Filtre de Kalman

Sarah Curtit

1 Presentation du Filtre de Kalman

1.1 Le filtre de Kalman continu

1.1.1 Le modèle

Le filtre de Kalman est un modèle d'état défini par deux équations :

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) + Mw(t) & \text{équation d'état} \\ y(t) = Cx(t) + Du(t) + v(t) & \text{équation de mesure} \end{cases}$$

x est un vecteur d'état, y un vecteur de mesures, u un vecteur de commandes (données déterminées), tandis que les signaux $w(k)$ et $v(k)$ sont des bruits blancs gaussiens centrés de densité spectrale de puissance W et V respectivement. On a

$$\begin{array}{lll} E[w(k)w(k+l)^T] & = W\delta(l) & \delta(l) = 1 \text{ si l=0 ; 0 sinon} \\ E[v(k)v(k+l)^T] & = V\delta(l) & \text{(V doit être inversible)} \\ E[w(k)v(k+l)^T] & = 0 & \end{array}$$

1.1.2 Minimisation de l'erreur d'estimation

En pratique, le filtre doit retourner en sortie l'état estimé du système, noté $\hat{x}$. L'équation d'état du filtre nous est donnée par

$$\dot{\hat{x}}(t) = A_f\hat{x}(t) + B_f u(t) + K_f y(t)$$

où A_f B_f et K_f sont des matrices à déterminer.

Soit $\varepsilon(t) = x(t) - \hat{x}(t)$ l'erreur d'estimation du système. On souhaite que cet estimateur soit non biaisé, c'est à dire que $\lim\limits_{t\to+\infty} \bar{\varepsilon} = 0$ quel que soit le profil de commande et quel que soit l'état initial ($\bar{\varepsilon}$ étant l'espérance mathématique de ε).

L'écriture de ces conditions nous donne finalement l'équation du filtre de Kalman

$$\dot{\hat{x}}(t) = (A\hat{x} + Bu) + K_f(y - C\hat{x} - Du)$$

K_f est appelé le gain du filtre. C'est lui qui traduit la confiance que l'on a dans le modèle ou les mesures. Si on considère que le modèle est très fiable et les mesures très bruitées on accordera peu d'importance à celles-ci, et K_f sera petit. Si l'on fait au contraire plus confiance aux mesures qu'au modèle, K_f sera grand.

Afin d'obtenir le filtre le plus fiable possible, le gain K_f doit minimiser la variance de l'erreur d'estimation, c'est à dire la trace de la matrice de covariance de l'erreur d'estimation,

$$P(t) = E[(x(t) - \hat{x}(t))(x(t) - \hat{x}(t))^T]$$

$P(t)$ vérifiant l'équation

$$\dot{P}(t) = (A - K_f C)P(t) + P(t)(A - K_f C)^T + MWM^T + K_f V K_f^T$$

En minimisant P on obtient

$$\begin{aligned} K_f &= P(t)C^T V^{-1} \\ \dot{P}(t) &= AP(t) + P(t)A^T - P(t)C^T V^{-1} C P(t) + MWM^T \end{aligned}$$

Cette deuxième équation étant appelée équation différentielle de Riccati.

Une fois les erreurs d'initialisation corrigées (régime constant), on obtient

$$\begin{aligned} K_f &= P(t)C^T V^{-1} && \text{Désormais constant.} \\ 0 &= AP(t) + P(t)A^T - P(t)C^T V^{-1} C P(t) + MWM^T && \text{P étant la solution positive de l'équation.} \end{aligned}$$

1.2 Le filtre de Kalman discret

1.2.1 Le modèle

Dans les faits, les appareils numériques fonctionnent de manière discrète.
On utilisera donc en pratique le modèle de Kalman discret

$$\begin{cases} x(k+1) = A_d x(k) + B_d u(k) + M_d w_d(k) & \text{équation d'état} \quad x \in \mathbb{R}^n, y \in \mathbb{R}^m, w_d \in \mathbb{R}^q \\ y(k) = C_d x(k) + D u(k) + v_d(k) & \text{équation de mesure} \quad y \in \mathbb{R}^m, v_d \in \mathbb{R}^p \end{cases}$$

