LOCALISATION ET CARTOGRAPHIE SIMULTANÉES EN ROBOTIQUE (SLAM)

Depuis les années 1990, le SLAM (Simultaneous localization and mapping ≡ Localisation et cartographie simultanées) est sans doute l'un des problèmes de Robotique ayant donné lieu au plus grand nombre de contributions. Considérant un robot en mouvement dans un environnement inconnu (sans connaissance a priori), il consiste à construire un modèle du monde, et, concomitamment, à localiser le robot dans celui-ci. On obtient alors une carte métrique, pouvant être exploitée à des fins de navigation ou pour d'autres objectifs.

Le SLAM combine des données issues de capteurs proprioceptifs, qui délivrent une information relative au déplacement du robot (codeurs de roues, etc.) ainsi que des capteurs extéroceptifs, qui renseignent le robot sur son environnement (vision, télémètre laser, etc.). La version la plus élémentaire de l'algorithme exploite des primitives géométriques de l'environnement fixes et identifiables sans ambiguïté entre deux perceptions. Reprenant une terminologie utilisée en navigation maritime, ces primitives sont appelées « amers ». Lorsque le SLAM est basé sur des séquences d'images, ces amers peuvent être des cadres de portes, des posters quadrangulaires, etc.

De nombreuses approches du SLAM ont reposé sur le filtrage de Kalman et ses extensions. Le vecteur d'état x à estimer est naturellement constitué de la situation x^R du robot et des situations $\mathsf{x}^1, \ldots, \mathsf{x}^N$ des N primitives du modèle dans un repère lié au monde. La dynamique a priori de x^R est souvent basée sur l'odométrie du robot, *i.e.*, sur la reconstitution de son mouvement global à partir de son modèle cinématique et des déplacements (connus) des roues. Le sous-ensemble $\mathsf{x}^1, \ldots, \mathsf{x}^N$ relatif aux amers n'évolue pas au fil du temps, ceux-ci étant fixes. Le vecteur des observations y assimilées par le filtre est constitué des données brutes perçues. Ces mesures doivent être appariées avec leur contrepartie dans $\mathsf{x}^1, \ldots, \mathsf{x}^N$, *i.e.* telle perception doit être associée à tel amer correspondant à tel sous-ensemble de composantes de x. Ce problème d'*association de données* peut être significativement simplifié (voire éliminé) si les amers sont convenablement sélectionnés.

On réalise alors itérativement le cycle *[1. bouger le robot]* → *[2. apparier les données perçues]* → *[3. mettre à jour le modèle]*. L'étape *[1.]* sert de base à la prédiction de x entre deux instants successifs $k-1$ et k préalablement au prélèvement d'une mesure $\mathsf{y}_k = y_k$ à l'instant k ; l'étape *[2.]* consiste à associer y_k au sous-vecteur x^n correspondant dans x, $n \in \{1, \ldots, N\}$, en augmentant si besoin N si l'amer relatif à y_k ne figure pas déjà dans x ; l'étape *[3.]* consiste à assimiler y_k. La dimension de x croît donc au fil du temps tant qu'apparaissent de nouveaux amers. La dimension de y peut ne pas être fixe.

En général, le problème est fortement non linéaire. On en propose toutefois ici une version extrêmement simplifiée. Ainsi, le robot est doté d'une locomotion omnidirectionnelle, et est assimilable à un point R. Il effectue un mouvement circulaire plan. L'environnement comporte deux amers A_1 et A_2. Le robot mobile et les amers fixes sont repérés par leurs coordonnées (u^R, v^R) et (u^1, v^1),(u^2, v^2) relatives à un repère $(O, \overrightarrow{x}, \overrightarrow{y})$, où O désigne le centre de la trajectoire circulaire idéale que décrirait le robot en l'absence de perturbations sur sa dynamique (glissement, etc.).

