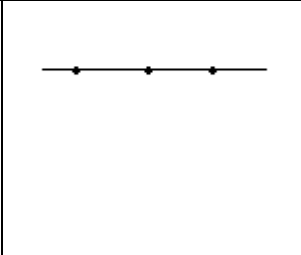
L'algorithme de Hough va se diviser en plusieurs étapes :

- Transformation des coordonnées cartésiennes en polaires
- Evaluation du gradient
- Détection des points de contours
- Accumulateur
- Affichage de la détection des droites

Dans le programme, on commence par transformer les coordonnées cartésiennes en polaire pour l'étape de l'accumulateur.

On passe ensuite à l'évaluation du gradient. Dans un algorithme de contour, on effectue un masque de 9 points. Ici, nous appliquons uniquement un masque sur un axe. Le gradient sera calculé à partir de la somme des différences de niveaux de gris de 4 voisins.

On en déduira tout de suite après selon le point dominant, l'orientation du gradient.

			
Si le point a est	Si le point b est	Si le point d est	Si le point c est

dominant, l'orientation sera verticale	dominant, l'orientation sera en diagonale positive	dominant, l'orientation sera en diagonale négative	dominant, l'orientation sera horizontale
--	--	--	--

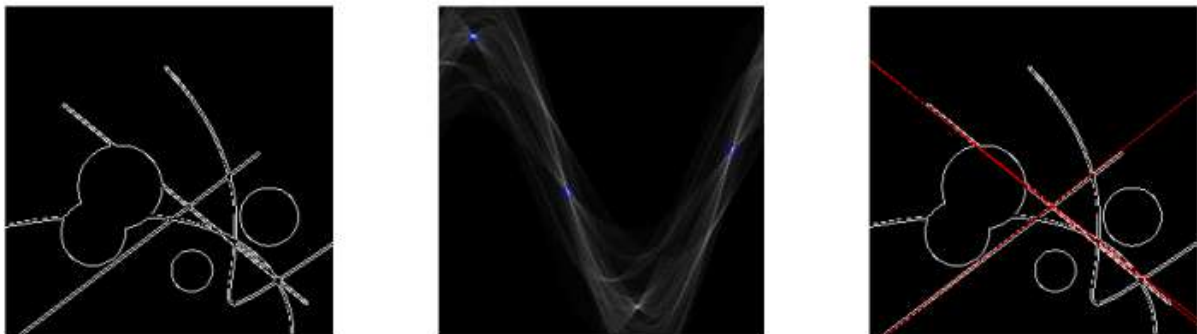
On compare ensuite la valeur du gradient au seuil défini dans le fichier de configuration afin d'effectuer la détection des points de contour.

L'intensité du gradient sera comparée au seuil. Si celui-ci est supérieur on considère que c'est un point important, il sera alors pris en compte.

Pour l'accumulateur, les points ont été passés en coordonnées polaires ce qui nous donne des sinusoidales :

- ϑ : l'angle
- ρ : la norme du vecteur (la longueur du segment perpendiculaire à la droite d'angle ϑ et passant par l'origine)

On remplit alors l'accumulateur de Hough à partir des points de contour détectés. Chaque courbe est associée à un point que l'on a sélectionné. Si les courbes associées à deux points se coupent, l'endroit où elles se coupent dans l'espace de Hough correspond aux paramètres d'une droite qui relie ces deux points, ainsi on obtient les droites correspondantes



On remarque bien sur la 2eme image (Espace de Hough) que les points où un maximum de courbes se coupent (points bleus) ont permis de construire les droites correspondantes.

2. Algorithme Evough

Tout comme pour l'algorithme d'Hough, on calcule le gradient de chaque point. La suite est basée sur un algorithme génétique. On commence donc par initialiser le tableau de sharring. On calcule ensuite la fitness pour chaque individu. On va alors pouvoir commencer le tournoi de sélection qui va nous permettre d'en éliminer les plus faibles, grâce aux permutations, puis aux croisements.

