Notes de cours

Robotique : Modélisation, Analyse et Commande

Préparé par

Pr. Alexandre Girard

Université de Sherbrooke



Dernière mise à jour le 17 février 2026

Table des matières

I Les robots manipulateurs **5**

1 Introduction à la science des robots manipulateurs **6**
1.1 L'importance de l'algèbre linéaire pour la robotique . . . 8
1.2 Organisation des notes de cours (partie I) . . . 8
1.3 Lexique . . . 9
1.4 Symboles et notations . . . 10

2 Les robots manipulateurs : définitions **11**
2.1 Mécanismes et composants . . . 11
2.2 Configurations . . . 13
2.3 Configurations classiques . . . 16

3 Cinématique des robots manipulateurs I : La position **17**
3.1 Systèmes de coordonnées . . . 17
3.2 L'approche présentée dans ce chapitre . . . 20
3.3 Vecteurs géométriques de positions . . . 22
3.4 Bases vectorielles et composantes d'un vecteur position . . . 26
3.5 Les matrices de rotations . . . 29
3.6 Coordonnées dans un repère et transformations homogènes . . . 41
3.7 Représentation de la pose d'un corps rigide . . . 45
3.8 Cinématique directe d'un manipulateur . . . 50
3.9 Cinématique inverse d'un manipulateur . . . 64
3.10 Résumé du chapitre . . . 68

4 Cinématique des robots manipulateurs II : Le mouvement **69**
4.1 Référentiel . . . 69
4.2 Vitesse et dérivée d'un vecteur position . . . 72
4.3 Vitesse angulaire et dérivée d'une matrice de rotation . . . 74
4.4 Accélération et dérivée seconde d'un vecteur position . . . 76
4.5 Cinématique différentielle des robots manipulateurs . . . 77
4.6 Cinématique différentielle inverse . . . 86
4.7 Manipulabilité d'un robot manipulateur . . . 92
4.8 Relation entre les variables d'accélération . . . 95
4.9 Résumé du chapitre . . . 96

5 Statique des robots manipulateurs **97**
5.1 Introduction . . . 97
5.2 Les vecteurs forces . . . 98
5.3 Relation entre les forces aux joints et les forces à l'effecteur . . . 98
5.4 Relation statique incluant les forces conservatrices . . . 102
5.5 Relation de la compliance aux joints et à l'effecteur . . . 103
5.6 Manipulabilité en force d'un robot manipulateur . . . 105
5.7 Résumé du chapitre . . . 107

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

6 Dynamique des robots manipulateurs 108
6.1 Introduction . . . 108
6.2 Structure des équations . . . 108
6.3 Variables, nomenclature et dimensions . . . 109
6.4 Équation des manipulateurs . . . 110
6.5 Conservation de l'énergie . . . 117
6.6 Dynamique inverse . . . 117
6.7 Systèmes sous-actionnés vs. complètement actionnés . . . 117
6.8 Manipulabilité dynamique . . . 118
6.9 Dérivation des équations avec la méthode de Lagrange . . . 119
6.10 Équations dans l'espace de la tâche . . . 120
6.11 Manipulateur en contact avec l'environnement . . . 121
6.12 Dynamique des actionneurs et effet des ratios de réduction . . . 123

7 Introduction à la commande des systèmes robotisés 124
7.1 Architectures . . . 126

8 Commande des robots manipulateurs I : quasi-statique 130
8.1 Commande en vitesse de l'effecteur . . . 131
8.2 Commande en position de l'effecteur . . . 132
8.3 Commande en force de l'effecteur . . . 137
8.4 Commande en impédance et en admittance . . . 138
8.5 Commande en admittance . . . 145

9 Commande des robots manipulateurs II : dynamique 150
9.1 Commande dé-localisée joint par joint . . . 151
9.2 Commande avec la méthode du couple calculé . . . 152
9.3 Commande robuste . . . 157
9.4 Méthode du mode glissant . . . 158
9.5 Commande adaptative . . . 161
9.6 Commande hybride en position et force . . . 165
9.7 Méthode d'analyse de la stabilité . . . 166
9.8 Commande optimale . . . 167

10 Génération de trajectoires 168
10.1 Introduction . . . 168
10.2 Chemin et profil temporel . . . 169
10.3 Planification cinématique pour les robots manipulateurs . . . 175

II Les véhicules robotisés 178

11 Modélisation des véhicules 179
11.1 Introduction : L'art de la simplification . . . 179
11.2 Les trois ingrédients de base d'un modèle de véhicule . . . 180
11.3 Équations du mouvement dans le repère du corps . . . 184
11.4 Nomenclature et Notation . . . 188
11.5 Véhicules Terrestres (Automobiles et Robots mobiles) . . . 189
11.6 Drones (Multirotors) . . . 198
11.7 Avions (Ailes fixes) . . . 199
11.8 Véhicules nautiques . . . 201

12 Véhicule : Analyse et stabilité 203
12.1 Linéarisation . . . 203
12.2 Stabilité statique . . . 203

13 Introduction à la commande optimale **204**
13.1 Formalisation mathématique . . . 204
13.2 Théorie de la commande optimale . . . 205
13.3 Grandes familles de méthodes . . . 206

14 Commande de véhicules **208**
14.1 Lois cinématiques . . . 208
14.2 Lois dynamiques . . . 208

15 Planification cinématique **209**
15.1 Espace configuration . . . 209
15.2 Fonction guidage . . . 209
15.3 RRT . . . 209
15.4 RRT* . . . 209

16 Optimisation de trajectoires **210**
16.1 Introduction et contexte . . . 210
16.2 Formalisation mathématique . . . 210
16.3 Méthodes de transcription en programme mathématique . . . 212
16.4 Méthodes de recherche globale . . . 216
16.5 Planéité différentielle (*Differential Flatness*) . . . 218

III Boîte à outils mathématique **219**

17 Calcul vectoriel **220**
17.1 Les Vecteurs . . . 220
17.2 Les bases vectorielles . . . 220
17.3 Opérations vectorielles avec les vecteur-colonnes . . . 222
17.4 Summary of vector operations in terms of components . . . 225
17.5 Column-vector and matrix differentiation . . . 226

18 Algèbre linéaire **230**
18.1 Opérations matricielles de base . . . 230
18.2 Classes de matrices particulières . . . 233
18.3 Interprétations de l'opération $\boldsymbol{y} = A\boldsymbol{x}$ . . . 235
18.4 L'opération matricielle A^{-1} . . . 238
18.5 Propriétés d'une matrice . . . 239
18.6 Les quatre espaces fondamentaux . . . 241
18.7 Solution des moindres carrés pour un système sur-contraint . . . 251
18.8 Solution de norme minimale pour un système sous-contraint . . . 254
18.9 Vecteurs et valeurs propres . . . 256
18.10Décomposition en valeurs singulières (SVD) . . . 262
18.11Matrices de Projection . . . 264

19 Trigonométrie **265**

20 Analyse de la stabilité **266**
20.1 Introduction . . . 266
20.2 Définitions de la stabilité . . . 267
20.3 Méthode Directe de Lyapunov . . . 268
20.4 Passivité . . . 271

21 Optimisation **273**
21.1 Formulation du problème . . . 273
21.2 Définitions importantes . . . 274
21.3 Classes de fonctions . . . 275
21.4 Optimalité . . . 277
21.5 Classes de problèmes . . . 278
21.6 Solutions analytiques . . . 279
21.7 Méthodes de résolution numériques . . . 281

IV Exercices **283**

A Cinématique **284**
A.1 Intro cinématique . . . 284
A.2 Traduire des énoncés en équations vectorielles . . . 284
A.3 Substitution de vecteurs unitaires . . . 284
A.4 Obtenir les composantes avec un produit scalaire . . . 285
A.5 Matrice de rotation calculée avec toutes les définitions . . . 285
A.6 Axe de rotation à partir d'une matrice de rotation . . . 285
A.7 Fonction pour un changement de repère . . . 286
A.8 Calcul de la cinématique d'un robot à deux joints . . . 286
A.9 Calcul de la position de l'effecteur d'un robot . . . 287
A.10 Calcul de la cinématique d'une grue . . . 288
A.11 Calcul de la cinématique d'une jambe . . . 288

B Exercices d'algèbre linéaire **292**
B.1 Moindres carrés pour estimer un ratio de transmission . . . 292
B.2 Solution optimale pour un tracteur avec 4 moteurs indépendants . . . 293
B.3 Identification des paramètres d'un circuit électrique . . . 294
B.4 Vecteur propre d'une matrice de rotation . . . 295

C Exercices sur la cinématique différentielle **296**
C.1 Cinématique différentielle inverse d'un bras à 2 DDL . . . 296
C.2 Cinématique différentielle d'un robot à 3 DDL . . . 297
C.3 Matrice jacobienne . . . 297
C.4 Rigidité d'un robot manipulateur . . . 300
C.5 Commande en impédance d'un robot manipulateur . . . 300
C.6 Génération de trajectoires . . . 301
C.7 Espace nul d'une matrice et robots redondants . . . 301

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Première partie

Les robots manipulateurs

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Chapitre 1

Introduction à la science des robots manipulateurs

La première partie de ces notes discute de la science pour modéliser et analyser le mouvement des robots manipulateurs. Dans un contexte d'ingénierie, modéliser un système robotique est essentiel lors de la phase de conception, par exemple pour valider qu'une géométrie de bras permet d'atteindre plusieurs positions désirées avec son effecteur (i.e. l'outil au bout du bras), et aussi lors de la programmation pour coordonner les différents moteurs et articulations d'un robot. Les outils présentés dans ces notes sont pertinents pour les mécanismes avec plusieurs articulations, on parlera toutefois de *robot* pour alléger le texte.

Capsule vidéo
Introduction à l'analyse des robots manipulateurs
`https://youtu.be/N9cYzmWeyKM`

Comme illustré à la figure 1.1, la modélisation des robots peut être séparée en quatre grandes familles d'analyse.

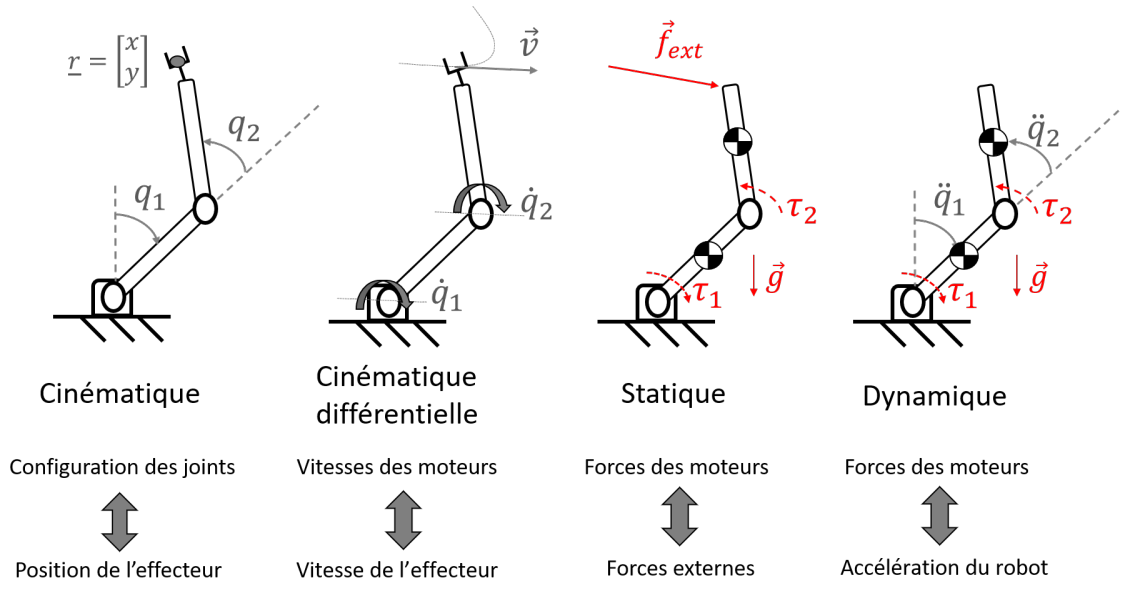


FIGURE 1.1 – Quatre grands domaines de base de l'analyse de robots manipulateurs

Les méthodes de **cinématique directe** visent à calculer la position finale de l'effecteur d'un robot en fonction du positionnement des articulations. Lorsqu'on parle de **cinématique inverse**, l'objectif est de déterminer le positionnement des articulations qui mène à une position désirée de l'effecteur du robot. Mathématiquement, l'analyse consiste à construire et solutionner la fonction non-linéaire qui relie les coordonnées de l'effecteur, groupées dans un vecteur-colonne $\boldsymbol{r}$, aux coordonnées des joints, groupées dans un vecteur-colonne $\boldsymbol{q}$:

Définition 1.1 Cinématique:

$$\text{directe :} \quad \boldsymbol{r} = f\left(\boldsymbol{q}\right) \quad \text{inverse :} \quad \boldsymbol{q} = f^{-1}\left(\boldsymbol{r}\right) \tag{1.1}$$

Un des grands défis est que la fonction inverse $f^{-1}\left(\boldsymbol{r}\right)$ peut avoir plusieurs ou aucune solution, selon les coordonnées de l'effecteur qui sont visées.

Le domaine appelé la **cinématique différentielle** vise à calculer la vitesse de l'effecteur du robot en fonction de la vitesse des moteurs qui actionnent les articulations, ou vice-versa. Mathématiquement, l'analyse consiste à construire la matrice Jacobienne $J(\boldsymbol{q})$, qui permet de relier le vecteur-colonne $\dot{\boldsymbol{r}}$ des vitesses de l'effecteur (la dérivée temporelle des coordonnées $\boldsymbol{r}$) et le le vecteur-colonne $\dot{\boldsymbol{q}}$ des vitesses des joints (la dérivée temporelle des coordonnées $\boldsymbol{q}$) :

Définition 1.2 Cinématique différentielle:

$$\text{directe :} \quad \dot{\boldsymbol{r}} = J\left(\boldsymbol{q}\right)\dot{\boldsymbol{q}} \quad \text{inverse :} \quad \dot{\boldsymbol{q}} = J^{\#}\left(\boldsymbol{q}\right)\dot{\boldsymbol{r}} \tag{1.2}$$

Un des défis est que pour certaines configurations du manipulateur la matrice $J(\boldsymbol{q})$ est singulière, i.e. son inverse J^{-1} n'existe pas. On dit alors que le robot est sur une singularité cinématique où il est impossible de faire bouger l'effecteur dans certaines directions. Un autre défis est que lorsque le nombre de joints est différent du nombre de coordonnées de l'effecteur, la matrice J est rectangulaire et on doit alors utiliser la notion de pseudo-inverse notée $J^{\#}$.

Pour les robots manipulateurs, les analyses de **statique** visent généralement à déterminer les forces/-couples nécessaires aux moteurs/actionneurs d'un robot pour le maintenir en place en fonction des forces externes, ou vice-versa. Mathématique, l'analyse peut aussi utiliser la matrice Jacobienne pour faire la relation entre le vecteur-colonne de couples des moteurs $\boldsymbol{\tau}$ et le vecteur-colonne des forces externes $\boldsymbol{f}_{ext}$:

Définition 1.3 Statique:

$$\boldsymbol{\tau} = J^{T}\left(\boldsymbol{q}\right)\boldsymbol{f}_{E} \tag{1.3}$$

Finalement, la **dynamique** est le domaine qui vise à déterminer les équations différentielles qui représentent la relation entre l'accélération des joints d'un robot et les forces appliquées. Mathématiquement, l'analyse consiste à construire et analyser une équation différentielle non-linéaire (reliant les coordonnées des joints $\boldsymbol{q}$, la vitesse des joints $\dot{\boldsymbol{q}}$, l'accélération des joints $\ddot{\boldsymbol{q}}$, les couples appliqués $\boldsymbol{\tau}$ et les forces externes $\boldsymbol{f}_{ext}$) qui prend la forme :

Définition 1.4 Dynamique:

$$H(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + D\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau} - J^{T}(\boldsymbol{q})\,\boldsymbol{f}_{E} \tag{1.4}$$

où les termes $H(\boldsymbol{q})\ddot{\boldsymbol{q}}$ et $C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$ représentent des effets inertiels, le terme $D\dot{\boldsymbol{q}}$ représente des forces dissipatives (ex. : friction) et le terme $\boldsymbol{g}(\boldsymbol{q})$ représente des forces gravitationnelles. Cette équation c'est $\vec{F} = m\vec{a}$ appliquée aux robots manipulateurs.

Hypothèses de travail :

L'approche de modélisation pour la cinématique, la statique et la dynamique utilisée dans cette partie (chapitres 3, 4, 5 et 6) considère les robots comme des systèmes de corps rigides reliés par des articulations qui permettent un nombre restreint de mouvements relatifs. Cette approche de modélisation fait l'hypothèse que **les déformations des sections rigides du robot sont négligeables**, ce qui est par exemple raisonnable pour les robots manipulateurs industriels dans la plupart des applications.

1.1 L'importance de l'algèbre linéaire pour la robotique

Contrairement à plusieurs machines et systèmes, les robots sont caractérisés par la présence de plusieurs dimensions. Les entrées sont multiples, par exemple les forces/vitesses des multiples moteurs, et les sorties aussi, par exemple la position (x, y, z) de l'outil. Les quantités intéressantes en robotique (entrées et sorties des analyses) peuvent donc être représentées par des vecteur-colonnes, et les relations entre ceux-ci peuvent être représentées par des opérations matricielles qui simplifient grandement les analyses. L'algèbre linéaire est donc un des outils mathématiques les plus importants pour les roboticiens.

1.2 Organisation des notes de cours (partie I)

Le chapitre 2 introduit la nomenclature et les concepts de base impliqués dans l'analyse et la modélisation des robots manipulateurs.

Le chapitre 3 présente des méthodes mathématiques pour modéliser la cinématique des robots manipulateurs. Premièrement, les systèmes de coordonnées importants pour les robots manipulateurs sont introduits. Ensuite, l'utilisation des vecteurs géométriques, des bases vectorielles et des repères pour représenter la position est présentée. Finalement, ces notions sont appliquées à la résolution des problèmes de cinématique pour les robots manipulateurs.

Le chapitre 4 présente des méthodes pour analyser la cinématique différentielle des robots manipulateurs.

Le chapitre 5 présente des méthodes pour analyser la statique des robots manipulateurs.

Le chapitre 6 présente des méthodes pour analyser la dynamique des robots manipulateurs.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

1.3 Lexique

Terme technique	En anglais	Définition
Robot Manipulateur	Manipulator Robot	Robot avec une base fixe qui a comme tâche de positionner dans l'espace un objet ou un outil.
Actionneur	Actuator	Dispositif qui transforme l'énergie en travail mécanique. Typiquement des moteurs électriques et des vérins pneumatiques ou hydrauliques pour les robots.
Effecteur	End-effector	L'endroit où est situé l'objet ou l'outil manipulé par un robot.
Corps rigide	Rigid body	Un modèle idéal d'un objet qui ne se déforme pas même lorsque soumis à des forces externes.
Joint	Joint	Jonction mécanique permettant un mouvement relatif entre deux pièces.
Système de coordonnées	Coordinate system	Ensemble de scalaires (angles ou positions) qui permettent de paramétrer la position/configuration d'un système.
Degrés-de-Liberté (DDL)	Degree-of-Freedom (DoF)	Le nombre de variable indépendante qui permettent de paramétrer dans l'espace la position/configuration d'un système.
Base vectorielle	Vector basis	Trois vecteurs unitaires qui définissent une orientation.
Base vectorielle orthonormée	Orthogonal Vector basis	Trois vecteurs unitaires orthogonaux qui définissent une orientation.
Origine	Origin	Un point de référence pour la mesure des positions.
Repère	Frame	Combinaison d'une origine et d'une base vectorielle orthonormée.
Système de coordonnées cartésiennes	Cartesian coordinate system	Ensemble de trois scalaires qui déterminent la position d'un point par rapport à une origine et trois axes orthogonaux.
Référentiel	Reference frame	Point de vue choisi pour analyser/observer un mouvement.

1.4 Symboles et notations

Symbole	Définition
Vecteurs géométriques	
$\vec{v}$	Vecteur
$\hat{a}$	Vecteur unitaire
$\vec{r}_{A/B}$	Vecteur position du point A par rapport au point B
Vecteur-colonnes et composantes scalaires	
$\boldsymbol{c} = \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} c_1 & c_2 & c_3 \end{bmatrix}^T$	Vecteur-colonne
$\boldsymbol{v}^a = \begin{bmatrix} v_1^a \\ v_2^a \\ v_3^a \end{bmatrix}$	Vecteur-colonne des composantes du vecteur $\vec{v}$ dans la base vectorielle a
v_i^a	Composante du vecteur $\vec{v}$ selon le vecteur unitaire i de la base vectorielle a
$\boldsymbol{q} = \begin{bmatrix} q_1 \\ \vdots \\ q_n \end{bmatrix}$	Vecteur-colonne qui regroupe n scalaires représentant la configuration d'un robot à n DDL dans l'espace des joints.
Bases et repères	
$\{\hat{a}_1, \hat{a}_2, \hat{a}_3\}$	Base vectorielle a définie par un ensemble de trois vecteur unitaires orthogonaux
$\{A_O, \hat{a}_1, \hat{a}_2, \hat{a}_3\}$	Repère A défini par un point d'origine A_O et la base vectorielle a
Matrices	
$M = \begin{bmatrix} M_{1,1} & \dots & M_{1,n} \\ \vdots & \ddots & \vdots \\ M_{m,1} & \dots & M_{m,n} \end{bmatrix}$	Matrice de m rangées et n colonnes ($m \times n$)
${}^aR^b = \begin{bmatrix} \boldsymbol{b}_1^a & \boldsymbol{b}_2^a & \boldsymbol{b}_3^a \end{bmatrix}$	Matrice de rotation (3×3) qui représente l'orientation de la base vectorielle b par rapport à la base vectorielle a.
${}^AT^B = \begin{bmatrix} {}^aR^b & \boldsymbol{r}^a_{B_o/A_o} \\ 0\,0\,0 & 1 \end{bmatrix}$	Matrice de transformation homogène (4×4) du repère B vers le repère A

Chapitre 2

Les robots manipulateurs : définitions

Ce chapitre introduit les fondations technologiques et conceptuelles des bras robotiques. Nous y définissons la nomenclature des composants physiques (liens, joints, effecteurs) ainsi que les notions mathématiques de base permettant de quantifier la capacité de mouvement d'un robot, notamment les degrés de liberté et l'espace de travail.

Les robots manipulateurs sont des bras artificiels constitués d'une chaîne de liens rigides connectés par des articulations. La grande majorité des robots industriels sont des manipulateurs conçus pour positionner des outils (soudure, peinture, assemblage). La figure 2.1 présente les composants principaux, tandis que la figure 2.2 illustre un prototype concret.

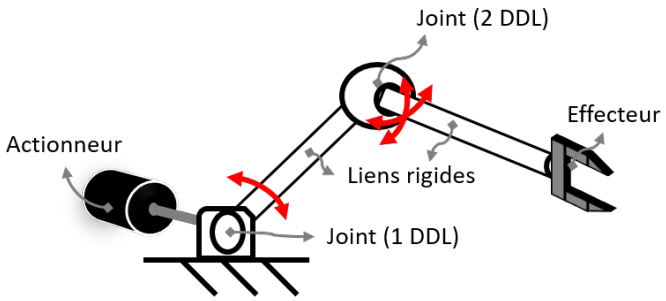


FIGURE 2.1 – Nomenclature pour l'analyse d'un robot manipulateur

2.1 Mécanismes et composants

2.1.1 Liens rigides

Un lien rigide est un ensemble de pièces fixes les unes par rapport aux autres formant un corps solide. D'un point de vue cinématique, si aucun mouvement relatif n'est possible entre deux pièces, elles sont considérées comme un seul lien.

2.1.2 Joints (articulations)

Les joints sont les points de pivot ou de glissement du robot, permettant un mouvement relatif entre deux liens rigides. Le nombre de variables nécessaires pour décrire la position relative entre les deux liens reliés à une articulation est appelé le nombre de **degrés de liberté** (DDL) de l'articulation. Contrairement au corps humain, les robots utilisent généralement des joints simples (un seul DDL par articulation). On va noter q_i la variable qui décrit la position (translation ou rotation) du joint i d'un robot.

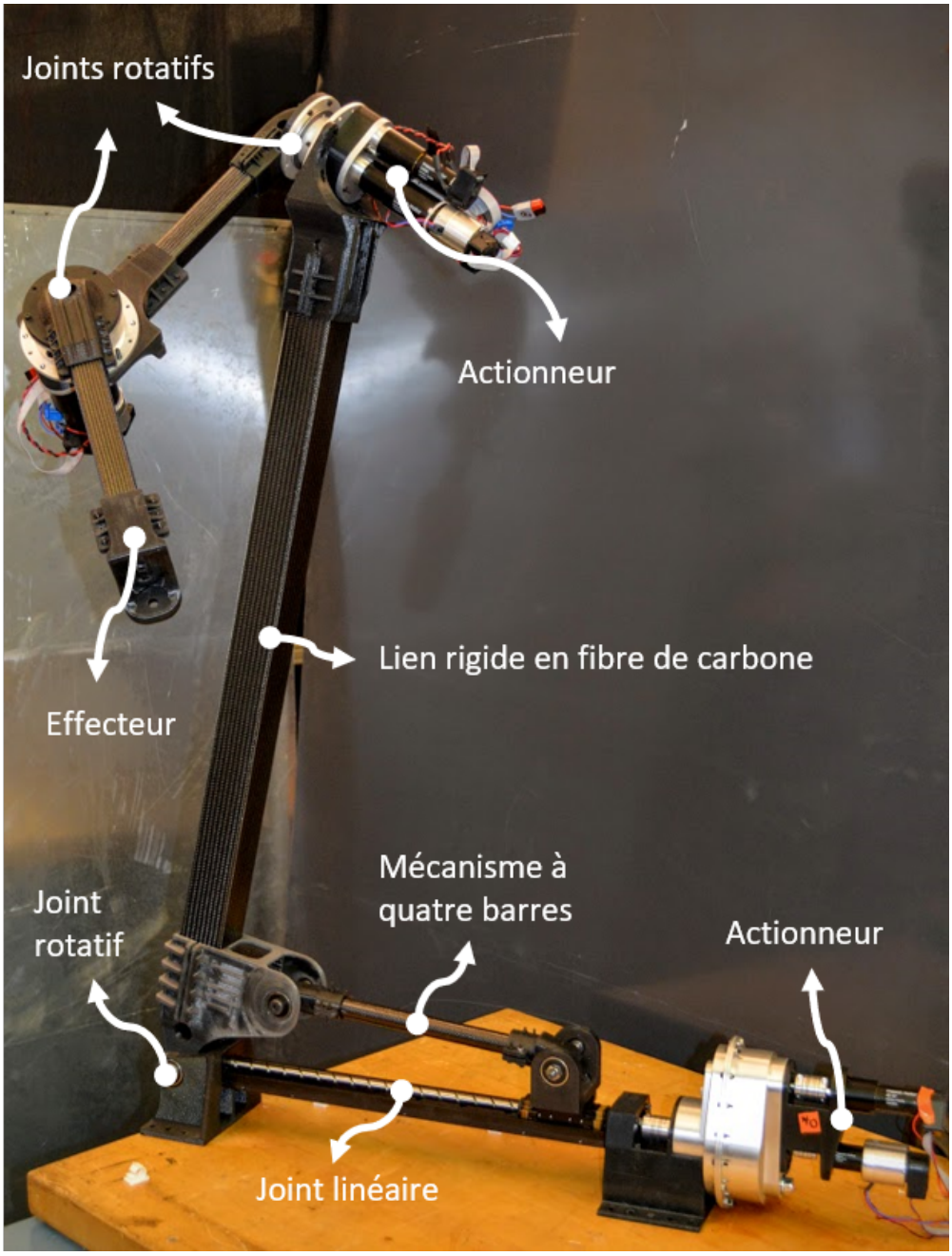


FIGURE 2.2 – Prototype de robot manipulateur avec trois degrés de liberté

Définition 2.1 Joint prismatique (linéaire):

Un joint prismatique permet un mouvement de translation relative entre deux liens selon un axe donné. La variable de configuration q_i représente la distance de coulissement :

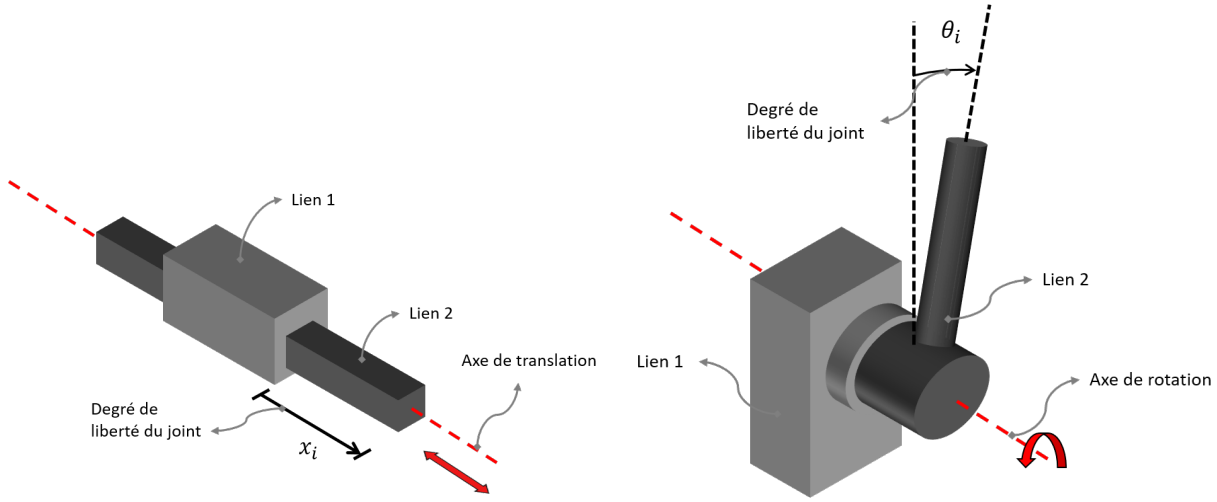
$$q_i = x_i \quad [m] \tag{2.1}$$

Définition 2.2 Joint rotatif (révolu):

Un joint rotatif (ou révolu) permet un mouvement de rotation relative entre deux liens autour d'un axe fixe. La variable de configuration q_i représente l'angle de rotation :

$$q_i = \theta_i \quad [rad] \tag{2.2}$$

Un joint est dit **actif** s'il est motorisé, et **passif** s'il est libre. On va noter τ_i une force (couple ou force) appliquée directement à une articulation i par un actionneur.



(a) Joint prismatique (b) Joint rotatif

FIGURE 2.3 – Joints à un degré de liberté

2.1.3 Effecteur

L'effecteur (ou organe terminal) désigne l'outil physique situé à l'extrémité de la chaîne cinématique du bras (pince, caméra, tête de soudure, etc.). C'est l'interface entre le robot et son environnement. Du point de vue de la modélisation, on s'intéresse principalement à la « pose » de cet effecteur, c'est-à-dire sa position et son orientation dans l'espace.

2.1.4 Actionneurs et capteurs

Les **actionneurs** transforment l'énergie en travail mécanique pour mouvoir les joints. Les **capteurs** collectent l'information sur l'état du robot. On utilise principalement des **encodeurs** pour mesurer les positions articulaires q_i.

2.2 Configurations

Il est souvent utile de regrouper l'information sur la position de tous les joints :

Définition 2.3 Vecteur de configuration:

L'ensemble des variables articulaires est regroupé dans le vecteur de configuration $\boldsymbol{q} \in \mathbb{R}^n$:

$$\boldsymbol{q} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix} \tag{2.3}$$

Ce vecteur définit de manière unique la « pose » de tous les liens du robot. L'espace formé par l'ensemble des vecteurs $\boldsymbol{q}$ possibles est appelé **l'espace articulaire**.

2.2.1 Degrés de liberté (DDL)

Définition 2.4 Degré de liberté:

Le nombre de degrés de liberté (DDL), noté n, correspond au nombre minimal de variables indépendantes (la dimension de $\boldsymbol{q}$) nécessaires pour décrire complètement la configuration d'un mécanisme.

Définition 2.5 Robot déterminé:

Un robot est dit **déterminé** lorsqu'il possède exactement 6 DDL. C'est le minimum requis pour contrôler indépendamment la position et l'orientation dans l'espace (soit $\dim(\boldsymbol{q}) = \dim(\boldsymbol{r})$).

Définition 2.6 Robot redondant:

Un robot est dit **redondant** lorsqu'il possède plus de 6 DDL ($n > 6$). Il existe alors une infinité de vecteurs $\boldsymbol{q}$ permettant d'atteindre une même pose de l'effecteur.

2.2.2 Espace opérationnel

L'espace opérationnel (aussi appelé espace de la tâche) est l'espace mathématique dans lequel est définie la tâche du robot. Bien qu'il corresponde le plus souvent à la pose de l'effecteur, il peut être défini de manière plus générique selon les besoins de l'application.

Définition 2.7 Vecteur de l'espace opérationnel:

La configuration du robot dans l'espace de la tâche est décrite par le vecteur colonne $\boldsymbol{r}$:

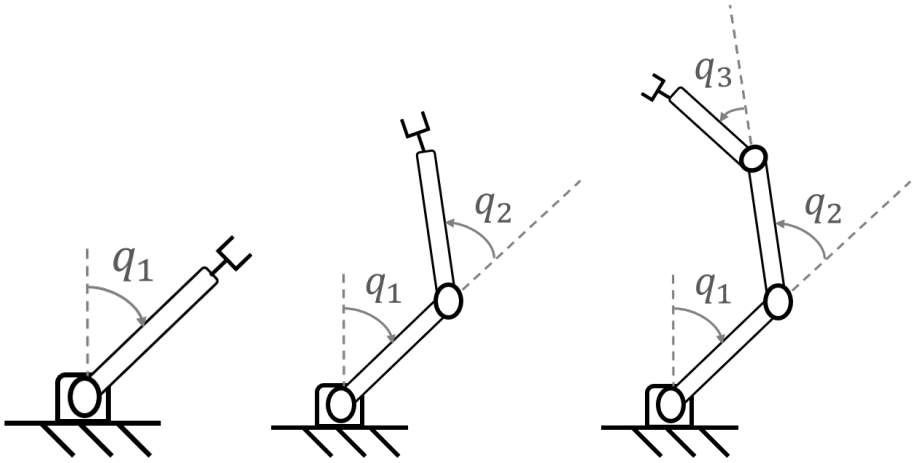
$$\boldsymbol{r} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_m \end{bmatrix} \in \mathbb{R}^m \tag{2.4}$$

où m est le nombre de variables nécessaires pour décrire la tâche.

Par exemple, pour un problème tridimensionnel complet avec la pose de l'effecteur à positionner précisément $m = 6$ (position et orientation). Alternativement, pour une tâche de dessin sur un mur, on pourrait définir $\boldsymbol{r}$ uniquement par les coordonnées (x, y) par rapport au coin de la feuille, donc on aurait $m = 2$, ignorant ainsi les orientations si le crayon reste toujours perpendiculaire à la surface.

Note :

Pour représenter l'orientation de l'effecteur en 3D, utilisez seulement 3 variables a des inconvénients (singularité). Dans cette situation on va typiquement utiliser des matrices de rotation ou des quaternions, et donc l'espace de la tâche ne sera pas juste un vecteur de réels. On peut toutefois faire abstraction de cette distinction pour la plupart des concepts.



(a) 1 DDL (b) 2 DDL (c) 3 DDL

FIGURE 2.4 – Manipulateurs planaires avec des joints rotatifs

2.2.3 Espace de travail

L'espace de travail décrit la portée physique du robot en termes de positions et d'orientations qu'il peut physiquement atteindre.

Définition 2.8 Espace de travail:

L'espace de travail est l'ensemble des poses atteignables par l'effecteur du robot.

Typiquement, l'espace de travail est défini par la pose complète de l'effecteur ou un sous-ensemble de celle-ci (par exemple, uniquement les positions (x, y, z) sans l'orientation). Cependant, il peut exister des définitions spécifiques selon le contexte :

- **Espace de travail atteignable :** L'ensemble des positions que l'effecteur peut atteindre avec au moins une orientation.
- **Espace de travail dextre :** L'ensemble des positions que l'effecteur peut atteindre avec n'importe quelle orientation.

Ce volume dépend de la longueur des liens et des limites articulaires (butées).

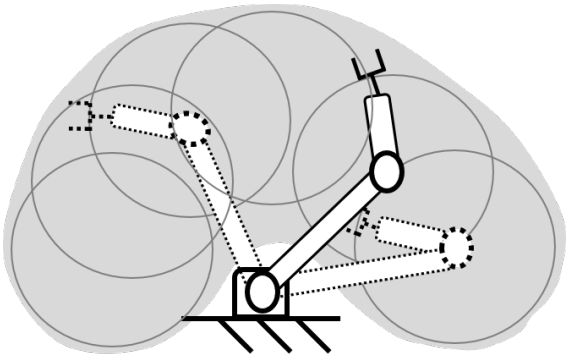


FIGURE 2.5 – Exemple de l'espace de travail d'un robot manipulateur planaire à deux joints

2.3 Configurations classiques

La disposition des trois premières articulations du robot détermine la forme de son espace de travail. On utilise souvent une notation compacte (ex. : RRR, PPP) pour désigner la séquence des joints.

Définition 2.9 Robot Cartésien (PPP):

Possède trois joints prismatiques orthogonaux. Son espace de travail est un prisme rectangulaire.

Définition 2.10 Robot SCARA (RRP):

Utilise deux joints rotatifs parallèles pour le mouvement planaire et un joint prismatique pour le vertical. Très rigide verticalement mais souple horizontalement.