Le capteur placé sur le robot (idéalisé, car n'ayant pas d'équivalent « physique ») délivre, à une erreur de mesure près, les composantes dans la base $(\overrightarrow{x}, \overrightarrow{y})$ des vecteurs $\overrightarrow{RA_1}$ et $\overrightarrow{RA_2}$ qui séparent le robot des amers *à condition que ces composantes soient positives*. En d'autres termes, le capteur délivre l'ensemble des composantes $l_n = \overrightarrow{RA_n}.\overrightarrow{x}$, $m_n = \overrightarrow{RA_n}.\overrightarrow{y}$ pour tout $n \in \{1, 2\}$ tel que A_n soit

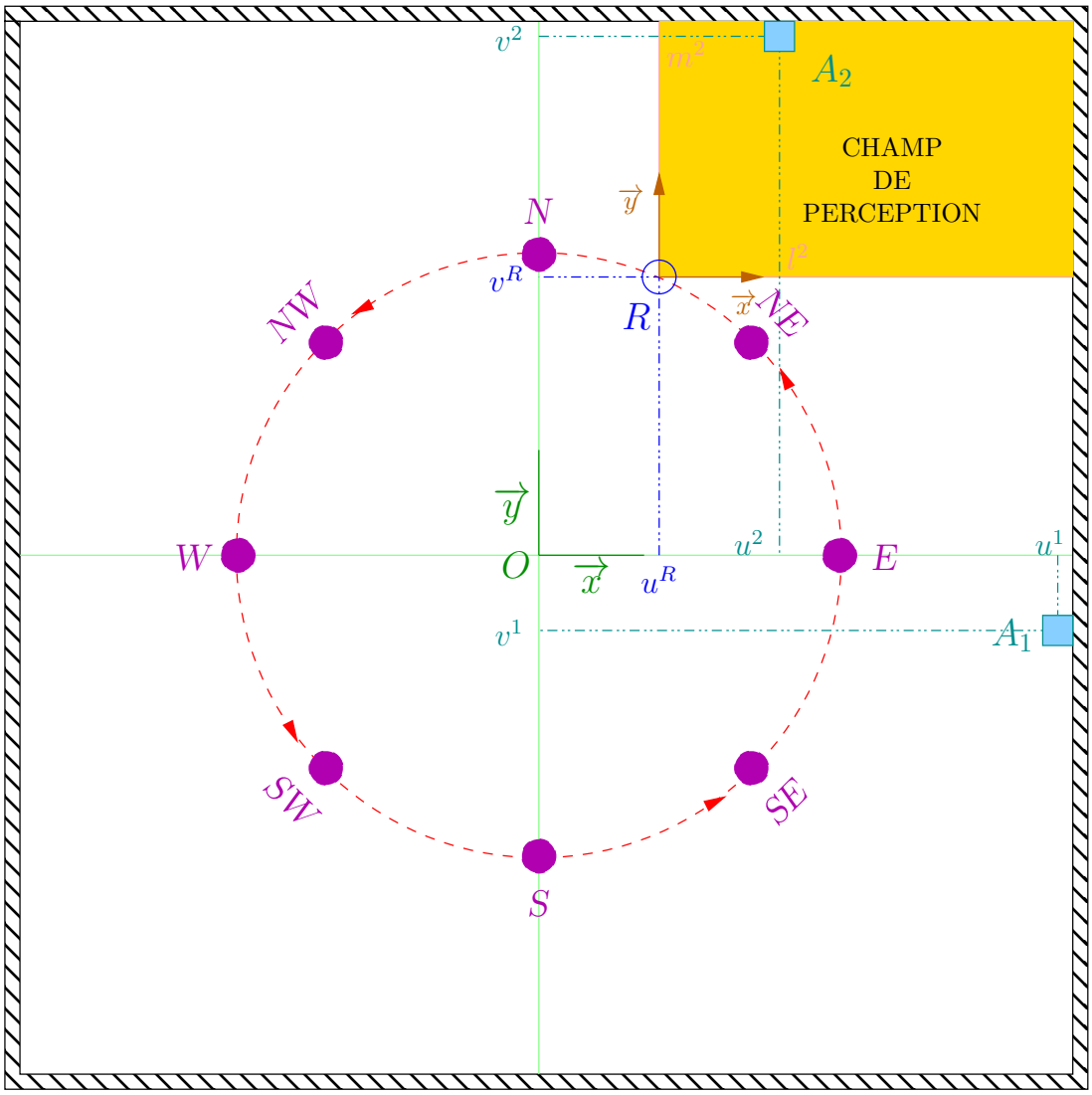

Figure 1 – Problème considéré.