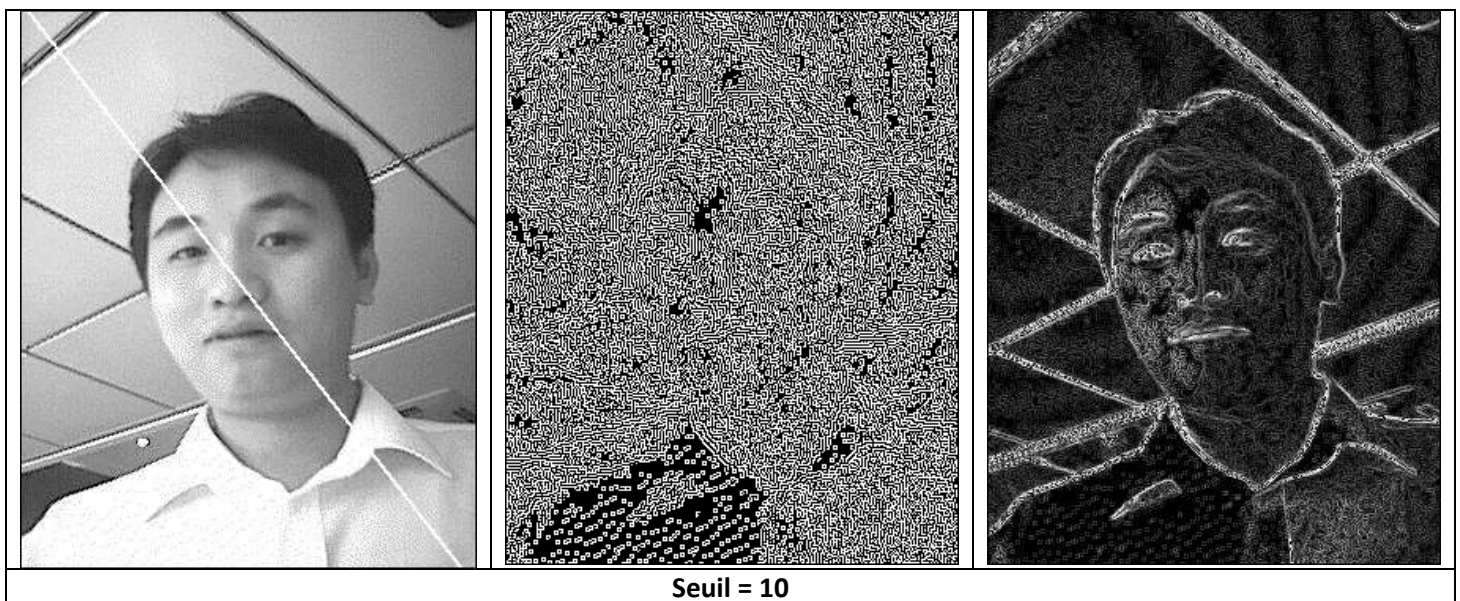
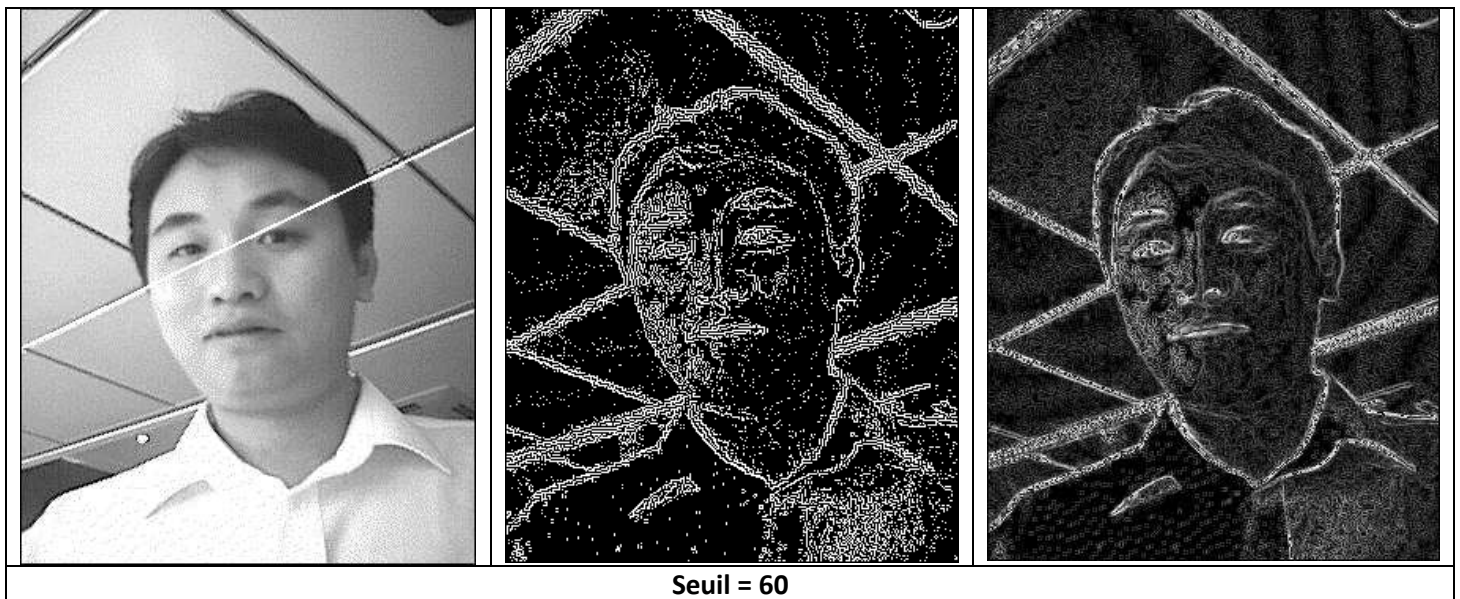
Une fois que ceci est fait, on va pouvoir activer les opérateurs génétiques, qui vont nous permettre de remplacer les individus éliminés par de nouveaux individus créés à partir de mutations ou de croisements.