Définition 2.11 Robot Anthropomorphe (RRR):

Configuration la plus courante des robots industriels, qui consiste en trois joints rotatifs (base avec axe verticale, épaule avec axe horizontal et coude avec axe horizontal). Offre un grand espace de travail pour un faible encombrement. Typiquement associé à un poignet (trois joint rotatif avec les axes qui se croisent en un point) avec trois DDLs supplémentaires pour pouvoir commander l'orientation de l'effecteur.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Chapitre 3

Cinématique des robots manipulateurs I : La position

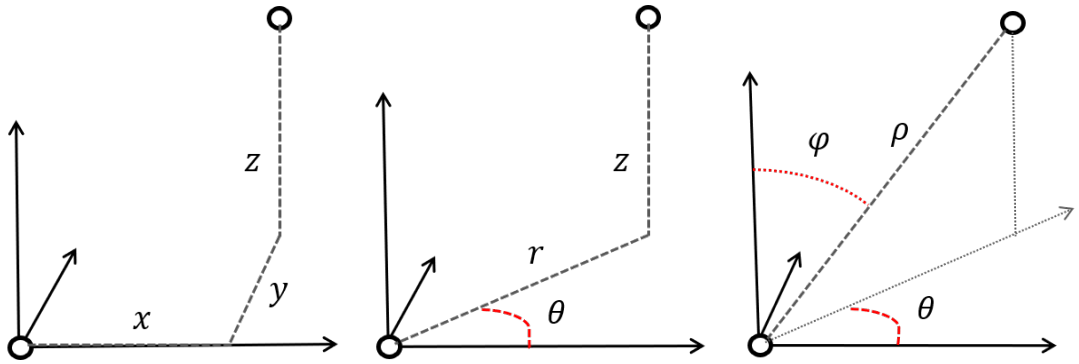
Ce chapitre présente des méthodes pour modéliser et analyser la position de systèmes à plusieurs degrés de liberté, donc bien adaptés aux robots manipulateurs. Dans un contexte de robotique, les problèmes de cinématique consistent premièrement à bien choisir et définir des systèmes de coordonnées pertinents. Par exemple, des coordonnées généralisées pour décrire la configuration des joints, des coordonnées cartésiennes pour décrire la tâche d'un robot en terme de trajectoire de l'effecteur, des coordonnées sphériques pour décrire les mesures d'un système de vision, etc. Ensuite, le défi est de **calculer les fonctions de transformations pour passer d'un système de coordonnées à un autre**. Par exemple, pour les robots manipulateurs, le principal défi est le calcul de la fonction de la cinématique directe et son inverse, i.e. une fonction avec comme entrées les coordonnées généralisées qui représentent la configuration des joints et comme sortie la pose de l'effecteur. Le calcul de ces transformations dans un contexte de robotique est le sujet principal de ce chapitre. La section 3.1 introduit aux différents systèmes de coordonnées. Les sections 3.3 et 3.4 introduisent les notions de vecteurs de position géométriques, bases vectorielles et vecteur-colonnes de composantes. Les sections 3.5 et 3.6 présentent les changements de base et les changements des repères. Ensuite, la section 3.7 présente les méthodes pour représenter mathématiquement la pose d'un corps rigide. Finalement, les sections 3.8 et 3.9 traitent de la cinématique des robots manipulateurs, i.e. le calcul des transformations pour passer de l'espace des joints à la pose de l'effecteur et vice-versa.

3.1 Systèmes de coordonnées

Un système de coordonnées c'est un ensemble de scalaires, généralement des distances et des angles, qui décrivent la position d'un système.

3.1.1 Position d'une particule

Une particule (un point) dans l'espace tri-dimensionnel possède trois DDL et sa position peut donc être complètement déterminée par trois scalaires qui forment un système de coordonnées. Trois types de systèmes de coordonnées sont généralement utilisés pour décrire la position d'une particule par rapport à des axes : **cartésien** (x, y, z), **cylindrique** (r, θ, z) ou **sphérique** (ρ, θ, ϕ), voir la figure 3.1.



(a) Cartésien (b) Cylindrique (c) Sphérique

FIGURE 3.1 – Trois systèmes de coordonnées standards pour représenter la position d'un point en 3D

Selon la situation, l'utilisation d'un système de coordonnées particulier peut simplifier l'analyse d'un problème. Par exemple, pour localiser un point géographique sur la Terre, des coordonnées sphériques (ex. : longitude et latitude) sont plus pratiques que des coordonnées cartésiennes (x,y,z) par rapport au centre de la Terre. Parfois l'utilisation d'un type de système de coordonnées est nécessaire car un capteur mesure une coordonnée particulière. Par exemple, le *LIDAR* sur une voiture autonome mesure des distances dans plusieurs directions grâce à un laser pivotant, les données brute sont des coordonnées sphériques. Il est donc parfois inévitable de faire la conversion d'un système de coordonnées à un autre. Le tableau 3.1 donne les conversions entre des coordonnées cartésiennes, cylindriques et sphériques, qui sont basées sur la même origine et le même système d'axes.

3.1.2 Pose d'un corps rigide

Pour décrire la position d'un corps rigide, trois scalaires qui représentent l'orientation sont nécessaires en plus des trois coordonnées qui spécifient la translation, pour un total de six DDL. La combinaison de la translation et de l'orientation est souvent appelée **la pose** d'un objet. La translation est typiquement spécifiée par la position d'un point sur le corps rigide (ex. : son centre) avec les différentes représentations discutées à la section 3.1.1 (cartésienne, cylindrique ou sphérique) et l'orientation est typiquement spécifiée par trois angles. Comme illustré à la figure 3.2, les six coordonnées suivantes pourraient être utilisées pour spécifier la pose d'un objet :

$$\text{La translation (3 DDL), ex. :} \quad [x, y, z] \tag{3.1}$$

$$\text{L'orientation (3 DDL), ex. :} \quad [\theta_{tangage}, \theta_{roulis}, \theta_{lacet}] \tag{3.2}$$

Il est à noter que **la représentation de l'orientation d'un corps rigide est un problème complexe** et qu'il existe plusieurs méthodes alternatives pour représenter l'orientation (matrices de rotations, angles de Euler, représentation axe-angle, quaternions, etc.) qui seront discutées à la section 3.7. Selon les domaines (robotique, aérospatiale, engins graphiques des jeux vidéos, vision par ordinateur, astronomie, etc.), différents standards et conventions existent en terme de coordonnées utilisées pour décrire l'orientation. Le point à retenir pour le moment est que trois coordonnées indépendantes sont nécessaires pour représenter une orientation arbitraire, donc six totales sont nécessaires pour représenter une pose arbitraire. Par exemple, un minimum de six coordonnées doivent être utilisées pour : spécifier la position de l'effecteur d'un robot, décrire la position d'un avion dans le ciel, décrire la position de la terre par rapport au soleil, etc.

3.1.3 Configuration d'un robot manipulateur

Pour représenter la configuration d'un robot manipulateur il faut suffisamment de variables pour décrire la position relative de chacun des liens rigides qui le compose. Le choix des coordonnées doit être adapté à la géométrie du robot étudié. Un ensemble de variables indépendantes, généralement des angles et des distances, qui permet de déterminer la configuration d'un robot est appelé **coordonnées généralisées**. L'adjectif

	De cartésien	De cylindrique	De sphérique
Vers cartésien	$x = x$ $y = y$ $z = z$	$x = r\cos\theta$ $y = r\sin\theta$ $z = z$	$x = \rho\sin\phi\cos\theta$ $y = \rho\sin\phi\sin\theta$ $z = \rho\cos\phi$
Vers cylindrique	$r = \sqrt{x^2 + y^2}$ $\theta = \arctan\frac{y}{x}$ $z = z$	$r = r$ $\theta = \theta$ $z = z$	$r = \rho\sin\phi$ $\theta = \theta$ $z = \rho\cos\phi$
Vers sphérique	$\rho = \sqrt{x^2 + y^2 + z^2}$ $\theta = \arctan\frac{y}{x}$ $\phi = \arctan\frac{\sqrt{x^2 + y^2}}{z}$	$\rho = \sqrt{r^2 + z^2}$ $\theta = \theta$ $\phi = \arctan\frac{r}{z}$	$\rho = \rho$ $\theta = \theta$ $\phi = \phi$

TABLE 3.1 – Conversions entre les systèmes de coordonnées cartésiennes, polaires et sphériques qui font référence aux même axes et à la même origine.

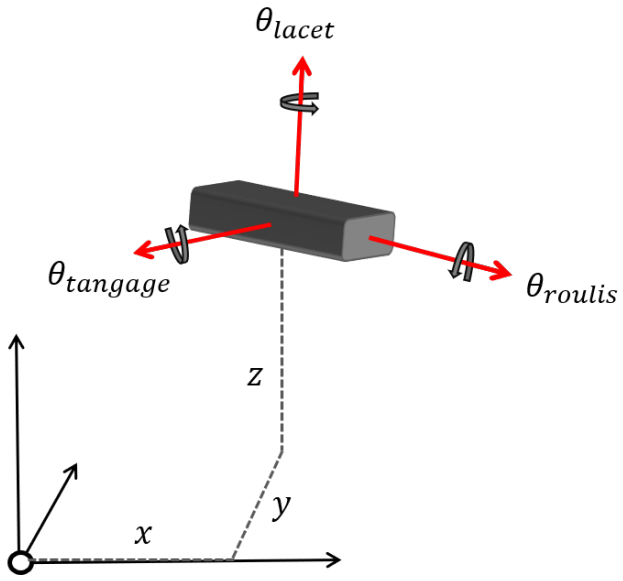
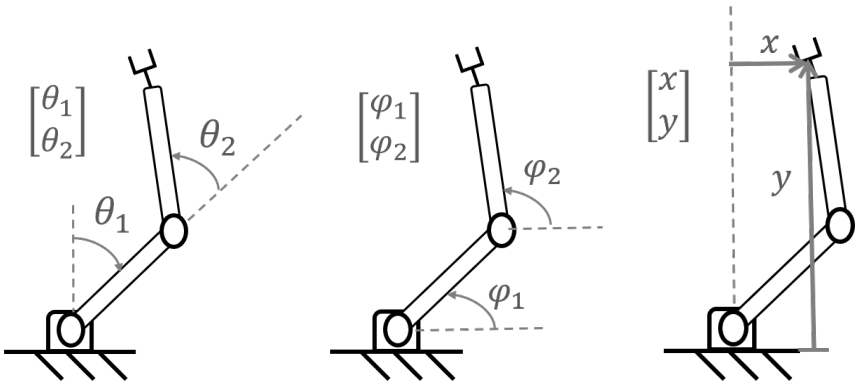


FIGURE 3.2 – Six coordonnées sont nécessaire pour représenter la pose d'un corps rigide, trois pour décrire la translation et trois pour décrire son orientation.

généralisées est historique et a ici le sens de *pas nécessairement cartésiennes*. Le nombre de coordonnées généralisées nécessaire correspond au nombre de **degrés-de-liberté** (DDL) du robot. La figure 3.3 montre trois exemples de deux coordonnées indépendantes qui peuvent être utilisées comme coordonnées généralisées pour décrire la configuration d'un robot à deux joints rotatifs.



(a) Deux angles relatifs (b) Deux angles absolus (c) Deux distances

FIGURE 3.3 – Trois possibilités de coordonnées indépendantes pour spécifier la position d'un robot à 2 DDL

Deux types de systèmes de coordonnées sont généralement utilisés en robotique :

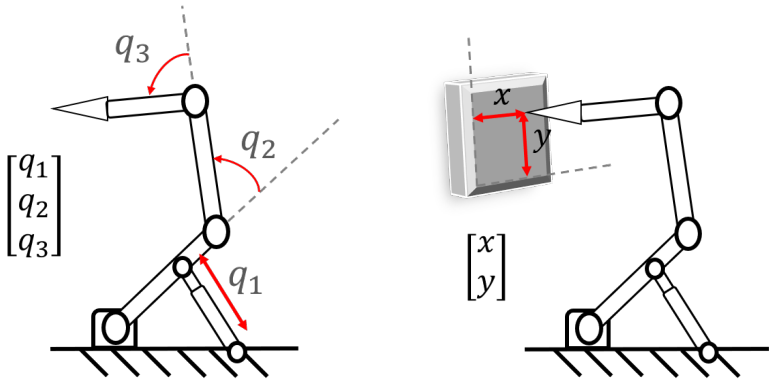
Espace des joints/actionneurs Pour les robots manipulateurs, les algorithmes de contrôle utilisent généralement des coordonnées généralisées qui correspondent à des mesures du déplacement relatif de chaque joint comme illustré à la figure 3.4a. Ces coordonnées sont généralement notées $\boldsymbol{q} = \begin{bmatrix} q_1 & q_2 & \dots & q_n \end{bmatrix}^T$ (où n est le nombre de DDL) et on réfère aussi à ce système comme **l'espace des joints**. Chaque coordonnée q_i représente alors une mesure locale du DDL qui relie deux liens rigides séquentiels. Pour les robots industriels par exemple, ce système de coordonnées est très utile car des encodeurs mesurent directement ces variables et les moteurs les contrôlent de façon indépendante.

Espace de la tâche La tâche d'un robot peut rarement être spécifiée facilement dans l'espace des joints. Un système de coordonnées naturel pour la tâche doit donc généralement être utilisé et on réfère aussi à ce système comme **l'espace de la tâche**. La tâche d'un robot manipulateur est généralement spécifiée plus naturellement en termes de coordonnées cartésiennes de l'effecteur. Il est à noter que le nombre de coordonnées nécessaires pour l'espace de la tâche peut être différent du nombre de DDL total du robot. Comme illustré à la figure 3.4b, pour un robot qui aurait comme tâche d'écrire sur une plaque, l'objectif serait naturellement exprimé par une trajectoire définie avec des coordonnées (x, y) sur la plaque.

3.2 L'approche présentée dans ce chapitre

L'approche de modélisation mise de l'avant dans ce chapitre, et illustrée à la Figure 3.5, ce résume à la séquence :

1. Décrire le problème avec des vecteurs géométriques symboliques ;
2. Choisir des bases appropriées ;
3. Traduire la relation vectorielle en relation matricielle (projection dans une base) ;
4. Déterminer la solution numérique avec les outils de l'algèbre linéaire.



(a) Espace des joints (b) Espace de la tache

FIGURE 3.4 – Exemple de coordonnées représentant l'espace des joints et l'espace tâche d'un robot manipulateur

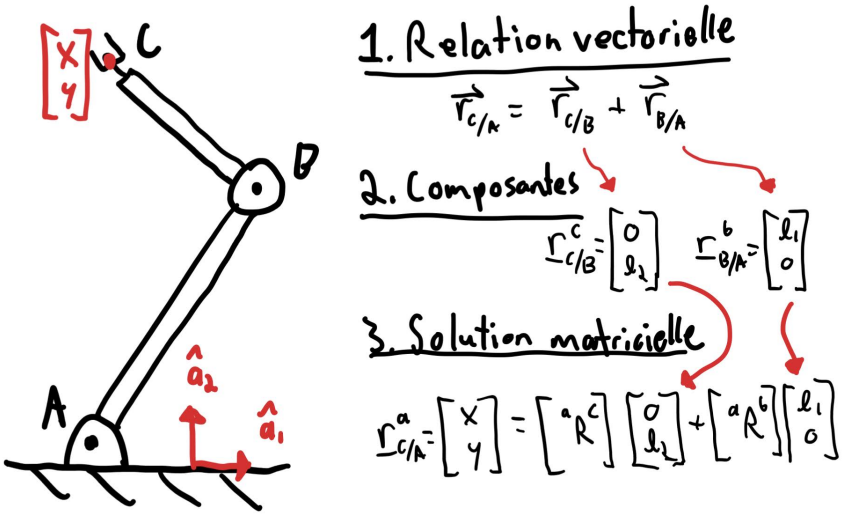


FIGURE 3.5 – Aperçu des grandes lignes de l'approche de modélisation présentée dans ces notes

3.3 Vecteurs géométriques de positions

La notion de vecteur géométrique permet de travailler avec des **vecteurs de position** qui sont indépendants d'un choix de système de coordonnées. Pour les problèmes de cinématique de robots, il est particulièrement adapté de travailler d'abord avec les vecteurs géométrique, plutôt que directement avec des coordonnées. Cette approche **1)** simplifie les calculs en 3D et **2)** permet de faire le saut facilement entre plusieurs systèmes de coordonnées. Un vecteur géométrique est une quantité définie dans l'espace et possédant une grandeur et une direction. Dans ces notes, les vecteurs géométriques, notés $\vec{v}$, seront explicitement distingués des vecteur-colonnes (groupe de plusieurs scalaires) qui seront notés $\boldsymbol{v} = \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix}^T$, voir chapitre 17.

Capsule vidéo
Les vecteurs géométrique de position
https://youtu.be/5DeO4NeFAO8

Un vecteur position nécessite deux points pour être défini, une origine et une destination. Notez que c'est particulier des vecteurs positions, la plupart des autres vecteurs utilisés en physique (vitesse, accélération, force, champ électrique, etc.) ont un point d'application mais sont indépendants d'un choix d'origine. Dans ces notes, les vecteurs positions seront notés :

Définition 3.1 Vecteur position:

$$\vec{r}_{B/A} \quad = \quad \text{Vecteur position du point } B \text{ par rapport au point } A \tag{3.3}$$

où A est le point d'origine et B le point de destination, tel qu'illustré à la figure 3.6. Parfois, la notation équivalente $\overrightarrow{AB}$ est utilisée dans la littérature. Ces vecteurs peuvent être utilisés pour faire des constructions géométriques indépendamment d'un choix de système de coordonnées.

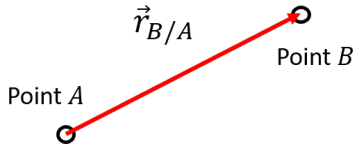
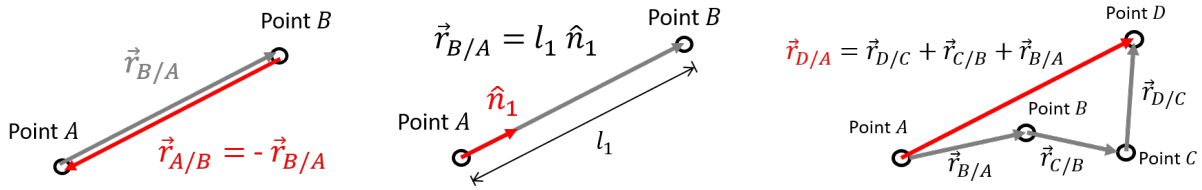


FIGURE 3.6 – Un vecteur position

3.3.1 Propriétés des vecteurs de position

La figure 3.7 illustre les propriétés de base des vecteurs de position.



(a) Inversion (b) Vecteur unitaire et distance (c) Addition

FIGURE 3.7 – Propiétés des vecteurs de position

Propriété 3.1: Addition

Un vecteur position peut être décomposé en une addition de vecteurs qui passent par des points intermédiaires :

$$\vec{r}_{Z/A} = \vec{r}_{Z/Y} + \vec{r}_{Y/X} + ... + \vec{r}_{C/B} + \vec{r}_{B/A} \tag{3.4}$$

Propriété 3.2: Inversion

L'inversion de l'origine et la destination est équivalente à la négation du vecteur de position :

$$\vec{r}_{B/A} = -\vec{r}_{A/B} \tag{3.5}$$

Propriété 3.3: Vecteurs unitaires

Les vecteurs positions peuvent être exprimés comme des **combinaisons de distances et directions** qui sont représentées respectivement par des paires de scalaires l_i et vecteurs unitaires $\hat{n}_i$ (qui représente des directions dans l'espace) :

$$\vec{r} = l_1\hat{n}_1 + l_2\hat{n}_2 + ... + l_n\hat{n}_n \tag{3.6}$$

Propriété 3.4: Norme

La norme d'un vecteur position, i.e. sa longueur, peut être calculée en effectuant un produit scalaire du vecteur avec lui même :

$$\|\vec{r}\|^2 = \vec{r} \cdot \vec{r} \tag{3.7}$$

Propriété 3.5: Projection

La longueur d d'un vecteur position $\vec{r}$ projeté selon un axe particulier correspond au produit scalaire du vecteur position avec un vecteur unitaire $\hat{n}$ aligné sur cet axe :

$$d = \vec{r} \cdot \hat{n} = \|\vec{r}\| \cos\angle(\vec{r}, \hat{n}) \tag{3.8}$$

Cette opération est aussi appelée **projection d'un vecteur sur un axe** et illustrée à la figure 3.8a.

Propriété 3.6: Angle

Le produit scalaire de deux vecteurs unitaires est égale au cosinus de l'angle qui les sépare puisqu'ils ont une dimension unitaire :

$$\cos\angle(\hat{n}_1, \hat{n}_2) = \hat{n}_1 \cdot \hat{n}_2 \tag{3.9}$$

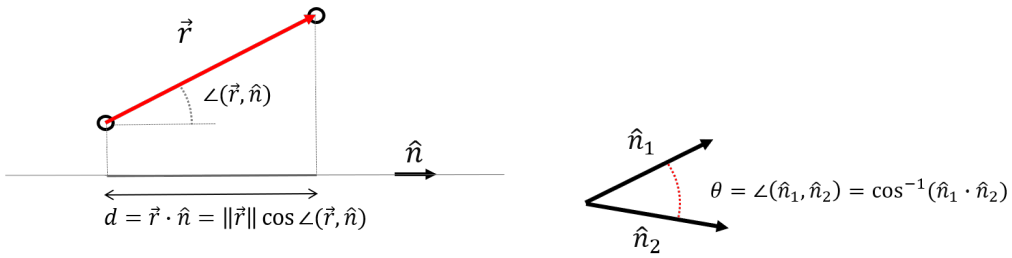
Il est donc possible d'utiliser le produit scalaire pour mesurer un angle entre deux vecteurs, comme illustré à la figure 3.8b.

3.3.2 Procédure d'utilisation pour le calcul de distances

La démarche utilisée pour calculer des positions avec les vecteurs géométriques se résume par les trois étapes suivantes :

1. Par inspection, construire le vecteur position $\vec{r}$ d'intérêt comme une addition/soustraction de plusieurs vecteurs positions $\vec{r}_i$ avec des longueurs et orientations connues :

$$\vec{r} = \sum_i \vec{r}_i \tag{3.10}$$



(a) Distance selon un axe (b) Angle entre deux vecteurs unitaires

FIGURE 3.8 – Utilisation du produit scalaire pour des mesures géométriques

2. Substituer les vecteurs positions symboliques $\vec{r}_i$ par des variables de distance l_i et des vecteurs unitaires $\hat{n}_i$ représentant des directions qui peuvent être mesurées :

$$\vec{r} = \sum_i l_i \hat{n}_i \tag{3.11}$$

3. Calculer les distances désirées d en effectuant un produit scalaire avec les vecteurs unitaires selon les axes désirés :

$$d_x = \vec{r}\cdot\hat{x} = \sum_i l_i(\hat{n}_i\cdot\hat{x}) = \sum_i l_i \cos\angle(\hat{n}_i,\hat{x}) \tag{3.12}$$

$$d_y = \vec{r}\cdot\hat{y} = \sum_i l_i(\hat{n}_i\cdot\hat{y}) = \sum_i l_i \cos\angle(\hat{n}_i,\hat{y}) \tag{3.13}$$

$$d_z = \vec{r}\cdot\hat{z} = \sum_i l_i(\hat{n}_i\cdot\hat{z}) = \sum_i l_i \cos\angle(\hat{n}_i,\hat{z}) \tag{3.14}$$

Capsule vidéo
Exemple de calcul avec les vecteurs géométrique de position
https://youtu.be/YqmyBJPJmP0

Exemple 3.1 Calcul de la hauteur d'un robot:

Un exemple est ici donné pour le calcul de la hauteur de l'effecteur d'un robot à 3 DDL, illustré à la figure 3.9a, avec l'aide des vecteurs géométriques de position. Comme illustré à la Figure 3.9b, par inspection il est possible de déterminer la relation vectorielle suivante :

$$\vec{r}_{D/A} = \vec{r}_{D/C} + \vec{r}_{C/B} + \vec{r}_{B/A} \tag{3.15}$$

Ensuite, comme illustré à la figure 3.9c, des vecteurs unitaires $\hat{n}_i$ alignés avec chacun des liens rigides du robot manipulateur permettent de préciser l'équation (3.15) avec des longueurs et des directions :

$$\vec{r}_{D/A} = l_1\hat{n}_1 + l_2\hat{n}_2 + l_3\hat{n}_3 \tag{3.16}$$

Finalement, pour calculer la hauteur h de ce robot par exemple, il suffit de prendre le produit scalaire avec un vecteur unitaire vertical $\hat{y}$:

$$h = \vec{r}_{D/A}\cdot\hat{y} \tag{3.17}$$

$$h = (l_1\hat{n}_1 + l_2\hat{n}_2 + l_3\hat{n}_3)\cdot\hat{y} \tag{3.18}$$

$$h = l_1(\hat{n}_1\cdot\hat{y}) + l_2(\hat{n}_2\cdot\hat{y}) + l_3(\hat{n}_3\cdot\hat{y}) \tag{3.19}$$

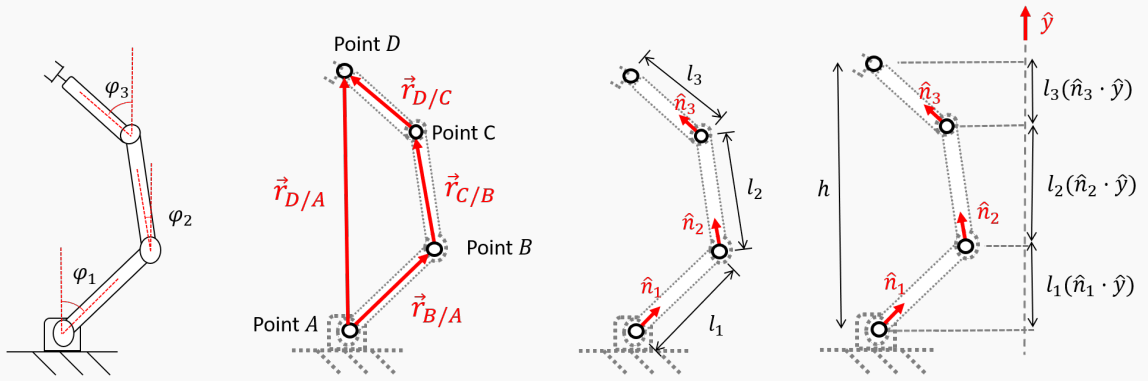
$$h = l_1\cos\angle(\hat{n}_1,\hat{y}) + l_2\cos\angle(\hat{n}_2,\hat{y}) + l_3\cos\angle(\hat{n}_3,\hat{y}) \tag{3.20}$$

À noter ici que le produit scalaire de deux vecteurs unitaires correspond au cosinus de l'angle entre les deux vecteurs, voir équation (3.9). Donc avec l'exemple ci-dessus le produit scalaire $\hat{n}_i\cdot\hat{y}$ correspond

simplement au cosinus de l'angle du joint i avec l'axe vertical, donné par les angles φ_i à la figure 3.9a. L'équation (3.20) peut donc être précisée comme une expression des variables de distances l_i et des variables d'angles φ_i :

$$h = l_1 \cos(\varphi_1) + l_2 \cos(\varphi_2) + l_3 \cos(\varphi_3) \tag{3.21}$$

Dans un cas simple planaire comme celui-ci il aurait été assez simple de calculer la hauteur du robot directement par trigonométrie. L'approche vectorielle est toutefois plus systématique et se généralise beaucoup plus facilement à des systèmes complexes et tri-dimensionnels comme les robots industriels.



(a) Robot à 3 DLL (b) Vecteurs géométrique (c) Vecteur unitaires (d) Projection sur l'axe vertical

FIGURE 3.9 – Exemple de calcul de la hauteur d'un robot.

3.4 Bases vectorielles et composantes d'un vecteur position

Les vecteurs positions tri-dimensionnels $\vec{r}$ peuvent être décris par trois scalaires qui représentent des distances (r_1, r_2 et r_3) qui multiplient trois vecteur unitaires orthogonaux ($\hat{a}_1$, $\hat{a}_2$ et $\hat{a}_3$) qui forment une base vectorielle. Les scalaires r_i qui multiplient chaque vecteur unitaire sont appelés composantes et peuvent être regroupés sous la forme d'un vecteur-colonne.

Définition 3.2 Vecteur-colonne de composantes:

On notera $\boldsymbol{r}^a$ un vecteur-colonne de composante associée à une vecteur-géométrique $\vec{r}$ exprimée selon une base a :

$$\textbf{Vecteur-géométrique :} \quad \vec{r} = r_1^a\,\hat{a}_1 + r_2^a\,\hat{a}_2 + r_3^a\,\hat{a}_3 \qquad \textbf{Vecteur-colonne :} \quad \boldsymbol{r}^a = \begin{bmatrix} r_1^a \\ r_2^a \\ r_3^a \end{bmatrix} \tag{3.22}$$

Note sur la séquence de lecture :

Il est suggérer au lecteur de lire les sections 17.1, 17.2 et 17.3 dans le chapitre 17 qui couvre le calcul vectoriel de façon plus générique. Ces sections couvrent plus en détails les fondements qui sont ici utilisés dans un contexte de cinématique.

Capsule vidéo
Les bases vectorielles et composantes d'un vecteur
https://youtu.be/pZdoFz5PpKU

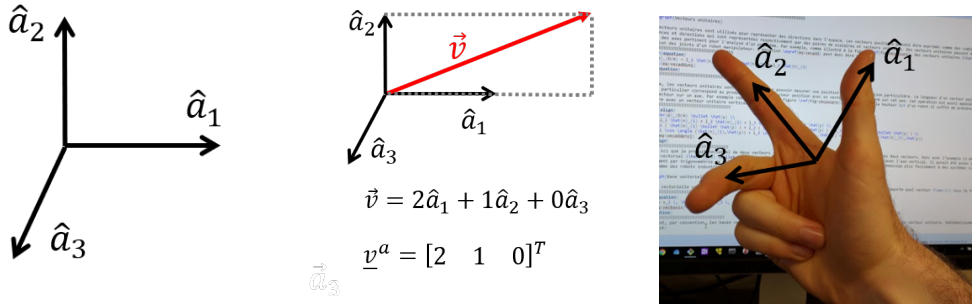
3.4.1 Rappel sur les bases vectorielles

Une base vectorielle correspond à un ensemble de trois vecteurs unitaires (en 3D) orthogonaux qui déterminent l'orientation de trois axes dans l'espace et forment une base qui permet de décrire n'importe quel vecteur position $\vec{r}$ sous la forme :

$$\vec{r} = r_1^a\,\hat{a}_1 + r_2^a\,\hat{a}_2 + r_3^a\,\hat{a}_3 \tag{3.23}$$

Définition 3.3 Base vectorielle:

On appellera *base vectorielle* a, une base formée par un ensemble de vecteurs unitaires $\{\hat{a}_1, \hat{a}_2, \hat{a}_3\}$, qui sont typiquement orthogonaux.



(a) Base a (b) Exemple (c) Convention de la main droite

FIGURE 3.10 – Les bases vectorielles

Les bases vectorielles permettent de faire des opérations et combiner des vecteurs en les exprimant avec une base commune de vecteurs unitaires. Dans la littérature, la notation $(\hat{i},\hat{j},\hat{k})$ ou $(\hat{x},\hat{y},\hat{z})$ est parfois utilisée. Dans ces notes, on utilisera des lettres (a,b,c,...) pour identifier des bases, ainsi que des indices (1,2,3) pour spécifier les trois axes (voir figure 17.1a). Cette approche simplifie la notation lorsqu'un grand nombre de bases sont utilisées simultanément, comme c'est souvent le cas en robotique.

Base vs. repère :

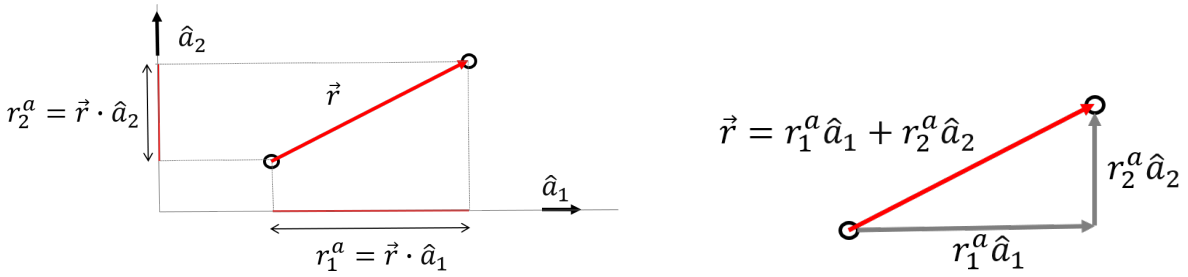
Les bases vectorielles représentent seulement des orientations, la position d'une base est donc sans signification. Dans les dessins et schémas, on dessine une base vectorielle sur un corps rigide pour signifier que celle-ci est attachée au corps rigide et tourne avec celui-ci, toutefois la position de la base vectorielle sur le corps est arbitraire. On parlera de repère, voir section 3.6, lorsqu'un point d'origine est associé à une base vectorielle.

Comme illustré à la figure 17.2a, il est possible de calculer les composantes d'un vecteur exprimé dans une base par un produit scalaire du vecteur géométrique avec chacun des vecteurs unitaires de la base :

$$r_i^a = \vec{r}\cdot\hat{a}_i \quad \Rightarrow \quad \boldsymbol{r}^a = \begin{bmatrix} \vec{r}\cdot\hat{a}_1 \\ \vec{r}\cdot\hat{a}_2 \\ \vec{r}\cdot\hat{a}_3 \end{bmatrix} \tag{3.24}$$

Inversement, comme illustré à la figure 17.2b, le vecteur position géométrique peut être reconstruit en effectuant la somme des composantes multipliées avec leurs vecteurs unitaires respectifs :

$$\vec{r} = \sum_i r_i^a \hat{a}_i = r_1^a\hat{a}_1 + r_2^a\hat{a}_2 + r_3^a\hat{a}_3 \tag{3.25}$$



(a) Projection sur une base vectorielle

(b) Décomposition selon les directions de la base

FIGURE 3.11 – Relations entre le vecteur géométrique de position et les composantes du vecteur-colonne

3.4.2 Transfert d'une équation vectorielle vers une équation matricielle

Lorsque les composantes de plusieurs vecteurs-géométrique de position sont exprimées avec la même base, il est possible de faire des opérations directement avec les vecteur-colonnes de composantes. Cela permet de traiter simultanément les calculs de plusieurs axes, et aussi de faire des calculs numériques efficaces en utilisant les outils de l'algèbre linéaire. Par exemple, les additions et soustractions de vecteurs géométriques peuvent être calculées directement en termes des vecteur-colonnes dans une base :

$$\vec{r}_{A/C} = \vec{r}_{A/B} + \vec{r}_{B/C} \quad \Rightarrow \quad \boldsymbol{r}^a_{A/C} = \boldsymbol{r}^a_{A/B} + \boldsymbol{r}^a_{B/C} \tag{3.26}$$

Ce transfert d'une seule équation vectorielle à une équation matricielle (équivalent à un système de trois équations scalaires) correspond à une projection de l'équation vectorielle sur chacun des axes de la base :

$$\vec{r}_{A/C} = \vec{r}_{A/B} + \vec{r}_{B/C} \quad \Rightarrow \quad \begin{bmatrix} (\vec{r}_{A/C} = \vec{r}_{A/B} + \vec{r}_{B/C})\cdot\hat{a}_1 \\ (\vec{r}_{A/C} = \vec{r}_{A/B} + \vec{r}_{B/C})\cdot\hat{a}_2 \\ (\vec{r}_{A/C} = \vec{r}_{A/B} + \vec{r}_{B/C})\cdot\hat{a}_3 \end{bmatrix} \quad \Rightarrow \quad \boldsymbol{r}^a_{A/C} = \boldsymbol{r}^a_{A/B} + \boldsymbol{r}^a_{B/C} \tag{3.27}$$

Les équations vectorielles avec des vecteurs de position peuvent donc être substituées par des équations matricielles équivalentes avec les vecteur-colonnes, **à condition que tous les vecteur-colonnes soient exprimés dans une base commune**.

Programmation de calculs de cinématique :

Lors de l'écriture d'un programme informatique pour effectuer des calculs de cinématique, les calculs numériques doivent être fait en termes de composantes. En effet, les ordinateurs sont adaptés à manipuler des chiffres, des vecteur-colonnes de chiffres, des matrices de chiffres, etc. Toutefois, un ordinateur ne peut manipuler directement des éléments conceptuels comme les vecteurs unitaires.

3.4.3 Calcul de longueurs, projections et angles avec les composantes

Lorsqu'on connaît les composantes d'un vecteur position $\vec{r}$, le produit intérieur peut être utilisé pour calculer sa longueur :

$$\|\vec{r}\|^2 = \vec{r} \cdot \vec{r} = \boldsymbol{r}^T \boldsymbol{r} = r_1^2 + r_2^2 + r_3^2 \tag{3.28}$$

$$\|\vec{r}\| = \sqrt{\boldsymbol{r}^T \boldsymbol{r}} \tag{3.29}$$

pour calculer une distance projetée selon un axe décrit par un vecteur unitaire $\hat{n}$:

$$d_n = \vec{r} \cdot \hat{n} = \boldsymbol{r}^T \boldsymbol{n} = \begin{bmatrix} r_1 & r_2 & r_3 \end{bmatrix} \begin{bmatrix} n_1 \\ n_2 \\ n_3 \end{bmatrix} = r_1 n_1 + r_2 n_2 + r_3 n_3 \tag{3.30}$$

pour calculer un angle entre deux vecteurs unitaires $\hat{x}$ et $\hat{y}$:

$$\cos \angle(\hat{x}, \hat{y}) = \hat{x} \cdot \hat{y} = \boldsymbol{x}^T \boldsymbol{y} \tag{3.31}$$

$$\angle(\hat{x}, \hat{y}) = \arccos\left(\boldsymbol{x}^T \boldsymbol{y}\right) \tag{3.32}$$

et pour calculer entre deux vecteurs position $\vec{r}_A$ et $\vec{r}_B$:

$$\vec{r}_A \cdot \vec{r}_B = \|\vec{r}_A\| \|\vec{r}_B\| \cos \angle(\vec{r}_A, \vec{r}_B) = \boldsymbol{r}_A^T \boldsymbol{r}_B \tag{3.33}$$

$$\angle(\vec{r}_A, \vec{r}_B) = \arccos\left(\frac{\boldsymbol{r}_A^T \boldsymbol{r}_B}{\sqrt{\boldsymbol{r}_A^T \boldsymbol{r}_A} \sqrt{\boldsymbol{r}_B^T \boldsymbol{r}_B}}\right) \tag{3.34}$$

Il est à noter que dans ces équations les vecteur-colonnes de composantes doivent tous être dans la même base. Les résultats sont des scalaires qui ne dépendent pas de la base utilisée.