« recouvert » par le premier quadrant délimité par $(R, \overrightarrow{x}, \overrightarrow{y})$. Notons que le repère $(R, \overrightarrow{x}, \overrightarrow{y})$ est en translation circulaire mais conserve une orientation constante. Les mesures sont prélevées lorsque le robot se trouve dans l'une des huit zones *Est*, *Nord-Est*, *Nord*, *Nord-Ouest*, *Ouest*, *Sud-Ouest*, *Sud*, *Sud-Est*, respectivement désignées par *E*, *NE*, *N*, *NW*, *W*, *SW*, *S*, *SE*. Le robot a en permanence connaissance du nombre d'amers situés dans le champ de perception de son capteur. Le problème est résumé sur la Figure 1.

Notations et Données

On désigne par $\mathbb{I}$ toute matrice identité et par $\mathbb{O}$ toute matrice constituée de zéros. Les dimensions de ces matrices sont sous-entendues lorsqu'elles découlent du contexte. La notation $\text{blkdiag}(A_1, \ldots, A_N)$ consiste en la matrice bloc-diagonale constitutée de l'assemblage de $A_1, \ldots, A_N$. L'opérateur de transposition est noté $'$. Enfin, $\mathsf{x} \sim \mathcal{G}(\bar{x}, P)$ signifie que la variable aléatoire (a priori multidimensionnelle) x suit une loi normale de moyenne $\bar{x}$ et de covariance P.

Soient $\mathsf{x}^R = (u^R, v^R)'$, $\mathsf{x}^1 = (u^1, v^1)'$, $\mathsf{x}^2 = (u^2, v^2)'$, et $\mathsf{x} = \big((\mathsf{x}^R)', (\mathsf{x}^1)', (\mathsf{x}^2)'\big) \in \mathbb{R}^6$ le vecteur d'état (caché) complet. Les valeurs de ces quantités à l'instant k sont respectivement désignées par x_k^R, u_k^R,

v_k^R, x_k^1, u_k^1, v_k^1, x_k^2, u_k^2, v_k^2, x_k.

Dans l'ensemble du sujet, on note

$$\mathsf{x}_{k+1}^R = F^R \mathsf{x}_k^R \tag{1}$$

l'équation d'état qui unirait les valeurs de x^R à deux instants successifs (*i.e.*, passage $E \to NE \to N \to NW \to W \to SW \to S \to SE$) si la trajectoire du robot était parfaitement circulaire. Les amers sont supposés admettre une dynamique nulle, et satisfaire l'équation

$$\forall n \in \{1, 2\},\ \mathsf{x}_{k+1}^n = \mathsf{x}_k^n. \tag{2}$$

Du fait de phénomènes inévitables (glissement, etc.) qui font que le robot ne suit pas parfaitement la trajectoire idéale décrite par (1), le vecteur d'état complet $\mathsf{x} = \left((\mathsf{x}^R)', (\mathsf{x}^1)', (\mathsf{x}^2)'\right)$ obéit à une équation d'état stochastique à temps discret de la forme

$$\mathsf{x}_{k+1} = F\mathsf{x}_k + \mathsf{w}_k, \tag{3}$$

où le bruit de dynamique w est blanc stationnaire et vérifie $\forall k,\ \mathsf{w}_k \sim \mathcal{G}(0, Q_w)$. La matrice de dynamique F et la matrice de covariance de bruit Q_w (données) admettent donc une structure bloc-diagonale, *i.e.*, $F = \text{blkdiag}(F^R, \mathbb{I}, \mathbb{I})$ et $Q_w = \text{blkdiag}(Q_w^R, \mathbb{O}, \mathbb{O})$.

Le robot part de la zone E, de sorte que le vecteur aléatoire d'état à l'instant initial $k = 0$ vérifie

$$\mathsf{x}_0 \sim \mathcal{G}(m_{x_0}, P_0), \tag{4}$$

avec m_{x_0} et P_0 donnés. Pour tout k, x_0 et w_k sont supposés mutuellement indépendants.

Soient l_k^1, m_k^1, l_k^2, m_k^2 les mesures (à valeurs réelles) délivrées par le capteur à tout instant $k \geq 1$, et z_k le vecteur regroupant celles-ci. Selon les valeurs de k, la visibilité des amers change, de sorte que la dimension de z_k varie dans $\{0, 2, 4\}$. L'équation de mesure stochastique qui unit z et x s'écrit

$$\mathsf{z}_k = H_k \mathsf{x}_k + \mathsf{v}_k, \tag{5}$$

avec v un bruit de mesure blanc Gaussien stationnaire, tel que $\mathsf{v}_k \sim \mathcal{G}(0, R_v)$. Pour tous k, k', x_0, w_k, $\mathsf{v}_{k'}$ sont supposés mutuellement indépendants. Les dimensions des matrices (données) d'observation H_k et de covariance R_v du bruit varient éventuellement en chaque instant k avec la dimension de z_k.