Les signaux w_d et v_d sont cette fois associés à des matrices de covariance W_d et V_d telles que

$$\begin{aligned} E[w_d(k)w_d(k+l)^T] &= W_d \delta(l) \\ E[v_d(k)v_d(k+l)^T] &= V_d \delta(l) \quad (V_d \text{ doit être inversible}) \\ E[w_d(k)v_d(k+l)^T] &= 0 \end{aligned}$$

On note dt le pas de temps associé au système discret.

1.2.2 Passage du modèle continu au modèle discret

Le modèle de Kalman discret est très proche du modèle continu, et on peut obtenir les nouvelles matrices discrètes à partir de celles du modèle continu. On a

$$A_d = e^{Adt} \qquad B_d = \int_0^{dt} e^{Av} B \, \mathrm{d}v \qquad M_d = I_n \qquad C_d = C \qquad V_d = \frac{V}{dt}$$

L'expression de W_d est plus compliquée puisqu'on a $W_d = \int_0^{dt} e^{Av} M W M^T e^{A^T v} \, \mathrm{d}v$
Cependant dans l'hypothèse où dt est petit par rapport au temps de réponse du système, on peut écrire

$$W_d \approx dt M W M^T$$

1.2.3 Représentation d'état, implémentation du filtre en pratique

Comme dans le modèle continu, on souhaite optimiser le gain K_f afin de minimiser l'erreur d'estimation.
Dans la suite de cette section, on distinguera

l'état prédit à l'instant k+1 connaissant toutes les mesures jusqu'à l'instant k	$\hat{x}(k+1\|k)$
l'état estimé connaissant la mesure à l'instant k+1 (après recalage)	$\hat{x}(k+1\|k+1)$

L'état prédit est calculé de façon déterministe

$$\hat{x}(k+1|k) = A_d\hat{x}(k|k) + B_du(k)$$

Tandis que l'équation d'état du filtre

$$\hat{\hat{x}}(k|k) = A_d\hat{x}(k|k) + B_du(k) + K_f(k+1)\big(y(k) - C_d\hat{x}(k|k) - D\hat{x}(k|k)\big)$$

nous donne

$$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + K_f(k+1)\big(y(k) - C_d\hat{x}(k|k) - D\hat{x}(k|k)\big)$$

La combinaison de ces deux équations nous permet d'obtenir la représentation d'état du filtre de Kalman permanent :

$$\left\{\begin{array}{lcl} \hat{x}(k|k) & = & (I - K_fC_d)\hat{x}(k|k-1) + \begin{pmatrix} K_f & -K_fD \end{pmatrix}\begin{pmatrix} y(k) \\ u(k) \end{pmatrix} \\ \hat{x}(k+1|k) & = & A_d(I - K_fC_d)\hat{x}(k|k-1) + \begin{pmatrix} A_dK_f & B_d - A_dK_fD \end{pmatrix}\begin{pmatrix} y(k) \\ u(k) \end{pmatrix} \end{array}\right.$$

C'est la deuxième équation, calculant l'état prédit à l'instant k+1 qui nous intéresse et qui sera implémenté dans l'algorythme du filtre de Kalman.

Afin de totalement déterminer cette équation, il reste cependant à déterminer K_f, choisi encore une fois afin de minimiser l'erreur d'estimation $P(k+1|k+1)$ (dont l'expression est extrêmement compliquée).

