TRAITEMENT D'IMAGES

TP4 : Evaluation des détecteurs de contours

réalisé par LE Viet Man - promotion 15

FONCTIONNEMENT DU PROGRAMME

Compilation du programme

Pour compiler le programme, il faut modifier la variable INCLUDEDIR dans le fichier Makefile pour mettre le bon chemin pour le répertoire de OpenCV.

Ensuite, dans le répertoire du programme, vous tapez la commande `make clean`. Et puis, vous tapez la commande `make` pour compiler le programme.

Usage du programme

On peut utiliser une seule commande `contours` pour extraire les contours et calculer trois mesures P, TFP et TFN. Le programme possède la commande suivante :

```
./contours [options] <fichier>
```

où :

- `<fichier>` : le chemin et le nom du fichier d'image, le type de l'image doit être en niveaux de gris. Si vous passez une image en couleur, le programme la transformera en niveaux de gris.
- `[options]` : le programme fournit trois arguments :
  - `-a` : cet argument sert à afficher les informations de l'aide du programme
  - `-s <int>` : le seuil de contours. La valeur en défaut est 100.
  - `-r <fichier>` : le nom du fichier de la référence tracée à la main.

    Si l'on ne donne pas cette option, le programme ne calcule pas de trois mesures P, TFP et TFN.

    Les images sorties ont toujours le fond en blanc et les contours en noirs. C'est le type de la référence tracée à la main. Donc, l'évaluation est simple pour les images de contours et aussi les références tracée à la main.

    Les images sorties sont quatre images : l'image de contours Sobel dont le nom fini avec «_sobel.pgm», l'image de contours Prewitt dont le nom fini avec «_prewitt.pgm», l'image de contours Laplace dont le nom fini avec «_laplace.pgm» et l'image de contours Canny dont le nom fini avec «_canny.ppm». Les images sorties sont sauvegardées sous forme pgm et dans le répertoire du programme.

Quelques exemples

- Extraire les contours de l'image 12003.pgm avec le seuil 80 :

```
./contours -s 80 12003.pgm
```

- Extraire les contours de l'image basket.pgm avec le seuil 80 et évaluer les résultats avec la référence tracée à la main basket_gt_binary.pgm :

```
./contours -s 80 -r basket_gt_binary.pgm basket.pgm
```