On fournit la fonction MATLAB `simulationDonnees.m`, dont le prototype est décrit Figure 2. Cette fonction permet de simuler une expérience. Par conséquent, elle fournit un résultat différent lors de chacun de ses appels. Outre le nombre d'instants `N`, l'horizon temporel `T`, les trajectoires temporelles `Z` et `X` des processus de mesure et d'état `z` et `x` pour l'expérience considérée, elle fournit également la matrice `F` (F), la matrice `Hfull` (*i.e.*, H lorsque les deux amers sont visibles), ainsi que les statistiques `mX0`,`PX0`,`Qw`,`Rv` (respectivement m_{X_0}, P_0, Q_w, R_v). Il convient de préciser que si l'expérience était menée sur un robot réel, seuls `N`,`T`,`Z` seraient accessibles (on pourrait toutefois obtenir `X` *via* un système de capture de mouvement par exemple).

On fournit également la fonction `ellipse`, qui, selon le prototype reporté Figure 3, permet de tracer l'ensemble de volume minimal dans lequel se réalise une variable aléatoire Gaussienne à deux dimensions avec une probabilité de 99%.

Travail demandé

Élaboration de la représentation d'état stochastique du système

1. Montrer que dans (1), F^R s'écrit

$$F^R = \begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}. \tag{6}$$

```
function [N,T,Z,F,Hfull,mX0,PX0,Qw,Rv,X] = simulationDonnees(plot_p);
%SIMULATIONDONNEES
%  Simulation d'une expérience : déplacement du robot, recueil des mesures, etc.
%  Syntaxe d'appel : [N,T,Z,F,Hfull,mX0,PX0,Qw,Rv,X] = simulationDonnees(plot_p);
%
%  Entrée :
%  . plot_p : si égal à 1, produit un affichage animé, sinon n'affiche rien
%
%  Sorties :
%  . N : nombre d'échantillons temporels
%  . T : vecteur des instants d'échantillonnage (1xN)
%  . Z : réalisation du processus aléatoire de mesure (4xN, éventuellement avec composantes de type NaN)
%  -> et comme ceci est de la simulation, sont également accessibles
%     . F, Hfull : matrices du modèle (respectivement (6x6), (4x6))
%     . mX0 : espérance du vecteur d'état à l'instant 0 (6x1)
%     . PX0 : covariance du vecteur d'état à l'instant 0 (6x6)
%     . Qw  : covariance du bruit de dynamique (supposé stationnaire) (6x6)
%     . Rv  : covariance du bruit de mesure (supposé stationnaire) (4x4)
%     . X   : réalisation du processus aléatoire d'état (6xN)
%
%  Z et X admettent AUTANT DE COLONNES QUE D'INSTANTS
```

FIGURE 2 – Prototype de `simulationDonnees.m`

```
function h = ellipse(mx,Px,color)
%ELLIPSE
%  Trace l'ellipse contenant 99% des réalisations d'une variable aléatoire
%  Gaussienne 2D de moyenne mx et de covariance Px.
```

FIGURE 3 – Prototype de `ellipse.m`

2. Établir l'expression de la matrice H_k dans (5) selon que (i) A_1 et A_2 sont tous deux visibles ; (ii) seul A_1 se situe dans le champ de perception du capteur ; (iii) seul A_2 se situe dans le champ de perception du capteur.
3. Exécuter un "run" de la fonction `simulationDonnees`.
   (a) Expliquer les informations fournies par l'animation graphique.
   (b) Expliquer le contenu des variables `T`,`Z`,`X`.
   (c) Donner un sens *physique* à `mX0`,`P0`,`Qw`,`Rv`.

Élaboration des équations du filtre de Kalman

4. Écrire les équations du filtre de Kalman correspondant à chaque transition du robot (parmi $[E \to NE]$, $[NE, \to N]$, etc.) en supposant que les mesures sont assimilées *en fin* de transition.
5. Isoler les équations qui peuvent être prétabulées de celles qui ne peuvent être exécutées qu'en ligne, dans le cas où tous les amers seraient visibles simultanément, et dans le cas considéré ici.