La résolution de l'équation différentielle nous donne

$$\begin{array}{rcl} K_f(k+1) & = & P(k+1|k)C_d^T(C_dP(k+1|k)C_d^T + V_d)^{-1} \\ P(k+1|k) & = & A_dP(k|k-1)A_d^T - A_dP(k|k-1)C_d^T\big(C_dP(k|k-1)C_d^T + V_d\big)^{-1}C_dP(k|k-1)A_d^T \\ & & +M_dW_dM_d^T \end{array}$$

En régime permanent la matrice de covariance de l'erreur de prédiction est constante et vérifie l'équation discrète de Riccati

$$P_p = A_dP_pA_d^T - A_dP_pC_d^T\big(C_dP_pC_d^T + V_d\big)^{-1}C_dP_pA_d^T + M_dW_dM_d^T$$

On peut en déduire le gain du filtre et la matrice de covariance de l'erreur d'estimation

$$\begin{array}{rcl} K_f & = & P_pC_d^T(C_dP_pC_d^T + V_d)^{-1} \\ P_e & = & (I - K_fC_d)P_p \end{array}$$

La plupart des solveurs actuels proposent des fonctions permettant de résoudre l'équation de Riccati. Dans le cadre de ce projet, nous avons travaillé avec Python et la fonction scipy.linalg.solve_discrete_are.

2 Application au problème

On souhaite asservir en effort un robot 6 axes. L'objectif est d'appliquer une force choisie (constante ou non) le long d'une trajectoire parcourant une surface irrégulière. Un capteur 6 axe fixé au bout du bras du robot nous indique les forces et moments appliqués par l'outil (un poussoir à bille) sur le capteur.
Afin d'effectuer la tâche demandée, nous définissons une trajectoire de base censée suivre la forme de la surface. L'outil supporté par le robot est censé tourner avec la surface pour rester en permanence normal à celle-ci. Nous fonctionnerons dans la suite dans le repère du Tool Center Point.
Avant de démarrer, nous descendons lentement le robot à la verticale du point de départ jusqu'à ce que le capteur mesure la force désirée (selon z donc). Nous démarrons ensuite la trajectoire que nous corrigeons au fur et à mesure en fonction des données renvoyées par le capteur afin de rester en permanence le plus proche possible de la force objectif. Nous pensions à l'origine utiliser un filtre de Kalman, mais la rigidité du robot devant le poussoir à bille (de raideur 3.1 N/mm) entraîne plusieurs problèmes : nous ne pouvons plus écrire de modèle dynamique et les erreurs sur la position du robot et les aspérités de la surface deviennent en apparence impossible à intégrer correctement au problème. Nous sommes donc dans l'impossibilité de définir mathématiquement le gain du Filtre.

2.1 Problème 1D

Pour simplifier le problème, considérons tout d'abord que l'outil utilisé est parfaitement perpendiculaire à la surface au point de contact et qu'il n'y a pas de frottement. Dans ce cas, on peut se permettre de considérer seulement les données selon z (repère orienté selon la normale à la surface au point courant).

On prend alors comme vecteurs d'état, de mesure et de commande respectivement

$$x = \begin{pmatrix} z & \dot{z} \end{pmatrix}, \quad y = \begin{pmatrix} F_z \end{pmatrix}, \quad u = \begin{pmatrix} \dot{z}_{robot} \\ F_{obj} \end{pmatrix}$$

ce qui simplifie grandement le problème

2.1.1 Détermination des données du problème dans le cas continu

v et w sont ici respectivement le vecteur des signaux aléatoires qui polluent les mesures y et le vecteur des signaux inconnus qui viennent perturber directement l'équation d'état du système. Ici v représente l'imprécision du capteur et w rend compte des aspérités de la surface et de l'imprécision du robot.
La matrice M relie le bruit d'état w_x à w par la relation $w_x = Mw$

En faisant pour l'instant abstraction des vecteurs d'erreurs (et donc du dernier terme de

chaque égalité) le système s'écrit dans notre cas comme

$$\begin{cases} \begin{pmatrix} \dot{z} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} . \begin{pmatrix} z \\ \dot{z} \end{pmatrix} \\ \\ \begin{pmatrix} F_z \end{pmatrix} = \begin{pmatrix} k & 0 \end{pmatrix} . \begin{pmatrix} z \\ \dot{z} \end{pmatrix} + \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} \dot{z}_{robot} \\ F_{obj} \end{pmatrix} \end{cases}$$

Le choix a été fait ici de considérer l'accélération comme nulle car le déplacement du bras du robot est commandé en fonction de la force exercée sur le ressort, il n'en dépend pas directement. –> Premier problème : le système n'est en fait pas dynamique.
La matrice C est elle déterminée à partir d'une simple loi de Hooke : $F_z = -k * d_z$.

