TP Reconstruction
Master PPMD
ENSG

Bruno VALLET - IGN/MATIS

February 28, 2020

1 Introduction

L'objectif de ce TP est de vous faire appréhender certains enjeux de l'extraction de primitives par l'intermédiaire du problème de l'extraction de droites dans une image de contours. Il s'agira donc d'implémenter des algorithmes permettant de retrouver une puis plusieurs droites dans un ensemble de pixels donné:

$$\mathcal{X} = \{\mathbf{x}_i = (l_i, c_i)^t \in \mathbb{R}^2\}_{i=1..n}$$

2 Estimation d'une droite aux moindres carrés

2.1 La fonction `Coords(img,seuil)`

- Ecrivez une fonction `Coords(img, seuil)` qui liste les coordonnées $\mathbf{x} = (l, c)$ d'une image RGB dont le niveau du premier canal est supérieur au seuil.
- Appliquez cette fonction à l'image `hyst.png` avec un seuil de 200. Les points résultants sont approximativement répartis sur une droite $\mathcal{D}$.

2.2 La fonction `EstimationDroiteL2(coords)`

La façon la plus simple de retrouver $\mathcal{D}$ est de minimiser l'erreur (aussi appelée norme L_2):

$$\mathcal{E}_{L_2}(\mathcal{D}) = \sum_{i=1}^{n} dist(\mathbf{x}_i, \mathcal{D})^2$$

La droite estimée $\mathcal{D}$ sera cherchée sous la forme $\mathbf{n}.\mathbf{x} - r = 0$, c'est a dire que nos inconnues sont les coordonnées de $\mathbf{n}$ et r. L'énergie L_2 est alors:

$$\mathcal{E}_{L_2}(\mathbf{n}, r) = \sum_{i=1}^{n} (\mathbf{n}.\mathbf{x} - r)^2$$

sous la contrainte $||\mathbf{n}|| = 1$. La solution est (cf cours) donnée par:

$$r = \mathbf{n}.\mathbf{g} \quad \mathbf{g} = \sum_{i=1}^{n} \mathbf{x}_i / n$$

Et $\mathbf{n}$ est le vecteur propre de:

$$E_{L_2} = \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{g})^t (\mathbf{x}_i - \mathbf{g})$$

associée à sa plus petite valeur propre.

- Construisez le vecteur $\mathbf{g}$ et la matrice d'énergie E_{L_2} dans la fonction `EstimationDroiteL2(coords)`.
- Calculez les valeurs de r et $\mathbf{n}$ définissant la meilleure droite. On utilisera une fonction permettant de calculer les vecteurs/valeurs propres (eigen vector/values) d'une matrice.
- Trouvez les points extrêmes $\mathbf{x}_{min}$ et $\mathbf{x}_{max}$ qui maximisent et minimisent le déterminant $det(\mathbf{x}, \mathbf{n})$
- Projetez ces points sur la droite. La fonction `EstimationDroiteL2(coords)` doit retourner ces deux point qui définissent un segment confondu avec la droite
- Dessinez ce segment sur l'image `hyst.png`. On pourra utiliser pour cela le module python PIL.

3 Estimation d'un droite par RANSAC

- Appliquez la fonction `Coords(img,seuil)` à l'image `hyst.png` avec un seuil de 100 pour obtenir la liste $\mathcal{X}$ des coordonnées de tous les points de contour. Certains sont approximativement répartis sur plusieurs segments de droites mais pas tous.

Nous allons maintenant implémenter un algorithme pour retrouver ces doites très employé dans la communauté de la vision par ordinateur: le RANdom SAmple Consensus (RANSAC).

RANSAC va tirer de nombreux échantillons de 2 points au hasard dans la donnée $\mathcal{X}$ et chercher si la droite défini par ces échantillons explique bien $\mathcal{X}$, c'est a dire si il est proche de beaucoup de points de $\mathcal{X}$.

3.1 Selection d'un échantillon

- Ecrivez une fonction `RANSAC(coords, Niter, seuil)` qui contient pour l'instant uniquement une boucle `for` tournant un nombre donné $n_{iter} = 100$ par exemple de fois. C'est la boucle principale de tirage d'échantillons de RANSAC.
- Dans cette boucle principale de RANSAC, tirez deux points (les échantillons) de la liste `coords` au hazard.

3.2 Distance d'un point a une droite

RANSAC nécessite de calculer les distances entre les points de données et les droites définies par les échantillons. On rappelle que la distance du point A à la droite $\mathcal{D}$ définie par deux points P_1 et P_2 est:

$$dist(A, \mathcal{D}) = |\overrightarrow{P_1A}.\overrightarrow{n})| \quad \overrightarrow{n} = \frac{\overrightarrow{P_1P_2}^{\perp}}{||\overrightarrow{P_1P_2}||}$$

$\overrightarrow{n}$ est la normale (unitaire) de $\mathcal{D}$

- Ecrivez une fonction `Normale(P1, P2)` qui renvoie le vecteur $\overrightarrow{n}$ défini ci-dessus.

3.3 Sélection de la meilleure droite

Le score de la droite est son nombre d'inliers, c'est a dire le nombre de points dont la distance à la droite est inférieure au seuil.

- Dans la boucle principale de RANSAC, comptez le nombre d'inliers pour chaque droite.
- Utilisez ce calcul pour retenir la meilleure droite (sous la forme des deux échantillons qui la définissent), c'est a dire de celle qui a le plus d'inliers.
- Après la boucle principale, construisez la liste des inliers de cette meilleure droite
- Réestimez cette meilleure droite aux moindres carrés en appliquant la fonction `EstimationDroiteL2(coords)` aux inliers.
- Dessinez cette droite sur l'image.
- Testez votre algorithme, jouez avec les paramètres.

4 Estimation de plusieurs Droites par RANSAC

- Modifiez la fonction `RANSAC(coords, Niter, seuil)` pour qu'elle renvoie les listes d'inliers et outliers de la meilleure droite trouvée.
- Ecrivez une fonction `MultiRANSAC(coords, Niter, seuil, Nmin)` qui itère la fonction `RANSAC(coords, Niter, seuil)` tant que celle ci renvoie plus de n_{min} inliers. A chaque itération, il faut lui passer les outliers de l'itération précédente.
- A chaque itération, ré-estimez la meilleure droite aux moindres carrés et ajouter le résultat dans une liste. `MultiRANSAC` renvoie cette liste.
- Dessinez toutes les droites renvoyées sur l'image.
- Testez votre algorithme, jouez avec les paramètres.