Capsule vidéo
Opérations vectorielles avec les composantes de vecteurs
`https://youtu.be/7brvShmayAk`

3.5 Les matrices de rotations

Une matrice de rotation notée ${}^aR^b$, est une matrice (3×3 en 3D) qui permet de calculer directement les composants d'un vecteur position dans une base a avec les composants connus dans une base b, i.e. effectuer un **changement de base**, par une multiplication de la forme :

$$\boldsymbol{r}^a = {}^aR^b\boldsymbol{r}^b \tag{3.35}$$

La matrice de rotation est un raccourci pour projeter un vecteur position sur trois nouveaux axes en une seule opération matricielle, plutôt que d'effectuer trois opérations de projection indépendantes. Cette matrice représente l'orientation relative de deux bases vectorielles et **l'opération de multiplication d'un vecteur-colonne $\boldsymbol{r}^a$ par la matrice de rotation ${}^bR^a$ correspond à la projection du vecteur géométrique associé $\vec{r}$ sur la base vectorielle b.**

Deux rôles pour la cinématique :

Une matrice de rotation a deux fonctions majeures dans un contexte de cinématique : **1)** effectuer des opérations de changement de base et **2)** représenter une orientation relative. Cette section présente le rôle des matrices de rotation pour les opérations de changement de base, les matrices de rotation seront discutées à nouveau dans le contexte de la représentation de l'orientation d'un corps rigide à la section 3.7.

Capsule vidéo
Les matrices de rotation et les changements de base
https://youtu.be/hVmcRYD1KFQ

3.5.1 Définitions

Une matrice de rotation est définie par les produits scalaires des vecteurs unitaires qui forment deux bases vectorielles :

Définition 3.4 Matrice de rotation:

$${}^aR^b = \begin{bmatrix} \hat{a}_1\cdot\hat{b}_1 & \hat{a}_1\cdot\hat{b}_2 & \hat{a}_1\cdot\hat{b}_3 \\ \hat{a}_2\cdot\hat{b}_1 & \hat{a}_2\cdot\hat{b}_2 & \hat{a}_2\cdot\hat{b}_3 \\ \hat{a}_3\cdot\hat{b}_1 & \hat{a}_3\cdot\hat{b}_2 & \hat{a}_3\cdot\hat{b}_3 \end{bmatrix} = \begin{bmatrix} \cos\angle(\hat{a}_1,\hat{b}_1) & \cos\angle(\hat{a}_1,\hat{b}_2) & \cos\angle(\hat{a}_1,\hat{b}_3) \\ \cos\angle(\hat{a}_2,\hat{b}_1) & \cos\angle(\hat{a}_2,\hat{b}_2) & \cos\angle(\hat{a}_2,\hat{b}_3) \\ \cos\angle(\hat{a}_3,\hat{b}_1) & \cos\angle(\hat{a}_3,\hat{b}_2) & \cos\angle(\hat{a}_3,\hat{b}_3) \end{bmatrix} \tag{3.36}$$

Les produits scalaires correspondent aussi aux cosinus des angles entre les vecteurs unitaires, c'est pourquoi la matrice de rotation est aussi parfois appelée la **matrice des cosinus directeurs**. Par inspection de l'équation (3.36), on remarque que les colonnes de la matrice ${}^aR^b$ correspondent aux vecteur-colonnes des composantes des vecteurs unitaires $\hat{b}_i$ exprimées dans la base a et que les rangés de la matrice ${}^aR^b$ correspondent à la transposée des vecteur-colonnes des composantes des vecteurs unitaires $\hat{a}_i$ exprimées dans la base b. Cela permet donc des définitions équivalentes plus compactes :

$${}^aR^b = \begin{bmatrix} \begin{bmatrix} \\ \boldsymbol{b}_1^a \\ \\ \end{bmatrix} & \begin{bmatrix} \\ \boldsymbol{b}_2^a \\ \\ \end{bmatrix} & \begin{bmatrix} \\ \boldsymbol{b}_3^a \\ \\ \end{bmatrix} \end{bmatrix} = \begin{bmatrix} [\quad (\boldsymbol{a}_1^b)^T \quad] \\ [\quad (\boldsymbol{a}_2^b)^T \quad] \\ [\quad (\boldsymbol{a}_3^b)^T \quad] \end{bmatrix} \tag{3.37}$$

Finalement, la matrice de rotation peut aussi être définie en termes de composantes et d'indices par :

$${}^aR^b_{ij} = \hat{a}_i\cdot\hat{b}_j \tag{3.38}$$

une équation qui est simple à se rappeler et permet d'éviter de mélanger l'ordre de rotation ${}^aR^b$ vs. ${}^bR^a$ avec la notation utilisée dans ces notes.

Démonstration. Si les composantes d'un vecteur $\vec{r}$ sont connus dans une base a :

$$\boldsymbol{r}^a = \begin{bmatrix} r_1^a \\ r_2^a \\ r_3^a \end{bmatrix} \tag{3.39}$$

le vecteur géométrique associé est donné par :

$$\vec{r} = r_1^a \hat{a}_1 + r_2^a \hat{a}_2 + r_3^a \hat{a}_3 \tag{3.40}$$

Les composantes dans une base b peuvent être calculées par des produits scalaires suivants :

$$r_1^b = \vec{r} \cdot \hat{b}_1 \tag{3.41}$$
$$r_2^b = \vec{r} \cdot \hat{b}_2 \tag{3.42}$$
$$r_3^b = \vec{r} \cdot \hat{b}_3 \tag{3.43}$$

Si on substitue $\vec{r}$ par l'équation (3.40) :

$$r_1^b = (r_1^a \hat{a}_1 + r_2^a \hat{a}_2 + r_3^a \hat{a}_3) \cdot \hat{b}_1 \tag{3.44}$$
$$r_2^b = (r_1^a \hat{a}_1 + r_2^a \hat{a}_2 + r_3^a \hat{a}_3) \cdot \hat{b}_2 \tag{3.45}$$
$$r_3^b = (r_1^a \hat{a}_1 + r_2^a \hat{a}_2 + r_3^a \hat{a}_3) \cdot \hat{b}_3 \tag{3.46}$$

et on réarrange les termes :

$$r_1^b = (\hat{a}_1 \cdot \hat{b}_1) r_1^a + (\hat{a}_2 \cdot \hat{b}_1) r_2^a + (\hat{a}_3 \cdot \hat{b}_1) r_3^a \tag{3.47}$$
$$r_2^b = (\hat{a}_1 \cdot \hat{b}_2) r_1^a + (\hat{a}_2 \cdot \hat{b}_2) r_2^a + (\hat{a}_3 \cdot \hat{b}_2) r_3^a \tag{3.48}$$
$$r_3^b = (\hat{a}_1 \cdot \hat{b}_3) r_1^a + (\hat{a}_2 \cdot \hat{b}_3) r_2^a + (\hat{a}_3 \cdot \hat{b}_3) r_3^a \tag{3.49}$$

On obtient trois équations qui peuvent être regroupées sous forme matricielle :

$$\underbrace{\begin{bmatrix} r_1^b \\ r_2^b \\ r_3^b \end{bmatrix}}_{\boldsymbol{r}^b} = \underbrace{\begin{bmatrix} \hat{a}_1 \cdot \hat{b}_1 & \hat{a}_2 \cdot \hat{b}_1 & \hat{a}_3 \cdot \hat{b}_1 \\ \hat{a}_1 \cdot \hat{b}_2 & \hat{a}_2 \cdot \hat{b}_2 & \hat{a}_3 \cdot \hat{b}_2 \\ \hat{a}_1 \cdot \hat{b}_3 & \hat{a}_2 \cdot \hat{b}_3 & \hat{a}_3 \cdot \hat{b}_3 \end{bmatrix}}_{{}^b R^a} \underbrace{\begin{bmatrix} r_1^a \\ r_2^a \\ r_3^a \end{bmatrix}}_{\boldsymbol{r}^a} \tag{3.50}$$

La matrice de rotation ${}^b R^a$ consiste donc à des produits scalaires entre les vecteurs unitaires des bases vectorielles a et b comme définit à l'équation (3.36). □

Exemple 3.2 Matrice de rotation dans le plan calculée par trigonométrie:

La figure 3.12 montre un exemple d'un vecteur $\vec{r}$ en 2 dimensions exprimé dans deux base vectorielles b et a. Comme illustré à la figure 3.13, avec une construction de triangles il est possible de déterminer par trigonométrie les relations entre les composantes dans les deux bases, et donc la matrice de rotation ${}^a R^b$. L'approche par trigonométrie est toutefois limitée pour les rotations en 3D, une approche vectorielle plus simple et plus universelle pour calculer des matrices de rotation est présentée à la section 3.5.2.

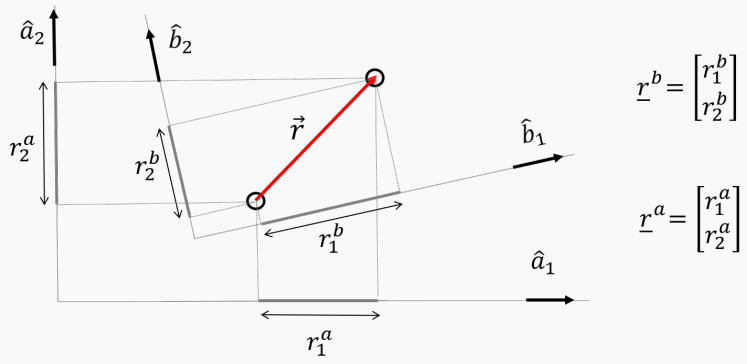


FIGURE 3.12 – Vecteur position exprimé dans deux bases vectorielles

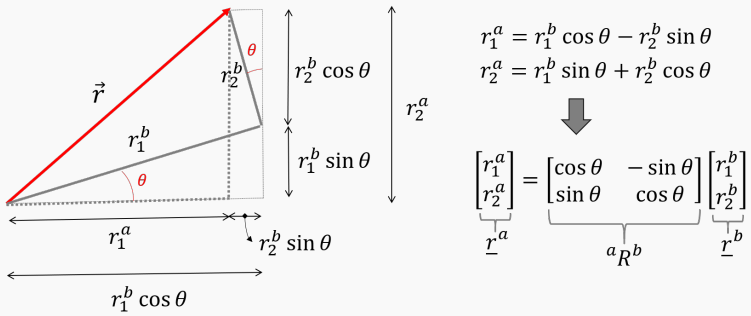


FIGURE 3.13 – Calcul par trigonométrie de la matrice de rotation 2x2 ${}^aR^b$ pour le changement de base illustré à la figure 3.12

3.5.2 Méthode de calcul des matrices de rotation basé sur les colonnes

La définition des matrices de rotation basée sur les vecteurs unitaires de eq.(3.37) est particulièrement utile pour calculer une matrice de rotation. Une astuce pour ne pas mélanger l'ordre de rotation (${}^bR^a$ vs. ${}^aR^b$) est d'avoir en tête que la multiplication matricielle c'est une combinaison linéaire de vecteur-colonnes :

$$\boldsymbol{r}^a = \underbrace{\begin{bmatrix} \begin{bmatrix} \\ \boldsymbol{b}_1^a \\ \\ \end{bmatrix} & \begin{bmatrix} \\ \boldsymbol{b}_2^a \\ \\ \end{bmatrix} & \begin{bmatrix} \\ \boldsymbol{b}_3^a \\ \\ \end{bmatrix} \end{bmatrix}}_{{}^aR^b} \underbrace{\begin{bmatrix} r_1^b \\ r_2^b \\ r_3^b \end{bmatrix}}_{\boldsymbol{r}^b} = \begin{bmatrix} \\ \boldsymbol{b}_1^a \\ \\ \end{bmatrix} r_1^b + \begin{bmatrix} \\ \boldsymbol{b}_2^a \\ \\ \end{bmatrix} r_2^b + \begin{bmatrix} \\ \boldsymbol{b}_3^a \\ \\ \end{bmatrix} r_3^b \tag{3.51}$$

et donc que si on cherche la matrice de rotation ${}^aR^b$ pour calculer le passage de la base b vers la base a, il faut trouver les directions spatiales $\hat{b}_1$, $\hat{b}_2$ et $\hat{b}_3$ qui seront combinées linéairement selon les coeficients r_1^b, r_2^b, r_3^b.

Une méthode rapide et universelle pour calculer une matrice de rotation est de construire par inspection les vecteur unitaires $\hat{b}_i$ de la base vectorielle initiale, comme une combinaison de vecteurs unitaires $\hat{a}_i$ de la base cible :

$$\hat{b}_1 = a\,\hat{a}_1 + b\,\hat{a}_2 + c\,\hat{a}_3 \qquad \hat{b}_2 = d\,\hat{a}_1 + e\,\hat{a}_2 + f\,\hat{a}_3 \qquad \hat{b}_3 = g\,\hat{a}_1 + h\,\hat{a}_2 + i\,\hat{a}_3 \tag{3.52}$$

pour ensuite convertir les équations en vecteur-colonnes $\boldsymbol{b}_i^a$:

$$\boldsymbol{b}_1^a = \begin{bmatrix} a \\ b \\ c \end{bmatrix} \qquad \boldsymbol{b}_2^a = \begin{bmatrix} d \\ e \\ f \end{bmatrix} \qquad \boldsymbol{b}_3^a = \begin{bmatrix} g \\ h \\ i \end{bmatrix} \tag{3.53}$$

et assembler la matrice avec la définition de l'équation (3.37) :

$$ {}^{a}R^{b} = \begin{bmatrix} \boldsymbol{b}_1^a & \boldsymbol{b}_2^a & \boldsymbol{b}_3^a \end{bmatrix} = \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & i \end{bmatrix} \tag{3.54} $$

Pour les matrices de rotation élémentaires, i.e. les matrices qui représentent une rotation d'un angle θ autour de l'axe $\hat{b}_1$, $\hat{b}_2$ ou $\hat{b}_3$, les neufs éléments $a, b, c, d, ..$ de la matrice peuvent seulement prendre comme valeurs $\cos\theta$, $\pm\sin\theta$, 1 ou 0. Il est donc relativement facile de les identifier par inspection, comme il est illustré à la section suivante.

Capsule vidéo
Les matrices de rotation : propriétés et exemple de calcul
https://youtu.be/CNIN7UpyXvo

3.5.3 Matrices de rotation élémentaires

Les matrices de rotation élémentaires représentent des rotations relatives d'un angle θ autour de l'axe 1, 2 ou 3 entre deux bases vectorielles. Ces matrices seront notées $R_i(\theta)$ où i est l'indice de l'axe de rotation. Ces matrices sont caractérisées par le fait que le vecteur unitaire $\hat{b}_i$ de la base mobile coïncide avec le vecteur unitaire $\hat{a}_i$ de la base fixe, et la relation entre les autres vecteurs unitaires peut être facilement représentée dans le plan perpendiculaire à l'axe de rotation.

Notation simplifiée pour les calculs trigonométriques :

Les symboles $c\theta$ et $s\theta$ sont ici introduits pour alléger les équations et représentent les fonctions trigonométriques $\cos(\theta)$ et $\sin(\theta)$ respectivement :

$$ s\theta = \sin(\theta) \tag{3.55} $$
$$ c\theta = \cos(\theta) \tag{3.56} $$

Pour les calculs de cinématique qui implique plusieurs angles, une notation encore plus compacte sera utilisée avec seulement les lettres s ou c et des indices qui spécifie les angles :

$$ s_i = \sin(\theta_i) \tag{3.57} $$
$$ c_i = \cos(\theta_i) \tag{3.58} $$
$$ s_{ijk} = \sin(\theta_i + \theta_j + \theta_k) \tag{3.59} $$

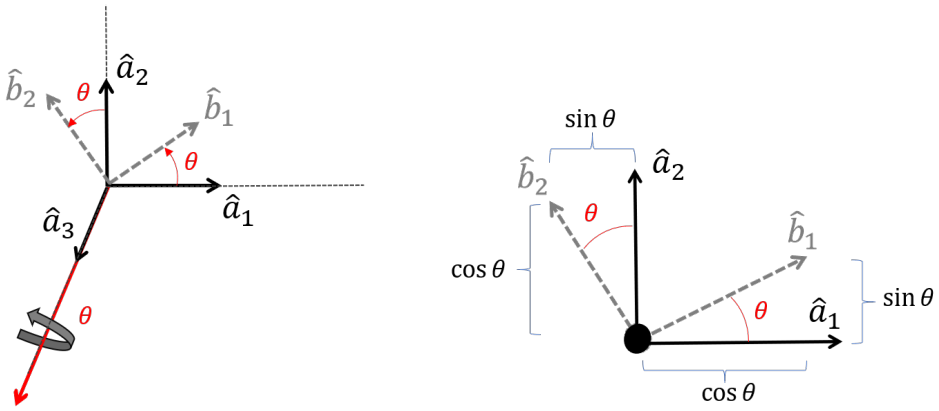
Rotation selon l'axe 3 Si les bases vectorielles a et b ont une orientation relative décrite par une rotation d'un angle θ autour de l'axe 3, les vecteurs unitaires $\hat{a}_3$ et $\hat{b}_3$ sont égaux et, comme illustré à la figure 3.14, les quatre autres vecteurs unitaires ($\hat{a}_1$, $\hat{b}_1$, $\hat{a}_2$ et $\hat{b}_2$) sont dans le plan formé par l'axe 1 et l'axe 2. Les vecteurs unitaires sont reliés par des fonctions trigonométriques simples :

$$ \hat{b}_1 = c\theta\,\hat{a}_1 + s\theta\,\hat{a}_2 \qquad \hat{b}_2 = -s\theta\,\hat{a}_1 + c\theta\,\hat{a}_2 \qquad \hat{b}_3 = \hat{a}_3 \tag{3.60} $$

La matrice ${}^{a}R^{b}$ peut donc être formée grâce à la définition (3.37) :

$$ \boldsymbol{b}_1^a = \begin{bmatrix} c\theta \\ s\theta \\ 0 \end{bmatrix} \quad \boldsymbol{b}_2^a = \begin{bmatrix} -s\theta \\ c\theta \\ 0 \end{bmatrix} \quad \boldsymbol{b}_3^a = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad \Rightarrow \quad {}^{a}R_3^b(\theta) = \begin{bmatrix} c\theta & -s\theta & 0 \\ s\theta & c\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.61} $$

À noter que le coin supérieur gauche de la matrice ${}^{a}R_3^b(\theta)$ correspond à une matrice de rotation 2×2 dans la plan.



(a) 3D (b) Vue de face

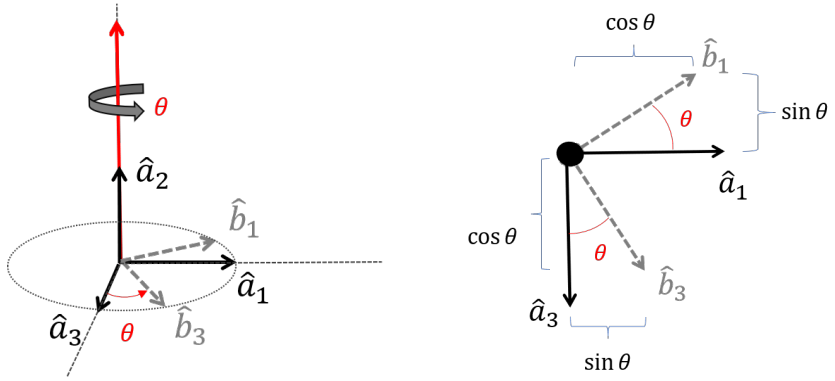
FIGURE 3.14 – Rotation relative selon l'axe 3

Rotation selon l'axe 2 Si les bases vectorielles a et b ont une orientation relative décrite par une rotation d'un angle θ autour de l'axe 2, les vecteurs unitaires $\hat{a}_2$ et $\hat{b}_2$ sont égaux et, comme illustré à la figure 3.15, les quatre autres vecteurs unitaires ($\hat{a}_1$, $\hat{b}_1$, $\hat{a}_3$ et $\hat{b}_3$) sont dans le plan formé par l'axe 1 et l'axe 3. Les vecteurs unitaires sont reliés par des fonctions trigonométriques simples :

$$\hat{b}_1 = c\theta\,\hat{a}_1 - s\theta\,\hat{a}_3 \qquad \hat{b}_2 = \hat{a}_2 \qquad \hat{b}_3 = s\theta\,\hat{a}_1 + c\theta\,\hat{a}_3 \tag{3.62}$$

La matrice ${}^aR^b$ peut donc être formée grâce à la définition (3.37) :

$$\boldsymbol{b}_1^a = \begin{bmatrix} c\theta \\ 0 \\ -s\theta \end{bmatrix} \quad \boldsymbol{b}_2^a = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad \boldsymbol{b}_3^a = \begin{bmatrix} s\theta \\ 0 \\ c\theta \end{bmatrix} \quad \Rightarrow \quad {}^aR_2^b(\theta) = \begin{bmatrix} c\theta & 0 & s\theta \\ 0 & 1 & 0 \\ -s\theta & 0 & c\theta \end{bmatrix} \tag{3.63}$$



(a) 3D (b) Vue de dessus

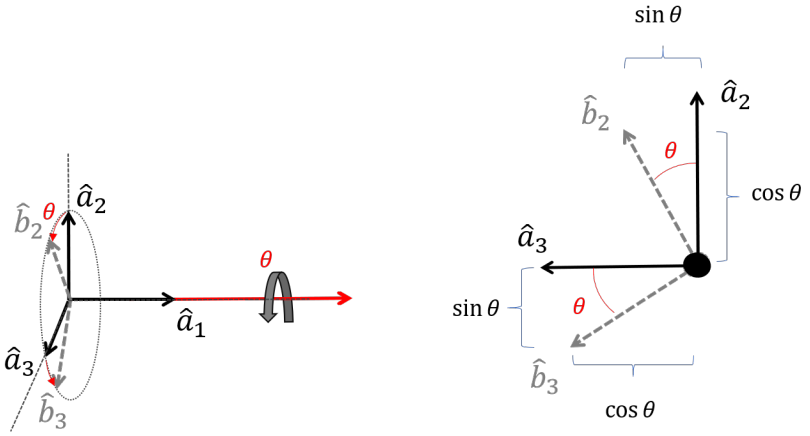
FIGURE 3.15 – Rotation relative selon l'axe 2

Rotation selon l'axe 1 Si les bases vectorielles a et b ont une orientation relative décrite par une rotation d'un angle θ autour de l'axe 1, les vecteurs unitaires $\hat{a}_1$ et $\hat{b}_1$ sont égaux et, comme illustré à la figure 3.14, les quatre autres vecteurs unitaires ($\hat{a}_2$, $\hat{b}_2$, $\hat{a}_3$ et $\hat{b}_3$) sont dans le plan formé par l'axe 2 et l'axe 3. Les vecteurs unitaires sont reliés par des fonctions trigonométriques simples :

$$\hat{b}_1 = \hat{a}_1 \qquad \hat{b}_2 = c\theta\,\hat{a}_2 + s\theta\,\hat{a}_3 \qquad \hat{b}_3 = -s\theta\,\hat{a}_2 + c\theta\,\hat{a}_3 \tag{3.64}$$

La matrice ${}^aR^b$ peut donc être formée grâce à la définition (3.37) :

$$\boldsymbol{b}_1^a = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \boldsymbol{b}_2^a = \begin{bmatrix} 0 \\ c\theta \\ s\theta \end{bmatrix} \quad \boldsymbol{b}_3^a = \begin{bmatrix} 0 \\ -s\theta \\ c\theta \end{bmatrix} \quad \Rightarrow \quad {}^aR_1^b(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\theta & -s\theta \\ 0 & s\theta & c\theta \end{bmatrix} \tag{3.65}$$



(a) 3D (b) Vue de côté

FIGURE 3.16 – Rotation relative selon l'axe 1

3.5.4 Propriétés des matrices de rotation

Cette section présente plusieurs propriétés des matrices de rotation qui sont utiles dans un contexte de changement de bases vectorielles.

Propriété 3.7: Norme des colonnes et rangés

Les colonnes $\boldsymbol{c}_i$ et les rangés $\boldsymbol{r}_i$ d'une matrice de rotation ont une norme égale à 1 :

$$R = \begin{bmatrix} \begin{bmatrix} \\ \boldsymbol{c}_1 \\ \\ \end{bmatrix} & \begin{bmatrix} \\ \boldsymbol{c}_2 \\ \\ \end{bmatrix} & \begin{bmatrix} \\ \boldsymbol{c}_3 \\ \\ \end{bmatrix} \end{bmatrix} = \begin{bmatrix} [\quad \boldsymbol{r}_1^T \quad] \\ [\quad \boldsymbol{r}_2^T \quad] \\ [\quad \boldsymbol{r}_3^T \quad] \end{bmatrix} \quad \Rightarrow \quad (\boldsymbol{c}_i)^T \boldsymbol{c}_i = 1 \quad \& \quad (\boldsymbol{r}_i)^T \boldsymbol{r}_i = 1 \quad \forall i \tag{3.66}$$

L'identité trigonométrique suivante est souvent utile pour ce type de calculs :

$$\sin^2(\theta) + \cos^2(\theta) = 1 \tag{3.67}$$

Propriété 3.8: Matrice identité

Une matrice de rotation est réduite à une matrice identitée lorsque les bases vectorielle sont coïncidentes. Par exemple, pour les matrices de rotation élémentaires des équations (3.61), (3.63) et (3.65), lorsque l'angle est égale à zéro, les matrices sont réduites à des matrices identités :

$$R_1(\theta) = R_2(\theta) = R_3(\theta) = I_{3\times 3} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{si} \quad \theta = 0 \tag{3.68}$$

Propriété 3.9: Inversion

Pour inverser la direction d'une opération de changement de base, il suffit de multiplier par l'inverse de la matrice de rotation :

$$\boldsymbol{r}^a = {}^aR^b\,\boldsymbol{r}^b \tag{3.69}$$
$$({}^aR^b)^{-1}\,\boldsymbol{r}^a = ({}^aR^b)^{-1}\,{}^aR^b\,\boldsymbol{r}^b \tag{3.70}$$
$$({}^aR^b)^{-1}\,\boldsymbol{r}^a = \boldsymbol{r}^b \tag{3.71}$$

L'inversion d'une matrice de rotation est équivalente à inverser le signe de l'angle θ utilisé dans le calcul de la matrice, à prendre la transposée de la matrice et, en termes de notation, à inverser les bases :

$$R(\theta)^{-1} = R(-\theta) \tag{3.72}$$
$$R^{-1} = R^T \tag{3.73}$$
$$({}^bR^a)^{-1} = {}^aR^b \tag{3.74}$$

Propriété 3.10: Changements de base successifs

Des changements de base successifs $a \to b \to c \to ... \to z$ peuvent être combinés en une seule matrice de rotation en multipliant les matrices de rotation intermédiaires par la gauche :

$${}^cR^a = {}^cR^b\,{}^bR^a \tag{3.75}$$
$${}^dR^a = {}^dR^c\,{}^cR^b\,{}^bR^a \tag{3.76}$$
$$\vdots \tag{3.77}$$
$${}^zR^a = {}^zR^y\,{}^yR^x\,\ldots\,{}^cR^b\,{}^bR^a \tag{3.78}$$

Notez que selon la notation utilisée dans ces notes, **les lettres des bases intermédiaires doivent être côte-à-côte dans les équations**, et la matrice de rotation totale conserve la première et la dernière base de la séquence dans le même ordre. Cette propriété découle des définitions, en combinant deux changements de base intermédiaires, une matrice totale peut être obtenue :

$$\left.\begin{aligned}\boldsymbol{r}^b &= {}^bR^a\boldsymbol{r}^a\\ \boldsymbol{r}^c &= {}^cR^b\boldsymbol{r}^b\\ \boldsymbol{r}^c &= {}^cR^a\boldsymbol{r}^a\end{aligned}\right\} \quad\Rightarrow\quad \boldsymbol{r}^c = \underbrace{{}^cR^b\,{}^bR^a}_{{}^cR^a}\boldsymbol{r}^a \quad\Rightarrow\quad {}^cR^a = {}^cR^b\,{}^bR^a \tag{3.79}$$

Propriété 3.11: Non-commutativité

Le produit des matrices de rotation n'est toutefois pas commutatif, sauf pour des petites rotations infinitésimales. L'ordre de multiplication des matrices est donc important et généralement :

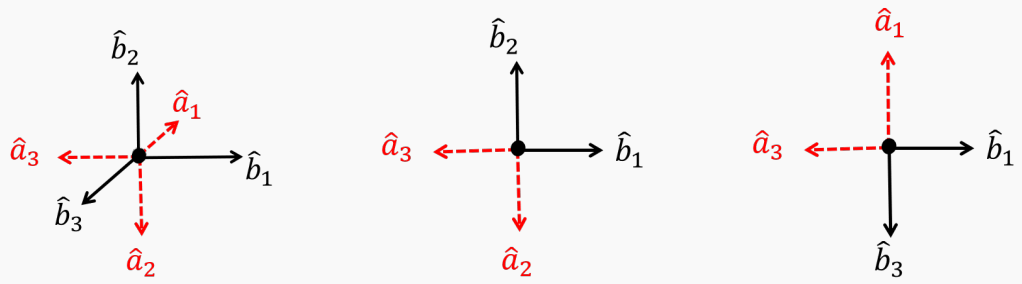
$${}^cR^b\,{}^bR^a \neq {}^bR^a\,{}^cR^b \tag{3.80}$$

Une **exception à cette règle** est pour les rotations successives selon le même axe de rotation. Par exemple pour deux matrices de rotation élémentaires selon un axe i :

$$R_i(\theta_1)R_i(\theta_2) = R_i(\theta_2)R_i(\theta_1) = R_i(\theta_1+\theta_2) \tag{3.81}$$

Exemple 3.3 Réorganisation des vecteurs unitaires:

La figure 3.17 illustre deux bases vectorielles qui ont des vecteurs unitaires parallèles mais avec des directions et un ordre différents. Pour calculer la matrice de rotation associée ${}^{b}R^{a}$, la première étape est d'identifier la relation entre les vecteurs unitaires $\hat{a}_i$ et $\hat{b}_i$.



(a) Vue 3D (b) Vue de face (c) Vue de dessus

FIGURE 3.17 – Réorganisation des directions des vecteurs unitaires

On peut constater que le vecteur unitaire $\hat{a}_1 = -\hat{b}_3$, le vecteur unitaire $\hat{a}_2 = -\hat{b}_2$ et le vecteur unitaire $\hat{a}_3 = -\hat{b}_1$. Avec ces résultats il est possible d'assembler les vecteur-colonnes $\boldsymbol{a}_i^b$:

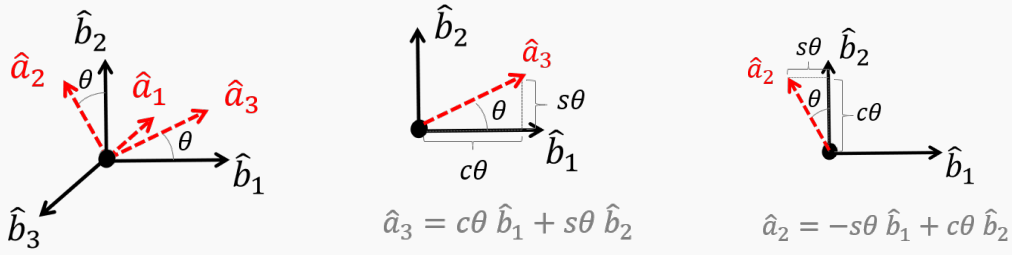
$$\boldsymbol{a}_1^b = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \quad \boldsymbol{a}_2^b = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} \quad \boldsymbol{a}_3^b = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \tag{3.82}$$

Il est donc maintenant possible d'assembler la matrice de rotation ${}^{b}R^{a}$ basée sur l'équation (3.37) :

$${}^{b}R^{a} = \begin{bmatrix} \boldsymbol{a}_1^b & \boldsymbol{a}_2^b & \boldsymbol{a}_3^b \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & -1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \tag{3.83}$$

Exemple 3.4 Réorganisation des vecteurs unitaires et rotation selon un axe:

Un exemple de calcul d'une matrice de rotation ${}^{b}R^{a}$ est donné pour une base a qui subit une révolution de θ degrés autour de l'axe $\hat{b}_3$ par rapport à une base b, mais ici avec des vecteurs unitaires qui sont définis avec des directions différentes, tel qu'illustré à la figure 3.18a, et ce n'est donc pas un cas de rotation élémentaire. Pour construire la matrice de rotation, la première étape est de dessiner les deux bases avec une origine commune, dans une configuration où la rotation θ est relativement faible (pour simplifier la visualisation). On peut constater que le vecteur unitaire $\hat{a}_1$ est aligné avec l'axe de rotation, donc indépendant de θ et toujours donné par $\hat{a}_1 = -\hat{b}_3$. Ensuite, par inspection, il est possible d'exprimer chaque vecteur unitaire $\hat{a}_i$ comme une combinaison des vecteurs unitaires $\hat{b}_i$, avec des relations trigonométriques simples comme illustrés par les figures 3.18b et 3.18.



(a) Base vectorielles a et b (b) Calcul de $\boldsymbol{a}_3^b$ (c) Calcul de $\boldsymbol{a}_2^b$

FIGURE 3.18 – Rotation selon l'axe 3 et réorganisation des directions des vecteurs unitaires

Avec ces résultats il est possible d'assembler les vecteur-colonnes $\boldsymbol{a}_i^b$:

$$\boldsymbol{a}_1^b = \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix} \quad \boldsymbol{a}_2^b = \begin{bmatrix} -s\theta \\ c\theta \\ 0 \end{bmatrix} \quad \boldsymbol{a}_3^b = \begin{bmatrix} c\theta \\ s\theta \\ 0 \end{bmatrix} \tag{3.84}$$

Il est donc maintenant possible d'assembler la matrice de rotation ${}^bR^a$ basé sur l'équation (3.37) :

$${}^bR^a = \begin{bmatrix} \boldsymbol{a}_1^b & \boldsymbol{a}_2^b & \boldsymbol{a}_3^b \end{bmatrix} = \begin{bmatrix} 0 & -s\theta & c\theta \\ 0 & c\theta & s\theta \\ -1 & 0 & 0 \end{bmatrix} \tag{3.85}$$

Exemple 3.5 Calcul de la position globale de l'effecteur d'un robot:

Dans cet exemple, le robot précédemment étudié (Figure 3.9) est ici analysé à nouveau mais cette fois avec l'aide des bases vectorielles. L'objectif est ici de calculer la position du point D par rapport au point A exprimée dans la base vectorielle a. Comme illustré à la figure **??**, la première étape est de décrire la position qui doit être calculée en terme de vecteurs de position géométriques :

$$\vec{r}_{D/A} = \vec{r}_{D/C} + \vec{r}_{C/B} + \vec{r}_{B/A} \tag{3.86}$$

Cette relation vectorielle peut ensuite être convertie en une équation matricielle entre les composantes dans une base a fixe qui correspond aux axes selon lesquels la position doit être exprimée :

$$\boldsymbol{r}_{D/A}^a = \boldsymbol{r}_{D/C}^a + \boldsymbol{r}_{C/B}^a + \boldsymbol{r}_{B/A}^a \tag{3.87}$$

Ensuite, comme illustré à la Figure 3.19, les différents vecteurs positions qui représentent chaque joint ont des composantes connues dans des bases vectorielles locales attachées aux différents liens rigides du robot :

$$\boldsymbol{r}_{B/A}^b = \begin{bmatrix} l_1 \\ 0 \\ 0 \end{bmatrix} \qquad \boldsymbol{r}_{C/B}^c = \begin{bmatrix} 0 \\ l_2 \\ 0 \end{bmatrix} \qquad \boldsymbol{r}_{D/C}^d = \begin{bmatrix} 0 \\ l_3 \\ 0 \end{bmatrix} \tag{3.88}$$

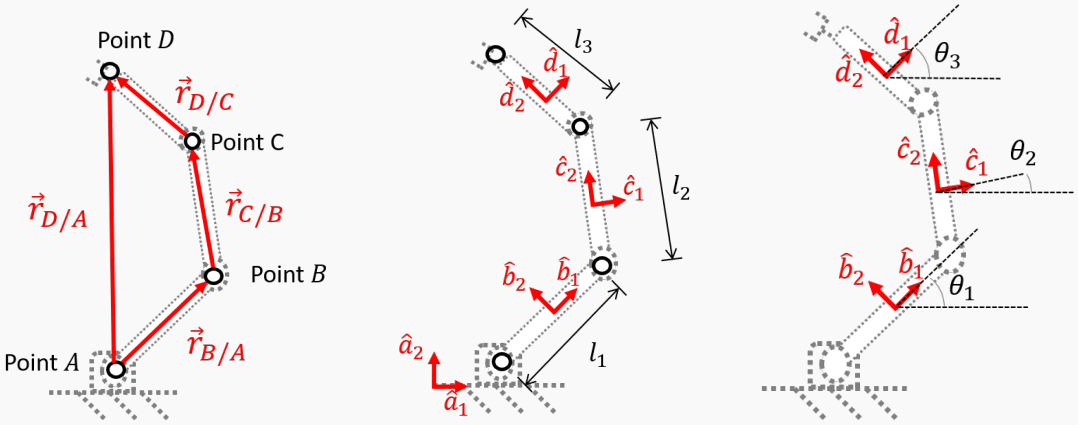
Il est à noter que ces vecteur-colonnes qui utilisent les bases locales sont constants et indépendants des angles des joints car les base locales tournent avec les liens rigides. Il est ensuite possible de convertir ces vecteur-colonnes de positions dans la base a par des opérations de changement de base avec les matrices de rotation appropriées :

$$\boldsymbol{r}_{D/C}^a = {}^aR^d\,\boldsymbol{r}_{D/C}^d \qquad \boldsymbol{r}_{C/B}^a = {}^aR^c\,\boldsymbol{r}_{C/B}^c \qquad \boldsymbol{r}_{B/A}^a = {}^aR^b\,\boldsymbol{r}_{B/A}^b \tag{3.89}$$

Finalement, en substituant dans l'équation (3.87), l'équation matricielle nécessaire pour obtenir les composantes du vecteur position $\vec{r}_{D/A}$ exprimé avec la base a est obtenue :

$$\boldsymbol{r}_{D/A}^a = {}^aR^d\,\boldsymbol{r}_{D/C}^d + {}^aR^c\,\boldsymbol{r}_{C/B}^c + {}^aR^b\,\boldsymbol{r}_{B/A}^b \tag{3.90}$$

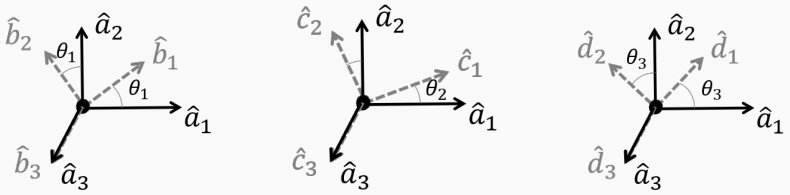
Les matrices de rotation doivent ensuite être calculées pour solutionner cette équation.