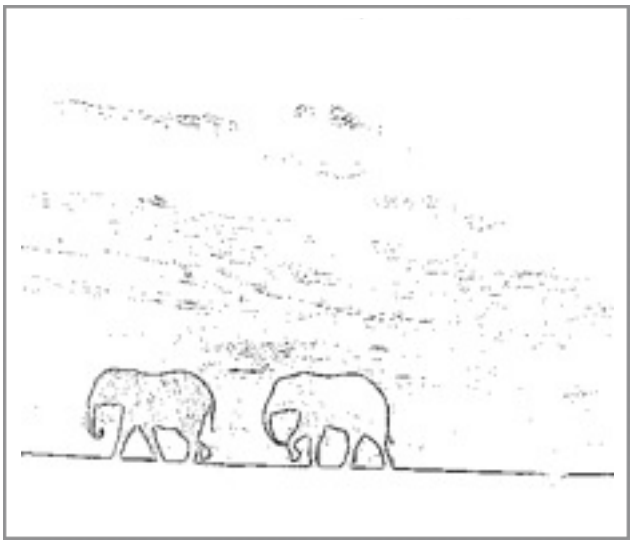
EVALUATION QUANTITATIVE DES DETECTEURS

Détecteurs

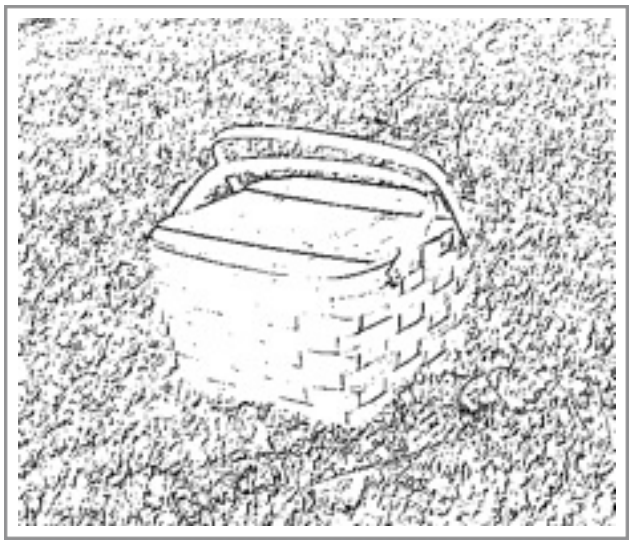
Dans le cadre de ce TP, je dois faire une comparaison entre quatre détecteurs de gradient de Sobel, Laplace, Canny et une autre de mon choix. Trois détecteurs de Sobel, Laplace et Canny est disponibles dans la librairie OpenCV. Donc, je dois seulement implémenter mon détecteur. J'ai choisi d'implémenter l'algorithme Prewitt parce que cet algorithme est simple et comme Sobel. Sa différence est que la méthode Prewitt utilise un moyenneur au lieu d'une gaussienne pour éliminer des bruits.

Calcul du gradient et seuillage

Tout d'abord, avec les algorithmes Sobel, Prewitt et Laplace, choisir le meilleur seuil est très difficile. J'ai testé avec les seuils de 10 à 120 pour chaque algorithme, les résultats sont différents avec les entrées différentes. Par exemple, avec l'image qui a moins de bruits (elephants.pgm), le contour est très bon avec un petit seuil. Par contre, avec l'image qui a plusieurs bruits (les images des poils des animaux et celles des herbes), le contour est encore mal voire le seuil dépasse la valeur 120.



Appliquer la méthode Laplace avec le seuil 80.
La mesure P = 0.256426.



Appliquer la méthode Sobel avec le seuil 120.
La mesure P = 0.071799.

Pour choisir efficacement un bon seuil, j'ai calculé la mesure P (= contour_corrects / (contour_corrects + faux_positifs + faux_negatifs)) pour tous les trois algorithmes. Le seuil de 70 à 80 ayant la plus haute mesure P était meilleur pour toutes les images. Donc, j'ai choisi la valeur 80 comme un seuil pour trois algorithmes Sobel, Prewitt et Laplace à faire la comparaison quantitative.

Avec l'algorithme Canny, dans la librairie OpenCV, la fonction cvCanny utilise deux seuils (min threshold et max threshold) où min threshold a pour but de relier les contours faibles et max threshold a pour but de trouver des segments des contours où le gradient est grand. Je l'ai testé avec les valeurs différentes pour ces seuils. J'ai trouvé que le premier seuil est moins important. La variation dans les résultats est moins quand on change le premier. Cependant, quand le deuxième est

180, les contours sont bons et moins de bruits. Donc, j'ai fixé deux valeurs 0 et 180 pour le seuil 1 et le seuil 2 dans la méthode cvCanny.

Dans ces quatre détecteurs, j'ai utilisé le masquage de la même taille 3.

Comparaison d'une image de contours avec la référence tracée à la main

Pour la comparaison est objective et bonne, j'ai sélectionné dix images à partir des images de l'Université de Groningen selon des critères suivants :

- la valeur P est plus grand pour chaque méthode Sobel, Prewitt, Laplace et Canny
- la valeur P est plus moins pour chaque méthode
- la valeur P a l'écart entre les méthodes

Ces images peuvent diviser en deux groupes :

- cinq images ayant moins de bruits : brush, elephants, golfcart, tire, turtle
- cinq images ayant plusieurs bruits : bear_8, elephant, elephants_2, goat_6, goat

Après avoir calculé trois mesures P, TFP et TFN, j'obtiens les résultats suivants :

	SOBEL			PREWITT			LAPLACE			CANNY		
bear_8	0.041	0.928	0.031	0.049	0.887	0.063	0.041	0.909	0.050	0.039	0.947	0.014
brush	0.150	0.620	0.230	0.114	0.516	0.370	0.172	0.625	0.203	0.194	0.708	0.098
elephant	0.093	0.649	0.259	0.077	0.487	0.436	0.114	0.663	0.222	0.191	0.775	0.033
elephants_2	0.055	0.759	0.185	0.049	0.634	0.317	0.066	0.684	0.250	0.126	0.856	0.018
elephants	0.224	0.509	0.267	0.210	0.377	0.413	0.256	0.245	0.498	0.395	0.476	0.130
goat_6	0.049	0.932	0.018	0.073	0.884	0.043	0.061	0.901	0.038	0.038	0.957	0.005
goat	0.210	0.682	0.108	0.249	0.542	0.209	0.251	0.612	0.137	0.204	0.765	0.031
golfcart	0.221	0.637	0.142	0.268	0.480	0.253	0.242	0.580	0.178	0.188	0.782	0.030
tire	0.171	0.739	0.090	0.201	0.609	0.189	0.176	0.746	0.078	0.194	0.769	0.037
turtle	0.214	0.633	0.153	0.229	0.516	0.255	0.244	0.459	0.297	0.264	0.605	0.131