Déterminons désormais les matrices associées vecteurs d'erreur v et w.
on note

$$\begin{array}{lll} v & = \begin{pmatrix} v_{F_z} \end{pmatrix}^T & \text{où } v_{F_z} \text{ représente l'incertitude sur la mesure } F_z \text{ du capteur} \\ w & = \begin{pmatrix} w_z \end{pmatrix} & \text{où } w_z \text{ représente l'incertitude de position du robot selon z} \end{array}$$

Deux sources supplémentaires d'erreur sont l'incertitude sur la position du robot et l'incertitude sur les aspérités de la surface. Problème : comment les prendre en compte ? Intuitivement il s'agit de d'erreurs liées à l'état du système, cependant le système n'est pas dynamique et elles ne viennent pas s'insérer dans l'équation. Si on les compte comme des contributions à l'erreur de mesure l'erreur de mesure sera disproportionnée devant l'erreur d'état (notamment à cause des aspérités de surface).

w(t) et v(t) sont des vecteurs gaussiens centrés de densité spectrale de puissance W et V respectivement. On a :

$$\begin{array}{ll} E[w(t)w(t+\tau)^T] & = W\delta(\tau) \\ E[v(t)v(t+\tau)^T] & = V\delta(\tau) \\ E[w(t)v(t+\tau)^T] & = 0 \end{array}$$

V peut-être calculé à partir des données d'échantillonnage du capteur. Le certificat d'étalonnage nous indique en effet les précisions suivantes :

$\delta\mathbf{F_x}$	$\delta\mathbf{F_y}$	$\delta\mathbf{F_z}$	$\delta\mathbf{M_x}$	$\delta\mathbf{M_y}$	$\delta\mathbf{M_z}$
1.00%	1.25%	0.75%	1.00%	1.25%	1.50%

On estime également les ordres de grandeur des forces et moments qui seront rencontrés par le capteur (avec les hypothèses émises précédemment)

$\bar{\mathbf{F}}_\mathbf{x}$	$\bar{\mathbf{F}}_\mathbf{y}$	$\bar{\mathbf{F}}_\mathbf{z}$	$\bar{\mathbf{M}}_\mathbf{x}$	$\bar{\mathbf{M}}_\mathbf{y}$	$\bar{\mathbf{M}}_\mathbf{z}$
$0N$	$0N$	$20N$	$0N.m$	$0N.m$	$0N.m$

La précision du robot et des aspérités de la surface sont

$\delta \mathbf{z_{robot}}$	$\delta \mathbf{z_{asp}}$
$0.02mm$	$1mm$

L'ordre de grandeur des aspérités représente ici la différence de hauteur de surface non prévue maximale qui puisse être rencontrée entre deux pas de temps.

V est la matrice de densité spectrale de puissance du vecteur $\underline{v}$. Chaque source d'erreur étant indépendante, cela signifie que la matrice est diagonale, et on a

$$\begin{aligned} V_{ii} &= \mathcal{L}[\phi_{ii}(\tau)] \\ &= \int_{-\infty} +\infty E[v_i(t)v_i(t+\tau)^t]\,\mathrm{d}\tau \end{aligned}$$