(a) Construction géométrique (b) Bases vectorielles (c) Angles

FIGURE 3.19 – Utilisation des bases vectorielles pour les vecteurs de position

Grâce à une inspection des mouvements relatifs des bases vectorielles, comme illustrée à la figure 3.19b, les matrices de rotation peuvent être calculées. Puisque le robot est planaire, les bases vectorielles b, c et d fixées aux pièces mobiles du robot ont tous une rotation relative autour de l'axe $\hat{a}_3$ par rapport à la base vectorielle fixe a, tel qu'illustré à la figure 3.20.



(a) ${}^a R^b$ (b) ${}^a R^c$ (c) ${}^a R^d$

FIGURE 3.20 – Calcul des matrices de rotation pour l'exemple 3.5.4

En fonction des angles définis à la figure 3.19c, les matrices de rotation sont donc toutes des matrices élémentaires de rotation autour de l'axe 3 :

$$ {}^a R^b = \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 \\ s\theta_1 & c\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad {}^a R^c = \begin{bmatrix} c\theta_2 & -s\theta_2 & 0 \\ s\theta_2 & c\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad {}^a R^d = \begin{bmatrix} c\theta_3 & -s\theta_3 & 0 \\ s\theta_3 & c\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.91} $$

Ces résultats peuvent donc être substitués dans l'équation matricielle qui avait été obtenue à l'exemple 3.5.4 :

$$ \boldsymbol{r}^a_{D/A} = {}^a R^d \, \boldsymbol{r}^d_{D/C} + {}^a R^c \, \boldsymbol{r}^c_{C/B} + {}^a R^b \, \boldsymbol{r}^b_{B/A} \tag{3.92} $$

et on obtient l'expression précise qui permet de calculer la position du point D par rapport au point de référence A exprimée dans la base vectorielle a en fonction des variables de distances l_i et d'angles

θ_i :

$$\boldsymbol{r}^a_{D/A} = \begin{bmatrix} c\theta_3 & -s\theta_3 & 0 \\ s\theta_3 & c\theta_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ l_3 \\ 0 \end{bmatrix} + \begin{bmatrix} c\theta_2 & -s\theta_2 & 0 \\ s\theta_2 & c\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ l_2 \\ 0 \end{bmatrix} + \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 \\ s\theta_1 & c\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} l_1 \\ 0 \\ 0 \end{bmatrix} \tag{3.93}$$

$$\boldsymbol{r}^a_{D/A} = \begin{bmatrix} -s\theta_3 l_3 - s\theta_2 l_2 + c\theta_1 l_1 \\ c\theta_3 l_3 + c\theta_2 l_2 + s\theta_1 l_1 \\ 0 \end{bmatrix} \tag{3.94}$$

À noter que ici les angles θ_i sont tous des angles absolus qui représentent l'orientation des bases b, c et d du robot par rapport à la base fixe a directement. Lorsqu'on contrôle un bras robotisé, c'est généralement les angles relatifs entre chacun des joints qui sont les variables mesurées et contrôlées.

Exercice de lecture : La hauteur du robot correspond à la deuxième ligne de l'équation (3.94), qui est la distance $\vec{r}_{D/A}$ selon le vecteur unitaire $\hat{a}_2$:

$$h = c\theta_3 l_3 + c\theta_2 l_2 + s\theta_1 l_1 \tag{3.95}$$

Alternativement, à l'exemple 3.3.2, l'équation suivante avait été obtenue pour la hauteur du robot :

$$h = l_1 \cos(\varphi_1) + l_2 \cos(\varphi_2) + l_3 \cos(\varphi_3) \tag{3.96}$$

Trouvez la correspondance entre les angles θ_i et φ_i et vérifiez que les deux expressions sont équivalentes.

Exemple 3.6 Distance projetée calculée avec les vecteur-colonnes:

Comme illustré à la figure 3.21, cet exemple démontre comment calculer la distance projetée d_n d'un vecteur $\vec{r}_{C/A}$ selon un axe décrit par $\hat{n}$, connaissant les vecteur-colonnes de composantes :

$$\boldsymbol{r}^b_{C/B} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad \boldsymbol{r}^a_{B/A} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \qquad \boldsymbol{n}^b = \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} \tag{3.97}$$

Premièrement, l'équation vectorielle de vecteurs géométriques de position est développée :

$$\vec{r}_{C/A} = \vec{r}_{C/B} + \vec{r}_{B/A} \tag{3.98}$$

pour ensuite effectuer le produit scalaire pour calculer la distance projetée :

$$d_n = \vec{r}_{C/A} \cdot \hat{n} = (\vec{r}_{C/B} + \vec{r}_{B/A}) \cdot \hat{n} \tag{3.99}$$

Deuxièmement, on choisi la base b pour transformer la relation vectorielle en équation matricielle :

$$d_n = (\boldsymbol{r}^b_{C/B} + \boldsymbol{r}^b_{B/A})^T \boldsymbol{n}^b \tag{3.100}$$

$$d_n = (\boldsymbol{r}^b_{C/B})^T \boldsymbol{n}^b + (\boldsymbol{r}^b_{B/A})^T \boldsymbol{n}^b \tag{3.101}$$

Tous les vecteur-colonnes sont connus sauf le vecteur-colonne $\boldsymbol{r}^b_{C/B}$, qui doit être substitué par $\boldsymbol{r}^b_{C/B} = {}^bR^a \, \boldsymbol{r}^a_{C/B}$ car les composantes de $\vec{r}_{C/B}$ sont connues seulement dans la base a. On obtient alors :

$$d_n = ({}^bR^a \, \boldsymbol{r}^a_{C/B})^T \boldsymbol{n}^b + (\boldsymbol{r}^b_{B/A})^T \boldsymbol{n}^b \tag{3.102}$$

$$d_n = (\boldsymbol{n}^b)^T \, {}^bR^a \, \boldsymbol{r}^a_{C/B} + (\boldsymbol{r}^b_{B/A})^T \boldsymbol{n}^b \tag{3.103}$$

et ce qui correspond en termes de composantes à l'équation :

$$d_n = \begin{bmatrix} n_1^b & n_2^b & n_3^b \end{bmatrix} \begin{bmatrix} {}^bR_{11}^a & {}^bR_{12}^a & {}^bR_{13}^a \\ {}^bR_{21}^a & {}^bR_{22}^a & {}^bR_{23}^a \\ {}^bR_{31}^a & {}^bR_{32}^a & {}^bR_{33}^a \end{bmatrix} \begin{bmatrix} r_{C/B,1}^a \\ r_{C/B,2}^a \\ r_{C/B,3}^a \end{bmatrix} + \begin{bmatrix} r_{B/A,1}^b & r_{B/A,2}^b & r_{B/A,3}^b \end{bmatrix} \begin{bmatrix} n_1^b \\ n_2^b \\ n_3^b \end{bmatrix} \tag{3.104}$$

si on substitue par les composantes connues définies à l'équation (3.97) :

$$d_n = \begin{bmatrix} n_x & n_y & n_z \end{bmatrix} \begin{bmatrix} {}^bR_{11}^a & {}^bR_{12}^a & {}^bR_{13}^a \\ {}^bR_{21}^a & {}^bR_{22}^a & {}^bR_{23}^a \\ {}^bR_{31}^a & {}^bR_{32}^a & {}^bR_{33}^a \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} X & Y & Z \end{bmatrix} \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} \tag{3.105}$$

C'est donc cette équation qui devrait être programmée pour faire ce calcul numériquement sur un ordinateur.

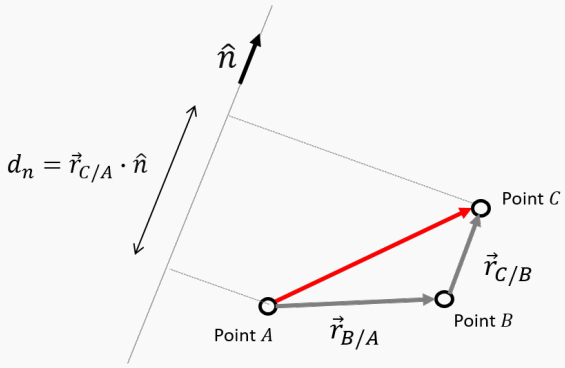


FIGURE 3.21 – Exemple de calcul d'une distance projetée

3.6 Coordonnées dans un repère et transformations homogènes

En robotique, il est souvent utile de placer un point de référence ainsi qu'une base vectorielle sur chaque corps rigide du robot. Bien qu'on peut travailler avec des points de référence et des bases vectorielles de façon indépendante, il est souvent pratique de les jumeler en paires puisqu'il faut ces deux types de références pour exprimer des coordonnées. **La combinaison d'un point d'origine et d'une base vectorielle est appelée repère dans ces notes**, le terme *Frame* est généralement utilisé en anglais.

Définition 3.5 Repère:

Un repère A est défini comme un point d'origine A_o et une base vectorielle a (qui consiste des trois vecteurs unitaires $\{\hat{a}_1, \hat{a}_2, \hat{a}_3\}$). Autrement dit, on appellera *repère* A le repère formé par l'ensemble suivant :

$$\textbf{Repère } A = \{A_o, \hat{a}_1, \hat{a}_2, \hat{a}_3\} \tag{3.106}$$

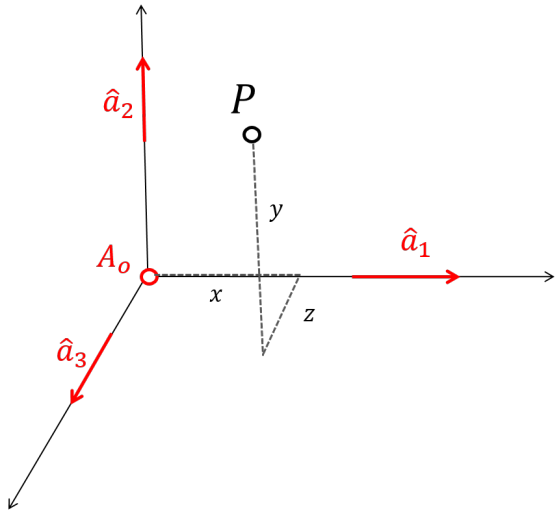


FIGURE 3.22 – Système de coordonnées cartésiennes (x, y, z) basé sur le repère $\{A_o, \hat{a}_1, \hat{a}_2, \hat{a}_3\}$

Comme illustré à la figure 3.22, les coordonnées cartésiennes $(x,\ y,\ z)$ d'un point P correspondent directement aux composantes du vecteur position avec comme origine l'intersection des axes du système de coordonnées (le point A_o) et exprimé dans une base vectorielle a alignée sur les axes du système de coordonnées :

Définition 3.6 Coordonnées cartésiennes dans un repère:

Des coordonnées cartésiennes (x, y, z) basé sur le repère $\{A_o, \hat{a}_1, \hat{a}_2, \hat{a}_3\}$ représentent le vecteur suivant :

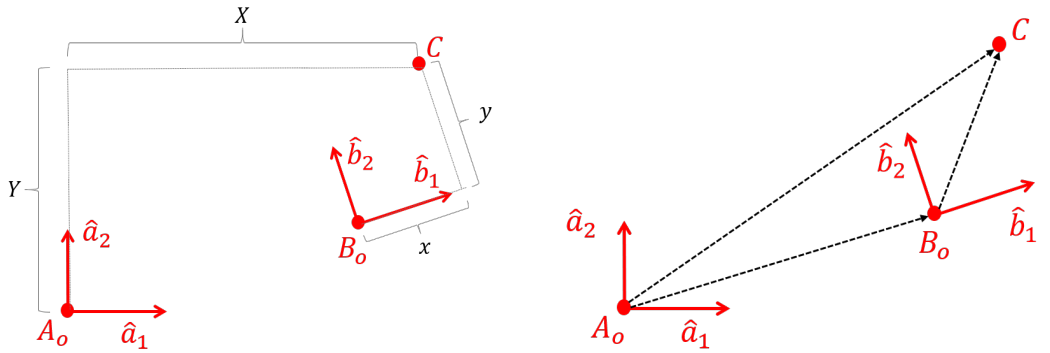
$$\boldsymbol{r}^a_{P/A_o} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \Leftrightarrow \quad \vec{r}_{P/A_o} = x\,\hat{a}_1 + y\,\hat{a}_2 + z\,\hat{a}_3 \tag{3.107}$$

Un système de coordonnées est toujours implicitement attaché à un repère, même s'il n'est pas explicitement défini. L'ensemble de l'origine des axes et des vecteurs unitaires alignés sur ces axes forment un repère.

Capsule vidéo
Les repères et les matrices de transformation
https://youtu.be/yTPA1R8b8XE

3.6.1 Changement de repère

Un changement de repère implique deux opérations : **1)** un changement de l'origine utilisée pour les vecteurs de position et **2)** un changement de la base vectorielle utilisée pour exprimer les composantes de ce vecteur position. Comme illustré à la figure 3.23 pour une transformation d'un repère B vers un repère A,



(a) Deux systèmes de coordonnées (b) Construction géométrique

FIGURE 3.23 – Changement du repère $\{B_o, \hat{b}_1, \hat{b}_2, \hat{b}_3\}$ vers le repère $\{A_o, \hat{a}_1, \hat{a}_2, \hat{a}_3\}$

les coordonnées cartésiennes d'un point C correspondent au vecteur-colonne $\boldsymbol{r}^b_{C/B_o}$ dans le repère B et au vecteur-colonne $\boldsymbol{r}^a_{C/A_o}$ dans le repère A :

$$\boldsymbol{r}^b_{C/B_o} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad \boldsymbol{r}^a_{C/A_o} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.108}$$

Pour effectuer la transformation des coordonnées (x, y, z) vers les coordonnées (X, Y, Z), la première étape est de construire la relation vectorielle qui relie les trois points :

$$\vec{r}_{C/A_o} = \vec{r}_{C/B_o} + \vec{r}_{B_o/A_o} \tag{3.109}$$

et ensuite il suffit de tout exprimer ces vecteurs dans la base a pour effectuer l'addition :

$$\boldsymbol{r}^a_{C/A_o} = \boldsymbol{r}^a_{C/B_o} + \boldsymbol{r}^a_{B_o/A_o} \tag{3.110}$$

$$\boldsymbol{r}^a_{C/A_o} = {}^aR^b \boldsymbol{r}^b_{C/B_o} + \boldsymbol{r}^a_{B_o/A_o} \tag{3.111}$$

ce qui correspond en termes de coordonnées à :

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = {}^aR^b \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \boldsymbol{r}^a_{B_o/A_o} \tag{3.112}$$

Un changement de base est nécessaire puisque que le vecteur $\vec{r}_{C/B_o}$ est connu dans la base vectorielle b. Donc les informations nécessaires pour effectuer la transformation sont : **1)** les composantes $\boldsymbol{r}^a_{B_o/A_o}$ du vecteur position qui relie les origines A_o et B_o et 2) la matrice de rotation ${}^aR^b$ qui donne l'orientation relative des bases vectorielles a et b.

L'**opération inverse** s'obtient à partir de l'équation (3.111), d'abord on soustrait le vecteur translation $\boldsymbol{r}^a_{B_o/A_o}$ et on multiplie par la gauche l'inverse de la matrice de rotation pour obtenir :

$$\boldsymbol{r}^b_{C/B_o} = {}^bR^a\, \boldsymbol{r}^a_{C/A_o} - {}^bR^a\, \boldsymbol{r}^a_{B_o/A_o} \tag{3.113}$$

ce qui correspond en termes des coordonnées à :

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = {}^bR^a \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + {}^bR^a\, \boldsymbol{r}^a_{B_o/A_o} \tag{3.114}$$

3.6.2 Coordonnées et transformations homogènes

Dans les logiciels de cinématique et de graphique, les opérations de changement de repère peuvent être très fréquentes. Il existe une méthode standard pour combiner l'opération d'addition vectorielle et la multiplication avec la matrice de rotation en une seule opération équivalente qui consiste à multiplier une matrice de transformation 4×4 notée ${}^AT^B$, avec un vecteur-colonne 4×1 de *coordonnées homogènes*. Les coordonnées homogènes sont tout simplement les coordonnées cartésiennes pour les trois premières composantes et la valeur 1 pour la quatrième composante.

Le changement de repère décrit à la section 3.6.1 correspond alors à l'opération suivante :

$$\begin{bmatrix} \boldsymbol{r}^a_{C/A_o} \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} {}^aR^b & \boldsymbol{r}^a_{B_o/A_o} \\ 0\,0\,0 & 1 \end{bmatrix}}_{{}^AT^B} \begin{bmatrix} \boldsymbol{r}^b_{C/B_o} \\ 1 \end{bmatrix} \tag{3.115}$$

ce qui correspond en termes de coordonnées à :

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} {}^aR^b & \boldsymbol{r}^a_{B_o/A_o} \\ 0\,0\,0 & 1 \end{bmatrix}}_{{}^AT^B} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{3.116}$$

Les trois premières rangés des équations matricielles (3.115) et (3.116) correspondent exactement à l'équation (3.111), et la dernière ligne correspond simplement à une équation 1=1. Les transformations homogènes c'est donc rien de fondamentalement nouveau, mais une astuce bien utile dans un contexte de programmation pour grouper deux opérations en une seule.

Si on substitut les définitions, la matrice de transformation peut être définie directement en termes de vecteurs de vecteurs unitaires et du vecteur géométrique qui définie la translation d'une origine à l'autre :

Définition 3.7 Matrice de transformation:

$${}^AT^B = \begin{bmatrix} {}^aR^b & \boldsymbol{r}^a_{B_o/A_o} \\ 0\,0\,0 & 1 \end{bmatrix} = \begin{bmatrix} \hat{a}_1 \cdot \hat{b}_1 & \hat{a}_1 \cdot \hat{b}_2 & \hat{a}_1 \cdot \hat{b}_3 & \hat{a}_1 \cdot \vec{r}_{B_o/A_o} \\ \hat{a}_2 \cdot \hat{b}_1 & \hat{a}_2 \cdot \hat{b}_2 & \hat{a}_2 \cdot \hat{b}_3 & \hat{a}_2 \cdot \vec{r}_{B_o/A_o} \\ \hat{a}_3 \cdot \hat{b}_1 & \hat{a}_3 \cdot \hat{b}_2 & \hat{a}_3 \cdot \hat{b}_3 & \hat{a}_3 \cdot \vec{r}_{B_o/A_o} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.117}$$

L'**opération inverse** de changement de repère avec les coordonnées homogènes est formée par :

$$\begin{bmatrix} \boldsymbol{r}^b_{C/B_o} \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} {}^bR^a & -{}^bR^a\, \boldsymbol{r}^a_{B_o/A_o} \\ 0\,0\,0 & 1 \end{bmatrix}}_{({}^AT^B)^{-1}} \begin{bmatrix} \boldsymbol{r}^a_{C/A_o} \\ 1 \end{bmatrix} \tag{3.118}$$

Comme pour les matrices de rotation l'inverse de la matrice de transformation correspond à inverser les repères avec la notation utilisée :

$$({}^AT^B)^{-1} = {}^BT^A \tag{3.119}$$

Dernièrement, les **changements de repères successifs** $A \to B \to C \to ... \to Z$ peuvent être combinés en une seule matrice de transformation en multipliant les matrices de transformation intermédiaires par la gauche :

$$^{C}T^{A} = {}^{C}T^{B}\,{}^{B}T^{A} \tag{3.120}$$

$$^{D}T^{A} = {}^{D}T^{C}\,{}^{C}T^{B}\,{}^{B}T^{A} \tag{3.121}$$

$$\vdots$$

$$^{Z}T^{A} = {}^{Z}T^{Y}\,{}^{Y}T^{X}\,...\,{}^{C}T^{B}\,{}^{B}T^{A} \tag{3.122}$$

Avec la notation utilisée dans ces notes, **les lettres des repères intermédiaires doivent être côte-à-côte dans les équations**, et la matrice de transformation totale conserve le premier et le dernier repère de la séquence dans le même ordre.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

3.7 Représentation de la pose d'un corps rigide

De façon générale, six variables sont nécessaires pour définir la positon d'un corps rigide par rapport à un repère, trois pour décrire la translation et trois pour décrire l'orientation. Pour un corps rigide sans aucune contrainte, par exemple un avion en vol, le nombre de DDL est donc de six.

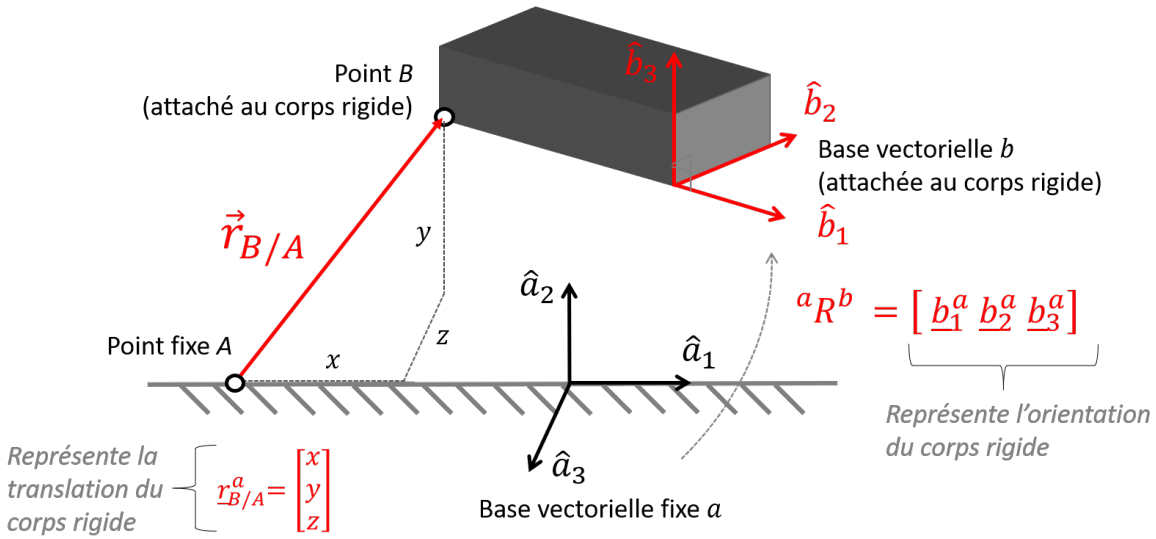


FIGURE 3.24 – Représentation de la pose d'un corp rigide

Capsule vidéo
Pose (position et orientation) d'un corps rigide
https://youtu.be/wrw4T8OE36U

Translation : Les trois DDL de translation peuvent être représentés par un vecteur position qui a comme origine un point fixe A et comme cible un point arbitraire B sur le corps rigide. Ce vecteur position $\vec{r}_{B/A}$, est typiquement représenté avec trois coordonnées cartésiennes (x, y, z) qui utilisent une base vectorielle fixe a :

$$\boldsymbol{r}_{B/A}^a = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \quad \Leftrightarrow \quad \vec{r}_{B/A} = x\,\vec{a}_1 + y\,\vec{a}_2 + z\,\vec{a}_3 \tag{3.123}$$

Orientation : Ensuite, même avec un point défini dans l'espace par un vecteur translation, un corps rigide a plusieurs orientations possibles. Pour définir l'orientation, une autre base vectorielle b attachée au corps rigide doit être utilisée. L'orientation d'un corps rigide est alors représentée par la direction des vecteurs unitaires $\hat{b}_i$, qui peuvent être représentés par leurs composantes dans la base fixe a. Les trois vecteur-colonnes $\boldsymbol{b}_i^a$ de trois composantes peuvent être regroupés sous la forme d'une matrice de rotation, comme il a été discuté à la section 3.5 dans le contexte des changements de base :

$${}^aR^b = \begin{bmatrix} \boldsymbol{b}_1^a & \boldsymbol{b}_2^a & \boldsymbol{b}_3^a \end{bmatrix} = R(\phi, \theta, \psi) \tag{3.124}$$

Malgré les neuf composants d'une matrice de rotation, puisque les vecteurs $\hat{b}_i$ sont unitaires et orthogonaux seulement trois variables suffisent pour la définir ; il y a un total de neuf composantes et six équations de contraintes, voir les équations (17.4) et (17.5). Il est donc possible de paramétrer la matrice ${}^aR^b$ avec trois variables, par exemple trois angles ϕ, θ et ψ, sans limiter les orientations possibles. On appelle trois angles qui paramétrisent une matrice de rotation des angles de Euler, diverses conventions de paramétrisation sont possibles et c'est le sujet de la section 3.7.1. Selon la situation, d'autres représentations alternatives aux

matrices de rotation et aux angles de Euler peuvent aussi être utiles, chaque représentation a ses avantages et inconvénients. La section 3.7.2 présente la représentation axe-angle et la section 3.7.3 présente les quaternions.

Pose : La pose d'un corps rigide, c'est la combinaison de sa translation et de son orientation. Il est possible d'utiliser des représentations indépendantes pour la translation et l'orientation, mais alternativement on peut regrouper une matrice de rotation (pour représenter l'orientation) et un vecteur-colonne (pour représenter la translation) sous la forme d'une matrice de transformation 4x4, qui relie le repère $\{A_o, \hat{a}_1, \hat{a}_2, \hat{a}_3\}$ formé par le point fixe et la base fixe, au repère $\{B_o, \hat{b}_1, \hat{b}_2, \hat{b}_3\}$ formé par le point mobile et la base mobile attachés au corps rigide. Cette matrice peut alors être paramétrée par 6 variables qui représent tous les DDL du corps rigide :

$$ {}^{A}T^{B}(x,y,z,\phi,\theta,\psi) = \begin{bmatrix} {}^{a}R^{b} & \boldsymbol{r}^{a}_{B/A} \\ 0\,0\,0 & 1 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \\ R(\phi,\theta,\psi) \\ \\ \end{bmatrix} & \begin{bmatrix} x \\ y \\ z \end{bmatrix} \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix} \tag{3.125} $$

La matrice de transformation ${}^{A}T^{B}$ a donc deux fonctions : **1)** elle peut être utilisée pour effectuer des changements de repères comme vue à la section 3.6.2, et **2)** elle peut aussi être utilisée pour représenter la pose d'un repère B (donc aussi la pose du corps rigide sur lequel il est attaché) relativement à un repère A.

3.7.1 Angles de Euler

Une matrice de rotation arbitraire peut être paramétrée avec trois variables. La méthode la plus simple pour y parvenir est de construire une matrice de rotation comme une multiplication de trois matrices élémentaires. Par exemple, l'orientation arbitraire d'une base vectorielle d par rapport à une base vectorielle a peut être décrite par les angles (ϕ, θ, ψ), avec la convention 3-1-3 :

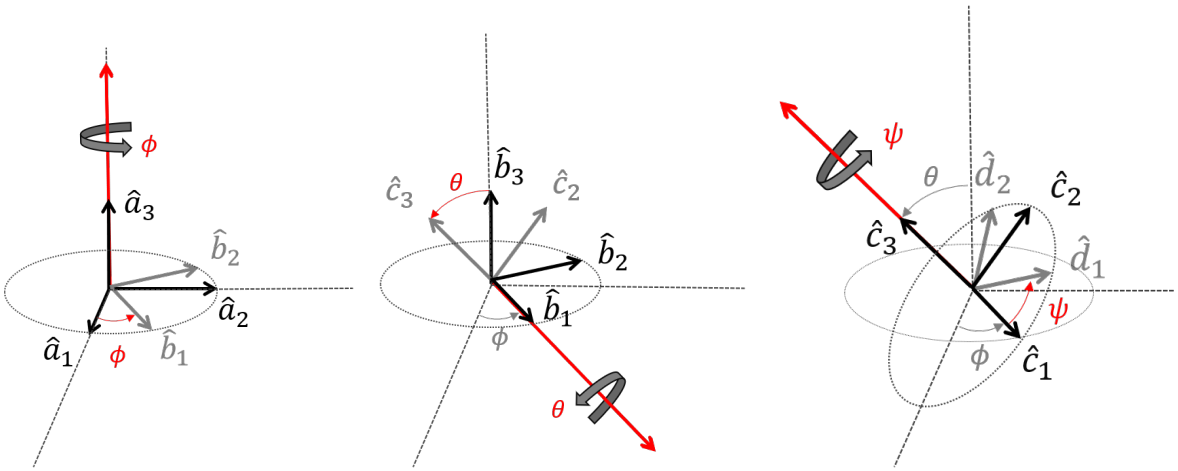
$$ {}^{a}R^{d}(\phi,\theta,\psi) = R_3(\phi)\,R_1(\theta)\,R_3(\psi) \tag{3.126} $$

$$ {}^{a}R^{d}(\phi,\theta,\psi) = \begin{bmatrix} c\phi & -s\phi & 0 \\ s\phi & c\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\theta & -s\theta \\ 0 & s\theta & c\theta \end{bmatrix} \begin{bmatrix} c\psi & -s\psi & 0 \\ s\psi & c\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.127} $$

$$ {}^{a}R^{d}(\phi,\theta,\psi) = \begin{bmatrix} +c\phi\,c\psi - s\phi\,c\theta\,c\psi & -c\phi\,s\psi - s\phi\,c\theta c\psi & +s\phi\,s\theta \\ +s\phi\,c\psi + c\phi\,c\theta\,c\psi & -s\phi\,s\psi + c\phi\,c\theta c\psi & -c\phi\,s\theta \\ +s\theta\,s\psi & +s\theta\,c\psi & +c\theta \end{bmatrix} \tag{3.128} $$

Cette matrice peut être interprétée comme une suite de trois rotations, avec des bases vectorielles intermédiaires, illustrées à la figure 3.25.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard



(a) ${}^{a}R^{b} = R_3(\phi)$ (b) ${}^{b}R^{c} = R_1(\theta)$ (c) ${}^{c}R^{d} = R_3(\psi)$

FIGURE 3.25 – Angles de Euler avec la convention 3-1-3 : toutes les orientations possibles de la base d peuvent être décrites par la matrice de rotation ${}^{a}R^{d}(\phi, \theta, \psi) = R_3(\phi)R_1(\theta)R_3(\psi)$ paramétrisée avec trois angles.

Une autre convention de trois angles utilisée fréquemment en robotique (typiquement pour les véhicules) sont les angles de roulis, tangage et lacet. Ces angles correspondent à une matrice formée par la multiplication de matrices élémentaires dans un ordre 3-2-1, si l'axe 1 correspond à la direction avant-arrière du véhicule et l'axe 2 vers à la direction haut-bas du véhicule :

$$R_{321}(\theta_{tangage}, \theta_{lacet}, \theta_{roulis}) = R_3(\theta_{tangage})\, R_2(\theta_{lacet})\, R_1(\theta_{roulis}) \tag{3.129}$$

Comme l'ordre des matrices de rotation qui sont multipliées influence la matrice résultante, il y a plusieurs possibilités de définitions des trois angles de Euler qui paramétrisent la matrice. Différents auteurs, domaines et organisations utilisent des définitions différentes, il est donc toujours **important de vérifier la convention utilisée si on travail avec les angles de Euler**.

3.7.2 Représentation axe-angle

Une orientation relative entre deux bases vectorielles peut aussi être représentée par un axe de rotation, décris par un vecteur unitaire $\hat{a}$, et un angle θ de rotation autour de celui-ci, comme illustré à la figure 3.26.

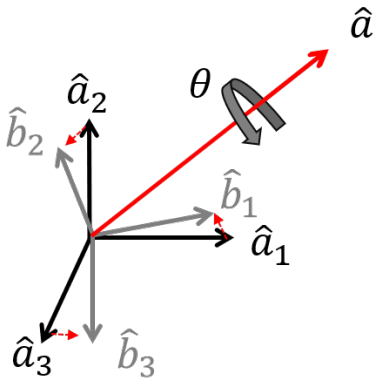


FIGURE 3.26 – Représentation axe-angle d'une orientation relative

Cette représentation nécessite quatre paramètres, trois pour un vecteur-colonne $\boldsymbol{a}$ qui représente les composantes du vecteur unitaire qui décris l'axe de rotation, et un pour l'angle θ. La représentation axe-angle permet donc d'utiliser seulement 4 variables plutôt que les 9 d'une matrice 3×3 pour représenter une

rotation dans un espace tridimensionnel. Un paramètre est toutefois redondant, car le vecteur-colonne $\boldsymbol{a}$ est unitaire et trois variables sont reliés par l'équation $a_1^2+a_2^2+a_3^2=1$. De plus, cette représentation présente des singularités lorsque $sin(\theta)=0$. L'avantage principal de la représentation axe-angle est le sens physique clair. Les inconvénients sont qu'il faut convertir vers des matrices de rotation (ou alternativement des quaternions) pour effectuer des calculs numériques de changement de base et pour combiner des rotations successives.

Vecteur propre de la matrice de rotation

L'axe de rotation $\hat{a}$ est directement lié à la matrice de rotation qui représente la même orientation relative entre deux bases vectorielles. Le vecteur unitaire $\hat{a}$ aligné avec l'axe de rotation a les mêmes composantes dans les deux bases vectorielles :

$$\boldsymbol{a} = \boldsymbol{a}^b = {}^bR^a\boldsymbol{a}^a = {}^aR^b\boldsymbol{a}^b = \boldsymbol{a}^a \tag{3.130}$$

Le vecteur-colonne qui représente **l'axe de rotation est donc un vecteur propre de la matrice de rotation**, et la valeur propre associée est de valeur unitaire car le vecteur-colonne est inchangé par la multiplication avec la matrice de rotation. Il est donc possible de calculer l'axe de rotation associé à une matrice de rotation en calculant les valeurs et vecteurs propre de la matrice.

Convertions

La représentation axe-angle peut être calculée à partir d'une matrice de rotation R de la façon suivante :

$$\theta = cos^{-1}\left[\frac{R_{11}+R_{22}+R_{33}-1}{2}\right] \qquad \boldsymbol{a} = \frac{1}{2\,sin(\theta)}\begin{bmatrix} R_{32}-R_{23} \\ R_{13}-R_{31} \\ R_{21}-R_{12} \end{bmatrix} \tag{3.131}$$

où

$$R = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \tag{3.132}$$

Inversement, la matrice de rotation peut être déterminée à partir de la représentation axe-angle :

$$R = cos(\theta)\,I + (1-cos(\theta))\,\boldsymbol{a}\;\boldsymbol{a}^T - sin(\theta)\boldsymbol{a}^\times \tag{3.133}$$

où

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \qquad \boldsymbol{a}^T = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \qquad \boldsymbol{a}^\times = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix} \tag{3.134}$$

3.7.3 Quaternions

Les quaternions sont une autre façon de représenter une orientation avec 4 paramètres, physiquement moins représentative mais beaucoup plus pratique mathématiquement. Similairement à la représentation axe-angle, un quaternion est constitué d'un scalaire et de 3 composantes vectorielles qui peuvent être groupés dans un vecteur colonne $[\eta,\ e_1,\ e_2,\ e_3]^T$. Les quatre paramètres sont reliés par l'équation suivante $\eta^2+e_1^2+e_2^2+e_3^2=1$. Cette représentation se dérive facilement à partir de la représentation axe-angle.

$$\eta = \cos\left(\frac{\theta}{2}\right) \tag{3.135}$$

$$\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \sin\left(\frac{\theta}{2}\right)\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \tag{3.136}$$

Les grands avantages de cette représentation sont **1)** l'absence de singularité et **2)** des méthodes numériquement efficaces pour calculer des rotations successives et des changements de base directement avec le vecteur colonne $[\eta,\ e_1,\ e_2,\ e_3]^T$.

Détails à venir !

