# SUIVI DE TRAJECTOIRE D'UN ROBOT MOBILE PAR COMMANDE PRÉDICTIVE

Version du 3/12/2022, 11:12 CET

*Merci de communiquer toute erreur que vous décèleriez à* `patrick.danes@laas.fr`

## I Modélisation à temps continu

Dans le plan muni du repère $\mathcal{F}_0 = (O, \vec{x}_0, \vec{y}_0)$, on associe à un robot mobile non holonome $\mathcal{P}$ le repère $\mathcal{F}_P = (P, \vec{x}_P, \vec{y}_P)$, où $P$ est le point milieu de l'axe de ses roues et $\vec{x}_P$ est tangent à sa trajectoire. L'objectif est de synthétiser une commande en vitesse qui, appliquée à $\mathcal{P}$, permet d'asservir sa trajectoire sur celle d'un robot de référence virtuel $\mathcal{R}$. Ce robot virtuel est semblable à $\mathcal{P}$, et son repère $\mathcal{F}_R = (R, \vec{x}_R, \vec{y}_R)$ associé constitue la référence/consigne que doit atteindre $\mathcal{F}_P$.

À tout instant $t$, les variables d'état et de commande de $\mathcal{P}$ sont respectivement regroupées dans les vecteurs $q(t) = (x(t), y(t), \theta(t))^T$ et $u(t) = (v(t), \omega(t))^T$, avec : $x(t) = \overrightarrow{OP}.\vec{x}_0$ ; $y(t) = \overrightarrow{OP}.\vec{y}_0$ ; $\theta(t) = (\widehat{\vec{x}_0, \vec{x}_P})$ ; $\left[\frac{\mathrm{d}\overrightarrow{OP}}{\mathrm{dt}}\right]_{\mathcal{F}_0} = v(t)\vec{x}_P$ ; $\left[\frac{\mathrm{d}\vec{x}_P}{\mathrm{dt}}\right]_{\mathcal{F}_0} = \omega(t)\vec{y}_P$ ; $\left[\frac{\mathrm{d}\vec{y}_P}{\mathrm{dt}}\right]_{\mathcal{F}_0} = -\omega(t)\vec{x}_P$. L'équation d'état s'écrit

$$\dot{x}(t) = v(t)\cos\theta(t)\,; \qquad \dot{y}(t) = v(t)\sin\theta(t)\,; \qquad \dot{\theta}(t) = \omega(t). \tag{1}$$

La condition initiale de $\mathcal{P}$ n'est pas spécifiée. Le robot de référence $\mathcal{R}$ est décrit de manière similaire par $q_R(t) = (x_R(t), y_R(t), \theta_R(t))^T$, obtenue par intégration de

$$\dot{x}_R(t) = v_R(t)\cos\theta_R(t)\,; \qquad \dot{y}_R(t) = v_R(t)\sin\theta_R(t)\,; \qquad \dot{\theta}_R(t) = \omega_R(t)\,; \tag{2}$$

depuis la condition initiale $q_R(0) = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}^T$ en appliquant le vecteur de commande (connu) $u_R(t) = (v_R(t), \omega_R(t))^T$.

On définit l'*erreur de suivi* $e(t)$ comme la superposition des coordonnées de $\overrightarrow{PR}$ exprimées dans $\mathcal{F}_P$ et de l'erreur angulaire $\theta_R(t) - \theta(t)$, soit

$$e(t) = \begin{pmatrix} e_1(t) \\ e_2(t) \\ e_3(t) \end{pmatrix} := \begin{pmatrix} \cos\theta(t) & \sin\theta(t) & 0 \\ -\sin\theta(t) & \cos\theta(t) & 0 \\ 0 & 0 & 1 \end{pmatrix} \big(q_R(t) - q(t)\big). \tag{3}$$

En boucle ouverte, l'équation d'état du vecteur d'erreur est alors

$$\dot{e}(t) = \begin{pmatrix} \cos e_3(t) & 0 \\ \sin e_3(t) & 0 \\ 0 & 1 \end{pmatrix} u_R(t) + \begin{pmatrix} -1 & e_2(t) \\ 0 & -e_1(t) \\ 0 & -1 \end{pmatrix} u(t). \tag{4}$$

