Introduction to computer vision

Mean-shift

par Nicolas Jeannerod

1 décembre 2014

J'ai codé l'algorithme `mean-shift` en `Octave` (encore une fois, je n'ai pas testé la compatibilité `Matlab`) selon les instructions trouvés dans le `.pdf` joint à votre mail.

Je me suis attaché à faire de nombreuses mesures de l'efficacité en temps de l'algorithme.

Basic implementation

Cette partie était assez facile, puisqu'il n'y avait pas vraiment de réflexion à fournir.

J'ai choisi de coder `findpeak` récursivement, et de ne pas y mettre de boucle. Je trouvais le code plus propre comme ça. Je ne sais pas si l'efficacité est au rendez-vous, il faudrait tester ça. À mon avis, les performances doivent être médiocres, car je vois mal `Octave` faire de l'optimisation de fonctions récursives. Pour la rendre récursive, il a fallu lui permettre d'accepter un `id` de point *ou* un point directement. Ce n'est pas très propre, et j'aurais préféré le faire avec des arguments optionnels.

Le codage de `meanshift` s'est fait sans encombres, et les résultats ont été concluants : sur `two_clusters.mat`, je trouve des résultats extrêmement proches de ceux de l'énoncé.

L'état des scripts à ce moment là se trouve dans le dossier `1/`.

Optimizations

J'ai suivi ici l'évolution de l'efficacité de mon script avec de nombreuses mesures. Le principe de ces mesures est simplement de lancer `100` fois l'algorithme sur `two_clusters.mat`, pour ne pas prendre en considération le temps de chargement d'`Octave`, ou des données de `two_clusters.mat`. J'ai fait ceci de nombreuses fois, et je vous met ici les résultats.

Optimisations		**Temps (s)**
/		206.05
Mean		205.34
ml_sqrDist		118.60
First bassin optimization		12.05
	$c=3$	60.51
Second bassin optimization	$c=2$	33.12
	$c=1$	

Tableau 1. Gain de temps des optimisations

Quelques remarques sur ces temps :

Tout d'abord, je suis content de voir que mon script de la partie `1/` était déjà assez propre au niveau des boucles, puisque la modification du calcul de la moyenne ne me fait presque rien gagner. En revanche, le calcul des distances euclidiennes aurait pu être meilleur. Je pense en particulier au choix de passer toute la matrice à la racine carrée, alors que je pourrais passer au carré le scalaire avec lequel on la compare.

On peut trouver les scripts produits à ce stade dans `2.1/`.

La première optimisation en utilisant les bassins d'abstraction est vraiment très efficace. Bien sur, il faut penser à ne pas exécuter la boucle `for` quand la valeur a déjà été calculée. La deuxième optimisation me parait moins efficace, ce qui me laisse à penser que j'ai fait une erreur dans son code, ou un mauvais choix. J'ai tout de même travaillé pour la suite avec cette optimisation là.

On peut trouver les scripts produits à ce stade dans `2.2/`, et dans la racine du dossier.

Image segmentation

Malgré quelques difficultés à utiliser correctement `rgb2luv` et `luv2rgb`, cette partie s'est revelée assez rapide, le code de la fonction `segment_meanshift` n'étant qu'une légère surcouche à `meanshift`.

Pour les résultats, tous se trouvent dans le dossier `results/`. Pour les `testdata*.mat`, le `r` choisi est `3` (et `c` est égal à `4` comme demandé), même si ce n'est pas forcément très satisfaisant pour le troisième jeu de données.

Les images produites pas `segment_meanshift` se trouvent dans `results/images/`. Elles sont toutes nommées de la même façon : `<nom-original>_r<r>_c<c>.png` où `<nom-original>` est le nom de l'image avant traitement (`hat`, `man`, `sheep`, `star`), `<r>` est la valeur de `r`, `<c>` est la valeur de `c`.

r	5	10	20
c			
1	14.180	1.788	0.612
2	102.296	18.660	3.312
4	457.280	101.840	62.724

Tableau 2. Temps d'exécution de `segment_meanshift` sur `hat.png`

J'ai fait des mesures du temps passé à faire ce calcul en fonction de `r` et `c`. J'affiche ici les résultats pour `hat.png` et `man.png`. On constate que l'action des paramètres joue vraiment violemment sur le temps de calcul. Cependant, la qualité s'en ressent beaucoup. Pour `sheep` par exemple, une grande valeur de `r` ne laisse même plus apparaitre les moutons sur l'image.

r	5	10	20
c			
1	30.832	4.636	0.716
2	151.928	33.068	4.348
4	477.420	151.712	19.592

Tableau 3. Temps d'exécution de `segment_meanshift` sur `man.png`

Voilà. C'était bien, mais c'est fini.