Exemple 3.7 Cinématique d'un robot mobile:

La figure illustre un exemple où un véhicule et les points d'intérêts qu'il trouve doivent être localisés par rapport à une base. La cinématique de ce problème peut se résumer à calculer la matrice de transformation ${}^{A}T^{B}$, qui peut être construite à partir de la définition :

$$ {}^{A}T^{B} = \begin{bmatrix} {}^{a}R^{b} & \boldsymbol{r}^{a}_{B_o/A_o} \\ 0\,0\,0 & 1 \end{bmatrix} = \begin{bmatrix} \hat{a}_1 \cdot \hat{b}_1 & \hat{a}_1 \cdot \hat{b}_2 & \hat{a}_1 \cdot \hat{b}_3 & \hat{a}_1 \cdot \vec{r}_{B_o/A_o} \\ \hat{a}_2 \cdot \hat{b}_1 & \hat{a}_2 \cdot \hat{b}_2 & \hat{a}_2 \cdot \hat{b}_3 & \hat{a}_2 \cdot \vec{r}_{B_o/A_o} \\ \hat{a}_3 \cdot \hat{b}_1 & \hat{a}_3 \cdot \hat{b}_2 & \hat{a}_3 \cdot \hat{b}_3 & \hat{a}_3 \cdot \vec{r}_{B_o/A_o} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.137} $$

Cette matrice représente la pose du véhicule par rapport au repère de la base, et peut aussi être utilisée pour transformer des coordonnées cartésiennes relatives au repère du véhicule en des coordonnées cartésiennes relatives au repère de la base :

$$ \begin{bmatrix} \boldsymbol{r}^{a}_{C/A_o} \\ 1 \end{bmatrix} = {}^{A}T^{B} \begin{bmatrix} \boldsymbol{r}^{b}_{C/B_o} \\ 1 \end{bmatrix} \tag{3.138} $$

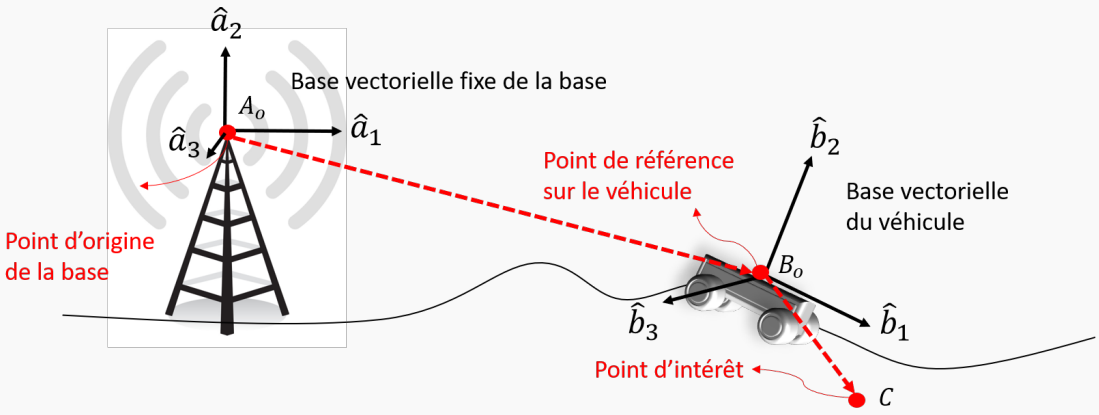


FIGURE 3.27 – Cinématique d'un robot mobile

3.8 Cinématique directe d'un manipulateur

La cinématique directe, c'est le calcul de la fonction de transformation qui permet de passer de l'espace des joints, i.e. le vecteur-colonne $\boldsymbol{q}$, vers l'espace de la tâche $\boldsymbol{r}$.

Définition 3.8 Cinématique directe:

La cinématique directe est la fonction mathématique qui permet les coordonnées dans l'espace opérationnel ($\boldsymbol{r}$), typiquement la position de l'effecteur, en fonction de la configuration articulaire du robot ($\boldsymbol{q}$) :

$$\boldsymbol{r} = f(\boldsymbol{q}) \tag{3.139}$$

Où :
- $\boldsymbol{q} \in \mathbb{R}^n$ est le vecteur des variables articulaires (angles ou distances).
- $\boldsymbol{r} \in \mathbb{R}^m$ est le vecteur décrivant les coordonnées dans l'espace opérationnel.
- f est une fonction dépendant de la géométrie du robot.

3.8.1 Cinématique directe de l'effecteur

Pour les robots manipulateurs, l'espace de la tâche est normalement décrit par la translation et l'orientation de l'effecteur. Si le point de référence de l'effecteur est noté T_o (souvent dans la littérature il est appelé le *TCP* pour *Tool Center Point*), et la base vectorielle associée à l'orientation de l'effecteur est notée t, voir figure 3.30, alors la fonction de cinématique directe pour la translation est :

$$\boldsymbol{r}^a_{T_o/A_o} = f_{Trans}\left(\boldsymbol{q}\right) \tag{3.140}$$

et la fonction de cinématique directe pour l'orientation est :

$${}^aR^t = f_{Orien}\left(\boldsymbol{q}\right) \tag{3.141}$$

où le point A_o et la base a décrive l'origine et la base vectorielle globale qui sont fixes par rapport à la base du robot. Les deux opérations peuvent être combinées en une seule opération qui consiste à calculer la matrice de transformation homogène du repère $\{T_o, \hat{t}_1, \hat{t}_2, \hat{t}_3\}$ par rapport au repère de global $\{A_o, \hat{a}_1, \hat{a}_2, \hat{a}_3\}$

$${}^AT^T = f_{Pose}\left(\boldsymbol{q}\right) \tag{3.142}$$

3.8.2 Chaîne cinématique ouverte

La plupart des robots manipulateurs sont caractérisés par une chaine cinématique ouverte de n joints, comme illustré à la figure 3.28. Pour modéliser cette chaîne cinématique, on associera des points de référence et des bases vectorielles sur chaque lien rigide. La fonction de **cinématique directe pour la translation** se résume à calculer l'addition des $n+1$ vecteurs géométriques de position qui caractérisent la chaîne cinématique, comme illustré à la figure 3.29 :

$$\vec{r}_{T_o/A_o} = \vec{r}_{T_o/H_o} + \ \ldots\ + \vec{r}_{C_o/B_o} + \vec{r}_{B_o/A_o} \tag{3.143}$$

Ce calcul peut être effectué en termes de composantes dans la base vectorielle globale :

$$\boldsymbol{r}^a_{T_o/A_o} = \boldsymbol{r}^a_{T_o/H_o} + \ \ldots\ + \boldsymbol{r}^a_{C_o/B_o} + \boldsymbol{r}^a_{B_o/A_o} \tag{3.144}$$

La fonction de **cinématique directe pour l'orientation** de l'effecteur se résume à multiplier les $n+1$ matrices de rotation qui caractérisent les orientations relatives de deux joints successifs, pour obtenir la matrice de rotation totale qui décrit l'orientation de l'effecteur par rapport à la base globale :

$${}^aR^t = {}^aR^b\,{}^bR^c\ \ldots\ {}^hR^t \tag{3.145}$$

Alternativement pour combiner les opérations, la fonction de **cinématique directe pour la pose** de l'effecteur peut être calculée en multipliant les $n+1$ matrices de transformations homogènes qui caractérisent la pose relative des repères associés à deux joints successifs :

$${}^AT^T = {}^AT^B\,{}^BT^C\ \ldots\ {}^HT^T \tag{3.146}$$

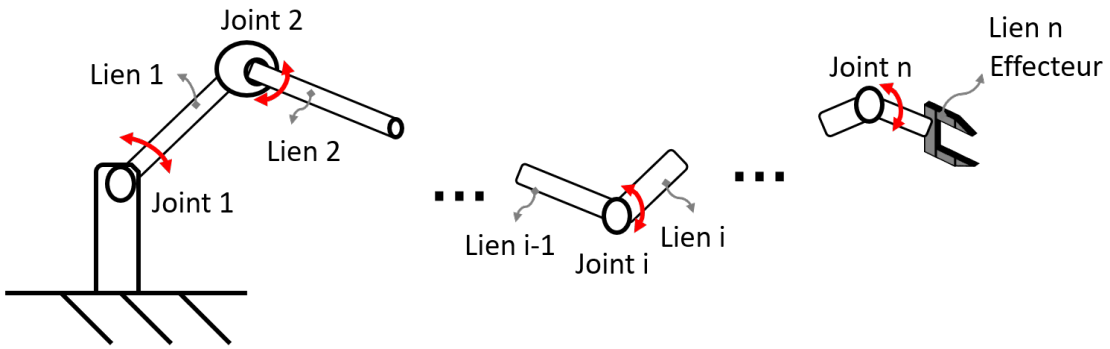


FIGURE 3.28 – Cinématique d'une chaîne ouverte de n joints

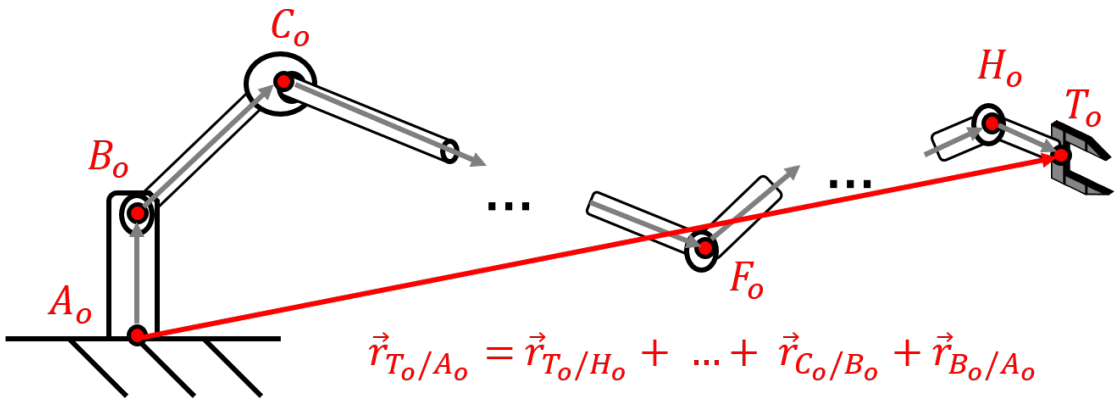


FIGURE 3.29 – Cinématique d'une chaîne ouverte de n joints : addition vectorielle de $n+1$ vecteurs géométriques de position.

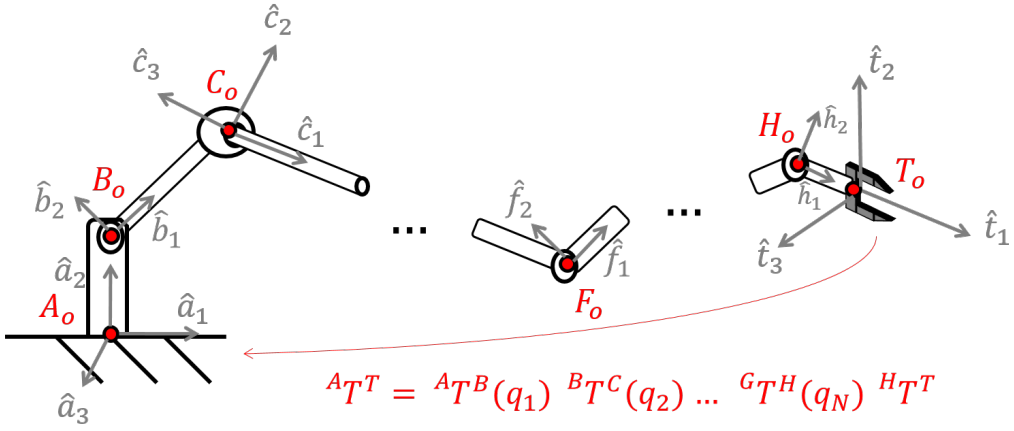


FIGURE 3.30 – Cinématique d'une chaîne ouverte de n joints : succession de $n+1$ transformations homogènes

3.8.3 Simplifications pour les chaînes à 1 DDL par joint

Pour une chaîne cinématique où chaque joint a un seul DDL qui est décrit par la variable q_i, l'équation (3.146) de la cinématique directe avec les coordonnées homogènes a la structure suivante :

$$ {}^{A}T^{T}(q_1, q_2, ..., q_n) = {}^{A}T^{B}(q_1)\,{}^{B}T^{C}(q_2)\,...\,{}^{G}T^{H}(q_n)\,{}^{H}T^{T} \tag{3.147} $$

Chaque matrice de transformation est une fonction d'une seule variable q_i. Notez que ici la dernière matrice de transformation ${}^{H}T^{T}$ est constante car elle décrit la positon de l'effecteur par rapport au dernier joint n. De plus, si un robot est constitué de **joints rotatifs seulement**, comme la plupart des robots manipulateurs industriels, alors l'équation pour l'orientation (3.145) a aussi une structure similaire :

$$ {}^{a}R^{t}(\theta_1, \theta_2, ..., \theta_n) = {}^{a}R^{b}(\theta_1)\,{}^{b}R^{c}(\theta_2)\,...\,{}^{g}R^{h}(\theta_n)\,{}^{h}R^{t} \tag{3.148} $$

où chaque matrice de rotation est une fonction de la variable q_i, qui est un angle θ_i.

3.8.4 Transformations relatives entre les joints à 1 DDL

Pour un joint rotatif, le DDL est une rotation relative selon un axe. Le mouvement peut donc être représenté par une matrice de rotation variable qui décrit l'orientation relative des deux liens rigides reliés par le joint. Pour un joint prismatique, il n'y a aucune rotation relative, seulement une translation selon un axe. La mouvement est donc représenté par un vecteur position de longueur variable. Donc pour représenter le mouvement du joint i d'un robot, qui est décrit par la pose relative entre le repère $\{F_o, \hat{f}_1, \hat{f}_2, \hat{f}_3\}$ et le repère $\{E_o, \hat{e}_1, \hat{e}_2, \hat{e}_3\}$, c'est le vecteur position $\vec{r}_{F_o/E_o}$ qui est une fonction de q_i pour un joint prismatique et la matrice ${}^{e}R^{f}$ qui est une fonction de q_i pour un joint rotatif :

$$ \textit{Joint prismatique} : \begin{cases} \boldsymbol{r}^{e}_{F_o/E_o} = f(q_i) \\ {}^{e}R^{f} = \textit{constante} \end{cases} \qquad \textit{Joint rotatif} : \begin{cases} \boldsymbol{r}^{e}_{F_o/E_o} = \textit{constante} \\ {}^{e}R^{f} = f(q_i) \end{cases} \tag{3.149} $$

de façon équivalente en termes des transformations homogènes :

$$ \textit{Prismatique} : {}^{E}T^{F}(q_i) = \begin{bmatrix} {}^{e}R^{f} & \boldsymbol{r}^{a}_{F_o/E_o}(q_i) \\ 0\;0\;0 & 1 \end{bmatrix} \qquad \textit{Rotatif} : {}^{E}T^{F}(q_i) = \begin{bmatrix} {}^{e}R^{f}(q_i) & \boldsymbol{r}^{a}_{F_o/E_o} \\ 0\;0\;0 & 1 \end{bmatrix} \tag{3.150} $$

3.8.5 Procédure de calcul d'une chaîne cinématique directe

Une procédure pour calculer la cinématique d'un chaîne ouverte arbitraire de n joints et $n+1$ liens rigides, est ici présentée. La notation utilisée est présentée aux figures 3.28, 3.29 et 3.30.

Capsule vidéo
Cinématique directe d'un robot manipulateur
`https://youtu.be/qgUkhZxMTpM`

I : Définition des références La première étape est de définir des coordonnées généralisées (q_1, q_2, q_2, ..., q_m) qui décrive l'état des m DDL des n joints du robot. En générale, pour la plupart des robots manipulateurs, $m = n$ car des joints standards à 1 DDL prismatique ou rotatif sont normalement utilisés. Ensuite, la seconde étape est de définir des points de références (A_o, B_o, C_o, ..., T_o) et des bases vectorielles (a, b, c, ..., t) sur chaque lien rigide.

Astuces :

Pour grandement simplifier le calcul des matrices de rotation, les bases vectorielles locales de deux liens rigides adjacents à un joint rotatif devraient être définies avec un vecteurs unitaire parallèle à l'axe de rotation du joint. Ensuite, l'orientation des deux autres vecteurs unitaires devrait être choisie

de façon à maximiser le nombre de composantes nulles dans les vecteurs positions.

II : Calcul des vecteurs positions locaux Ensuite, basé sur les points et bases vectorielles définies, les $n+1$ vecteurs positions locaux peuvent être calculés par inspection selon la géométrie et les dimensions du robot. Pour un robot qui est constitué de joints rotatifs seulement, toutes les composantes de ces vecteurs positions exprimées dans les bases vectorielles locales sont constantes :

$$\boldsymbol{r}^a_{B_o/A_o} = \begin{bmatrix} l_{1,x} \\ l_{1,y} \\ l_{1,z} \end{bmatrix}, \quad \boldsymbol{r}^b_{C_o/B_o} = \begin{bmatrix} l_{2,x} \\ l_{2,y} \\ l_{3,z} \end{bmatrix}, \quad \ldots \quad \boldsymbol{r}^h_{T_o/H_o} = \begin{bmatrix} l_{n+1,x} \\ l_{n+1,y} \\ l_{n+1,z} \end{bmatrix} \tag{3.151}$$

où les variables l représentent les dimensions des liens rigides. Toutefois, si un robot utilise des joints prismatiques, certains vecteurs positions locaux vont être des fonctions des variables q_i représentants l'état de ces joints prismatiques.

III : Calcul des matrices de rotations relatives Ensuite, les matrices de rotation qui décrive l'orientation relative des bases vectorielles de joints adjacents sont calculées en suivant la démarche décrite à la section 3.5.2 :

$${}^aR^b(q_1) = \begin{bmatrix} \boldsymbol{b}^a_1 & \boldsymbol{b}^a_2 & \boldsymbol{b}^a_3 \end{bmatrix}, \quad {}^bR^c(q_2) = \begin{bmatrix} \boldsymbol{c}^b_1 & \boldsymbol{c}^b_2 & \boldsymbol{c}^b_3 \end{bmatrix}, \quad \ldots \tag{3.152}$$

Pour un robot qui est constitué de joints rotatifs seulement, les matrices de rotation relatives entre deux bases locales vont être des fonctions des variables q_i qui décrivent l'état des joints rotatifs.

IV : Calcul des translations et orientations absolues Premièrement, les matrices de rotation qui décrivent l'orientation des bases vectorielles mobiles par rapport à la base vectorielle fixe a sont calculées en multipliant les matrices de rotation relatives :

$$\begin{aligned}
{}^aR^b &= {}^aR^b && (3.153)\\
{}^aR^c &= {}^aR^b\,{}^bR^c && (3.154)\\
{}^aR^d &= {}^aR^b\,{}^bR^c\,{}^cR^d && (3.155)\\
&\vdots \\
{}^aR^t &= {}^aR^b\,{}^bR^c\,{}^cR^d \ldots {}^hR^t && (3.156)
\end{aligned}$$

où la dernière ligne donne le résultat de cinématique directe pour l'orientation. Ensuite, avec ces matrices, il est possible d'effectuer des changements de base pour exprimer tous les vecteurs de position dans la base fixe a :

$$\begin{aligned}
\boldsymbol{r}^a_{B_o/A_o} &= \boldsymbol{r}^a_{B_o/A_o} && (3.157)\\
\boldsymbol{r}^a_{C_o/B_o} &= {}^aR^b\,\boldsymbol{r}^b_{C_o/B_o} && (3.158)\\
\boldsymbol{r}^a_{D_o/C_o} &= {}^aR^c\,\boldsymbol{r}^c_{D_o/C_o} && (3.159)\\
&\vdots \\
\boldsymbol{r}^a_{T_o/H_o} &= {}^aR^h\,\boldsymbol{r}^h_{T_o/H_o} && (3.160)
\end{aligned}$$

Finalement, avec tous les vecteurs de position relatifs connus dans la base fixe a, il est maintenant possible de calculer la position de tous les points de références par rapport à l'origine A_o

$$\boldsymbol{r}^a_{B_o/A_o} = \boldsymbol{r}^a_{B_o/A_o} \tag{3.161}$$
$$\boldsymbol{r}^a_{C_o/A_o} = \boldsymbol{r}^a_{C_o/B_o} + \boldsymbol{r}^a_{B_o/A_o} \tag{3.162}$$
$$\boldsymbol{r}^a_{D_o/A_o} = \boldsymbol{r}^a_{D_o/C_o} + \boldsymbol{r}^a_{C_o/B_o} + \boldsymbol{r}^a_{B_o/A_o} \tag{3.163}$$
$$\vdots$$
$$\boldsymbol{r}^a_{T_o/A_o} = \boldsymbol{r}^a_{T_o/H_o} + ... + \boldsymbol{r}^a_{C_o/B_o} + \boldsymbol{r}^a_{B_o/A_o} \tag{3.164}$$

où la dernière ligne donne le résultat de cinématique directe pour la translation de l'effecteur. Il est à noter que cette procédure donne aussi la position de tous les points de références (A_o, B_o, C_o, ..., T_o) dans le repère fixe A, ce qui est utile pour tracer le squelette du robot et visualiser la configuration.

IV* : Alternative de calcul avec les transformations homogènes Alternativement, l'étape IV peut être effectuée grâce aux transformations homogènes. Les points de références et bases vectorielles sur chaque lien rigide définissent des repères. Les matrices de transformation relative à deux repères adjacents à un joint peuvent être construites à partir des vecteurs positions locaux et des matrices de rotation relatives :

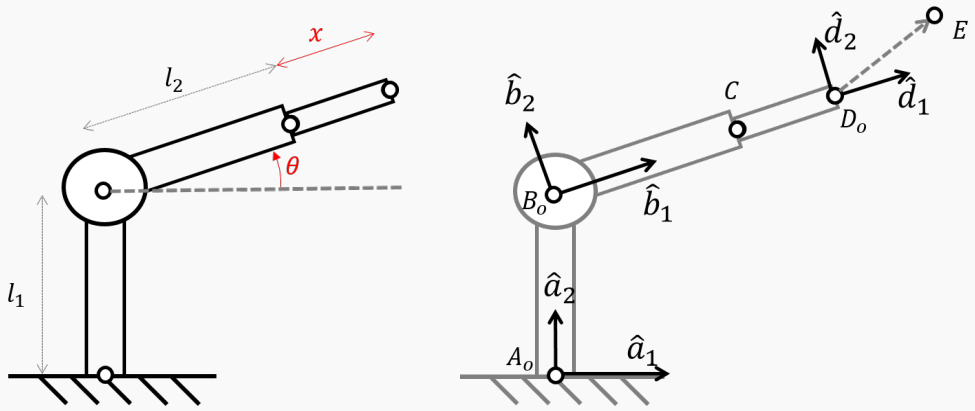
$${}^AT^B(q_1) = \begin{bmatrix} {}^aR^b(q_1) & \boldsymbol{r}^a_{B_o/A_o} \\ 0\,0\,0 & 1 \end{bmatrix}, \quad {}^BT^C(q_2) = \begin{bmatrix} {}^bR^c(q_2) & \boldsymbol{r}^b_{C_o/B_o} \\ 0\,0\,0 & 1 \end{bmatrix}, \quad ... \tag{3.165}$$

Les matrices de transformation globales de tous les repères mobiles vers le repère fixe A peuvent ensuite être calculées par la multiplication des matrices de transformation intermédiaires :

$${}^AT^B = {}^A\,T^B(q_1) \tag{3.166}$$
$${}^AT^C = {}^A\,T^B(q_1)\,{}^BT^C(q_2) \tag{3.167}$$
$$\vdots$$
$${}^AT^T = {}^A\,T^B(q_1)\,{}^BT^C(q_2)\,...\,{}^GT^H(q_n)\,{}^HT^T \tag{3.168}$$

Exemple 3.8 Cinématique pour un robot avec deux joints:

Un exemple est ici présenté de calcul de la cinématique directe pour le robot illustré à la figure 3.31 qui possède un joint rotatif et un joint prismatique. L'objectif est ici de calculer les coordonnées du point D_o dans le repère fixe $\{A_o, \hat{a}_1, \hat{a}_2, \hat{a}_3\}$ comme un fonction de la configuration des joints. Trois méthodes de résolution sont présentées, une méthode minimaliste qui utilise seulement les équations vectorielles, une méthode matricielle qui utilise les vecteur-colonnes et les matrices de rotation, ainsi qu'une méthode utilisant les coordonnées homogènes.
La première étape, commune aux trois méthodes, est de définir des bases vectorielles et des points de référence sur le robot comme illustré à la figure 3.31b. Ici les bases vectorielles a et b ont été définies de sorte que les vecteurs unitaires $\hat{a}_3$ et $\hat{b}_3$ soient parallèles à l'axe de rotation du joint rotatif (axe perpendiculaire au plan de la page). De plus, le vecteur unitaire $\hat{b}_1$ a été défini en ligne avec le deuxième lien rigide et l'axe de translation du joint prismatique. Les points de référence secondaires ont été choisis ainsi : un point B_o au centre du joint rotatif et un point C à la base du joint prismatique. Pour les coordonnées généralisées qui décrivent la configuration du robot, l'angle θ avec la vertical du deuxième lien rigide ainsi que la distance x de translation entre le point C et D_o ont été choisis.



(a) Géométrie (b) Bases vectorielles et points de références

FIGURE 3.31 – Robot à deux joints

Méthode vectorielle Premièrement, avec les points et bases qui ont été définis, il est possible de décrire les vecteurs positions qui représentent la géométrie des liens rigides et ne dépendent pas directement des joints :

$$\text{Éléments constants :} \quad \vec{r}_{B_o/A_o} = l_1\,\hat{a}_2 \quad \text{et} \quad \vec{r}_{C/B_o} = l_2\,\hat{b}_1 \tag{3.169}$$

Ensuite, la direction $\hat{b}_1$ dépend de l'état du joint rotatif donc de l'angle θ, et le vecteur position du point D_o par rapport au point C a une longueur variable égale à x :

$$\text{Éléments variables :} \quad \hat{b}_1(\theta) = c\theta\,\hat{a}_1 + s\theta\,\hat{a}_2 \quad \text{et} \quad \vec{r}_{D/C}(x) = x\,\hat{b}_1 \tag{3.170}$$

Tous les éléments sont maintenant en place pour effectuer la résolution. Le vecteur pertinent pour la résolution est décomposé en une addition de vecteurs connus :

$$\vec{r}_{D_o/A_o} = \vec{r}_{D_o/C} + \vec{r}_{C/B_o} + \vec{r}_{B_o/A_o} \tag{3.171}$$

qui peuvent être manipulés de sorte à garder seulement les vecteurs unitaires $\hat{a}_i$:

$$\vec{r}_{D_o/A_o} = x\,\hat{b}_1 + l_2\,\hat{b}_1 + l_1\,\hat{a}_2 \tag{3.172}$$

$$\vec{r}_{D_o/A_o} = (x + l_2)\,\hat{b}_1 + l_1\,\hat{a}_2 \tag{3.173}$$

$$\vec{r}_{D_o/A_o} = (x + l_2)\,(c\theta\,\hat{a}_1 + s\theta\,\hat{a}_2) + l_1\,\hat{a}_2 \tag{3.174}$$

$$\vec{r}_{D_o/A_o} = \underbrace{[(x + l_2)\,c\theta]}_{r_1^a}\,\hat{a}_1 + \underbrace{[(x + l_2)s\theta + l_1]}_{r_2^a}\,\hat{a}_2 \tag{3.175}$$

Puisque l'équation vectorielle a été réduite aux directions $\hat{a}_i$ seulement, il est possible d'extraire directement les composantes dans la base a :

$$\boldsymbol{r}^a_{D_o/A_o} = \begin{bmatrix} (x + l_2)\,c\theta \\ (x + l_2)s\theta + l_1 \\ 0 \end{bmatrix} \tag{3.176}$$

Méthode matricielle Premièrement, avec les points et bases qui ont été définis, il est possible de décrire des vecteurs de position locaux constants. Les vecteur-colonnes $\boldsymbol{r}^a_{B_o/A_o}$ et $\boldsymbol{r}^b_{C/B_o}$ représentent la géométrie des liens rigides et sont donc constants :

$$\text{Éléments constants :} \quad \boldsymbol{r}^a_{B_o/A_o} = \begin{bmatrix} 0 \\ l_1 \\ 0 \end{bmatrix} \quad \text{et} \quad \boldsymbol{r}^b_{C/B_o} = \begin{bmatrix} l_2 \\ 0 \\ 0 \end{bmatrix} \tag{3.177}$$

Toutefois, la matrice de rotation ${}^{a}R^{b}$ qui décrit l'orientation relative des bases a et b est variable en fonction de l'état du joint rotatif décrit par la variable θ, et le vecteur position $\boldsymbol{r}^{b}_{D_o/C}$ est associé à la translation du joint prismatique et fonction de la variable x :

$$\text{Éléments variables :} \quad {}^{a}R^{b}(\theta) = R_3(\theta) = \begin{bmatrix} c\theta & -s\theta & 0 \\ s\theta & c\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{et} \quad \boldsymbol{r}^{b}_{D_o/C}(x) = \begin{bmatrix} x \\ 0 \\ 0 \end{bmatrix} \tag{3.178}$$

La matrice de rotation peut être construite comme décrit à la section 3.5.2, alternativement il est possible d'identifier directement qu'elle correspond ici à une matrice de rotation élémentaire selon l'axe 3.
L'équation de la cinématique pour calculer la positon de l'effecteur est obtenue à partir de la relation vectorielle :

$$\vec{r}_{D_o/A_o} = \vec{r}_{D_o/C} + \vec{r}_{C/B_o} + \vec{r}_{B_o/A_o} \tag{3.179}$$

que l'on exprime dans la base a :

$$\boldsymbol{r}^{a}_{D_o/A_o} = \boldsymbol{r}^{a}_{D_o/C} + \boldsymbol{r}^{a}_{C/B_o} + \boldsymbol{r}^{a}_{B_o/A_o} \tag{3.180}$$

$$\boldsymbol{r}^{a}_{D_o/A_o} = {}^{a}R^{b}\,\boldsymbol{r}^{b}_{D_o/C} + {}^{a}R^{b}\,\boldsymbol{r}^{b}_{C/B_o} + \boldsymbol{r}^{a}_{B_o/A_o} \tag{3.181}$$

$$\boldsymbol{r}^{a}_{D_o/A_o} = \underbrace{{}^{a}R^{b}(\theta)}_{\text{joint rotatif}} \left(\underbrace{\boldsymbol{r}^{b}_{D_o/C}(x)}_{\text{joint prismatique}} + \boldsymbol{r}^{b}_{C/B_o} \right) + \boldsymbol{r}^{a}_{B_o/A_o} \tag{3.182}$$

Il est ensuite possible de substituer par les composantes connues et résoudre l'équation matricielle :

$$\boldsymbol{r}^{a}_{D_o/A_o} = \begin{bmatrix} c\theta & -s\theta & 0 \\ s\theta & c\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \left(\begin{bmatrix} x \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} l_2 \\ 0 \\ 0 \end{bmatrix} \right) + \begin{bmatrix} 0 \\ l_1 \\ 0 \end{bmatrix} = \begin{bmatrix} (x+l_2)\,c\theta \\ (x+l_2)s\theta + l_1 \\ 0 \end{bmatrix} \tag{3.183}$$

Méthode avec les transformations homogènes Alternativement, une façon plus systématique de procéder, est d'associer un repère à chaque lien rigide et de calculer toutes les matrices de transformation entre ces repères.
La matrice de transformation du repère B vers le repère A implique une matrice de rotation variable selon l'état du joint rotatif, et un vecteur de translation constant :

$${}^{A}T^{B}(\theta) = \begin{bmatrix} {}^{a}R^{b}(\theta) & \boldsymbol{r}^{a}_{B_o/A_o} \\ 0\,0\,0 & 1 \end{bmatrix} = \begin{bmatrix} c\theta & -s\theta & 0 & 0 \\ s\theta & c\theta & 0 & l_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.184}$$

La matrice de transformation du repère D vers le repère B implique seulement un vecteur de translation variable en fonction de l'état du joint prismatique. Puisque que les bases vectorielles d et b sont alignées la matrice de rotation ${}^{b}R^{d}$ est réduite à la matrice identité :

$${}^{B}T^{D}(x) = \begin{bmatrix} {}^{b}R^{d} & \boldsymbol{r}^{b}_{D_o/B_o} \\ 0\,0\,0 & 1 \end{bmatrix} = \begin{bmatrix} I_{3\times 3} & \left(\boldsymbol{r}^{b}_{D_o/C}(x) + \boldsymbol{r}^{b}_{C/B_o}\right) \\ 0 \quad 0 \quad 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & (x+l_2) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.185}$$

Ensuite la pose de l'effecteur peut être décrite par la matrice de transformation du repère D vers le

repère A qui est obtenu en multipliant les deux matrices de transformation locales de chaque joint :

$$ {}^{A}T^{D} = {}^{A}T^{B}(\theta)\,{}^{B}T^{D}(x) \tag{3.186} $$

$$ {}^{A}T^{D} = \begin{bmatrix} c\theta & -s\theta & 0 & 0 \\ s\theta & c\theta & 0 & l_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & (x+l_2) \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.187} $$

$$ {}^{A}T^{D} = \begin{bmatrix} c\theta & -s\theta & 0 & (x+l_2)c\theta \\ s\theta & c\theta & 0 & (x+l_2)s\theta + l_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.188} $$

Avec la dernière colonne de la matrice ${}^{A}T^{D}$, il est possible d'extraire les composantes qui représentent la translation du point D_o dans le repère A :

$$ \boldsymbol{r}^{a}_{D_o/A_o} = \begin{bmatrix} (x+l_2)\,c\theta \\ (x+l_2)s\theta + l_1 \\ 0 \end{bmatrix} \tag{3.189} $$

3.8.6 Les paramètres *Denavit–Hartenberg*

Les paramètres *DH* (*Denavit–Hartenberg*) consistent en quatre paramètres associés à une convention particulière pour attacher des repères aux liens rigides d'un robot manipulateur. Plutôt que de définir arbitrairement les bases vectorielles et les points d'origine des repères, la convention impose une méthodologie standardisée pour définir les repères et les matrices de transformations associées. Plusieurs librairies de cinématique de robot utilise cette convention, qui a comme principal avantage de décrire la cinématique d'un robot avec un nombre de paramètres et repères minimum.

Capsule vidéo
Les paramètres DH
https://youtu.be/LT5Fa1tbDFQ

Convention de positionnement relatif des repères

Avec la convention *DH*, il faut placer les axes 3 (ou z) des repères sur les axe de rotation des joints révolus et sur les axes de translation des joints prismatiques. Ensuite les axes 1 (ou X) d'un joint sont positionnés relativement à la position du joint précédent (parallèle à la normale commune des axes 3), et ensuite les axes 2 sont placés selon la règle de la main droite. L'axe 1 du premier repère est un choix arbitraire. Il y a aussi plusieurs options lorsque les axes 3 de deux joints consécutifs sont parallèles ou coïncident, donc la convention de permet pas de standardisé la représentation de la cinématique d'un à 100%. La Figure 3.32 illustre cette convention de définition des repères d'un robot.

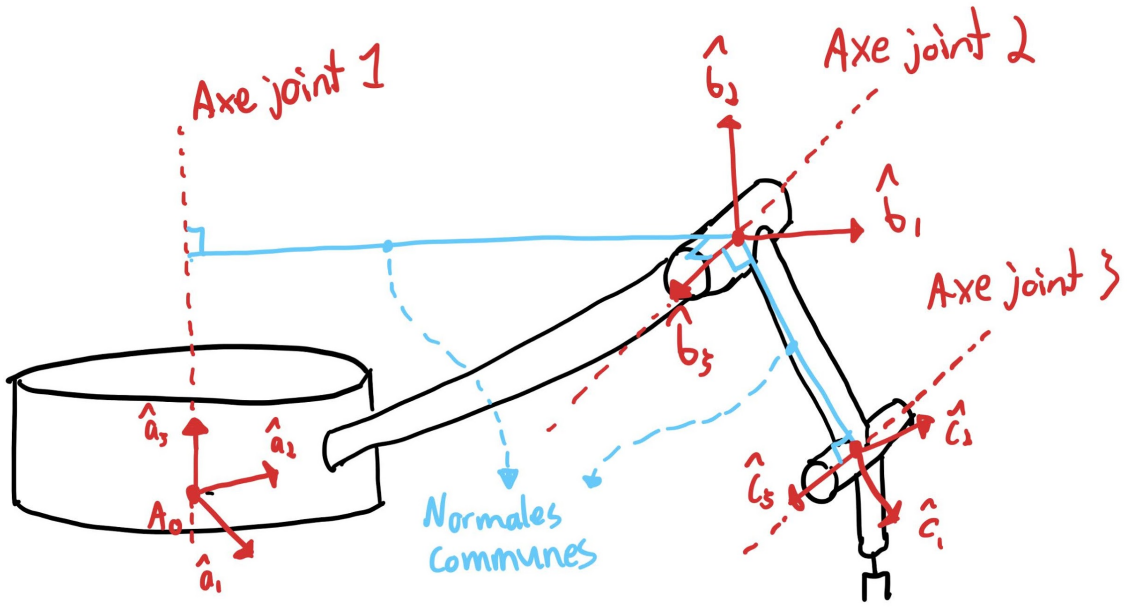


FIGURE 3.32 – Positionnement des repères selon la convention des paramètres *DH*

Repère attachés au lien rigide précédent

Il est à noter que avec cette convention, chaque repère est attaché au lien rigide précédent le joint associé à son axe 3, ce qui diffère de la méthode présentée à la section 3.8.5. Par exemple à la figure 3.32, le repère A est fixe et le repère B bouge seulement en fonction de la rotation du 1er joint autour de l'axe $\hat{a}_3$.

Les quatre paramètres *DH*

Avec la convention de positionnement des repères, seuls quatre scalaires (deux distance et deux angles) sont suffisants pour décrire la pose relative des deux repères sur deux joints adjacents. Les quatre paramètres sont illustrés à la Figure 3.33, pour le premier joint d'un robot ou le repère A est fixe et aligné avec l'axe du joint 1, et le le repère B bouge avec le premier lien rigide et est aligné avec le joint 2. Un repère intermédiaire A' est défini à l'intersection de la normale commune et l'axe 3 du repère A pour bien illustrer le sens des 4 paramètres. Le paramètre d représente la distance entre le point d'origine du premier repère A_o et l'intersection de l'axe $\hat{a}_3$ avec la normale commune (A'_o). Le paramètre θ est l'angle entre le vecteur unitaire $\hat{a}_1$ et la normale commune. Le paramètre r est la longueur de la normale commune. Le paramètre α est l'angle de rotation autour de la normale entre les deux axes des joints $\hat{a}_3$ et $\hat{b}_3$.