Le problème peut être significativement simplifié en constituant le signal de commande $u(t)$ du robot $\mathcal{P}$ comme la somme d'un terme en feedforward $u_F(t)$ –permettant un suivi parfait en l'absence de perturbation depuis une configuration où $\mathcal{F}_P$ et $\mathcal{F}_R$ sont superposés– et d'un terme en feedback $u_B(t)$ à déterminer. Ainsi, si on pose

$$u(t) = u_F(t) + u_B(t),\ u_F(t) = \begin{pmatrix} v_r(t)\cos e_3(t) \\ \omega_R(t) \end{pmatrix},\ u_B(t) \text{ à déterminer,} \tag{5}$$

alors (4) devient

$$\dot{e}(t) = \begin{pmatrix} \omega_R(t)e_2(t) \\ -\omega_R(t)e_1(t) + v_R(t)\sin e_3(t) \\ 0 \end{pmatrix} + \begin{pmatrix} -1 & e_2(t) \\ 0 & -e_1(t) \\ 0 & -1 \end{pmatrix} u_B(t). \tag{6}$$

Pour $u_B(t) = (0,0)^T$, $e(t) = e^\star = (0,0,0)^T$ est un point d'équilibre de (6). Par linéarisation locale de (6) en $e^\star$, on obtient le système linéaire variant dans le temps

$$\dot{e}(t) = \begin{pmatrix} 0 & \omega_R(t) & 0 \\ -\omega_R(t) & 0 & v_R(t) \\ 0 & 0 & 0 \end{pmatrix} e(t) + \begin{pmatrix} -1 & 0 \\ 0 & 0 \\ 0 & -1 \end{pmatrix} u_B(t). \tag{7}$$

*Dans tout le sujet, on suppose que les vitesses linéaire et angulaire de référence sont égales à des constantes $v_R$ et $\omega_R$, auquel cas : soit $\mathcal{R}$ est immobile, soit $\mathcal{R}$ pivote sur place, soit $\mathcal{R}$ effectue un mouvement linéaire, soit $\mathcal{R}$ effectue un mouvement circulaire. Sous cette hypothèse, le modèle d'état* (7) *devient*

$$[\mathsf{Gc}]\ :\ \dot{e}(t) = A_c e(t) + B_c u_B(t),\ \textit{avec } A_c = \begin{pmatrix} 0 & \omega_R & 0 \\ -\omega_R & 0 & v_R \\ 0 & 0 & 0 \end{pmatrix}\ \textit{et } B_c = \begin{pmatrix} -1 & 0 \\ 0 & 0 \\ 0 & -1 \end{pmatrix}. \tag{8}$$

---

**Nota : l'ensemble du TP peut être réalisé sous MATLAB ou Octave (ou Python dès lors qu'un solveur de programmation quadratique est à disposition) – Cependant, pour la toute fin (Question n°8, Section VIII), il faut bien avouer que SIMULINK peut simplifier la simulation et la visualisation du robot de référence et du robot suiveur commandé par calculateur...**

---

# II Questions préliminaires

1. Montrer que [Gc] est linéaire invariant, puis répondre aux questions suivantes.
   (a) Montrer que [Gc] est commandable dès lors que $v_R$ ou $\omega_R$ est non nulle.
   (b) En déduire qu'une commande par retour d'état statique et de gain constant permet la stabilisation de $\mathcal{P}$ sur la trajectoire de référence.
   (c) Vérifier que ces conclusions sont en accord avec le cours de robotique mobile.

---

# III Modélisation à temps discret

Le robot $\mathcal{P}$ est commandé par calculateur avec une période d'échantillonnage $T_e$, de sorte que pour tout instant $\tau$ compris dans l'intervalle $[t_k; t_{k+1})$, où $t_k = kT_e$ et $t_{k+1} = (k+1)T_e$, l'équation (5) est modifiée en

$$u(\tau) = u_F(\tau) + u_B(\tau),\ u_F(\tau) = u_F[k] = \begin{pmatrix} v_r\cos e_3[k] \\ \omega_R \end{pmatrix},\ u_B(\tau) = u_B[k] \text{ à déterminer,} \tag{9}$$

où pour tout signal $x$ on adopte la convention d'écriture $x[k] = x(kT_e)$. Bien que les propriétés permises par le terme feedforward $u_F(t)$ ne soient plus obtenues lorsqu'on utilise sa contrepartie $u_F[k]$ bloquée à l'ordre 0, on admet qu'il est pertinent de synthétiser le terme feedback $u_B[k]$ sur la base de la mise en série d'un bloqueur d'ordre 0 et de (8), dont l'équation à temps discret s'écrit