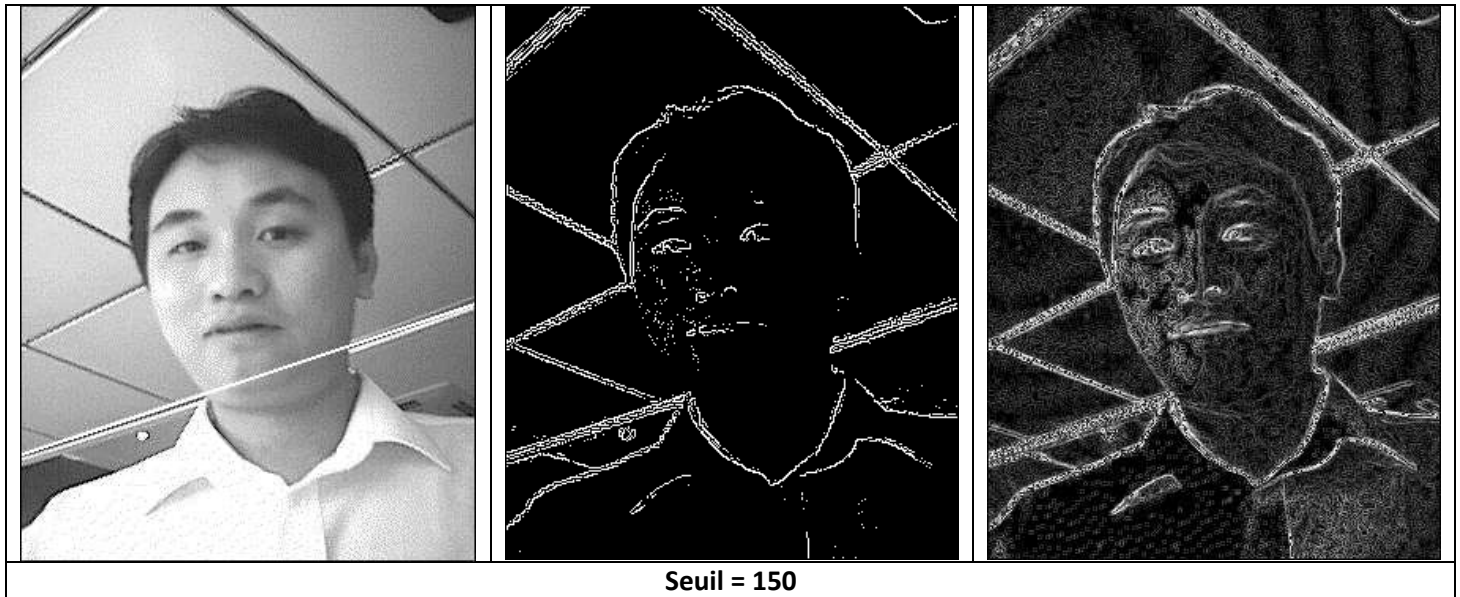
On cherche alors le nombre de solutions en comparant la fitness maximum multipliée par l'exigence définie dans les paramètres (comprise entre 0 et 1) et la fitness du point en question. On analyse ensuite la population donnée.