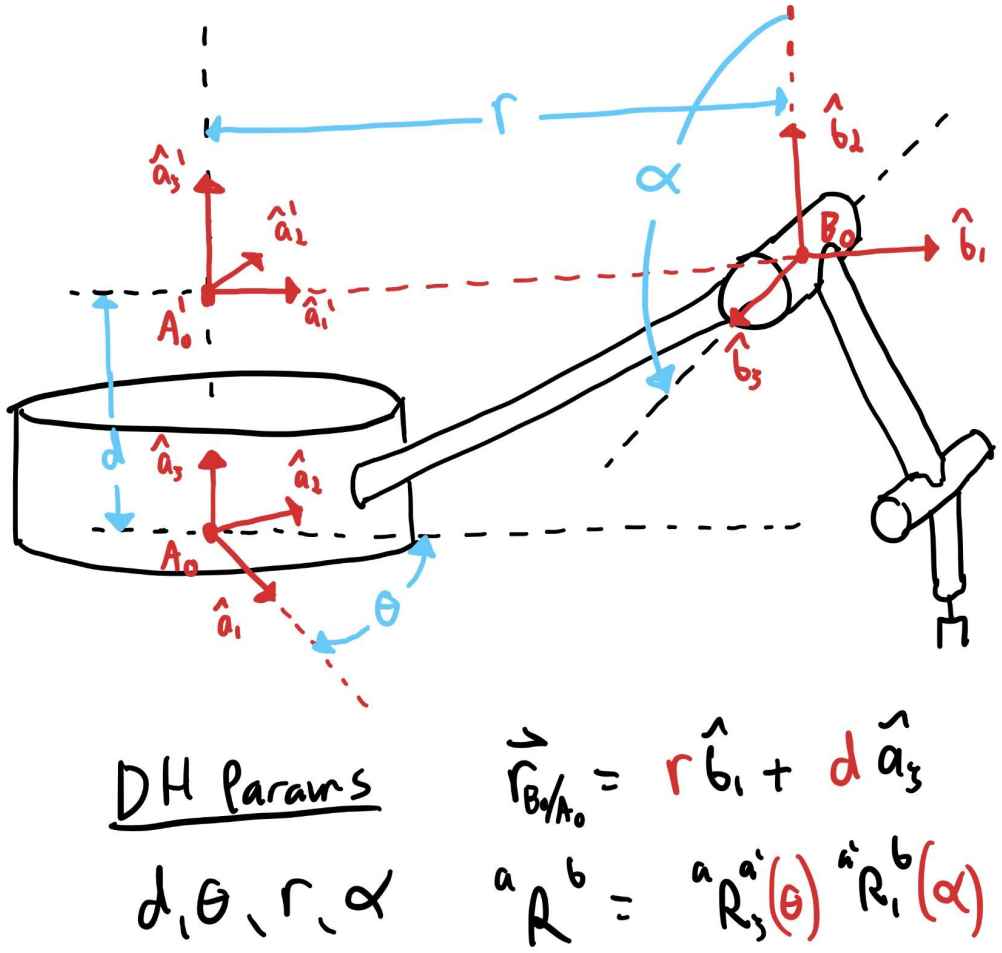


FIGURE 3.33 – Les quatre paramètres *DH*

Consultez une animation vidéo

Les paramètres *DH* sont plus facile à comprendre en visualisant une animation 3D, il est fortement conseillé de consulter un vidéo explicatif pour solidifier vos connaissances.

Pose relative des repères selon les 4 paramètres

La matrice de rotation qui relie la pose des deux repère associés à deux joints successifs peut être paramétrée par les quatre paramètres d, θ, r et α. Le vecteur position qui relie les origines est :

$$\vec{r}_{B_o/A_o} = d\,\hat{a}_3 + r\,\hat{b}_1 \tag{3.190}$$

L'orientation relative peut être exprimée comme la combinaison de deux rotation selon les axes $\hat{a}_3$ et $\hat{b}_1$:

$$^{a}R^{b} = {}^{a}R_3^{a'}(\theta)\,{}^{a'}R_1^{b}(\alpha) \tag{3.191}$$

$$^{a}R_3^{a'}(\theta) = \begin{bmatrix} c\theta & -s\theta & 0 \\ s\theta & c\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.192}$$

$$^{a'}R_1^{b}(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\alpha & -s\alpha \\ 0 & s\alpha & c\alpha \end{bmatrix} \tag{3.193}$$

Les composantes du vecteur position dans la base a sont donc :

$$\boldsymbol{r}^{a}_{B_o/A_o} = \begin{bmatrix} 0 \\ 0 \\ d \end{bmatrix} + {}^{a}R^{a'} \begin{bmatrix} r \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} rc\theta \\ rs\theta \\ d \end{bmatrix} \tag{3.194}$$

La matrice de transformation peut alors être assemblée :

Définition 3.9 Matrice de transformation avec les paramètres DH:

$$^{A}T^{B}(d,\theta,r,\alpha) = \begin{bmatrix} c\theta & -s\theta c\alpha & s\theta s\alpha & rc\theta \\ s\theta & c\theta c\alpha & -c\theta s\alpha & rs\theta \\ 0 & s\alpha & c\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.195}$$

Donc lorsque la convention est utilisée, la matrice de transformation entre deux repères successifs a toujours la forme ci-dessus, et il suffit d'identifier les quatre paramètres DH de chaque joints.

Capsule vidéo
Exemple d'utilisation des paramètres DH
https://youtu.be/cAOjxfkOgaI

Exemple 3.9 Cinématique d'un robot anthropomorphique avec les paramètres *DH*:

La Figure 3.34 illustre l'utilisation de la convention *DH* pour déterminer la matrice de transformation qui représente la pose de l'effecteur.

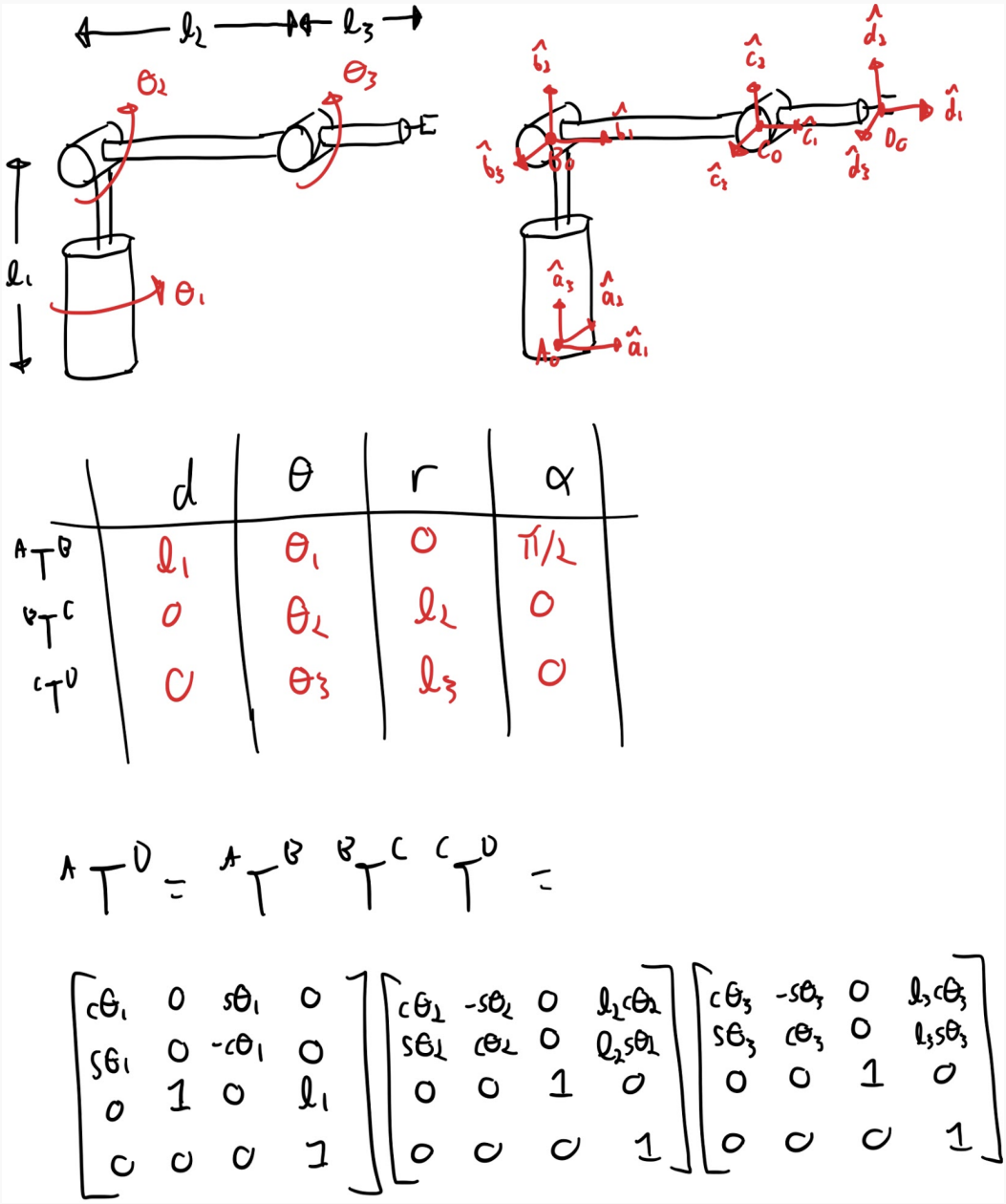


FIGURE 3.34 – Exemple de modélisation de la cinématique d'un robot manipulateur anthropomorphique avec la convention des paramètres *DH*

Joint 1 Premièrement l'axe de rotation du 1er joint doit être repéré, et le repère fixe A doit être placé de sorte à aligner sont vecteur unitaire $\hat{a}_3$ avec cet axe. Les autres variables sont arbitraires, ici on a choisi de position l'origine au sol et l'aligner le vecteur unitaire $\hat{a}_1$ avec les autres liens du robot pour la configuration nominale. Ensuite l'axe de rotation du 2e joint doit être repéré et le repère B doit être fixé de sorte à aligner sont vecteur $\hat{b}_3$. L'orientation de $\hat{b}_1$ doit être alignée avec la normale commune entre $\hat{a}_3$ et $\hat{b}_3$, ici comme les axes se croisent la normale est réduite à un seul point et donc l'orientation de $\hat{b}_1$ est arbitraire, il a donc été choisi de l'aligner avec les liens rigides en position neutre. Avec les repères A et B positionnés selon la convention, il est maintenant possible de déterminer les quatre paramètres de la première matrice de transformation. La distance entre

l'origine A_o et la normale commune (réduite au point B_o) est ici la longueur du premier lien $d = l_1$ et la longueur de la normale commune est nulle $r = 0$. L'angle θ entre les vecteurs unitaires $\hat{a}_1$ et $\hat{b}_1$ est nul pour la position neutre, mais lorsque le joint 1 va tourner l'angle correspond en fait à la position active θ_1. Finalement l'angle entre $\hat{a}_3$ et $\hat{b}_3$ autour de l'axe $\hat{b}_1$ est égale à $\pi/2$ radians. **Joint 2** Pour la prochaine transformation qui caractérise le joint 2, il faut d'abord positionner le repère C selon la convention. Le vecteur unitaire $\hat{c}_3$ est aligné avec l'axe de rotation du 2e joint, et le vecteur unitaire $\hat{c}_1$ est aligné avec la normale commune qui est directement alignée avec $\hat{b}_1$. Donc ici l'orientation de la base c est complètement fixée par la convention. Toutefois comme les axes de rotation du joint 2 et 3 sont parallèle, c'est l'origine C_o qui pourrait être placée n'importe ou sur l'axe de rotation du joint 3 (il y a une infinité de normales communes entre deux axes parallèles). Par simplicité ici on fixe arbitrairement l'origine C_o directement dans l'axe formé par $\hat{b}_1$. Avec le repère C positionné selon la convention il est maintenant possible de déterminer les quatre paramètres *DH*. Le paramètre d est ici nul car le point B_o est coïncident avec la normale commune choisie et le paramètre $r = l_2$ car la normale commune coïncide directement avec le 2e lien rigide. Ensuite, l'angle θ entre les vecteurs unitaires $\hat{b}_1$ et $\hat{c}_1$ est nul pour la position neutre, mais lorsque le joint 1 va tourner l'angle correspond en fait à la position active du 2e joint θ_2. Finalement l'angle entre $\hat{b}_3$ et $\hat{c}_3$ est nul donc $\alpha = 0$. **Joint 3** Pour la dernière transformation, il n'y a pas d'axe de rotation de joint puisque que c'est l'effecteur qui reste à localiser, on choisi ici arbitrairement de garder la même orientation que la base précédente. Avec ce choix, la pose du repère D par rapport au repère C a exactement la même structure que celle du repère C par rapport au repère B. La seule différence est les variables impliquées qui sont θ_3 et l_3. On trouve donc $d = 0$, $\theta = \theta_3$, $r = l_3$ et $\alpha = 0$ comme paramètres *DH* pour le joint 3. Ensuite, il suffit d'utiliser la formule donnée à l'équation (3.231) pour calculer les matrices de transformation entre chaque repère, et de les multiplier ensemble pour calculer la matrice de transformation totale. Avec la méthode des paramètres *DH*, le plus long de la démarche est donc de bien positionner et déterminer les paramètres.

3.8.7 Chaîne cinématique fermée

Les chaînes cinématiques fermées, sont des assemblages mécaniques où les liens et articulations ne sont pas assemblés dans une configuration série pure, mais plutôt en parallèle comme illustré à la figure 3.35.

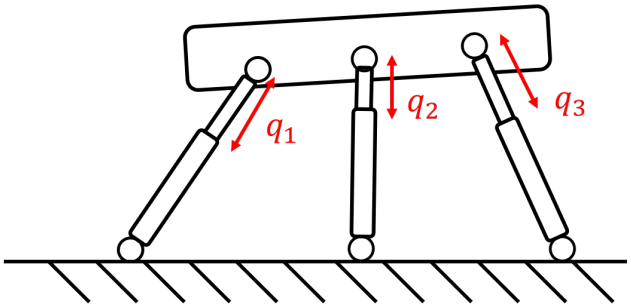


FIGURE 3.35 – Chaine cinématique fermée (robot parallèle)

Le calcul de la cinématique directe de ce type de mécanisme peux être plus compliqué car chaque chemin parallèle qui relie l'effecteur à la base d'un robot rajoute une contrainte au système. Les mouvements des joints ne sont alors pas nécessairement indépendants. Pour une configuration $\boldsymbol{q}$ des joints, il peut ne pas y avoir aucune pose de l'effecteur qui satisfait toutes les contraintes ou potentiellement plusieurs poses de l'effecteur qui satisfont les contraintes.

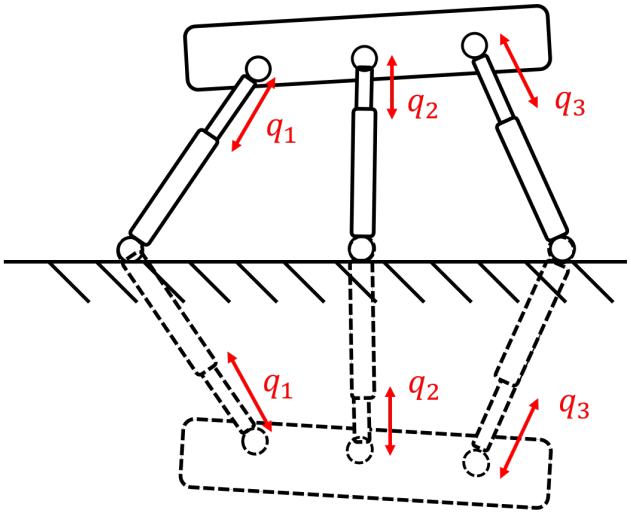


FIGURE 3.36 – Chaîne cinématique fermée avec deux solutions (théoriques) pour la pose de l'effecteur selon une configuration q_1, q_2 et q_3 des joints.

Détails à venir !

3.9 Cinématique inverse d'un manipulateur

La cinématique inverse consiste à calculer des coordonnées de l'espace des joints $\boldsymbol{q}$ qui mènent à des coordonnées désirés dans l'espace opérationnel.

Définition 3.10 Cinématique inverse:

La cinématique inverse est le processus mathématique permettant de déterminer le (ou les) vecteur(s) de configuration articulaire $\boldsymbol{q}$ qui mène à des coordonnées désirés dans l'espace opérationnel $\boldsymbol{r}$, typiquement la position de l'effecteur :

$$\boldsymbol{q} = f^{-1}(\boldsymbol{r}) \tag{3.196}$$

Contrairement à la cinématique directe, ce problème présente trois défis majeurs :

1. **Existence :** La pose $\boldsymbol{r}$ peut être hors de l'espace de travail.
2. **Multiplicité :** Il peut exister plusieurs configurations $\boldsymbol{q}$ pour une même position $\boldsymbol{r}$ (ex : coude en haut vs coude en bas).
3. **Complexité :** Ce problème n'admet pas toujours de solution analytique.

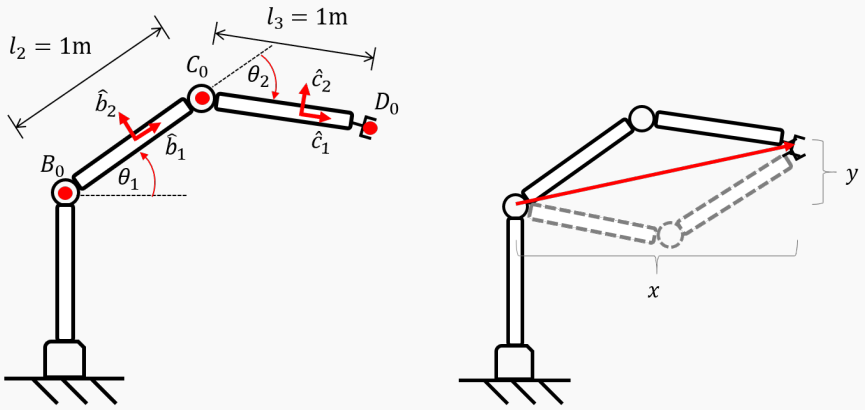
3.9.1 Chaîne cinématique ouverte

Pour les chaînes cinématiques ouvertes, comme la plupart des robots manipulateurs, le calcul de la cinématique directe est sans ambiguïté : une configuration $\boldsymbol{q}$ mène toujours à une seule position de l'effecteur possible. Toutefois, il existe plusieurs configurations $\boldsymbol{q}$ qui peuvent mener à la même position de l'effecteur, voir figure 3.37b par exemple. De plus, pour certaines positions de l'effecteur (ceux qui ne sont pas dans l'espace de travail), l'équation (3.196) n'aura pas de solutions. La cinématique inverse est donc généralement plus compliquée que la cinématique directe, elle consiste à résoudre une fonction hautement non-linéaire qui possède potentiellement aucune ou plusieurs solutions. De plus, même lorsque qu'une ou plusieurs solutions existent, il n'est pas toujours possible de résoudre analytiquement l'équation (3.196). Il est alors nécessaire d'utiliser des méthodes numériques pour trouver des solutions, comme la méthode de Newton-Raphson par exemple. Pour la plupart des architectures de robots industriels, la cinématique inverse peut être résolue analytiquement. Pour les bras manipulateurs à 6 DLL, le critère pour que la cinématique inverse puisse être résolue analytiquement est que l'axe de trois joints rotatifs séquentiels se croise en un point. La plupart des robots industriels ont leurs 3 derniers DDL configurés en un poignet (*wrist* en anglais) qui consiste en trois joints rotatifs avec leurs trois axes qui se croisent en un seul point, et ils rencontrent donc ce critère.

En pratique, les contrôleurs de robot trouvent généralement indirectement les solutions à l'équation (3.196). Une méthode standard est de contrôler les déplacement locaux de l'effecteur en direction de la position $\boldsymbol{r}$ désirée, voir section 8.2, une méthode qui revient à faire une descente du gradient itérative comme avec la méthode de Newton-Raphson mais en temps réel avec le vrai système.

Exemple 3.10 Cinématique inverse d'un robot à deux joints rotatifs:

Cet exemple illustre une résolution analytique de la cinématique inverse d'un robot simple à deux joints rotatifs dans le plan, illustré à la figure 3.37.



(a) Bases et points de références

(b) Deux configurations possible pour une position (x,y) désirée

FIGURE 3.37 – Cinématique inverse d'un robot à deux joints rotatif

L'objectif est ici de calculer la fonction de cinématique inverse :

$$\boldsymbol{q} = f_{Trans}^{-1}\left(\boldsymbol{r}^{a}_{D_o/B_o}\right) \tag{3.197}$$

$$\begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} = f_{Trans}^{-1}\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) \tag{3.198}$$

Premièrement la relation vectorielle est établie :

$$\vec{r}_{D_o/B_o} = \vec{r}_{D_o/C_o} + \vec{r}_{C_o/B_o} \tag{3.199}$$

$$\vec{r}_{D_o/B_o} = l_3\hat{c}_1 + l_2\hat{b}_1 \tag{3.200}$$

Ensuite il est possible de relier la position x et y de l'effecteur avec l'angle θ_2 en calculant la longueur du vecteur $\vec{r}_{D_o/B_o}$:

$$\|\vec{r}_{D_o/B_o}\|^2 = \|\vec{r}_{D_o/C_o} + \vec{r}_{C_o/B_o}\|^2 \tag{3.201}$$

$$x^2 + y^2 = (l_3\hat{c}_1 + l_2\hat{b}_1)\cdot(l_3\hat{c}_1 + l_2\hat{b}_1) \tag{3.202}$$

$$x^2 + y^2 = l_2^2 + l_3^2 + 2l_2l_3(\hat{c}_1\cdot\hat{b}_1) \tag{3.203}$$

$$x^2 + y^2 = l_2^2 + l_3^2 + 2l_2l_3\cos\theta_2 \tag{3.204}$$

À noter que la dérivation ci-dessus est équivalente à utiliser la loi des *cosinus*. Il est donc possible d'obtenir l'angle θ_2 comme une fonctions des coordonnées désirées de l'effecteur et des paramètres géométriques du robot :

$$\cos\theta_2 = \frac{(x^2+y^2) - (l_2^2+l_3^2)}{2l_2l_3} \tag{3.205}$$

L'équation (3.205) peut avoir 0, 1 ou 2 solutions. Si la position désirée (x,y) est hors de l'espace de travail, il n'y a pas de solution :

$$\frac{(x^2+y^2) - (l_2^2+l_3^2)}{2l_2l_3} > 1 \quad \Rightarrow \quad \text{Aucune solution} \tag{3.206}$$

Si la position désirée (x,y) est sur la limite de l'espace de travail, il y a une seule solution où les deux liens rigides sont alignés :

$$\frac{(x^2+y^2) - (l_2^2+l_3^2)}{2l_2l_3} = 1 \quad \Rightarrow \quad \text{Une solution :} \quad \theta_2 = 0 \tag{3.207}$$

En fait, techniquement il faudrait plutôt dire qu'il y a une infinité de solution périodique pour tout les facteurs entiers multiplicateurs de 2π , i.e. $\theta_2 = 2i\,\pi$ avec $i = 0, 1, -1, 2, -2, ...$, si on considère que le 2e joint du robot peut illimité pour sa plage angulaires.
Si la position désirée $(x,\, y)$ est dans l'espace de travail, il y a deux solutions comme illustré à la figure 3.37b :

$$\frac{(x^2+y^2)-(l_2^2+l_3^2)}{2l_2l_3} < 1 \quad \Rightarrow \quad \text{Deux solutions :} \quad \theta_2 = \pm \arccos\left(\frac{(x^2+y^2)-(l_2^2+l_3^2)}{2l_2l_3}\right) \tag{3.208}$$

Il est aussi possible d'isoler l'angle θ_1 sous forme explicite, comme illustré à la figure 3.38 :

$$\theta_1 = \arctan\left(\frac{y}{x}\right) + \arctan\left(\frac{l_2 s\theta_2}{l_1 + l_2 c\theta_2}\right) \tag{3.209}$$

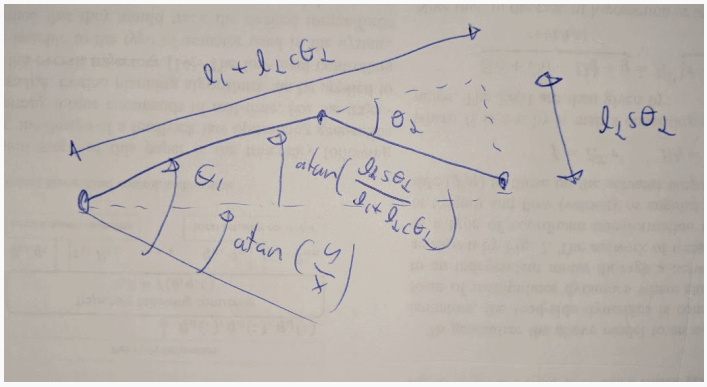


FIGURE 3.38 – Calcul de θ_1

3.9.2 Chaîne cinématique fermée

Contrairement aux chaînes cinématiques ouvertes, la cinématique inverse d'une chaîne cinématique fermée est généralement plus facile à calculer que la cinématique ouverte. Par exemple, comme illustré à la figure 3.39 pour un robot parallèle simple, si la pose de l'effecteur est précisée, alors les variables de joints q_i peuvent être facilement calculés et il y a une seule solution. L'état des joints prismatique peut être calculée sachant la longueurs des vecteurs $\vec{r}_{B_i/A_i}$:

$$q_i + l_i = \|\vec{r}_{B_i/A_i}\| \tag{3.210}$$

et la position des points B_i et A_i peut être déterminée avec la géométrie du robot et la pose de l'effecteur. Les variables q_i pourraient donc être calculées ainsi :

$$q_i = \|\vec{r}_{B_i/A_i}\| - l_i \tag{3.211}$$

$$q_i = \|\vec{r}_{B_i/T_o} + \vec{r}_{T_o/W_o} - \vec{r}_{A_i/W_o}\| - l_i \tag{3.212}$$

$$q_i = \sqrt{\left[\boldsymbol{r}^w_{B_i/T_o} + \boldsymbol{r}^w_{T_o/W_o} - \boldsymbol{r}^w_{A_i/W_o}\right]^T \left[\boldsymbol{r}^w_{B_i/T_o} + \boldsymbol{r}^w_{T_o/W_o} - \boldsymbol{r}^w_{A_i/W_o}\right]} - l_i \tag{3.213}$$

$$q_i = \sqrt{\left[{}^wR^t\, \boldsymbol{r}^t_{B_i/T_o} + \boldsymbol{r}^w_{T_o/W_o} - \boldsymbol{r}^w_{A_i/W_o}\right]^T \left[{}^wR^t\, \boldsymbol{r}^t_{B_i/T_o} + \boldsymbol{r}^w_{T_o/W_o} - \boldsymbol{r}^w_{A_i/W_o}\right]} - l_i \tag{3.214}$$

où ${}^wR^t$ et $\boldsymbol{r}^w_{T_o/W_o}$ sont des fonctions de la pose de l'effecteur et $\boldsymbol{r}^t_{B_i/T_o}$ et $\boldsymbol{r}^w_{A_i/W_o}$ sont des vecteur-colonnes constants et seulement fonction de la géométrie du robot.

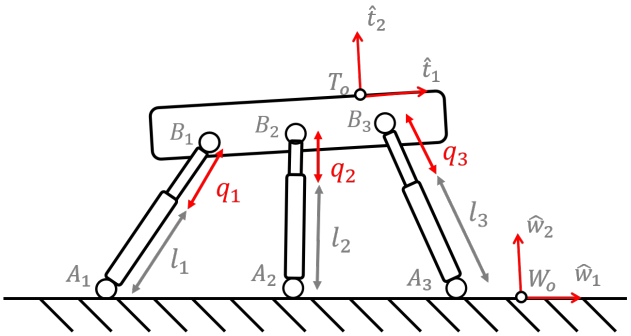


FIGURE 3.39 – Cinématique inverse d'un chaine cinématique fermée (robot parallèle)

3.10 Résumé du chapitre

Cinématique des robots manipulateurs :

$$\underbrace{\boldsymbol{r}}_{\text{Coordonnées dans l'espace de la tâche}} = f(\underbrace{\boldsymbol{q}}_{\text{Coordonnées dans l'espace des joints}}) \tag{3.215}$$

Vecteurs de position :

$$\vec{r}_{B/A} = \quad \text{Vecteur position du point } B \text{ par rapport au point } A \tag{3.216}$$
$$\vec{r}_{Z/A} = \vec{r}_{Z/Y} + \vec{r}_{Y/X} + ... + \vec{r}_{C/B} + \vec{r}_{B/A} \tag{3.217}$$
$$\vec{r}_{B/A} = -\vec{r}_{A/B} \tag{3.218}$$

Bases vectorielle :

$$\text{Base } a = \{\hat{a}_1, \hat{a}_2, \hat{a}_3\} \tag{3.219}$$

Vecteur géométrique vs. vecteur-colonne de composantes :

$$\vec{r} = r_1^a\,\hat{a}_1 + r_2^a\,\hat{a}_2 + r_3^a\,\hat{a}_3 \qquad \boldsymbol{r}^a = \begin{bmatrix} r_1^a \\ r_2^a \\ r_3^a \end{bmatrix} \tag{3.220}$$
$$\vec{r} = \sum_i r_i^a \hat{a}_i \qquad r_i^a = \vec{r} \cdot \hat{a}_i \tag{3.221}$$

Matrice de rotation :

$${}^aR^b = \begin{bmatrix} \hat{a}_1 \cdot \hat{b}_1 & \hat{a}_2 \cdot \hat{b}_1 & \hat{a}_3 \cdot \hat{b}_1 \\ \hat{a}_1 \cdot \hat{b}_2 & \hat{a}_2 \cdot \hat{b}_2 & \hat{a}_3 \cdot \hat{b}_2 \\ \hat{a}_1 \cdot \hat{b}_3 & \hat{a}_2 \cdot \hat{b}_3 & \hat{a}_3 \cdot \hat{b}_3 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \\ \boldsymbol{b}_1^a \\ \\ \end{bmatrix} & \begin{bmatrix} \\ \boldsymbol{b}_2^a \\ \\ \end{bmatrix} & \begin{bmatrix} \\ \boldsymbol{b}_3^a \\ \\ \end{bmatrix} \end{bmatrix} \tag{3.222}$$

Propriétés des matrices de rotation :

$$\boldsymbol{r}^a = {}^aR^b\,\boldsymbol{r}^b \tag{3.223}$$
$${}^aR^c = {}^aR^b\,{}^bR^c \tag{3.224}$$
$${}^aR^b_{ij} = \hat{a}_i \cdot \hat{b}_j \tag{3.225}$$
$$R(\theta)^{-1} = R(-\theta) \tag{3.226}$$
$$R^{-1} = R^T \tag{3.227}$$
$$({}^bR^a)^{-1} = {}^aR^b \tag{3.228}$$

Repère :

$$\text{Repère } A = \text{Base } a + \text{Origine } A_o = \{A_o, \hat{a}_1, \hat{a}_2, \hat{a}_3\} \tag{3.229}$$

Matrice de transformation :

$${}^AT^B = \begin{bmatrix} {}^aR^b & \boldsymbol{r}^a_{B_o/A_o} \\ 0\;0\;0 & 1 \end{bmatrix} \tag{3.230}$$

Transformation avec les paramètres DH :

$$T(d,\theta,r,\alpha) = \begin{bmatrix} c\theta & -s\theta c\alpha & s\theta s\alpha & rc\theta \\ s\theta & c\theta c\alpha & -c\theta s\alpha & rs\theta \\ 0 & s\alpha & c\alpha & d \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.231}$$

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Chapitre 4

Cinématique des robots manipulateurs II : Le mouvement

Dans le chapitre 3, les méthodes pour modéliser la position, l'orientation ont été présentée. Ce chapitre présente les méthodes associés pour représenter leurs variation dans le temps, c'est-à-dire la vitesse. Les premières sections décrivent les principes généraux et ensuite des méthodes spécifiques pour les robots manipulateurs sont présentées.

4.1 Référentiel

Un référentiel est un solide ou un ensemble de points par rapport auquel un observateur qui mesure un mouvement est fixe. Une position ou un mouvement ne peut être défini par rapport au vide et il n'y a pas de référentiel absolu, le choix du référentiel est donc un choix arbitraire de point de vue. Il est à noter que **contrairement au choix d'un repère, le choix du référentiel influence "la physique" du problème**, par exemple l'énergie cinétique d'un objet n'est pas la même selon le référentiel alors que cette quantité est indépendante du système de coordonnées. Un référentiel dit inertiel ou Galiléen (vitesse et orientation constante) est préférable si on souhaite éventuellement appliquer les équations de Newtons, car des facteurs correctifs (ex. : la force centrifuge) doivent être ajoutés aux équations dynamiques si un référentiel non-inertiel est utilisé. La notion de référentiel est critique pour le calcul de vitesses et accélérations, mais on peut en faire abstraction pour les problèmes de cinématique et statique qui n'impliquent pas la notion d'évolution dans le temps.

4.1.1 Référentiel vs. repère

Pour décrire le mouvement d'un objet, les concepts de référentiel, origine et base vectorielle sont important à distinguer. En quelques mots, un **référentiel** est un point de vue utilisé pour décrire un phénomène, une **origine** c'est un point par rapport auquel les mesures de position sont faites et une **base** vectorielle représente l'orientation du système d'axe utilisé. Ces trois éléments peuvent être choisi indépendamment. Ensuite, un **repère** c'est la combinaison d'une base et d'une origine. Par exemple, pour décrire la position d'un bateau, typiquement le référentiel utilisé serait la Terre, l'origine des mesures de position pourrait être le port le plus près, et la base pour exprimer cette position serait le système d'axe nord-sud/est-ouest. Il est important de noter que tous ces choix sont indépendants et les confondre est une source d'erreur courante en cinématique. Le reste de cette section discute de ce qui les distingue.



Capsule vidéo
Référentiel vs. repère vs. base vectorielle
`https://youtu.be/Uv_MbbfkiS4`

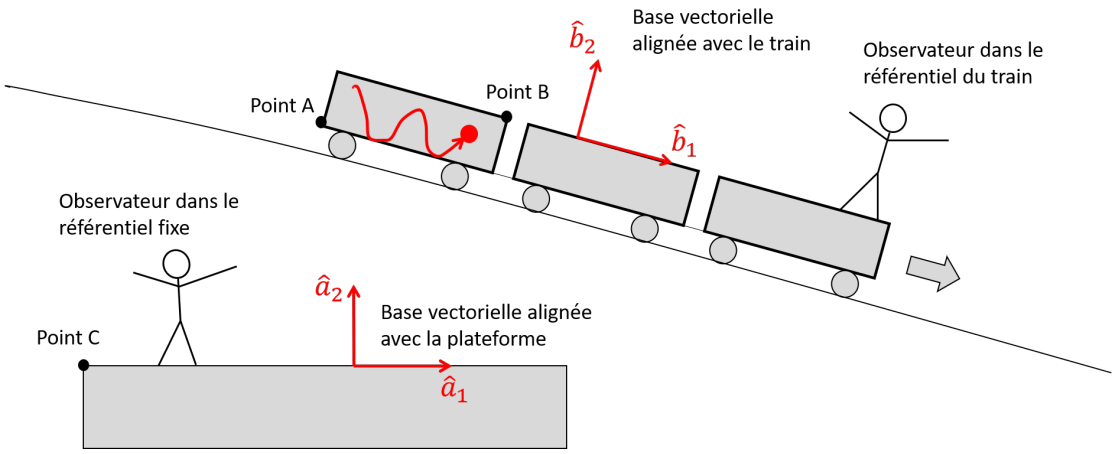
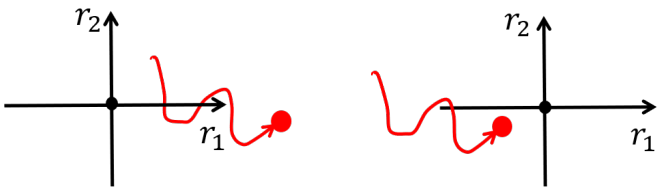


FIGURE 4.1 – Exemple de référentiels, points d'origine et bases vectorielles

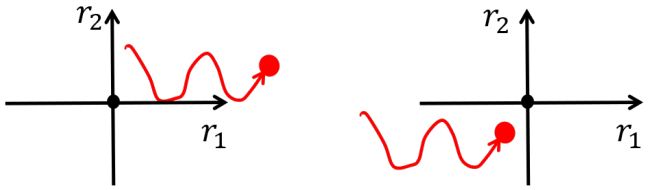
La figure 4.1 illustre ces concepts par un exemple où un train descend une pente à vitesse constante et un ballon rebondit sur le sol du troisième wagon. Deux bases vectorielles sont définies, une alignée avec l'horizontale et une alignée avec le train. Plusieurs points d'origine pour les mesures de position sont aussi disponibles. Finalement, deux référentiels sont définis, un référentiel attaché à la Terre (fixe) et un référentiel qui est attaché au train. Les coordonnées d'un vecteur-colonne qui décris la position du ballon :

$$\boldsymbol{r}_{Ball} = \begin{bmatrix} r_1 \\ r_2 \end{bmatrix} \tag{4.1}$$

sont illustrés pour différents choix d'origine et de base vectorielle, lorsque que le référentiel fixe est utilisé (figure 4.2) et lorsque que le référentiel du train est utilisé (figure 4.3). Il est à noter que les points d'origine sont ici fixés à la Terre lorsque le référentiel fixe est utilisé, et fixés au train lorsque que le référentiel du train est utilisé. Comme illustré pour cet exemple, **un changement d'origine est analogue à une translation** de la trajectoire, **un changement de base vectorielle est analogue à une rotation** de la trajectoire, et **un changement de référentiel est ici analogue à une dilatation** de la trajectoire dans la direction $\hat{b}_1$, qui correspond à la direction de la vitesse du train.