ANALYSE DES RESULTATS

Sobel vs Prewitt

Sobel et Prewitt sont deux méthodes pareils. La seule différence est que Prewitt utilise un moyenneur et Sobel utilise une gaussienne de lisser des bruits. A partir du tableau ci-dessus, je trouve que Prewitt donne les meilleures valeurs P pour la plupart des images, sauf quatre images : brush, elephant, elephants_2 et elephants.

En se basant sur ces images, on peut trouver que les bruits de ces images sont difficiles à éliminer. Par exemple, l'image brush a les bruits qui sont les plumes de la brosse, l'image elephants a les bruits qui sont les nuages. Ce sont les principales parties de l'image. Donc, on ne peut pas les éliminer complètement. La valeur P de Sobel pour ces quatre images est plus haute que celle de Prewitt parce que Sobel utilise une gaussienne qui est plus efficace avec les images comme ça.

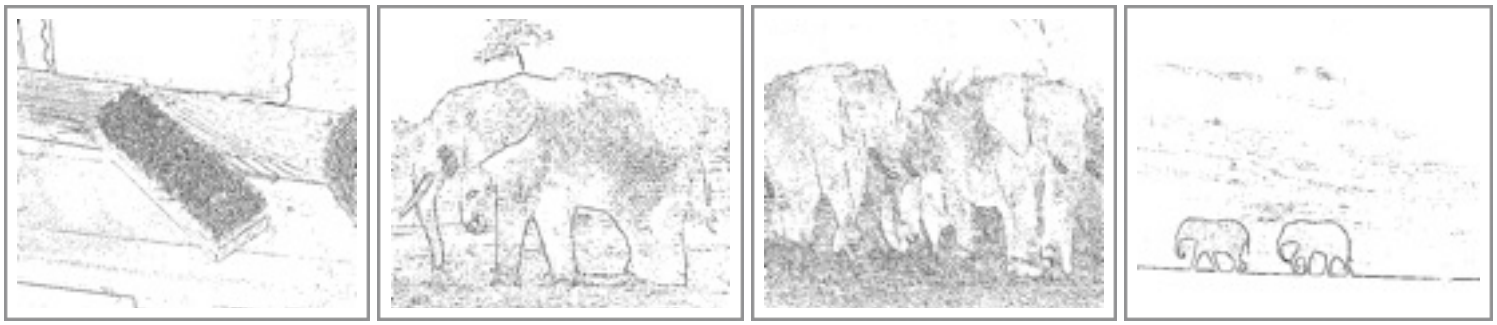
Les images de contours après avoir appliqué le filtre Prewitt avec le seuil 80.

Les images de contours après avoir appliqué le filtre Sobel avec le seuil 80. Les contours sont plus nets que ceux de Prewitt. Donc la valeur P de Sobel est plus haute que celle de Prewitt.

Mais tous les deux méthodes ont les mêmes hautes valeurs TFP et TFN. Cela montre que les contours de deux détecteurs ne sont pas bien détectés et ne sont pas détectés au bon endroit.

Laplace

Comme Sobel, Laplace est plus efficace avec ces quatre images et a les plus hautes valeurs P. Cela est facile à comprendre. Parce que Laplace utilise la seconde dérivée de l'image et les contours apparaissent toujours pour toutes les directions. Mais les valeurs TFP et TFN sont aussi hautes comme Prewitt et Sobel. Mais TFN est meilleure que Prewitt et Sobel. Cela montre que Laplace est mieux détecté.



Les images de contours après avoir appliqué le filtre Laplace avec le seuil 80. Les contours sont plus nets que ceux de Sobel. Donc la valeur P de Laplace est plus haute que celle de Sobel.

