TP3 – Estimation robuste

Exercice 1 : estimation du point de fuite dans une image

Lancez le script exercice_0, qui affiche l'image d'un parquet constitué de lames *parallèles*, ainsi que les segments détectés par l'algorithme du TP3 de Probabilités (méthode LSD, pour *Line Segment Detection*). Un ensemble de droites parallèles de l'espace 3D forment, par projection perspective dans l'image, un ensemble de droites concourantes qui se croisent en un point F appelé *point de fuite*. Le but de cet exercice est d'estimer la position de ce point de fuite.

Il a déjà été vu (cf. TP2 de Statistiques) qu'une droite D du plan est représentable par son *équation cartésienne normalisée* $x\cos\theta + y\sin\theta = \rho$, où (ρ,θ) sont les coordonnées polaires de la projection orthogonale Q sur D de l'origine O du repère. Il est rappelé que les coordonnées cartésiennes (x_Q, y_Q) de Q sont telles que :

- La distance à l'origine de Q vaut $\rho = \sqrt{x_Q^2 + y_Q^2} \in \mathbb{R}^+$;
- L'angle polaire $\theta \in]-\pi,\pi]$ de Q se calcule, en Matlab, à l'aide de l'expression $\text{atan2}(y_Q, x_Q)$.

Les paramètres (ρ_i,θ_i), $i \in \{1,\ldots,n\}$, des n droites D_i portant les alignements détectés dans l'image de parquet sont stockés dans deux vecteurs `rho` et `theta` de même taille $n\times 1$. Si une droite d'équation cartésienne normalisée $x\cos\theta + y\sin\theta = \rho$ passe par le point $F = (x_F, y_F)$, alors :

$$x_F\cos\theta + y_F\sin\theta = \rho \tag{1}$$

Par ailleurs, les coordonnées polaires (ρ_F,θ_F) de F sont liées à ses coordonnées cartésiennes (x_F, y_F) par :

$$x_F = \rho_F\cos\theta_F \qquad \text{et} \qquad y_F = \rho_F\sin\theta_F \tag{2}$$

et réciproquement :

$$\rho_F = \sqrt{x_F^2 + y_F^2} \qquad \text{et} \qquad \theta_F = \text{atan2}\,(y_F, x_F) \tag{3}$$

On déduit de (1) et (2) :

$$\rho = \rho_F\,(\cos\theta_F\cos\theta + \sin\theta_F\sin\theta) \tag{4}$$

c'est-à-dire :

$$\rho = \rho_F\cos(\theta - \theta_F) \tag{5}$$

Par conséquent, si l'on reporte les points $P_i = (\rho_i,\theta_i)$, $i \in \{1,\ldots,n\}$, ayant pour coordonnées les paramètres (ρ_i,θ_i) de n droites concourantes en F, dans un repère cartésien ayant comme axes θ en abscisse et ρ en ordonnée, ces points doivent être portés par une sinusoïde d'équation (5). L'estimation des paramètres (ρ_F,θ_F) peut donc être effectuée grâce à cette contrainte.

Complétez la fonction estimation_F, appelée par le script exercice_1, qui estime les coordonnées (ρ_F,θ_F) du point de fuite F. Pour cela, estimez les coordonnées cartésiennes (x_F, y_F) de F, en résolvant le système des n équations suivantes, correspondant aux n droites de paramètres (ρ_i,θ_i), au sens des moindres carrés (cf. TP2) :

$$x_F\,\cos\theta_i + y_F\,\sin\theta_i = \rho_i, \qquad i \in \{1,\ldots,n\} \tag{6}$$

qui peuvent être réécrites sous forme matricielle $A\,X = B$, avec $X = [x_F, y_F]^\top$, puis calculez (ρ_F,θ_F) grâce à (3).

Lorsque le résultat du script exercice_1 vous semble satisfaisant, relancez le script exercice_0 en lisant le fichier `bateau.mat` et au lieu de `parquet.mat`. Relancez ensuite le script exercice_1 : vous constatez que l'estimation du point de fuite n'est plus satisfaisante. L'explication est simple : cette nouvelle image contient deux ensembles de lignes parallèles, qui forment un quadrillage régulier sur le pont du bateau. Les points $Q_i = (\rho_i,\theta_i)$, $i \in \{1,\ldots,n\}$, se situent donc maintenant sur deux sinusoïdes, ayant deux équations de la forme (5), qui correspondent à deux points de fuite. Une façon d'estimer simultanément ces deux points de fuite est d'utiliser l'*estimation robuste*, dont la plus connue est l'*algorithme RANSAC*.