Le vecteur v étant un bruit blanc gaussien centré, les choses se simplifient. En effet on a $\forall v_i$

$$\phi(t,\tau)_{ii} = \sigma_i^2 \delta(0)$$

et donc

$$V_{ii} = \sigma_i^2 = Var(v_i)$$

Il ne reste plus qu'à calculer les écarts types associés à chaque erreur. L'intervalle de confiance à 95% étant l'intervalle $[m - 1.96\sigma, m + 1.96\sigma]$ (m étant la moyenne) on a

$$[-\delta F_z * \bar{F}_z \ ; \ \delta F_z * \bar{F}_z] = [-1.96 * \sigma_{F_z} \ ; \ 1.96 * \sigma_{F_z}]$$

Soit

$$\sigma_{F_z} = \frac{\delta F_z * \bar{F}_z}{1.96}$$

On en déduit

$$\begin{aligned} V &= \left(\sigma_{F_z}^2\right) \\ &= \left(\frac{(\delta F_z * \bar{F}_z)^2}{1.96^2}\right) \\ &= \left(5.857 * 10^{-3}\right) \end{aligned}$$

On remarque ici que la contribution de l'imprécision du robot à la matrice V est négligeable (de l'ordre de 10^{-8}) devant l'imprécision du capteur. Le robot est en effet extrêmement précis. La contribution des aspérités de surfaces est également ici négligeable, mais il serait envisageable d'augmenter l'ordre de grandeur de ces aspérités si notre système est efficace.

W est ensuite calculé de la même façon, en considérant que $\delta \dot{z}_{robot} = \delta z_{robot} = 0.02mm$

$$\begin{aligned} [-\delta \dot{z}_{robot} \ ; \ \delta \dot{z}_{robot}] &= [-1.96 * \sigma_{dotz_{robot}} \ ; \ 1.96 * \sigma_{\dot{z}_{robot}}] \\ \sigma_{\dot{z}_{robot}} &= \frac{\delta \dot{z}_{robot}}{1.96} \end{aligned}$$

$$\begin{aligned} W &= \left(\sigma^2_{\dot{z}_{robot}}\right) \\ &= \left(\left(\frac{\delta \dot{z}_{robot}}{1.96}\right)^2\right) \\ &= \left(1.041 * 10^{-8}\right) \end{aligned}$$

Déterminons enfin la matrice M qui permet de "relier" le vecteur w à l'équation d'état. Elle est ici triviale puisqu'on a tout simplement

$$M = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Toutes les données de l'équation ayant été définies, on peut désormais calculer les matrices associées au modèle discret

$$\begin{aligned} A_d &= e^{Adt} \\ &= e^{\begin{pmatrix} 0 & dt \\ 0 & 0 \end{pmatrix}} \\ &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 & dt \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} + \dots \\ &= \begin{pmatrix} 1 & dt \\ 0 & 1 \end{pmatrix} \end{aligned}$$

$$\begin{aligned} B_d &= \int_0^{dt} e^{Av} B \, \mathrm{d}v \\ &= 0 \end{aligned}$$

$$\begin{aligned} C_d &= C \\ &= \begin{pmatrix} k & 0 \end{pmatrix} \end{aligned}$$

$$\begin{aligned} W_d &= dtMWM^T \\ &= \begin{pmatrix} \left(\frac{\delta \dot{z}_{robot}}{1.96}\right)^2 * dt & 0 \\ 0 & 0 \end{pmatrix} \end{aligned}$$

$$\begin{aligned} V_d &= \frac{V}{dt} \\ &= \begin{pmatrix} \frac{(\delta F_z * \bar{F}_z)^2 + \delta z^2_{asp} + \delta z^2_{robot}}{1.96^2 * dt} \end{pmatrix} \end{aligned}$$

Et on peut calculer la matrice d'erreur P à partir de l'équation de Ricatti discrète.

$$P_p = A_d P_p A_d^T - A_d P_p C_d^T \left(C_d P_p C_d^T + V_d\right)^{-1} C_d P_p A_d^T + M_d W_d M_d^T$$