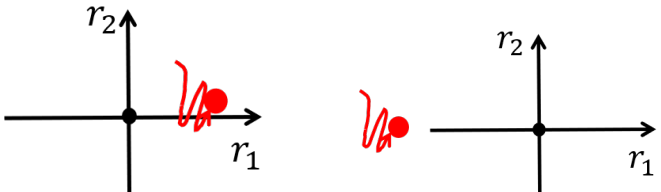


(a) Repère $\{A, \hat{a}_1, \hat{a}_2, \hat{a}_3\}$ (b) Repère $\{B, \hat{a}_1, \hat{a}_2, \hat{a}_3\}$

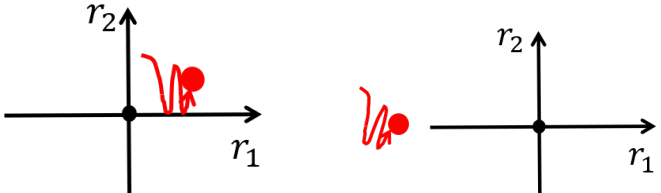


(c) Repère $\{A, \hat{b}_1, \hat{b}_2, \hat{b}_3\}$ (d) Repère $\{B, \hat{b}_1, \hat{b}_2, \hat{b}_3\}$

FIGURE 4.2 – Trajectoire du ballon avec un repère attaché au **référentiel fixe**



(a) Repère $\{A, \hat{a}_1, \hat{a}_2, \hat{a}_3\}$ (b) Repère $\{B, \hat{a}_1, \hat{a}_2, \hat{a}_3\}$



(c) Repère $\{A, \hat{b}_1, \hat{b}_2, \hat{b}_3\}$ (d) Repère $\{B, \hat{b}_1, \hat{b}_2, \hat{b}_3\}$

FIGURE 4.3 – Trajectoire du ballon avec un repère attaché au **référentiel du train**

4.2 Vitesse et dérivée d'un vecteur position

Un vecteur de vitesse est définit comme la dérivée temporelle d'une vecteur de position :

$$\vec{v}_{B/A} = \frac{d}{dt}\vec{r}_{B/A} \tag{4.2}$$

Toutefois, pour avoir la notion de vitesse précise qui correspond à un quantité de mouvement (momentum) qui peut être utilisée dans les divers lois de la physique (ex. conservation du mouvement, conservation de l'énergie, etc.), la dérivée doit être faite par rapport à un observateur Galiléen, aussi appelé un référentiel inertiel. Donc pour avoir la vitesse $\vec{v}_B$ d'un point B il faut 1) que la dérivée soit faite par rapport à une base vectorielle avec un orientation fixe dans le référentiel Galiléen, et 2) que le vecteur position dérivé soit par rapport à un point d'origine fixe dans le référentiel Galiléen.

$$\underbrace{\vec{v}_B}_{\text{Vitesse du point B}} = \underbrace{\frac{d}{dt}}_{\text{observateur Galiléen}} \vec{r}_{B/\underbrace{A}_{\text{Point fixe}}} \tag{4.3}$$

Avec cette définition, le vecteur vitesse est défini seulement par un seul point contrairement au vecteur position qui prend un sens seulement avec une origine. La méthode la plus simple pour déterminer le vecteur vitesse d'un point basé sur une expression connue pour son vecteur position est d'exprimer le vecteur position dans un repère fixe, et ensuite de dériver chaque composante. Donc si il est possible d'exprimer le vecteur position d'un point B par rapport à un point fixe A et en utilisant une base vectorielle fixe a :

$$\vec{r}_{B/A} = x\hat{a}_1 + y\hat{a}_2 + z\hat{a}_3 \tag{4.4}$$

il suffit de dériver l'expression de chaque composante dans cette base pour déterminer le vecteur vitesse du point B :

$$\vec{v}_B = \frac{d}{dt}\vec{r}_{B/A} = \dot{x}\hat{a}_1 + \dot{y}\hat{a}_2 + \dot{z}\hat{a}_3 \tag{4.5}$$

4.2.1 Dérivée d'un vecteur position exprimé avec une base mobile

Notation pour les repère fixes et mobiles

Pour les sections suivantes, les expressions sont développé en considérante que les repères notés A (base a et origine A) sont fixes dans le référentiel inertiel et que les repères notés B (base b et origine B) sont arbitraires donc potentiellement mobiles.

Dans certaines situations, le vecteur position est plus naturellement exprimé dans un repère mobile, cette section développe des expressions qui peuvent être utilisée pour déterminer le vecteur vitesse sans avoir à tout transférer dans un repère inertiel. Si on connaît les composantes un vecteur position dans une base mobile b :

$$\vec{r}_{B/A} = x\hat{b}_1 + y\hat{b}_2 + z\hat{b}_3 \tag{4.6}$$

Lorsqu'on dérive par rapport au temps, il faut tenir compte que les vecteurs unitaires de la base b on une direction qui change dans le temps, et donc une dérivée temporelle non-nulle :

$$\vec{v}_B = \frac{d}{dt}\vec{r}_{B/A} = \underbrace{\dot{x}\hat{b}_1 + \dot{y}\hat{b}_2 + \dot{z}\hat{b}_3}_{{}^b\frac{d}{dt}\vec{r}_{B/A}} + \underbrace{x\dot{\hat{b}}_1 + y\dot{\hat{b}}_2 + z\dot{\hat{b}}_3}_{\vec{w}_{b/a}\times\vec{r}_{B/A}} \tag{4.7}$$

Il y a donc deux sources de contribution au vecteur vitesse, le taux de variation des composantes et le taux de variation de la direction des vecteurs unitaires. La dérivée temporelle des vecteurs unitaires est directement reliée à la vitesse angulaire de la base b (voir section 4.3, et cette contribution peut être calculée avec le produit vectoriel d'un vecteur de vitesse angulaire de la base b par rapport à une base fixe a fois le vecteur de position :

$$\vec{v}_B = \underbrace{{}^b\frac{d}{dt}\vec{r}_{B/A}}_{\text{Observateur mobile}} + \underbrace{\vec{w}_{b/a}\times\vec{r}_{B/A}}_{\text{Effet de la vitesse angulaire}} \tag{4.8}$$

où b est une base mobile, a est une base fixe dans le référentiel inertiel, le terme ${}^{b}\frac{d}{dt}$ représente le taux de variation vu par un observateur dans le référentiel non-inertiel qui tourne avec la base b et $\vec{w}_{b/a}$ est la vitesse angulaire de la base b p/r à une base a inertielle.

Équation matricielle équivalente avec les composantes :

On peut trouver une relation équivalente qui implique les vecteur-colonnes de composantes et les matrices de rotation. Le vecteur vitesse est la dérivée temporelle des composantes dans la base fixe a, on peut donc substituer par le vecteur-colonne de composantes dans la base b fois la matrice de rotation ${}^{a}R^{b}$ et prendre la dérivée :

$$\boldsymbol{v}^{a}_{B} = \frac{d}{dt}\boldsymbol{r}^{a}_{B/A} = \frac{d}{dt}\left({}^{a}R^{b}\,\boldsymbol{r}^{b}_{B/A}\right) \tag{4.9}$$

Ici on applique la règle de dérivé d'un produit et comme avec la relation vectorielle de l'équation (4.7), on trouve deux termes :

$$\boldsymbol{v}^{a}_{B/A} = {}^{a}R^{b}\,\dot{\boldsymbol{r}}^{b}_{B/A} + {}^{a}\dot{R}^{b}\,\boldsymbol{r}^{b}_{B/A} \tag{4.10}$$

Toutefois ici le taux de variation de la direction de la base mobile b est représenté par la dérivée temporelle de la matrice de rotation ${}^{a}R^{b}$ (voir section 4.3). On peut utiliser une expression qui utilise les composantes de la vitesse angulaire de la base b pour déterminer la dérivée de la matrice de rotation

$$\boldsymbol{v}^{a}_{B} = {}^{a}R^{b}\,\dot{\boldsymbol{r}}^{b}_{B/A} + (\boldsymbol{w}^{a}_{b/a})^{\times}\,{}^{a}R^{b}\,\boldsymbol{r}^{b}_{B/A} \tag{4.11}$$

$$\boldsymbol{v}^{a}_{B} = {}^{a}R^{b}\,\dot{\boldsymbol{r}}^{b}_{B/A} + {}^{a}R^{b}\,(\boldsymbol{w}^{b}_{b/a})^{\times}\,\boldsymbol{r}^{b}_{B/A} \tag{4.12}$$

pour obtenir une forme équivalente à l'expression vectorielle précédemment obtenue (eq. (4.8), mais impliquant les vecteurs-colonnes de composantes et la matrice de rotation :

$$\vec{v}_{B} = {}^{b}\frac{d}{dt}\vec{r}_{B/A} + \vec{w}_{b/a} \times \vec{r}_{B/A} \quad \Leftrightarrow \quad \boldsymbol{v}^{a}_{B} = {}^{a}R^{b}\left[\dot{\boldsymbol{r}}^{b}_{B/A} + (\boldsymbol{w}^{b}_{b/a})^{\times}\boldsymbol{r}^{b}_{B/A}\right] \tag{4.13}$$

4.3 Vitesse angulaire et dérivée d'une matrice de rotation

La représentation de l'orientation a plusieurs difficultés mathématiques : pour représenter seulement 3 DDL les matrices de rotation ont 9 paramètres, les rotation ne sont pas commutative, etc. Toutefois, les petites rotations infinitésimales peuvent être représentées par un vecteur de trois paramètres et sont commutatives. Un vecteur rotation infinitésimale a comme direction l'axe de rotation et comme amplitude un angle autour de cet axe. Lorsque qu'on divise un vecteur rotation infinitésimal par dt ont obtient le concept de vecteur de vitesse angulaire :

$$\vec{w} = \underbrace{w\left[\frac{rad}{sec}\right]}_{\text{Taux de variation d'un angle}} \underbrace{\hat{a}}_{\text{axe de rotation instantané}} \tag{4.14}$$

Le vecteur de vitesse angulaire $\vec{w}$ a comme direction l'axe de rotation instantané $\hat{a}$ et comme amplitude un taux de variation w d'un angle autour de cet axe. D'un point de vue matricielle, le vecteur de vitesse angulaire est associé à la dérivée temporelle d'une matrice de rotation :

$$\dot{R} = \frac{d}{dt}R = (\boldsymbol{w})^x R \qquad \text{avec} \quad (\boldsymbol{w})^x = \begin{bmatrix} 0 & -w_3 & w_2 \\ w_3 & 0 & -w_1 \\ -w_2 & w_1 & 0 \end{bmatrix} \tag{4.15}$$

et du point de vue vectorielle, de façon équivalente à la dérivée temporelle des vecteurs unitaires de la base mobile peut être calculé avec un produit vectoriel du vecteur de vitesse angulaire et les vecteurs unitaires :

$$\dot{\hat{b}}_i = \vec{w} \times \hat{b}_i \tag{4.16}$$

Lorsqu'on utilise la représentation des angles de Euler, le vecteur de vitesse angulaire est définie directement avec la dérivée temporelle des trois angles de Euler et leur axe de rotation respectif :

$$\vec{w}_{d/a} = \dot{\phi}\hat{a}_3 + \dot{\theta}\hat{b}_1 + \dot{\psi}\hat{c}_3 \tag{4.17}$$

voir la définition des angles et axes à la section 3.7.1. Comme la base vectorielle de l'équation vectorielle précédente n'est pas orthogonale, il faut faire attention car :

$$\begin{bmatrix} \dot{w}_1 \\ \dot{w}_2 \\ \dot{w}_3 \end{bmatrix} \neq \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{4.18}$$

Il faut convertir le vecteur $\vec{w}_{d/a}$ dans la base approprié pour déterminer ses composantes.

4.3.1 Propriétés

Premièrement, une vitesse angulaire est le taux de variation d'une orientation relative entre deux bases. Tout comme les matrices de rotation pour être précis il faut donc noter les bases associés pour bien définir un vecteur de vitesse angulaire :

$$\vec{w}_{b/a} \tag{4.19}$$

est le taux de variation de l'orientation de la base b par rapport à la base a. Selon si le vecteur $\vec{w}_{b/a}$ est exprimé dans la base a ou b deux expressions permettre de calculer la dérivée temporelle de la matrice de rotation associée :

$${}^a\dot{R}^b = (\boldsymbol{w}^a_{b/a})^\times \, {}^aR^b = {}^aR^b \, (\boldsymbol{w}^b_{b/a})^\times \tag{4.20}$$

Ensuite, il est à noter lors du calcul des dérivées temporelles des vecteurs unitaires, que la drivée obtenue est selon un observateur dans la base d'origine pour laquelle le vecteur de vitesse angulaire est calculé :

$${}^a\frac{d}{dt}\hat{b}_i = \vec{w}_{b/a} \times \hat{b}_i \tag{4.21}$$

donc pour obtenir une dérivée temporelle de $\hat{b}_i$ dans un référentiel inertiel, la base a du vecteur de vitesse angulaire doit être inertielle.

Addition Une propriété utile pour calculer une vitesse angulaire totale, i.e. par rapport au référentiel inertiel, lorsqu'on a plusieurs vitesse angulaire relative (par exemple pour chacun des joints d'un robot), est qu'il suffit d'additionner les vecteur de vitesse angulaire.

$$\vec{w}_{d/a} = \vec{w}_{d/a} + \vec{w}_{c/b} + \vec{w}_{b/a} \tag{4.22}$$

Commutativité Contrairement aux rotations, la combinaison de vecteurs de vitesses angulaires est commutative, i.e. l'ordre n'a pas d'importance :

$$^{a}R^{b}\,{}^{b}R^{c} \neq {}^{b}R^{c}\,{}^{a}R^{b} \tag{4.23}$$

$$\vec{w}_{c/b} + \vec{w}_{b/a} = \vec{w}_{b/a} + \vec{w}_{c/b} \tag{4.24}$$

4.4 Accélération et dérivée seconde d'un vecteur position

$$\vec{a}_B = {}^b\frac{d}{dt}\vec{v}_B + \vec{w}_{b/a} \times \vec{v}_B \quad \Leftrightarrow \quad \boldsymbol{a}_B^a = {}^aR^b\left[\dot{\boldsymbol{v}}_B^b + (\boldsymbol{w}_{b/a}^b)^\times \boldsymbol{v}_B^b\right] \tag{4.25}$$

$$\vec{a}_B = {}^b\frac{d^2}{dt^2}\vec{r}_{B/A} + {}^b\frac{d}{dt}\vec{w}_{b/a} \times \vec{r}_{B/A} + \underbrace{2\vec{w}_{b/a} \times {}^b\frac{d^2}{dt^2}\vec{r}_{B/A}}_{\text{Coriolis}} + \underbrace{\vec{w}_{b/a} \times \vec{w}_{b/a} \times \vec{r}_{B/A}}_{\text{Centrifuge}} \tag{4.26}$$

$$\boldsymbol{a}_B^a = {}^aR^b\left[\ddot{\boldsymbol{r}}_{B/A}^b + (\dot{\boldsymbol{w}}_{b/a}^b)^\times \boldsymbol{r}_{B/A}^b + \underbrace{2(\boldsymbol{w}_{b/a}^b)^\times \dot{\boldsymbol{r}}_{B/A}^b}_{\text{Coriolis}} + \underbrace{(\boldsymbol{w}_{b/a}^b)^\times (\boldsymbol{w}_{b/a}^b)^\times \boldsymbol{r}_{B/A}^b}_{\text{Centrifuge}}\right] \tag{4.27}$$

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

4.5 Cinématique différentielle des robots manipulateurs

La cinématique différentielle c'est l'étude de la relation entre le mouvement des joints et le mouvement associé de l'effecteur. Dans le chapitre 3, les fonctions reliant la position de l'effecteur $\boldsymbol{r}$ et la configuration du robot dans l'espace des joints $\boldsymbol{q}$ ont été étudiées :

$$\text{Cinématique directe : } \quad \boldsymbol{r} = f(\boldsymbol{q}) \quad \text{inverse : } \quad \boldsymbol{q} = f^{-1}(\boldsymbol{r}) \tag{4.28}$$

Plusieurs méthodes en robotique sont plutôt basées sur les fonctions qui relient la vitesse de l'effecteur $\dot{\boldsymbol{r}}$ à la vitesse des joints $\dot{\boldsymbol{q}}$. Ces fonctions prennent la forme suivante :

Définition 4.1 Cinématique différentielle:

La cinématique différentielle est les relations mathématiques reliant le taux de variation des variables dans l'espace articulaire et opérationnel, typiquement des vitesses angulaires de joints et la vitesse de l'effecteur :

$$\text{Cinématique différentielle directe : } \quad \dot{\boldsymbol{r}} = J(\boldsymbol{q})\,\dot{\boldsymbol{q}} \quad \text{inverse : } \quad \dot{\boldsymbol{q}} = J^{-1}(\boldsymbol{q})\,\dot{\boldsymbol{r}} \tag{4.29}$$

Les fonctions de cinématique différentielle impliquent une matrice Jacobienne :

Définition 4.2 Matrice Jacobienne:

La matrice Jacobienne $\boldsymbol{J}(\boldsymbol{q})$ est une matrice de dimensions $m \times n$ qui relit des déplacements infinitésimales des joints aux déplacements infinitésimales dans l'espace opérationnel :

$$\text{Matrice Jacobienne : } \quad J(\boldsymbol{q}) = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{q}} = \begin{bmatrix} \frac{\partial r_1}{\partial q_1} & \cdots & \frac{\partial r_1}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_m}{\partial q_1} & \cdots & \frac{\partial r_m}{\partial q_n} \end{bmatrix}_{m \times n} \tag{4.30}$$

Autrement dit, chaque élément de la matrice correspond à la sensibilité d'une coordonnée opérationnelle par rapport à une articulation :

$$J_{ij}(\boldsymbol{q}) = \frac{\partial r_i(\boldsymbol{q})}{\partial q_j} \tag{4.31}$$

Plus de détails sur la dérivation dans un contexte multi-variable sont disponibles à la section 17.5.

Capsule vidéo
Cinématique différentielle et matrice jacobienne
https://youtu.be/jJqYWNSsJvE

La composante J_{ij} de la matrice correspond à la variation de la coordonnée r_i de l'effecteur, due à un déplacement unitaire du joint q_j. La relation entre les déplacements infinitésimales est données par :

$$\begin{bmatrix} d\boldsymbol{r} \end{bmatrix}_{m\times 1} = \begin{bmatrix} J(\boldsymbol{q}) \end{bmatrix}_{m\times n} \begin{bmatrix} d\boldsymbol{q} \end{bmatrix}_{n\times 1} \quad \Leftrightarrow \quad dr_j = \sum_i^n J_{ji} dq_i \qquad j \in \{1, ..., m\} \tag{4.32}$$

La relation qui relit les vitesses est relié à la relation entre les déplacements infinitésimales par le principe des dérivées en chaînes :

$$\text{Dérivées en chaînes : } \underbrace{\frac{d\boldsymbol{r}}{dt}}_{\dot{\boldsymbol{r}}} = \underbrace{\left(\frac{\partial \boldsymbol{r}}{\partial \boldsymbol{q}}\right)}_{J(\boldsymbol{q})} \underbrace{\frac{d\boldsymbol{q}}{dt}}_{\dot{\boldsymbol{q}}} \tag{4.33}$$

autrement dit, il suffit de diviser l'équation (4.32) par dt pour obtenir la relation entres les vitesses.

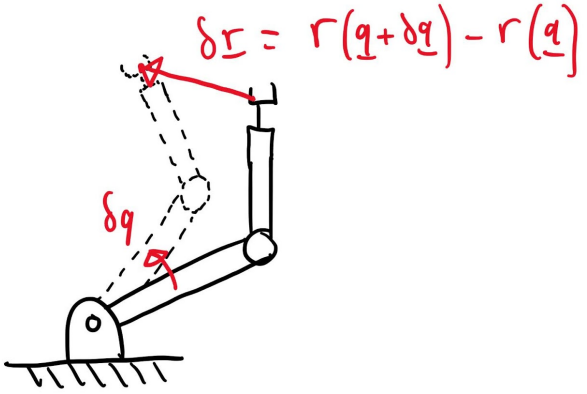


FIGURE 4.4 – Le Jacobien décrit la relation entre des déplacements infinitésimal

4.5.1 La matrice Jacobienne

Unités Typiquement pour les analyse de cinématique différentielle, les variables de sortie sont les DDL de translation de l'effecteur, la pose complète de l'effecteur ou bien des variables qui décrivent les DDL de l'espace de la tâche ; et les variables d'entrée sont les vitesse angulaire ou linéaire des joints du robot. Les unités des composantes J_{ij} de la matrice Jacobienne associée sont donc typiquement soit des unités de distance qui correspondent à des *bras de levier*, ou bien des ratios adimensionnels.

Non-linéarité Il est important de noter que généralement la matrice Jacobienne est une fonction de la configuration $\boldsymbol{q}$ du robot. Comme la fonction de cinématique directe est généralement hautement non-linéaire, l'effet d'un déplacement infinitésimal d'un joint sur l'effecteur dépend considérablement de la configuration initiale. La matrice Jacobienne est donc seulement valide pour décrire des petites variation déplacement autour de la configuration pour laquelle elle a été évaluée, ou bien pour la relation entre les vitesses momentanément à cette configuration.

Colonnes de la matrice Jacobienne Comme illustré à la Figure 4.5, dans le contexte de cinématique différentiel, la colonne j de la matrice Jacobienne correspond à un vecteur déplacement infinitésimal de l'effecteur due au mouvement du joint j :

$$J = \left[\frac{\partial \boldsymbol{r}}{\partial q_1} \cdots \frac{\partial \boldsymbol{r}}{\partial q_i} \cdots \frac{\partial \boldsymbol{r}}{\partial q_n} \right] \tag{4.34}$$

Chaque colonne a donc une interprétation vectorielle visuelle comme illustré à la Figure suivante.

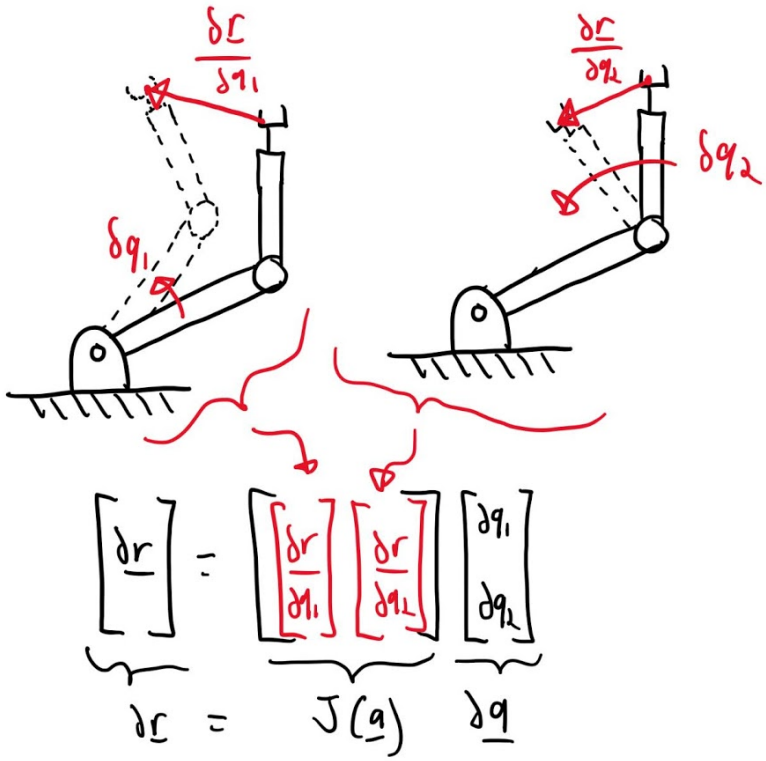


FIGURE 4.5 – Illustration vectorielle du Jacobian du manipulateur (relation de translation)

Produits vectoriels Lorsque l'espace de la tâche associé à un Jacobien correspond à la position cartésienne de l'effecteur d'un robot, chaque colonne du Jacobien peut aussi être déterminée basé sur un calcul vectoriel avec l'axe de rotation du joint associé. Si on considère d'abord un joint prismatique i, l'effet infinitésimal de son déplacement est une translation d'amplitude ∂q_i dans une direction qui correspond à l'axe de translation du joint $\hat{z}_i$. Pour un joint rotatif i, l'effet est plutôt une rotation des membrures en aval du joint i autour de l'axe de rotation du joint $\hat{z}_i$. Si on utile un repère mobile qui tourne avec toute les membrures en aval du joint i, les composantes de vecteurs position en aval ne varie pas, c'est la vitesse angulaire de la base vectorielle mobile qui explique la vitesse. Donc comme vue à la section 4.2.1, cet effet peut être représenté mathématiquement par un produit vectoriel. Le mouvement à l'effecteur (résultant de la rotation d'un axe seulement) peut donc ici être calculé par le produit vectoriel du vecteur de vitesse angulaire (l'axe de rotation fois le taux de variation de l'angle de rotation) avec le vecteur position entre l'effecteur et le centre de rotation du joint. En termes de déplacements infinitésimal de l'effecteur dus aux déplacements de joints, la relation est

$$\partial \vec{r} = \begin{cases} (\partial q_i)\ \hat{z}_i & \text{pour un joint } i \text{ prismatique} \\ (\partial q_i)\ \hat{z}_i \times \vec{r}_{T_o/Z_i} & \text{pour un joint } i \text{ rotatif} \end{cases} \tag{4.35}$$

où q_i est la position du joint i, $\hat{z}_i$ est l'axe de rotation ou de translation du joint, $\vec{r}$ est le vecteur position de l'effecteur par rapport à une origine fixe et $\vec{r}_{T_o/Z_i}$ est le vecteur position de l'effecteur par rapport à un point d'origine Z_i sur l'axe de rotation du joint i (centre de rotation de ce mouvement). Cette relation vectorielle différentielle peut être utilisée pour obtenir une définition équivalente des colonnes du Jacobien :

$$\text{Colonnes du Jacobien :} \quad \frac{\partial \boldsymbol{r}}{\partial q_i} = \begin{cases} \boldsymbol{z}_i & \text{pour un joint } i \text{ prismatique} \\ \boldsymbol{z}_i^{\times}\, \boldsymbol{r}_{T_o/Z_i} & \text{pour un joint } i \text{ rotatif} \end{cases} \tag{4.36}$$

ou toutes les composantes sont exprimée dans la base vectorielle fixe globale.

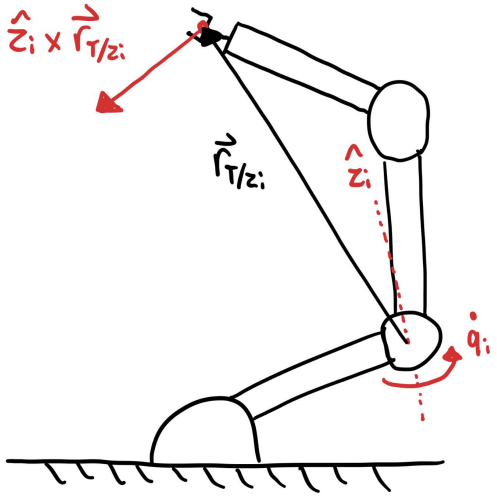


FIGURE 4.6 – Interprétation des colonnes du Jacobian comme un poduit vectoriel

Relation avec les paramètres *DH*

La définition des colonnes du Jacobien donnée à l'équation (4.36) est particulièrement utile pour calculer le Jacobien d'un robot manipulateur lorsqu'on a déjà déterminé les paramètres *DH*. La convention de positionnement des repères fait que les axes de rotation $\boldsymbol{z}_i$ et les vecteurs positions qui donnent la position des points Z_i sont déjà calculés dans les matrices de transformation homogènes. Dans ce cas, cette technique peut être plus rapide que de calculer les dérivées partielles basée sur la définition donnée à l'équation (4.30) pour construire la matrice Jacobienne.

Cinématique différentielle pour l'orientation de l'effecteur Lorsque ce n'est pas la position cartésienne de l'effecteur que l'on désire contrôler mais plutôt sont orientation, le vecteur sortie $\dot{\boldsymbol{r}}$ contient alors les composantes du vecteur de vitesse angulaire de l'effecteur qui définie le taux de variation de l'orientation d'une base vectorielle t fixée sur l'effecteur par rapport à une base fixe a par rapport à la base du robot manipulateur. L'équation de cinématique différentiel est alors caractérisé par l'addition des vecteurs de vitesse angulaire relative pour chacun des joints d'un système, basée sur l'équation (4.22), pour un robot à 6 joints :

$$\vec{w}_{t/w} = \vec{w}_{t/f} + \vec{w}_{f/e} + \vec{w}_{e/d} + \vec{w}_{d/c} + \vec{w}_{c/b} + \vec{w}_{b/a} \tag{4.37}$$

où les base b, c, d, e et f sont des bases vectorielles intermédiaires sur les joints (voire Figure 3.30). Donc le cas commun où le robot manipulateur a seulement des joints rotatifs à un degrée de liberté, chacun vecteur de vitesse angulaire est simplement l'axe de rotation $\hat{z}_i$ fois de taux de variation de l'angle :

$$\vec{w}_{t/w} = \hat{z}_6\dot{q}_6 + \hat{z}_5\dot{q}_5 + \hat{z}_4\dot{q}_4 + \hat{z}_3\dot{q}_3 + \hat{z}_2\dot{q}_2 + \hat{z}_1\dot{q}_1 \tag{4.38}$$

$$\boldsymbol{w} = \underbrace{\begin{bmatrix}\boldsymbol{z}_1 & \boldsymbol{z}_2 & \boldsymbol{z}_3 & \boldsymbol{z}_4 & \boldsymbol{z}_5 & \boldsymbol{z}_6\end{bmatrix}}_{J_{ori}} \dot{\boldsymbol{q}} \tag{4.39}$$

Dans ce cas, les colonnes du Jacobien sont simplement les axes de rotation de chacun des joints. Pour des joints prismatiques, ils n'influencent pas l'orientation de l'effecteur et les colonnes du Jacobien associées à un joint prismatique sont donc nulles :

$$\text{Colonnes du Jacobien (orientation)} : \begin{cases} \boldsymbol{0} & \text{pour un joint } i \text{ prismatique} \\ \boldsymbol{z}_i & \text{pour un joint } i \text{ rotatif} \end{cases} \tag{4.40}$$

Un exemple où on s'intéresserait plutôt à l'orientation de système robotique, serait pour asservir une antenne de communication qui doit toujours être alignée avec un satellite. Dans ce cas, l'effecteur du système est la parabole de l'antenne et c'est sont orientation spatiale qui est importante seulement.

Cinématique différentielle pour la pose de l'effecteur Si on désire décrire de taux de variation de la pose complète de l'effecteur, il suffit de combiner les systèmes linéaire la translation et l'orientation :

$$\dot{\boldsymbol{r}} = [J_{trans}]\ \dot{\boldsymbol{q}} \qquad J_{trans} = \left[\ldots \boldsymbol{z}_i^{\times}\, \boldsymbol{r}_{T_o/Z_i} \ldots\right] \tag{4.41}$$

$$\boldsymbol{w} = [J_{ori}]\ \dot{\boldsymbol{q}} \qquad J_{ori} = [\ldots \boldsymbol{z}_i \ldots] \tag{4.42}$$

$$\begin{bmatrix} \dot{\boldsymbol{r}} \\ \boldsymbol{w} \end{bmatrix} = [J_{pose}]\ \dot{\boldsymbol{q}} \qquad J_{pose} = \begin{bmatrix} J_{trans} \\ J_{ori} \end{bmatrix} \tag{4.43}$$

Cinématique différentielle pour un espace de la tâche Il est aussi possible de définir un matrice Jacobienne pour une relation spécifique à une tâche. Par exemple, voir Figure 3.4, où il serait désirable d'obtenir les équations qui relit la position/vitesse des joints et celle de la tâche décrite en termes de x et y sur un plan particulier. Si on a un espace de la tâche de dimension m (normalement égale ou inférieur au nombre de DDL de l'effecteur du robot), alors on peut déterminer matrice Jacobienne m par n.

$$\left[\boldsymbol{r}_e\right]_{m\times 1} = f_e(\boldsymbol{q}) \quad \Rightarrow \quad \left[\dot{\boldsymbol{r}}_{\boldsymbol{e}}\right]_{m\times 1} = \left[J_e\right]_{m\times n}\left[\dot{\boldsymbol{q}}\right]_{n\times 1} \tag{4.44}$$

Note de lecture

Pour les sections suivantes, s'il n'est pas préciser autrement on va toujours travailler avec la relation cinématique de translation seulement de l'effecteur pour l'explication des concepts. Les exemples sont visuellement plus clair dans cette situation. Toutefois, tout les concepts présentés se généralisent à toutes les situations.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Capsule vidéo
Exemple de calcul de la matrice jacobienne
https://youtu.be/qgqDscXSsMM

Exemple 4.1 Cinématique différentielle d'un robot cartésien:

La Figure 4.7 illustre un robot planaire constitué de deux actionneurs linéaires basé sur des vis à billes qui sont assemblés à 90 degrées.

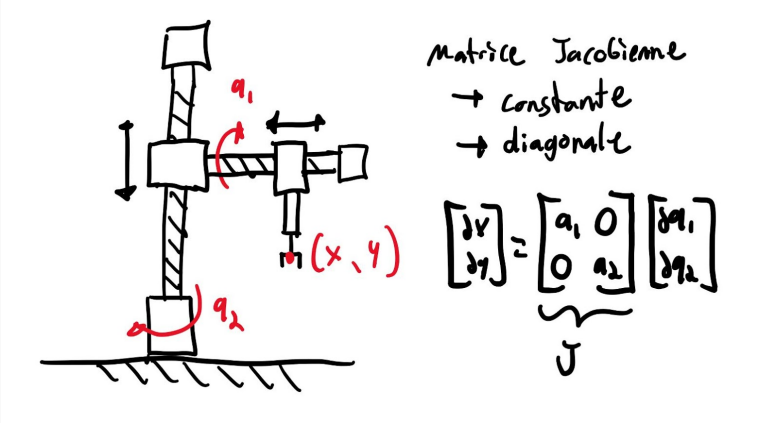


FIGURE 4.7 – Jacobien pour un robot cartésien

Pour ce robot, chaque actionneur linéaire contrôle de façon indépendante la translation x et y de l'effecteur, la fonction de cinématique directe est donnée par :

$$\boldsymbol{r} = f(\boldsymbol{q}) \tag{4.45}$$

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_1 q_1 \\ a_2 q_2 \end{bmatrix} \tag{4.46}$$

ou les variables a_i sont des constantes qui relient la rotation angulaire des vis aux déplacements linéaires. La relation différentielle peut alors être calculée :

$$\underbrace{\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}}_{\dot{\boldsymbol{r}}} = \underbrace{\begin{bmatrix} \frac{\partial x}{\partial q_1} & \frac{\partial x}{\partial q_2} \\ \frac{\partial y}{\partial q_1} & \frac{\partial y}{\partial q_2} \end{bmatrix}}_{J} \underbrace{\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}}_{\dot{\boldsymbol{q}}} \tag{4.47}$$

$$\underbrace{\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}}_{\dot{\boldsymbol{r}}} = \underbrace{\begin{bmatrix} a_1 & 0 \\ 0 & a_2 \end{bmatrix}}_{J} \underbrace{\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}}_{\dot{\boldsymbol{q}}} \tag{4.48}$$

(4.49)

Il est a noter ici que : **1)** la matrice Jacobienne est diagonale car chaque joint influence indépendamment un seul DDL de l'effecteur. **2)** la matrice Jacobienne est constante et indépendante de $\boldsymbol{q}$ car la relation de cinématique directe, voir équation (4.46), est linéaire.

Exemple 4.2 Cinématique différentielle d'un robot à deux joints:

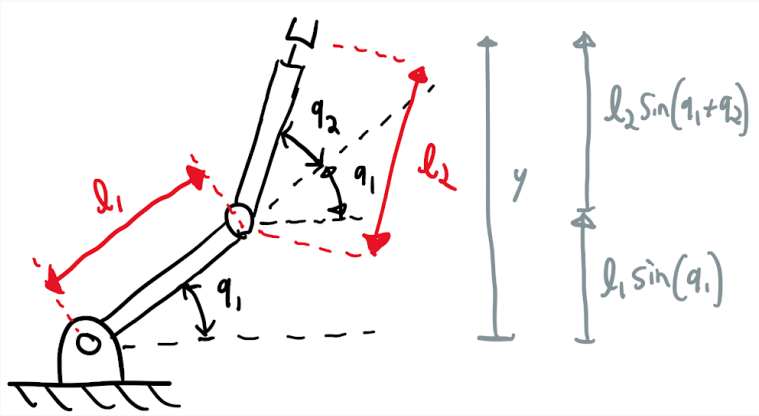


FIGURE 4.8 – Robot à deux joints

Pour un robot manipulateur planaire à deux DDL, comme illustré à la Figure 4.8, la fonction de cinématique directe est données par :

$$\boldsymbol{r} = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} l_1 c_1 + l_2 c_{12} \\ l_1 s_1 + l_2 s_{12} \end{bmatrix} \tag{4.50}$$

avec

$$c_1 = \cos(q_1) \tag{4.51}$$
$$s_1 = \sin(q_1) \tag{4.52}$$
$$c_{12} = \cos(q_1 + q_2) \tag{4.53}$$
$$s_{12} = \sin(q_1 + q_2) \tag{4.54}$$

On peut alors trouver la relation différentielle en dérivant par rapport au temps :

$$\dot{\boldsymbol{r}} = \frac{d\boldsymbol{r}}{dt} = \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} -l_1 s_1 \dot{q}_1 - l_2 s_{12}(\dot{q}_1 + \dot{q}_2) \\ l_1 c_1 \dot{q}_1 + l_2 c_{12}(\dot{q}_1 + \dot{q}_2) \end{bmatrix} \tag{4.55}$$

$$\dot{\boldsymbol{r}} = \underbrace{\begin{bmatrix} -l_1 s_1 - l_2 s_{12} & -l_2 s_{12} \\ l_1 c_1 + l_2 c_{12} & l_2 c_{12} \end{bmatrix}}_{J(\boldsymbol{q})} \underbrace{\begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}}_{\dot{\boldsymbol{q}}} \tag{4.56}$$

$$\dot{\boldsymbol{r}} = J(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{4.57}$$

Exemple 4.3 Cinématique différentielle d'un robot à trois joints:

Les Figures 84 et 4.10 illustre la cinématique différentielle d'un robot à trois joints dans une configuration qui permet de bien visualiser indépendamment les effets du mouvement de chaque joint.