Implémentation du filtre de Kalman

6. Implémenter le filtre de Kalman sous MATLAB, lorsque son initialisation ainsi que les modèles de bruits sur lesquels il est basé sont identiques à ceux exploités dans `simulationDonnees`. Sachant que chaque $(l+1)^{\text{ème}}$ colonne de `Z` contient l'observation z_l à l'instant l, on pourra par exemple définir 5 vecteurs (lignes) de cellules (cf. `https://fr.mathworks.com/help/matlab/matlab_prog/create-a-cell-array.html`).
   - `Xpred`,`Ppred`
   - `Xest`,`Pest`
   - `Gain`

dont les $(l+1)^{\text{ème}}$ colonnes contiennent respectivement
- la prédiction $\hat{x}_{l|l-1}$ de $\mathtt{x}_l$ sur la base des mesures disponibles jusqu'à l'instant $l-1$ et la matrice de covariance $P_{l|l-1}$ de l'erreur associée ;
- l'estimé $\hat{x}_{l|l}$ de $\mathtt{x}_l$ sur la base des mesures disponibles jusqu'à l'instant l et la matrice de covariance $P_{l|l}$ de l'erreur associée ;
- le gain K_l du filtre à l'instant l.

Cette structuration du code permettra un « alignement temporel » de tous les signaux manipulés sous MATLAB, au sens où toute colonne correspondent au même instant.

7. Exécuter le filtre sur la base d'une réalisation du processus de mesure.
   (a) Sur la base des statistiques théoriques délivrées par le filtre, établir des domaines dans lesquels doit se trouver le vecteur d'état (caché) avec une certaine probabilité.
   Par exemple,
   - à chaque instant k, sur la base de $\hat{x}_{k|k}$ et $P_{l|l}$, définir pour chaque composante du vecteur d'état (caché) $\mathtt{x}_k$, un « couloir » dans lequel elle doit se réaliser avec une probabilité théorique de 99 % ;
   - (mieux !) à chaque instant k, sur la base de $\hat{x}_{k|k}$ et $P_{l|l}$, définir les ellipses de confiance dans lesquels doivent se réaliser $\mathtt{x}_k^R$, $\mathtt{x}_k^1$, $\mathtt{x}_k^2$ avec une probabilité théorique de 99 % (fonction `ellipse`).

   (b) Vérifier la cohérence de la réalisation effective du vecteur d'état avec ces considérations théoriques.

   (c) Observer le phénomène de « fermeture de boucle », où l'observation d'un amer déjà vu dans le passé permet, grâce à des corrélations établies a posteriori par le filtre entre les composantes de x, d'apporter des informations sur la situation d'autres amers !

8. Renouveler cette étude en considérant le fait que le processus innovation Γ (qui se réalise à l'instant k en l'écart $\gamma_k = z_k - \hat{z}_{k|k-1}$, z_k et $\hat{z}_{k|k-1}$ désignant les sorties réelle et prédite à l'instant k) est caractérisé par

$$\forall k,\ E(\Gamma_k) = 0\,; \quad E(\Gamma_k \Gamma_{k'}^T) = (R_v + H_k P_{k|k-1} H_k^T)\delta_{k-k'},$$
$$\text{où } \delta_h = 0 \text{ (resp. 1) si } h \neq 0 \text{ (resp. } h = 0).$$

Analyse qualitative du filtre

9. Le filtre demeure basé sur une initialisation et des modèles de bruits conformes à ceux exploités dans `simulationDonnees`. Que se passe-t-il si Q_w ou R_v sont séparément multipliées par un facteur important ?

10. Que se passe-t-il si le filtre est basé sur des valeurs de m_{x_0}, P_0, Q_w ou R_v non conformes à celles exploitées dans `simulationDonnees` ?

Élaboration des statistiques a priori du vecteur d'état

11. Sur la base de (3) et (4), établir les valeurs de l'espérance m_{x_k} et de la matrice de covariance P_k de $\mathtt{x}_k$, *i.e.*, telles que $\mathtt{x}_k \sim \mathcal{G}(m_{x_k}, P_k)$. Ces quantités sont appelées *statistiques a priori* du vecteur d'état, car elles le caractérisent en l'absence de toute mesure.

12. Sur un "run" de la fonction `simulationDonnees`, reporter pour chaque instant k l'ellipse de confiance défini par m_{x_k}, P_k dans lequel soit se trouver la réalisation de $\mathtt{x}_k$ avec une probabilité de 99 % (fonction `ellipse`).

13. Vérifier la cohérence du résultat obtenu. Que remarque-t-on ?