$$[\mathsf{B0Gd}] \,:\, e[k+1] = A_d e[k] + B_d u_B[k], \tag{10}$$

$$\text{avec } A_d = \begin{pmatrix} \cos(\omega_R T_e) & \sin(\omega_R T_e) & \frac{v_R(1-\cos(\omega_R T_e))}{\omega_R} \\ -\sin(\omega_R T_e) & \cos(\omega_R T_e) & \frac{v_R \sin(\omega_R T_e)}{\omega_R} \\ 0 & 0 & 1 \end{pmatrix} \text{ et } B_d = \begin{pmatrix} -\frac{\sin(\omega_R T_e)}{\omega_R} & \frac{v_R(\sin(\omega_R T_e)-\omega_R T_e)}{\omega_R^2} \\ \frac{2\sin^2(\frac{\omega_R T_e}{2})}{\omega_R} & \frac{-2v_R \sin^2(\frac{\omega_R T_e}{2})}{\omega_R^2} \\ 0 & -T_e \end{pmatrix}.$$

---

## IV Commande DLQ : approche Riccati

On se propose tout d'abord de synthétiser une commande DLQ (Discrete-time Linear Quadratic control) sur le système $[\mathsf{B0Gd}]$ initialement en $e[0] = e_0$, indépendamment de tout problème de robotique. En d'autres termes, pour $N$ donné, on recherche la séquence de commandes $u_B^\star[0:N-1] := (u_B^\star[0], \ldots, u_B^\star[N-1])$ solution du problème de commande optimale à temps discret

$$u_B^\star[0:N-1] = \arg \min_{u_B[0:N-1]} J(e_0, 0, u_B[0:N-1]), \text{ où } u_B[0:N-1] := (u_B[0], \ldots, u_B[N-1]) \tag{11}$$

$$\text{et } J(e_0, 0, u_B[0:N-1]) := e[N]^T S e[N] + \sum_{k=0}^{N-1} \big(e^T[k] Q e[k] + u_B^T[k] R u_B[k]\big),\ Q, S \geq 0,\ R > 0 \text{ données,}$$

sous les contraintes $e[0] = e_0$ et $\forall k \in \{0, \ldots, N-1\}$, l'équation (10).

---

2. Soient $v_R = 1\,\mathrm{m.s}^{-1}$, $\omega_R = 0.2\,\mathrm{rad.s}^{-1}$, $T_e = 0.25\,\mathrm{s}$, $e_0 = \begin{pmatrix} 4 & 2 & \frac{\pi}{3} \end{pmatrix}^T$. Pour $N = 100$ et chacun des trois triplets $(S = 100\mathbb{I}_{3\times3},\, Q = 100\mathbb{I}_{3\times3},\, R = 1\mathbb{I}_{2\times2})$, $(S = 100\mathbb{I}_{3\times3},\, Q = 1\mathbb{I}_{3\times3},\, R = 100\mathbb{I}_{2\times2})$, $\left(S = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 100 \end{pmatrix}, Q = \begin{pmatrix} 100 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 1 \end{pmatrix}, R = \begin{pmatrix} 1 & 0 \\ 0 & 10 \end{pmatrix}\right)$, répondre aux questions suivantes.
   (a) Résoudre l'équation récurrente de Riccati rétrograde à temps discret puis en déduire l'expression de la solution du problème en tant qu'une commande par retour d'état de gain matriciel variant dans le temps que l'on explicitera.
   (b) Simuler la trajectoire d'état optimale correspondante. Caractériser qualitativement les comportements obtenus pour les divers choix des triplets $(S, Q, R)$.

---

## V Commande DLQ : approche par programmation quadratique sans contrainte

La littérature en optimisation permet de séparer les problèmes convexes (critère et contraintes convexes en le vecteur des variables de décision), dont la résolution numérique est généralement aisée s'ils sont de tailles raisonnables, des problèmes non convexes généralement plus difficiles. La programmation quadratique, qui consiste à résoudre

$$\xi^\star = \arg \min_{\xi \in \mathbb{R}^n} \frac{1}{2} \xi^T H \xi + f^T \xi \text{ sous la contrainte inégalité } A\xi \leq b, \text{ et égalité } A_e \xi = b_e, \tag{12}$$

où les matrices $H > 0$ et $A, A_e$ ainsi que les vecteurs $f$ et $b, b_e$ sont donnés, est convexe. On note accessoirement que les lieux d'iso-valeurs du critère sont des ellipsoïdes emboîtés et que l'ensemble admissible des variables de décision est un polytope (ou l'intersection d'un polytope avec des hyperplans s'il existe des contraintes égalité). La solution numérique d'un tel programme d'optimisation peut être obtenue au moyen de librairies dédiées, cf. par exemple la commande `quadprog` disponible dans l'Optimization Toolbox de MATLAB ou dans le package optim d'Octave.