Algorithme RANSAC

RANSAC (abréviation de *RANdom SAmple Consensus*) est un algorithme itératif d'estimation robuste, publié par Fischler et Bolles en 1981, qui consiste à effectuer une *partition* entre les données conformes au modèle (*inliers*) et les données dites « aberrantes » (*outliers*). Cet algorithme est non déterministe : le résultat n'est garanti qu'avec une certaine probabilité, qui croît avec le nombre $k_{\max}$ d'itérations.
Le principe de RANSAC consiste à tirer aléatoirement un sous-ensemble de données de cardinal égal au nombre minimal de données permettant d'estimer le modèle (par exemple 2 si l'on estime les paramètres d'une droite de régression, 3 si l'on estime le rayon et le centre d'un cercle, etc.). Ces données sont considérées comme des données conformes au modèle (cela reste à vérifier par la suite), puis la séquence suivante est répétée en boucle :

1. Les paramètres du modèle sont estimés à partir de ce sous-ensemble de données.
2. Toutes les autres données sont testées relativement au modèle estimé, afin de détecter les données conformes, c'est-à-dire celles dont l'écart au modèle est inférieur à un seuil S_1.
3. Le modèle estimé en 1 est accepté si la proportion de données conformes est supérieure à un seuil S_2.
4. Si le modèle est accepté, il est réestimé à partir de l'ensemble des données conformes.

Le modèle retenu est celui qui minimise l'écart moyen des données conformes.

Exercice 2 : estimation de la ligne de fuite dans une image

Complétez la fonction `RANSAC_2`, qui est appelée deux fois par le script `exercice_2`, et dont le rôle est d'effectuer l'estimation d'un point de fuite à l'aide de l'algorithme RANSAC, sachant que :

- Les valeurs des paramètres de l'algorithme, à savoir $S_1 = 5$, $S_2 = 0.3$ et $k_{\max} = \frac{\mathrm{C}_n^2}{n}$, sont passées en entrée par l'intermédiaire de `parametres`.
- L'expression `randperm(n,2)`, qui tire deux entiers aléatoires distincts entre 1 et n, permet de tirer aléatoirement les indices de deux points.
- L'estimation, à l'étape 4 de l'algorithme RANSAC, peut être effectuée par la fonction `estimation_F` déjà écrite pour l'exercice 1, mais cette fonction doit être légèrement modifiée, de manière à retourner un troisième paramètre égal à l'écart moyen des données conformes, soit $\frac{1}{m}\sum_{i=1}^{m} |\rho_i - \rho^* \cos(\theta_i - \theta^*)|$, où m désigne le nombre de données conformes et (ρ^*, θ^*) les paramètres estimés.
- Avant d'estimer le deuxième point de fuite, il est nécessaire de retirer les données conformes à la première sinusoïde, sans quoi le même point de fuite serait estimé deux fois !

Au vu du résultat obtenu, pensez-vous que le pont du bateau était horizontal au moment de la prise de vue ?

Exercice 3 : retour sur le TP1 (exercice facultatif)

Lancez le script `donnees_aberrantes`, qui affiche un nuage de points tirés aléatoirement au voisinage d'un cercle, auxquels sont ajoutés une certaine proportion de points tirés aléatoirement selon une loi uniforme à l'intérieur de la fenêtre d'affichage. Ces dernières constituent donc des données aberrantes vis-à-vis du cercle. Le but de cet exercice est d'effectuer une estimation robuste du cercle.

Dans un premier temps, complétez les fonctions `G_et_R_moyen` et `estimation_C_et_R` pour l'estimation du centre et du rayon, que vous avez déjà utilisées dans le TP1. En suivant, complétez la fonction `RANSAC_3`, appelée par le script `exercice_3`, qui doit effectuer l'estimation du centre et du rayon d'un cercle selon l'algorithme RANSAC, sachant que :

- Les valeurs des paramètres sont fixées à $S_1 = 2$, $S_2 = 0.5$ et $k_{\max} = \frac{\mathrm{C}_n^3}{n}$.
- Le cercle passant par trois points distincts est unique. Son centre C et son rayon R peuvent être déterminés en utilisant le prototype de la fonction `[C,R] = cercle_3_points(x,y)`, qui vous est fournie.
- L'estimation, à l'étape 4 de l'algorithme RANSAC, doit être effectuée par maximum de vraisemblance car, contrairement à l'exercice 2, ce nouveau problème n'est plus linéaire. Il vous est donc conseillé de vous inspirer de l'exercice 2 (ou de l'exercice 4) du TP1 de Statistiques pour écrire la fonction `RANSAC_3`.