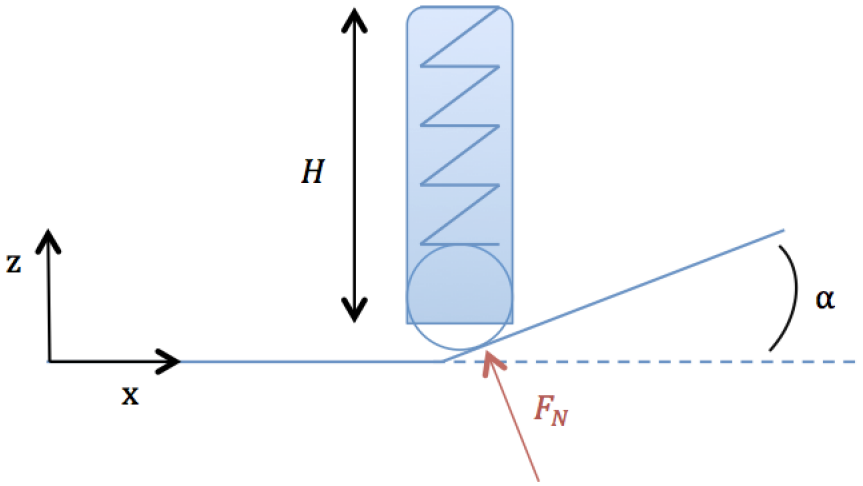
Une résolution numérique nous donne

$$P_p = \begin{pmatrix} 5.316 * 10^{-8} & -5.045 * 10^{-25} \\ -5.045 * 10^{-25} & 0 \end{pmatrix}$$

Ces valeurs ont été obtenues en prenant en compte les incertitudes de position et d'aspérité dans le vecteur de mesure : P est bien trop petit, et donc le gain l'est aussi

2.2 Modèle 2D

Afin d'être plus réalistes, considérons désormais des forces selon les axes x et z, ainsi que le moment autours de l'axe y. Nous négligerons cependant toujours les frottements.



Plaçons nous dans le cas où l'outil n'est plus perpendiculaire à la surface (voir schéma ci-dessus)

Etant donné qu'il n'est pas possible de changer l'orientation du robot XXX -> CARO, nous nous bornerons toujours à corriger la trajectoire du robot.

Les valeurs mesurées par le capteur dépendent désormais de l'angle α formé avec l'horizontale. On a

$$\begin{aligned} F_x &= F_N \sin(\alpha) \\ F_z &= F_N \cos(\alpha) \\ M_y &= F_x * H \qquad \text{en faisant l'hypothèse que la hauteur découverte de la bille est négligeable devant la hauteur du poussoir} \\ &= F_N \sin(\alpha) * H \end{aligned}$$

L'objectif force devant désormais être mesuré selon la nouvelle normale à la surface, il convient de corriger la matrice D de l'équation de mesure. On détermine tout d'abord α grâce aux données du capteur :

$$\alpha = \arctan\left(\frac{F_x}{F_z}\right)$$

On a

$$F_z = k\delta z + F_{obj} cos(\alpha)$$

soit

$$D = \begin{pmatrix} 0 & \cos(\alpha) \end{pmatrix}$$

Afin de tenir compte de la pente imprévue, il faut également ajuster le déplacement et la vitesse selon z.

Pour cela, on prend désormais en compte un vecteur commande u correspondant à la vitesse du robot. Afin de suivre la pente, on souhaite désormais imposer une vitesse verticale supplémentaire égale à $\sin\alpha * \dot{z}_{robot}$.
On pose donc

$$\begin{aligned} u &= \begin{pmatrix} \dot{z}_{robot} \\ F_{obj} \end{pmatrix} \\ B &= \begin{pmatrix} \sin(\alpha) & 0 \\ 0 & 0 \end{pmatrix} \end{aligned}$$

Les matrices V et W sont modifiées car une nouvelle source d'imprécision est apporté par l'introduction du paramètre alpha.

2.3 Modèle 2D avec prise en compte du frottement