En l'absence de contrainte, la solution analytique de ce problème est immédiate et s'écrit $\xi^\star = -H^{-1}f$.

---

3. Retrouver la solution du problème DLQ par résolution d'un programme d'optimisation quadratique sans contrainte (donc soluble analytiquement) en procédant selon les étapes suivantes, avec pour vecteur des variables de décision $\xi = \begin{pmatrix} u_B^T[0] & u_B^T[1] & \dots & u_B^T[N-1] \end{pmatrix}^T$.
   (a) Écrire le critère du problème (11) sous la forme

$$J(e_0, 0, u_B[0:N-1]) = e^T[0]Qe[0]\dots + \begin{pmatrix} e^T[1] & e^T[2] & \dots & e^T[N] \end{pmatrix} \overline{Q} \begin{pmatrix} e[1] \\ e[2] \\ \vdots \\ e[N] \end{pmatrix} + \begin{pmatrix} u_B^T[0] & u_B^T[1] & \dots & u_B^T[N-1] \end{pmatrix} \overline{R} \begin{pmatrix} u_B[0] \\ u_B[1] \\ \vdots \\ u_B[N-1] \end{pmatrix}, \quad (13)$$

   où $\overline{Q}$ et $\overline{R}$ sont des matrices bloc-diagonales à déterminer, fonctions de $Q, S$ et $R$, respectivement.
   (b) Exprimer les contraintes du problème (11) sous la forme

$$\begin{pmatrix} e[1] \\ e[2] \\ \vdots \\ e[N] \end{pmatrix} = \overline{S} \begin{pmatrix} u_B[0] \\ u_B[1] \\ \vdots \\ u_B[N-1] \end{pmatrix} + \overline{T} e_0, \quad (14)$$

   où la matrice bloc-triangulaire inférieure $\overline{S}$ doit s'exprimer en fonction de $A_d$ et $B_d$ et la matrice $\overline{T}$ doit être constituée de la superposition de puissances entières de $A_d$.
   (c) Incorporer (14) dans (13) de façon à obtenir la nouvelle expression du critère

$$J(e_0, 0, u_B[0:N-1]) = \mathcal{J}(\xi) := \frac{1}{2}\xi^T H \xi + f^T \xi + \frac{1}{2} e_0^T Y e_0, \quad (15)$$
$$\text{avec } H = 2(\overline{R} + \overline{S}^T \overline{QS}),\ f = 2\overline{S}^T \overline{QT} e_0,\ Y = 2(Q + \overline{T}^T \overline{QT}).$$

   En déduire l'optimum non contraint $\xi^\star = \begin{pmatrix} u_B^{\star T}[0] & \dots & u_B^{\star T}[N-1] \end{pmatrix}^T$. Vérifier qu'il correspond à la solution précédemment obtenue par résolution de l'équation récurrente de Riccati.

---

# VI Extension n° 1 : problème à horizon infini, commande DLQR

4. Répondre aux questions suivantes, qui permettraient la mise en place d'une commande DLQR.
   (a) Observer les « régimes permanents » (aux faibles instants $k$) de la matrice $P[k]$ solution de l'équation récurrente de Riccati ainsi que du gain de retour d'état $K[k]$.
   (b) Retrouver directement leurs valeurs $P$ et $K$, qui correspondent au cas où $N = +\infty$.

---

## VII Extension n° 2 : solution du problème DLQ sous contraintes sur la commande et/ou sur la sortie par programmation quadratique

On considère à nouveau le problème DLQ (11), mais en exigeant la satisfaction de contraintes

$$\forall k \in \{0, \ldots, N-1\},\ u_{\min} \leq u_B[k] \leq u_{\max}, \tag{16}$$

$$\forall k \in \{1, \ldots, N\},\ z_{\min} \leq z[k] \leq z_{\max}, \tag{17}$$

sur les commandes $\{u_B[k]\}$ ainsi que sur des sorties dites « critiques » $\{z[k] = Le[k]\}$, la matrice d'observation $L$ ainsi que les vecteurs de bornes $u_{\min}$, $u_{\max}$, $z_{\min}$, $z_{\max}$ étant préalablement définis, et la relation « $\leq$ » entre deux vecteurs étant interprétée composante par composante. Le problème (11) soumis aux contraintes complémentaires (16)–(17) est effectivement de la forme (12).