Analyses et résultats

1. Algorithme de Hough

Pour l'algorithme de Hough, un seul paramètre est pris dans le fichier de configuration : la valeur du seuil. De base le seuil est à 60.





Premièrement on remarque que quel que soit le seuil, le gradient ne change pas. En effet, celui-ci est calculé indépendamment de ce paramètre.

On peut également voir que plus le seuil est bas, moins la détection sera précise, en effet, dans le programme, lorsque la valeur du gradient est inférieure à la valeur du seuil, l'algorithme va se servir de l'orientation pour modifier le point présent sur cette même orientation et le mettre en blanc. Ainsi, comme on le voit pour le seuil à 10, presque tous les points possédaient un gradient inférieur au seuil ce qui donne cette détection de contour non exploitable.

Contrairement à une opération de seuillage classique, dans cette détection de contour, on sélectionne d'abord uniquement les points dont le gradient calculé est supérieur à la valeur du seuil. Il est ainsi considéré comme important. Ce point prendra ainsi une valeur selon le point voisin situé sur le même axe sélectionné (même orientation).

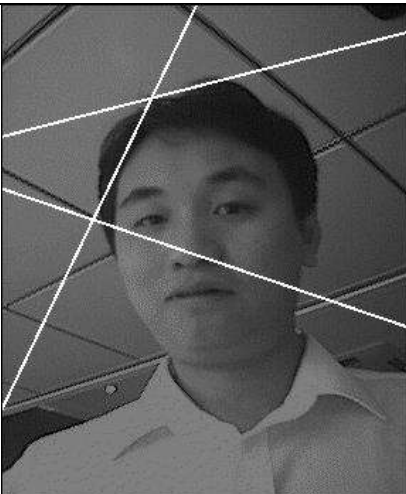
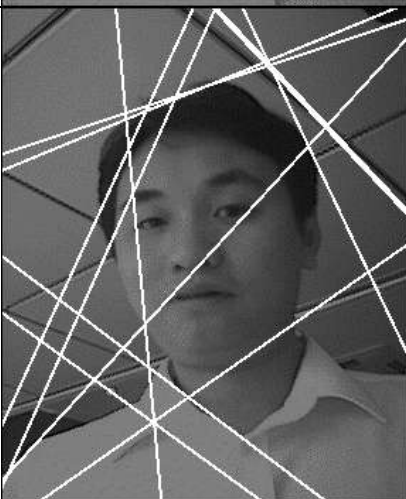
On peut voir dans le cas du seuil le plus bas, que l'algorithme détecte un contour là où il n'y en a pas. Ceci est dû à la valeur du seuil, en effet, celui-ci étant bas dans ce cas, de très nombreux points sont considérés comme des contours ce qui entraîne une mauvaise déduction d'affichage de droites correspondantes.

