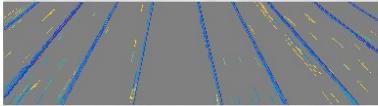


## TP3 – Estimation robuste

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.

### Ensembles candidats



### Alignements



### Localisation du point de fuite

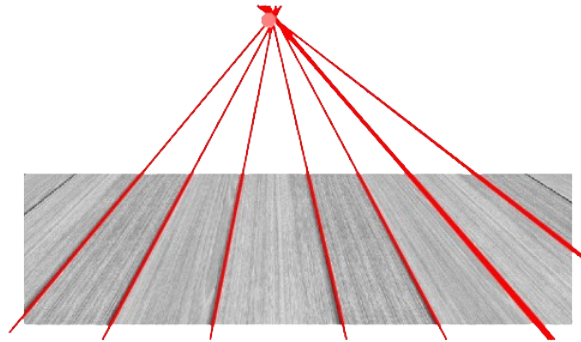


FIGURE 1 – À gauche : ensembles candidats  $E$  puis ceux conservés par détections d'alignements, comme vu dans le TP3 de Probabilités. À droite : point de fuite estimé à partir des droites conservées précédemment.

## Exercice 1 : estimation du point de fuite dans une image

Il a déjà été vu dans le TP1 de Statistiques qu'une droite  $D$  du plan est représentable par son *équation cartésienne normalisée*  $x \cos \theta + y \sin \theta = \rho$ , où  $(\rho, \theta)$  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 `atan2(y_Q, x_Q)`.

Les paramètres  $(\rho_i, \theta_i)$ ,  $i \in \{1, \dots, n\}$ , 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 :

$$x_F \cos \theta + y_F \sin \theta = \rho \quad (1)$$

Par ailleurs, les coordonnées polaires  $(\rho_F, \theta_F)$  de  $F$  sont liées à ses coordonnées cartésiennes  $(x_F, y_F)$  par :

$$x_F = \rho_F \cos \theta_F \quad \text{et} \quad y_F = \rho_F \sin \theta_F \quad (2)$$

et réciproquement :

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

On déduit alors de (1) et (2) :

$$\rho = \rho_F (\cos \theta_F \cos \theta + \sin \theta_F \sin \theta) = \rho_F \cos(\theta - \theta_F) \quad (4)$$

Par conséquent, si l'on reporte les points  $P_i = (\rho_i, \theta_i)$ ,  $i \in \{1, \dots, n\}$ , ayant pour coordonnées les paramètres  $(\rho_i, \theta_i)$  de  $n$  droites concourantes en  $F$ , dans un repère cartésien ayant comme axes  $\theta$  en abscisse et  $\rho$  en ordonnée, ces points doivent être portés par une sinusoïde d'équation (4). L'estimation des paramètres  $(\rho_F, \theta_F)$  peut donc être effectuée grâce à cette contrainte.

Complétez la fonction `estim_param_F`, appelée par le script `exercice_1`, qui estime les coordonnées  $(\rho_F, \theta_F)$  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  $(\rho_i, \theta_i)$ , au sens des moindres carrés (cf. TP1 de Statistiques) :

$$x_F \cos \theta_i + y_F \sin \theta_i = \rho_i, \quad i \in \{1, \dots, n\} \quad (5)$$

qui peuvent être réécrites sous forme matricielle  $AX = B$ , avec  $X = [x_F, y_F]^\top$ , puis calculez  $(\rho_F^*, \theta_F^*)$  grâce à (3). Calculez également l'écart moyen  $\bar{\varepsilon}$  des  $n$  droites au point de fuite estimé, correspondant aussi à la moyenne des valeurs absolues des résidus, et qui s'écrit :

$$\bar{\varepsilon} = \frac{1}{n} \sum_{i=1}^n |\rho_i - \rho_F^* \cos(\theta_i - \theta_F^*)| = \frac{1}{n} \sum_{i=1}^n |AX - B| \quad (6)$$

Lorsque le résultat du script `exercice_1` vous semble satisfaisant, relancez le script `exercice_0` en lisant le fichier `bateau.mat` 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  $P_i = (\rho_i, \theta_i)$ ,  $i \in \{1, \dots, n\}$ , se situent donc maintenant sur deux sinusoïdes, ayant deux équations de la forme (4), 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 *RAN*dom *SAM*ple *CON*sensus) 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 l'algorithme 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  $S_{\text{ecart}}$ .
3. Le modèle estimé en 1 est accepté si la proportion de données conformes est supérieure à un seuil  $S_{\text{prop}}$ .
4. Si le modèle est accepté, il est **ré-estimé à partir de l'ensemble des données conformes**.

Au final, le modèle retenu est celui qui minimise l'écart moyen  $\bar{\varepsilon}$  des données conformes.

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

Complétez ensuite la fonction `RANSAC_2droites`, qui est appelée une première fois par le script `exercice_2` pour estimer le premier point de fuite à l'aide de l'algorithme RANSAC, sachant que :

- Les valeurs des paramètres de l'algorithme, à savoir  $S_{ecart} = 5$ ,  $S_{prop} = 30\%$  et  $k_{\max} = \lfloor \frac{1}{n} \binom{n}{2} \rfloor$ , sont passées en entrée par l'intermédiaire de `parametres`.
- L'estimation, à l'étape 4 de l'algorithme RANSAC, peut être effectuée par la fonction `estim_param_F` déjà écrite pour l'exercice 1, en utilisant l'écart moyen des données conformes en sortie comme critère de sélection, soit  $\frac{1}{m} \sum_{i=1}^m |\rho_i - \rho_F^* \cos(\theta_i - \theta_F^*)|$ , où  $m$  désigne le **nombre de données conformes** et  $(\rho_F^*, \theta_F^*)$  les paramètres estimés.

Une fois le premier point de fuite estimé, lancez le script `exercice_2bis` qui estime en plus le second point de fuite avec l'algorithme RANSAC qui est appliqué une seconde fois, mais seulement sur les droites qui n'ont pas été retenues pour l'estimation du premier point de fuite. Au vu du résultat obtenu, pensez-vous que le pont du bateau était horizontal au moment de la prise de vue ?

## Exercice 3 : estimation des paramètres d'un cercle (TP de Probabilités)

Lancez le script `exercice_3`, 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, copiez les fonctions suivantes que vous avez déjà utilisées en TP de Probabilités pour l'estimation du centre et du rayon d'un cercle : `calcul_G_et_R_moyen`, `tirages_aleatoires_uniformes` et `estimation_C_et_R`.

En suivant, complétez la fonction `RANSAC_3points`, appelée par le script `exercice_3bis`, 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  $S_{ecart} = 3$ ,  $S_{prop} = 50\%$  et  $k_{\max} = \lfloor \frac{1}{n} \binom{n}{3} \rfloor$ .
- 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_3p,R_3p] = estim_param_cercle_3points(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.