---

5. Pour le triplet ($S = 100\mathbb{I}_{3\times3}$, $Q = 100\mathbb{I}_{3\times3}$, $R = 1\mathbb{I}_{2\times2}$) et pour la définition du vecteur de sorties critiques $z = e_3$ ($z$ est donc scalaire et égal à la troisième composante de $e$), rechercher la solution de divers problèmes DLQ sous contraintes spécifiés comme suit.
   (a) Examiner la trajectoire d'état $e^\star[0:N]$ de [B0Gd] soumis à la commande optimale $u_B^\star[0:N-1]$ non contrainte obtenue précédemment, depuis la condition initiale $e[0] = e_0$. Conclure que le jeu de bornes

   $$u_{\min} = \begin{pmatrix} -3 \\ -0.5 \end{pmatrix},\ u_{\max} = \begin{pmatrix} 3 \\ 2 \end{pmatrix},\ z_{\min} = -0.7,\ z_{\max} = \frac{\pi}{3} \tag{18}$$

   impose des contraintes significatives. Sélectionner un ou deux jeux de bornes $u_{\min}, u_{\max}, z_{\min}, z_{\max}$ supplémentaires exprimant des contraintes de plus en plus sévères.
   (b) Pour chaque jeu de bornes, résoudre le problème DLQ sous contraintes en sélectionnant, au choix, l'une des deux options suivantes :
   - Exprimer les contraintes (16)–(17) sous la forme

     $$G\xi \leq V + We_0,\ \text{où } \xi = \begin{pmatrix} u_B^T[0] & u_B^T[1] & \ldots & u_B^T[N-1] \end{pmatrix}^T, \tag{19}$$

     *via* la définition de matrices $G, W$ et d'un vecteur $V$ adéquats puis résoudre un problème semblable à (12) sans contrainte égalité.
   - Ou bien (option probablement plus simple mais ne permettant pas d'exploiter les résultats de la section précédente), définir le vecteur des variables de décision comme $\xi = \begin{pmatrix} e^T[1] & e^T[2] & \ldots & e^T[N] & u_B^T[0] & u_B^T[1] & \ldots & u_B^T[N-1] \end{pmatrix}^T$, puis : (1) réécrire le critère du problème (11) sous la forme $\frac{1}{2}\xi^T H \xi$ ; (2) réécrire les contraintes (10), pour $k \in \{0, \ldots, N-1\}$, sous la forme de contraintes égalité $A_e\xi = b_e$ ; (3) exprimer les contraintes (16)–(17) sous la forme de contraintes inégalité $A\xi \leq b$ ; (4) résoudre un problème semblable à (12).

   (c) Visualiser la séquence de commandes et la trajectoire d'état optimales. Vérifier que les contraintes sont satisfaites.

---

## VIII Suivi de trajectoire par MPC

On dispose désormais de tous les ingrédients permettant de résoudre et simuler le suivi par commande prédictive (Model Predictive Control –MPC) du robot virtuel $\mathcal{R}$ par le robot $\mathcal{P}$. On rappelle que cette commande est obtenue par résolution itérative d'un problème d'optimisation faisant intervenir deux horizons glissants préalablement définis : l'horizon de prédiction $P$ et l'horizon de commande $N \leq P$. Ainsi,

- En l'instant $k$, disposant de $e[k]$, établir la solution $u_B^\star[k:k+N-1]$ du problème de commande optimale défini par
  - le critère

$$J(e[k],k,u[k:k+N-1]) := e^T[k+P]Se[k+P]+\sum_{i=1}^{P-1} e^T[k+i]Qe[k+i]+\sum_{i=0}^{N-1} u_B^T[k+i]Ru_B[k+i], \quad (20)$$

  avec $Q,S \geq 0$, $R > 0$, et, implicitement, $\forall i \in \{k+N,\ldots,k+P-1\}, u_B[i]=0$ ;
  - la prédiction des $\{e[k+i]\}_{i=1,\ldots,P}$ sur la base du modèle d'état [B0Gd] défini en (10) ;
  - l'incorporation de contraintes inégalités linéaires (16)–(17).
- Appliquer $u^\star[k]$ ;
- Poser $k \leftarrow k+1$, puis répéter les deux étapes précédentes.