Canny

Canny utilise aussi un filtre Gaussien et appliquer le filtre de Sobel en X et en Y, mais il utilise deux seuils : un seuil haut (Sh) et un seuil bas (Sb). Pour chaque pixel de la norme du gradient : Si norme (x,y) < Sb, alors le pixel est mis à zéro (non-contour), si norme(x,y) > Sh, alors le pixel est contour, Sb <= norme(x,y) <= Sh, alors le pixel est contour s'il est connecté à un autre pixel déjà accepté comme contour. C'est pourquoi Canny a plusieurs avantages que Sobel.

A partir du tableau, avec les images ayant plusieurs bruits, Canny donne les meilleurs résultats avec les valeurs P est hautes et les valeurs TFN est moines (<10%). Canny est mieux détecté.

La meilleure méthode de gradient

Si l'on se base sur les valeurs P, la méthode de Prewitt semble meilleure parce que celles sont plus hautes pour la plupart des images. Mais quand l'on examine les valeurs TFP et TFN, on trouve que cette méthode n'est pas efficace.

Canny a les moines valeurs TFN. Donc, Canny est plus efficace que les autres. Selon moi, Canny semble la meilleure.

Les valeurs P de Canny est moines que celles de Prewitt pour la plupart des images à cause de choisir sensiblement la valeur max threshold (= 180), n'est pas de l'inefficacité de Canny.

Est-ce que vos résultats sont les mêmes pour toutes les images ?

J'ai testé avec toutes les images dans la base d'images de l'Université de Groningen et j'ai obtenu les mêmes résultats pour quatre détecteurs. Bien que les valeurs P de Canny soit moines que celles de Prewitt, les valeurs TFN sont les plus basses.

Est-ce que selon vous, la méthodes d'évaluation utilisée dans ce travail est bien adaptée ou non ?

Selon moi, la méthodes d'évaluation utilisée dans ce travail est bien adaptée. Calculer cinq grandeurs (contours_détectés, contours_référence, contours_corrects, faux_positifs, faux_négatifs) est simple et précis. Mais ici, il existe deux problème :

- Quand la référence tracée à la main n'est pas précis, le résultat n'est pas bon.
- Quand l'image de contours a beaucoup de bruits, un contours qui n'est pas contours dans l'image de référence, peut être un contours, vice versa.

Par conséquent, le résultat ici n'est pas absolument précis.

Est-ce que le choix de paramètres influence les résultats obtenus ?

Qui, avec le petit seuil, l'image de contours aura beaucoup de bruits et avec le haut seuil, elle perdra des contours. Avec la méthode Canny, j'ai choisi sensiblement la valeur max threshold. Donc, cela influence le résultat de la méthode Canny. Le résultat de Canny n'est pas peut-être correspondant à celui de Sobel, de Prewitt et de Laplace pour chaque image. Par conséquent, la valeur P de Canny est moine que celle de Prewitt est son incidence.

Est-ce que les trois mesures utilisées sont utiles ou est-ce que l'une semble mieux adaptée pour ce travail ?

Selon moi, deux valeurs P et TFN sont très utiles. La valeur P représente la performance d'un détecteur et la valeur TFN représente l'efficacité de trouver des contours. Quand la valeur TFN est basse, le détecteur est efficace. La valeur TFP est aussi utile, mais dans ce travail, les images de contours ont plusieurs bruits. La valeur TFP est donc toujours haute. Par conséquent, la valeur TFP n'est pas efficace dans ce travail.

CONCLUSION

Dans ce travail, j'ai implémenté quatre détecteurs de contours de gradient et j'ai fait une évaluation de ces quatre détecteurs. Dans la plupart de cas, le filtre Prewitt a montré qu'il est un bon détecteur. Mais sa capacité de détection des contours est très basse. La méthode Laplace étant aussi un bon

détecteur, a un avantage est qu'il existe toujours des contours pour toutes les directions, mais elle a aussi le faible comme Prewitt. Dans ce cas-là, Canny a deux points forts :

- la valeur TFN est la plus basse. Canny est bien détecté.
- la valeur P est la plus haute dans l'image ayant des bruits qui sont difficiles à éliminer.

Il n'existe pas le filtre parfait qui marche bien dans tous les cas avec la performance très haute. L'importance c'est que l'on sait choisir et appliquer quel détecteur à quel cas pour obtenir les résultats meilleurs.