Dans la dernière image comme dans le cas du seuil à 60, on remarque que la détection de contour est moyenne, on remarque qu'une ligne passe par-dessus le visage de l'homme. Ceci s'explique par la détection de contour moyenne qui va marquer fortement plusieurs lignes du plafond.

2. Algorithme Evough

	Seuil 60 Avec ce seuil, on peut voir que l'algorithme détecte bien l'un des contours du plafond et suit bien le tracé de celle-ci.
	Seuil 150 Comparé au précédent, le tracé est un peu plus approximatif et moins précis. Il suit toutefois une ligne présente dans l'image de base.
	Seuil 20 Avec un seuil 20, la détection de contour n'est pas exacte ni efficace, on peut voir que l'une des lignes suit approximativement une du plafond, l'autre non.

		Seuil 60 Population 700 Lorsqu'on augmente très largement la population, on a beaucoup plus de données de départ, mais la précision n'est pas améliorée. On le remarque très bien sur cette image, la détection a repéré une ligne étant donné que la valeur du seuil reste correcte, cependant la détection est encore très approximative.
		Seuil 60 Population 700 Generation 1000 En augmentant la génération, on augmente forcément les probabilités de mutation et de croisement, améliorant ainsi la précision de la détection de contour. On le remarque très bien sur cette image, la détection est plus précise que l'image précédente.
		seuil 60 population 700 generations 1000 pmutation 100 pcrossover 50 mutabilitero 40 mutabilitetheta 40 En favorisant un maximum le nombre de générations et la probabilité de mutation, le résultat devait théoriquement être plutôt précis. On peut voir sur cette image que la détection a repéré une ligne différente du cas précédent, d'une précision correcte.

	seuil 60 population 50 generations 1000 pmutation 100 pcrossover 50 mutabilitero 40 mutabilitetheta 40 Malgré une probabilité de mutation et un nombre de générations importantes, si la population n'est pas assez grande, la détection sera alors plus hasardeuse. Cependant le résultat n'est pas aussi catastrophique compte tenu du nombre de générations et croisements/mutations important.
	seuil 60 population 50 generations 10 pmutation 1 pcrossover 1 mutabilitero 2 mutabilitetheta 2 Pour illustrer les images précédentes, lorsque l'on diminue quasiment tous les paramètres, le résultat est alors évident. La détection n'est absolument pas exacte et très imprécise.
	seuil 60 population 100 generations 10 pmutation 40 pcrossover 10 mutabilitero 5 mutabilitetheta 5 En changeant la génération, la détection n'est absolument pas correcte. Ceci illustre les analyses précédentes confirmant ainsi que le nombre de générations est important pour la qualité du résultat.



seuil 60
population 100
generations 200
pmutation 40
pcrossover 10
mutabilitero 5
mutabilitetheta 5
exigence .01

Pour finir, la valeur de l'exigence reste très importante. Avec une valeur très faible, la détection n'est alors pas précise du tout.

Conclusion

Nous avons tout au long de ce projet étudié les caractéristiques à la fois de l'algorithme de Hough mais également de son évolution intégrant et utilisant un algorithme génétique dans le but d'obtenir de meilleurs résultats.

Nous avons pu remarquer que dans le cas de l'algorithme de Hough, le paramétrage du seuil est nécessaire au bon fonctionnement de notre algorithme de détection.

Dans le cas de son évolution, le nombre de paramètres étant plus élevé, il est plus difficile de se l'approprier mais les résultats observés semblent meilleurs si le paramétrage est effectué correctement. Cela s'explique notamment par l'utilisation de l'algorithme génétique permettant de faire évoluer les solutions vers des solutions encore plus précises. Il sera donc important de configurer correctement le programme pour que son efficacité n'en soit qu'augmentée.