---

6. Implémenter cette stratégie MPC sur le modèle (10). *On choisira $Q \leq 0$ et $R > 0$, avec $Q = M^TM$ et $(A,M)$ détectable. On définira $S$ comme la solution définie positive de l'équation de Riccati algébrique à temps discret du problème DLQR.* Simuler le comportement obtenu.
7. Visualiser les considérations qualitatives suivantes.
   - Les programmes d'optimisation résolus à chaque instant sont faisables quelles que soient les contraintes sur les entrées $u_B[.]$ s'il n'existe pas de contrainte sur des sorties critiques $z[.]$. Ils peuvent cependant devenir infaisables si ces contraintes sont trop sévères.
   - La commande est d'autant plus agressive que les $u_B[.]$ sont faiblement pondérés dans le critère.
   - Si l'horizon de prédiction $P$ est faible, alors la commande est plus agressive mais peut ne pas satisfaire l'objectif de suivi. Intuitivement, le choix de $P$ élevé a un effet stabilisant (accessoirement, pour $P=+\infty$, le problème MPC sans contrainte est équivalent au problème DLQR !).
   - Si l'horizon de commande $N$ est faible, alors la résolution du problème d'optimisation est moins complexe, mais ceci s'effectue au prix d'une perte de performance. Tester le compromis empirique consistant à fixer $N \approx \frac{P}{10}$.
8. Implémenter et simuler le suivi de la trajectoire du robot $\mathcal{R}$ par implémentation d'une commande MPC sur le robot $\mathcal{P}$. Idéalement, modifier la synthèse de la commande de telle sorte que $u_{\min}, u_{\max}$ constituent des bornes sur $u[k] = u_F[k] + u_B[k]$ plutôt que sur $u_B[k]$ seulement (ce qui est plus pertinent en pratique).

---

# IX Quelques remarques finales

Le problème considéré peut être facilement étendu au cas où le robot virtuel de référence $\mathcal{R}$ est soumis à des commandes $v_R, \omega_R$ variables dans le temps tout en étant bloquées à l'ordre 0 sur chaque période d'échantillonnage. Les matrices $A_d, B_d$ de (10) dépendent alors explicitement du temps. Ceci complique légèrement les écritures des matrices $\overline{S}, \overline{T}, G, W$, mais le raisonnement global reste inchangé.

Les matrices de pondération utilisées dans la partie intégrale du critère peuvent bien sûr ne pas être constantes afin de pondérer différemment l'erreur de suivi et la commande sur différentes zones de l'horizon temporel.

Il est possible (et utile !) d'imposer des contraintes sur les incréments de commande $\Delta u_B[k] := u_B[k] - u_B[k-1]$ (ou "slew-rate"). Pour cela, il suffit d'augmenter le vecteur d'état de

[$\mathsf{B0Gd}$] défini en (10) par $e_u[k] := u_B[k-1]$, ce qui conduit au modèle d'état augmenté (avec $\mathbb{I}$ [resp. $\mathbb{O}$] la matrice identité [resp. la matrice nulle] de dimensions adéquates)

$$[\mathsf{B0Gd}_{\mathsf{aug}}] \;:\; \begin{pmatrix} e[k+1] \\ e_u[k+1] \end{pmatrix} = \begin{pmatrix} A_d & B_d \\ \mathbb{O} & \mathbb{I} \end{pmatrix} \begin{pmatrix} e[k] \\ e_u[k] \end{pmatrix} + \begin{pmatrix} B_d \\ \mathbb{I} \end{pmatrix} \Delta u_B[k], \tag{21}$$

et au problème

$$\Delta u_B^\star[0:N-1] = \arg \min_{\Delta u_B[0:N-1]} J\Big( \big( \begin{smallmatrix} e[k] \\ u_B[k-1] \end{smallmatrix} \big), k, \Delta u_B[k:k+N-1] \Big) \tag{22}$$

$$\text{où } J\Big( \Big( \begin{smallmatrix} e[k] \\ u_B[k-1] \end{smallmatrix} \Big), k, \Delta u_B[k:k+N-1] \Big) := y^T[k+P] S y[k+P] + \sum_{i=1}^{P-1} y^T[k+i] Q y[k+i] + \sum_{i=0}^{N-1} \Delta u_B^T[k+i] R \Delta u_B[k+i],$$

avec $y[k] = e[k] = \begin{pmatrix} \mathbb{I}_{3\times 3} & \mathbb{O}_{3\times 2} \end{pmatrix} \Big( \begin{smallmatrix} e[k] \\ u_B[k-1] \end{smallmatrix} \Big)$, sous les contraintes égalité (21) $\forall k \in \{0, \ldots, N-1\}$ et les contraintes inégalités sur $u_B[.]$, $\Delta u_B[.]$, $z[.]$, etc.