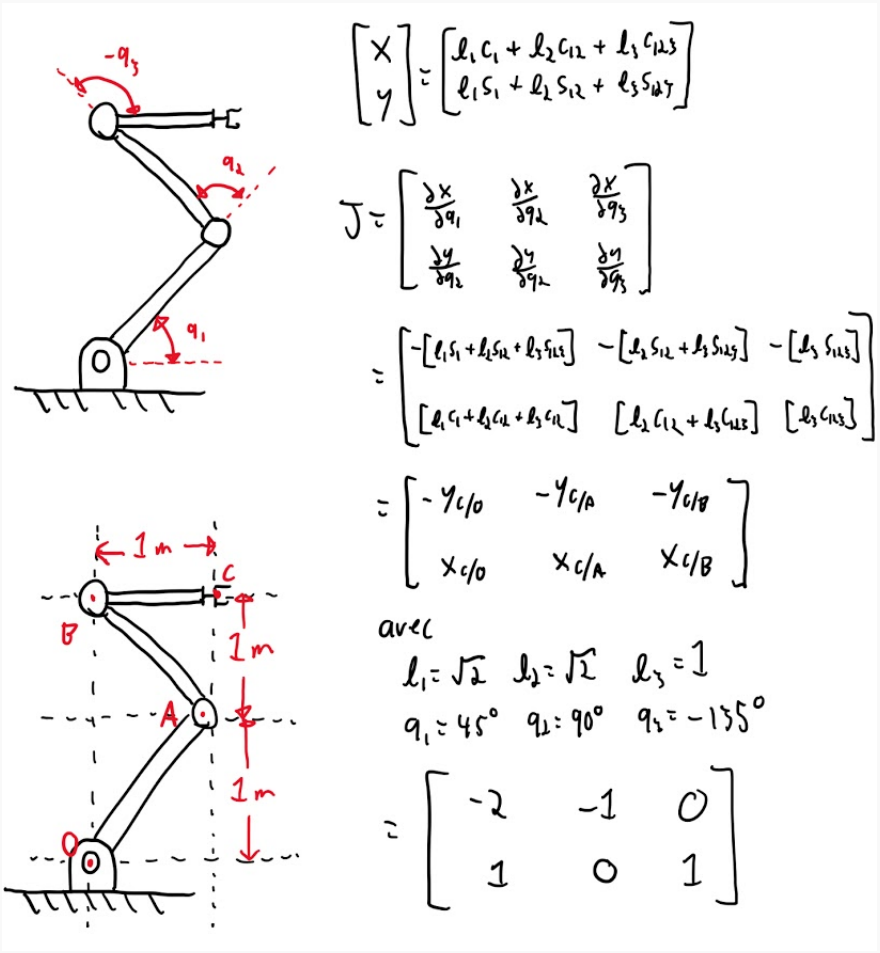


FIGURE 4.9 – Jacobien d'un robot à trois joints - partie a)

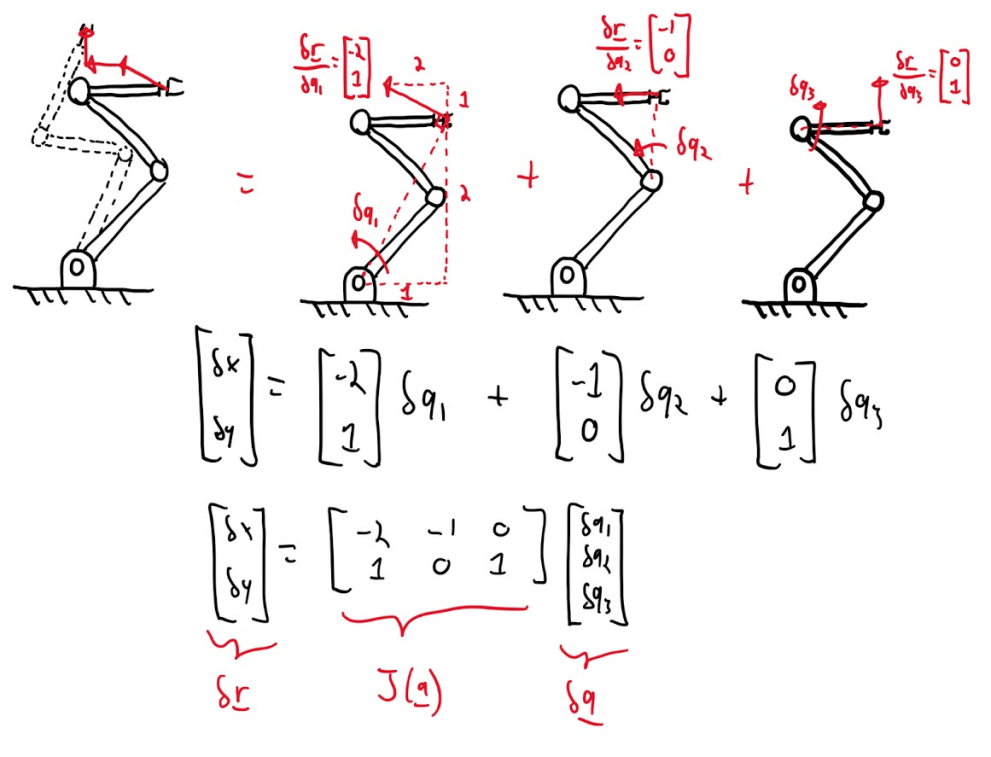


FIGURE 4.10 – Jacobien d'un robot à trois joints - partie b)

4.6 Cinématique différentielle inverse

Lorsque le problème de cinématique différentielle est inversé, c'est-à-dire lorsqu'on cherche un vecteur de vitesse aux joints $\dot{\boldsymbol{q}}$ qui va produire une vitesse de l'effecteur spécifiée $\dot{\boldsymbol{r}}$, il faut inverser la relation mathématique.

Définition 4.3 Problème de cinématique différentielle inverse:

Le problème de la cinématique différentielle inverse consiste à déterminer le vecteur des vitesses articulaires $\dot{\boldsymbol{q}}$ nécessaire pour produire une vitesse désirée dans l'espace opérationnel $\dot{\boldsymbol{r}}$, en inversant la relation linéaire définie par la Jacobienne. Si la jacobienne est carrée et non singulière ($\det(\boldsymbol{J}) \neq 0$), la solution est unique :

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(\boldsymbol{q})\dot{\boldsymbol{r}} \tag{4.58}$$

Toutefois, cette opération est impossible si le robot a une configuration singulière (voir section 4.6.1). De plus, si les dimensions de $\boldsymbol{q}$ et $\boldsymbol{r}$ sont différentes, la matrice jacobienne est rectangulaire et d'autres techniques doivent être utilisée (voir sections 4.6.2 et 4.6.3)

Capsule vidéo
Cinématique différentielle inverse des manipulateurs
https://youtu.be/YqYV10mVVBI

4.6.1 Les singularités

Certaine configuration $\boldsymbol{q}$ d'un robot manipulateur peuvent être singulière, c'est-à-dire des configuration pour lesquelles il est impossible d'inverser la matrice jacobienne.

Définition 4.4 Singularité cinématique:

Les configurations singulière sont l'ensemble des configurations pour lesquelles lesquels la matrice jacobienne subit une perte de rang. Pour un robot déterminé ($m = n$), cette condition est mathématiquement équivalente aux configurations qui mènent à une matrice jacobienne avec un déterminant nul :

$$\text{Singularités :} \quad \{\boldsymbol{q} \mid det(J(\boldsymbol{q})) = 0\} \tag{4.59}$$

Ces situations correspondent aussi à lorsque les colonnes de J sont co-linéaires, et qu'aucune combinaison linéaire ne permet de déplacer l'effecteur dans au moins une direction (la matrice jacobienne a un espace-nul gauche, voir section 18.6.2 pour les notions d'algèbre linéaire associées).

Exemple 4.4 Singularités d'un robot manipulateur planaire à 2 joints:

Pour un robot manipulateur à deux DDl comme décrit à l'exemple **??**, le Jacobien est donné par :

$$J = \begin{bmatrix} -(l_1 s_1 + l_2 s_{12}) & -l_2 s_{12} \\ (l_1 c_1 + l_2 c_{12}) & l_2 c_{12} \end{bmatrix} \tag{4.60}$$

Le déterminant de la matrice est donnée par :

$$\begin{aligned} det(J) &= l_2 s_{12}(l_1 c_1 + l_2 c_{12}) - l_2 c_{12}(l_1 s_1 + l_2 s_{12}) && (4.61)\\ &= l_1 l_2 s_{12} c_1 + l_2^2 s_{12} c_{12} - l_1 l_2 s_1 c_{12} - l_2^2 s_{12} c_{12} && (4.62)\\ &= l_1 l_2 (s_1 c_{12} - c_1 s_{12}) && (4.63)\\ &= l_1 l_2 s_2 && (4.64) \end{aligned}$$

Donc les singularités sont les configurations avec un angle q_2 égale à 0 ou 180 degrés. Comme illustré à la Figure 4.11, ces configurations correspondent à lorsque les deux liens rigides du robot sont alignés.

Dans ces configurations, il est seulement possible de faire bouger l'effecteur dans une direction perpendiculaire aux liens rigides. Comme illustré à la Figure 4.12, la direction pour laquelle le mouvement est possible correspond à l'espace colonne de la matrice Jacobienne, et la direction orthogonale pour laquelle il est impossible de déplacer l'effecteur correspond au left-nullspace de la matrice Jacobienne, voir section 18.6 pour les notions d'algèbre linéaire associées.

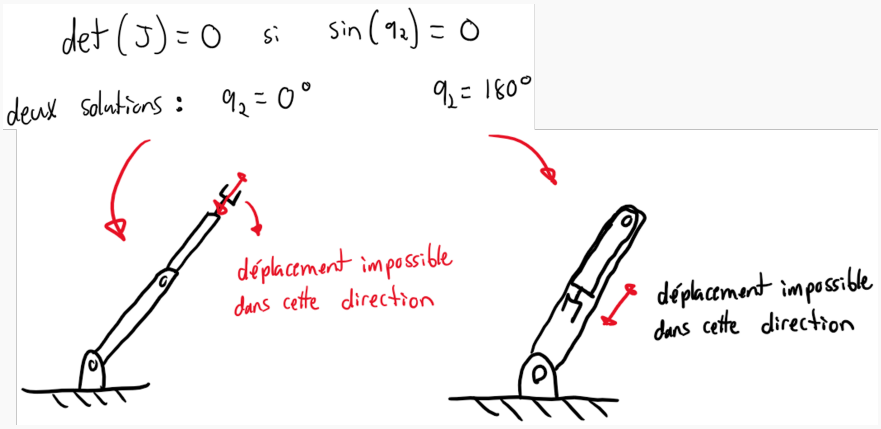


FIGURE 4.11 – Singularités pour un robot manipulateur à deux DDL

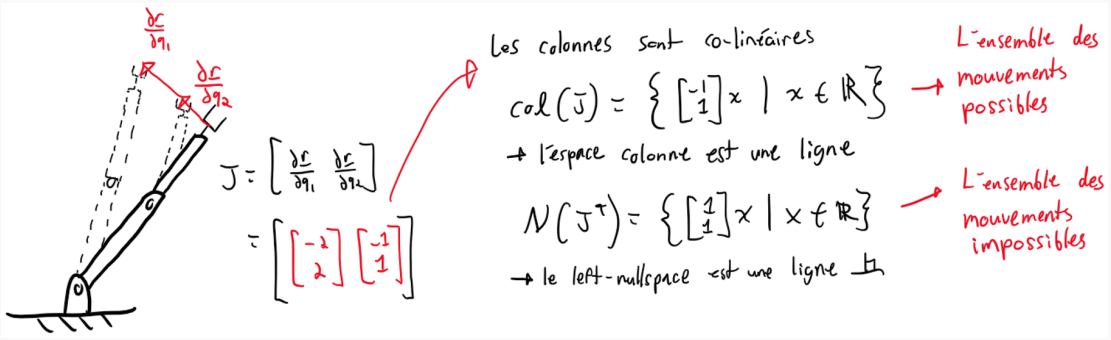


FIGURE 4.12 – Co-linéarité des colonnes du Jacobien pour une singularité

Limites de l'espace de travail

Un robot va nécessairement être sur une singularité aux limites de sont espace de travail, car par définition le robot est incapable de bouger son effecteur dans la direction normale à la surface qui définie la limite de l'espace de travail (sinon le robot pourrait sortir de l'espace de travail ce qui est une contradiction). Par exemple, les deux singularité du robot 2DDL de l'exemple précédant correspondent à des configurations où l'effecteur est à la limite intérieur de l'espace de travail ou à la limite extérieur de l'espace de travail.

Types de singularités

Détails à venir !

4.6.2 Cinématique différentielle inverse d'un robot redondant (situations sous-contraintes)

Si le nombre d'entrées n est plus grand que le nombre de sorties m :

$$n > m \tag{4.65}$$

avec

$$n = dim(\boldsymbol{q}) = \quad \text{Nombre de DDL du robot} \tag{4.66}$$
$$m = dim(\boldsymbol{r}) = \quad \text{Coordonnées de l'espace de la tâche} \tag{4.67}$$

le robot est dit redondant, et le problème de cinématique différentiel inverse est sous-contraint : il y a plusieurs solutions possible de déplacement des joints qui produisent un même déplacement de l'effecteur. Le Jacobien d'une telle situation est rectangulaire, il a plus de colonnes que de rangées :

$$\begin{bmatrix} \\ \dot{\boldsymbol{r}} \\ \\ \end{bmatrix}_{m\times 1} = \begin{bmatrix} \\ J(\boldsymbol{q}) \\ \\ \end{bmatrix}_{m\times n} \begin{bmatrix} \\ \dot{\boldsymbol{q}} \\ \\ \end{bmatrix}_{n\times 1} \tag{4.68}$$

Cette situation nous même à un problème d'algèbre linéaire sous-contraint qui peu être résolu avec une méthode qui utilise une matrice pseudo inverse, voir section 18.8 pour les notions d'algèbre linéaire. Cette situation est aussi caractérisée par la présence d'un nullspace : une combinaison de déplacement des joints peut avoir un effet net nul sur l'effecteur, voir section 18.6.3 pour les notions d'algèbre linéaire et l'exemple 18.6.

Une méthode pour travailler avec ce genre de situation consiste à utiliser l'équation de cinématique inverse suivante :

$$\begin{bmatrix} \\ \dot{\boldsymbol{q}} \\ \\ \end{bmatrix}_{n\times 1} = \underbrace{\begin{bmatrix} \\ J^{\#} \\ \\ \end{bmatrix}_{n\times m}}_{\text{Pseudo-inverse}} \begin{bmatrix} \\ \dot{\boldsymbol{r}} \\ \\ \end{bmatrix}_{m\times 1} + \underbrace{\begin{bmatrix} \\ I - J^{\#}J \\ \\ \end{bmatrix}_{n\times n}}_{\text{Projection sur le Nullspace}} \begin{bmatrix} \\ \boldsymbol{\psi} \\ \\ \end{bmatrix}_{n\times 1} \tag{4.69}$$

avec $J^{\#}$ qui est la matrice pseudo-inverse droite de Moore–Penrose, définie par :

$$J^{\#} = J^T \left(JJ^T\right)^{-1} \tag{4.70}$$

Le premier terme $J^{\#}\dot{\boldsymbol{r}}$ produit un vecteur $\dot{\boldsymbol{q}}$ qui produit une solution exacte pour le système d'équations (4.68) et minimise la norme du vecteur-colonne $\dot{\boldsymbol{q}}$. Le second terme produit une variation pour le vecteur $\dot{\boldsymbol{q}}$ qui réside dans l'espace nul de la matrice J, donc cette variation n'a aucun influence sur la sortie et n'affectera pas l'objectif principal d'obtenir un vecteur $\dot{\boldsymbol{q}}$ qui produit un mouvement de l'effecteur donné par $\dot{\boldsymbol{r}}$. Le vecteur $\boldsymbol{\psi}$ dans le second terme est arbitraire et peut être utilisé pour prioriser certaine solutions sans influencer la justesse de la solution.

Exemple 4.5 Exemple de cinématique différentielle inverse pour une situation sous-contrainte:

Le robot de l'exemple 4.5.1 est ici analysé en termes de cinématique différentielle, la Figure 4.13 illustre la présence de plusieurs solutions possibles et la Figure 4.14 illustre l'utilisation de la matrice pseudo-inverse pour obtenir la solution optimale ainsi que le nullspace.

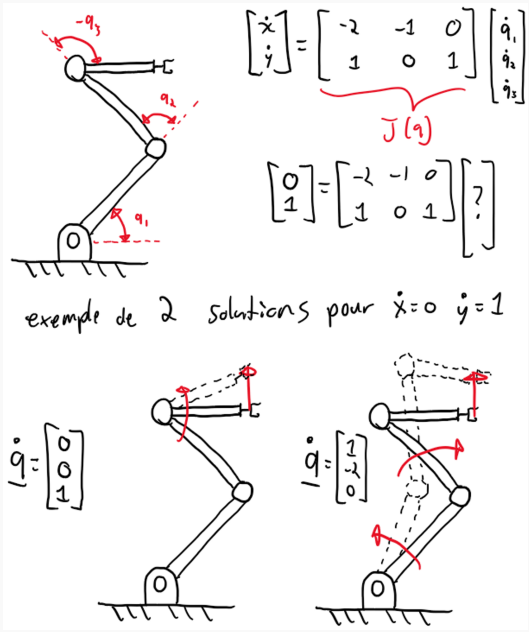


FIGURE 4.13 – Cinématique différentielle inverse d'un robot redondant ($n = 3 > m = 2$)

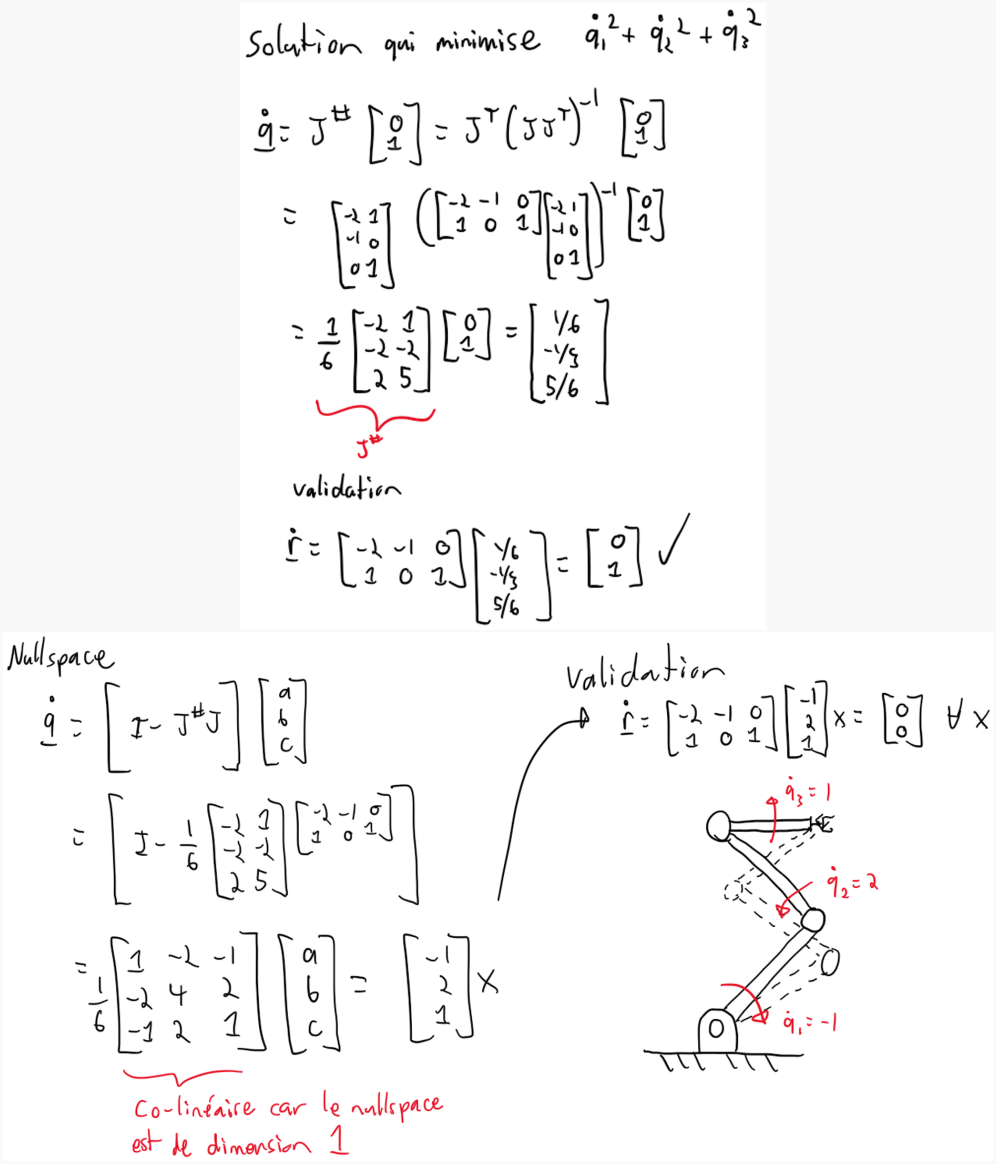


FIGURE 4.14 – Cinématique différentielle inverse d'un robot redondant : solution optimale et nullspace

4.6.3 Cinématique différentielle inverse pour une situation sur-contrainte

Si le nombre d'entrées n est plus petit que le nombre de sorties m :

$$n < m \tag{4.71}$$

avec

$$n = dim(\boldsymbol{q}) = \quad \text{Nombre de DDL du robot} \tag{4.72}$$
$$m = dim(\boldsymbol{r}) = \quad \text{Coordonnées de l'espace de la tâche} \tag{4.73}$$

le problème de cinématique différentiel inverse est sur-contraint : il n'y a pas de solutions exactes possibles sauf pour un sous-ensemble de déplacements de l'effecteur. Le Jacobien d'une telle situation est rectangulaire,

il a plus de rangées que de colonnes :

$$\begin{bmatrix} \\ \dot{\boldsymbol{r}} \\ \\ \end{bmatrix}_{m\times 1} = \begin{bmatrix} \\ J(\boldsymbol{q}) \\ \\ \end{bmatrix}_{m\times n} \begin{bmatrix} \\ \dot{\boldsymbol{q}} \\ \\ \end{bmatrix}_{n\times 1} \tag{4.74}$$

Un méthode pratique pour la cinématique inverse dans ce contexte est la méthode des moindres-carré, voir la section 18.7 pour les notions d'algèbre linéaire. L'utilisation de cette méthode permet de calculer explicitement un vecteur solution approximé $\hat{\dot{\boldsymbol{q}}}$ qui minimise la norme de l'erreur entre une vitesse à l'effecteur désiré et celle obtenue avec la solution approximée. La solution au sens des moindres-carrée est calculée ainsi :

$$\hat{\dot{\boldsymbol{q}}} = \underset{\dot{\boldsymbol{q}}}{\operatorname{argmin}} \|J\dot{\boldsymbol{q}} - \dot{\boldsymbol{r}}\|^2 = \left(J^T J\right)^{-1} J^T \dot{\boldsymbol{r}} \tag{4.75}$$

Exemple 4.6 Exemple de cinématique différentielle inverse pour une situation sur-contrainte:

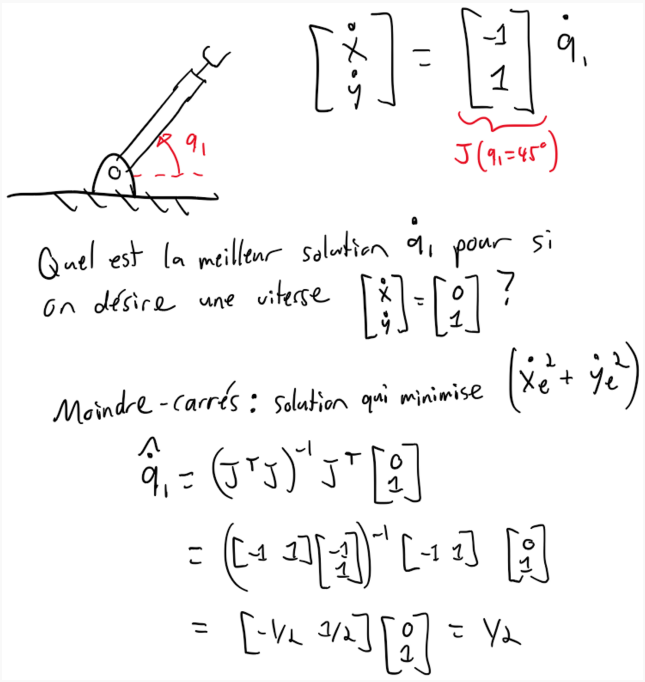


FIGURE 4.15 – Cinématique différentielle inverse pour une situation sur-contrainte ($n = 1 < m = 2$)

4.7 Manipulabilité d'un robot manipulateur

Lorsqu'on désire effectuer une tâche avec l'effecteur d'un robot à une certaine position, la première question est : est-ce que le robot est capable d'atteindre cette position, i.e. est-ce que la position cible est dans l'espace de travail ? Si la position est atteignable, une deuxième question doit être posée : est-ce que le robot est capable de bouger arbitrairement à cette position ? Cette deuxième question est la notion de manipulabilité du robot manipulateur.

Comme discuté à la section 4.6.1, lorsque le robot est à une configuration singulière l'effecteur est incapable de bouger dans certaines directions. On considère donc que la manipulabilité d'un robot n'est pas bonne sur une singularité. Toutefois, il est utile d'avoir une notion non-binaire de la manipulabilité, car même si un robot n'est pas parfaitement sur une singularité, certaines configurations peuvent nécessiter de très grandes vitesses aux joints pour satisfaire des vitesses cartésiennes de l'effecteur. Donc en considérant les limites de vitesse des joints le robot est en fait incapable d'executer certain vecteur vitesse à son effecteur proche d'une singularité. Les indices de manipulabilité sont des fonctions scalaires qui ont comme objectif de caractériser si un robot dans une certaine configuration est proche d'une singularité. Ces indicateurs peuvent être utile lorsque qu'on conçoit la cinématique d'un robot ou lorsque qu'on programme des tâches avec un robot existant.

Indices de manipulabilité d'un robot manipulateur

Pour les situations ou la matrice jacobienne J du robot est carré, le déterminant est un bon indice de la manipulabilité du robot :

$$m_1(\boldsymbol{q}) = det(J(\boldsymbol{q})) \tag{4.76}$$

Le déterminant correspond au volume du prisme généré par les colonnes du jacobien qui correspondent à des vecteurs vitesses de l'effecteur pour des vitesses de joints unitaires de chacun des axes :

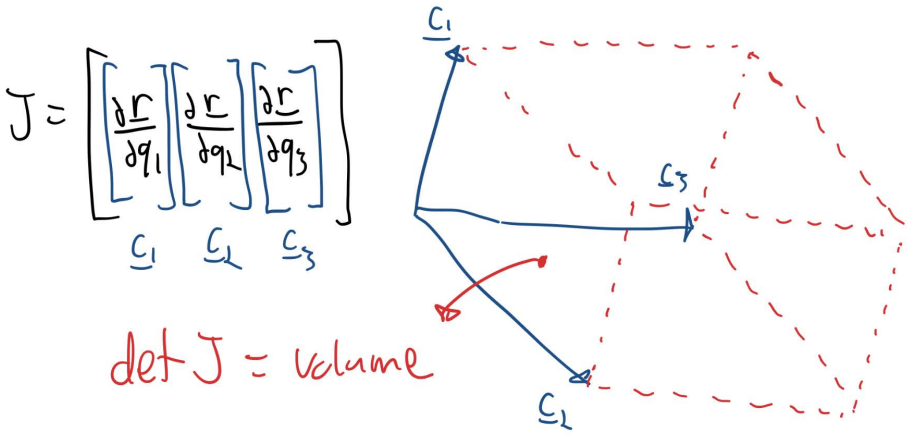


FIGURE 4.16 – Volume de la matrice jacobienne

Dès que un des vecteurs est très petit, ou bien quand les vecteurs sont colinéaires, le volume du prisme devient très petit ce qui correspond à une mauvaise manipulabilité.

Un autre indicateur souvent utilisé est les valeurs singulières de la matrice jacobienne. Les valeurs singulières μ sont relié aux valeurs propres λ du produit $J^T J$

$$\mu_i^2(J(\boldsymbol{q})) = \lambda_i(J(\boldsymbol{q})^T J(\boldsymbol{q})) \tag{4.77}$$

Pour relié à un indice de manipulabilité, on peut surveillez la plus petite valeur singulière ou bien le ratio

entre la plus petite et la plus grande :

$$m_2(\boldsymbol{q}) = \mu_{min}(J(\boldsymbol{q})) \tag{4.78}$$

$$m_3(\boldsymbol{q}) = \frac{\mu_{min}(J(\boldsymbol{q}))}{\mu_{max}(J(\boldsymbol{q}))} \tag{4.79}$$

$$\tag{4.80}$$

La valeur μ_{min} correspond à la norme minimal du vecteur vitesse de l'effecteur produit par un vecteur de vitesse aux joints unitaire.

Ellipsoïde de vitesse

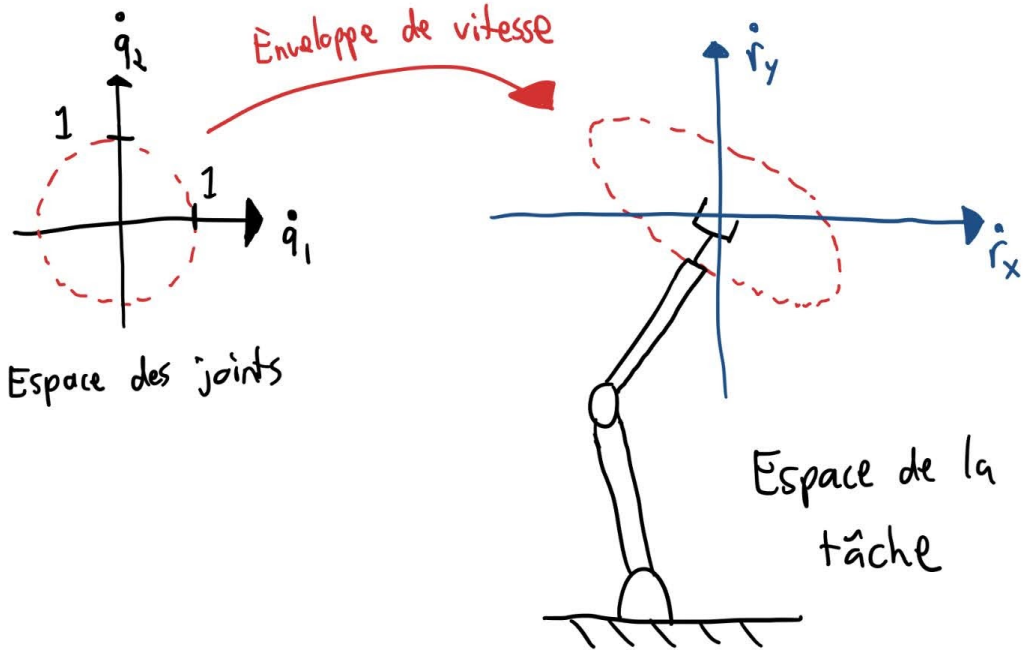


FIGURE 4.17 – Ellipsoïde de vitesse

L'ellipsoïde de vitesse d'un robot manipulateur est l'enveloppe des vecteurs vitesses possibles à l'effecteur pour des vecteurs de vitesse des joints unitaires, voir 4.17. L'enveloppe de vitesses possibles à l'effecteur, résultant d'un vecteur de vitesse des joints unitaire, peut être calculer en débutant avec une équation qui contraint la norme du vecteur de la vitesse des joint et en substituant $\dot{\boldsymbol{q}}$ par $J^{-1}\dot{\boldsymbol{r}}$ dans l'équation :

$$1 = \dot{\boldsymbol{q}}^T \dot{\boldsymbol{q}} \tag{4.81}$$

$$1 = \dot{\boldsymbol{r}}^T J^{-T} J^{-1} \dot{\boldsymbol{r}} \tag{4.82}$$

$$1 = \dot{\boldsymbol{r}}^T (JJ^T)^{-1} \dot{\boldsymbol{r}} \tag{4.83}$$

$$1 = \dot{\boldsymbol{r}}^T A^{-1} \dot{\boldsymbol{r}} \qquad \text{avec} \qquad A = JJ^T \tag{4.84}$$

On obtient donc une équation quadratique pour laquelle l'ensemble des solutions possibles forment une ellipse. Les axes principaux de l'ellipse correspondent aux valeurs propres et vecteurs propres de la matrice JJ^T.

FIGURE 4.18 – TODO : ellipse 3D

Un lien peu aussi être fait directement avec les valeurs singulière de la matrice J. Détails à venir !

Exemple 4.7 Indice de manipulabilité pour un robot à 2 DDL:

Une exemple simple pour illustré l'indice de manipulabilité est un robot planaire à 2 DDL comme illutré à la Figure 4.19, pour lequel on peut analyser la manipulabilité pour divers position de l'effecteur sur l'axe horizontal. La manipulabilité est nulle lorsque le coude est complètement plié (le robot est sur une singularité et ne peut bouger horizontalement), nulle lorsque le coude est complètement déplié (le robot est sur une singularité de ne peut bouger horizontalement) et optimale entre les deux lorsque le coude est partielement plié.

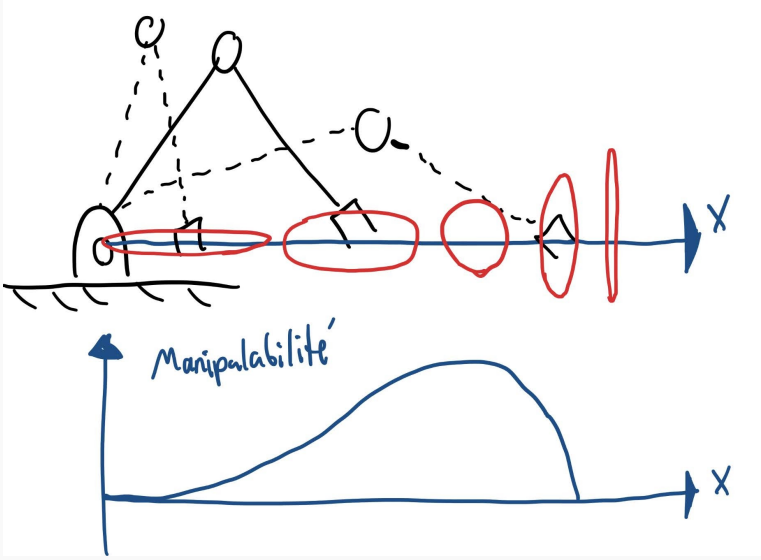


FIGURE 4.19 – Exemple de l'indice de manipulabilité pour un robot à 2 DDL

4.8 Relation entre les variables d'accélération

Pour faire le lien en l'accélération dans l'espace des joints et l'accélération dans l'espace de la tâche, il suffit de dériver par rapport au temps l'expression de la cinématique différentielle, autrement dit on dérive deux fois par rapport au temps l'équation de la cinématique directe :

$$\text{Cinématique directe :} \quad \boldsymbol{r} = f(\boldsymbol{q}) \tag{4.85}$$

$$\text{Cinématique différentielle :} \quad \dot{\boldsymbol{r}} = J(\boldsymbol{q})\,\dot{\boldsymbol{q}} \tag{4.86}$$

$$\text{Cinématique différentielle ordre-2 :} \quad \ddot{\boldsymbol{r}} = J(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + \dot{J}(\boldsymbol{q}, \dot{\boldsymbol{q}})\,\dot{\boldsymbol{q}} \tag{4.87}$$

Comme on dérive un produit avec deux termes qui varient dans le temps, l'accélération $\ddot{\boldsymbol{r}}$ est le résultat de deux effets représentés par deux termes. Le premier terme est simplement l'accélération des joints fois le Jacobien, i.e. les bras de leviers. Le deuxième terme est une contribution à l'accélération de l'effecteur qui vient de la variation temporelle du Jacobien. Comme le Jacobien a seulement une dépendance directe aux positions $\boldsymbol{q}$, on peut exprimer son taux de variation temporelle comme sa dérivée par rapport à $\boldsymbol{q}$ fois le vecteur-colonne de vitesse $\dot{\boldsymbol{q}}$, selon le principe de dérivée en chaînes :

$$\dot{J} = \frac{\partial J}{\partial \boldsymbol{q}}\,\dot{\boldsymbol{q}} \tag{4.88}$$

Toutefois ici, on dérive une matrice $m \times n$ par un vecteur-colonne $n \times 1$, le résultat est donc un tenseur (ses composantes si on veux être rigoureux) d'ordre 3, i.e. une matrice 3D $m \times n \times n$:

$$\begin{bmatrix} \\ \dot{J} \\ \\ \end{bmatrix}_{m\times n} = \underbrace{\begin{bmatrix} \\ \frac{\partial J}{\partial \boldsymbol{q}} \\ \\ \end{bmatrix}_{m\times n\times n}}_{\text{Tenseur d'ordre 3}} \begin{bmatrix} \\ \dot{\boldsymbol{q}} \\ \\ \end{bmatrix}_{n\times 1} \tag{4.89}$$

Lorsqu'on travail avec des tenseurs d'ordre 3 et plus, il est normalement plus facile de travailler avec des équations qui relient les composantes avec une notation indicielle. La composante ijk du tenseur qui résultent de la dérivation du Jacobien J par $\boldsymbol{q}$ est égale à la dérivée de l'élément ij du Jacobien par rapport à l'élément k du vecteur-colonne $\boldsymbol{q}$:

$$\left[\frac{\partial J}{\partial \boldsymbol{q}}\right]_{ijk} = \frac{\partial J_{ij}}{\partial q_k} \tag{4.90}$$

Donc le produit intérieur exprimé à l'équation (4.89) peut être exprimé directement comme une sommation à effectuer pour calculer l'élément ij de la matrice $\dot{J}$:

$$\dot{J}_{ij} = \sum_k \left(\frac{\partial J_{ij}}{\partial q_k}\dot{q}_k\right) \tag{4.91}$$

Finalement, il est possible d'exprimer directement l'équation (4.87) en termes de composantes, d'indices et de sommations :

$$\ddot{r}_i = \sum_j J_{ij}\ddot{q}_j + \sum_j \sum_k \frac{\partial J_{ij}}{\partial q_k}\dot{q}_k\dot{q}_j \tag{4.92}$$

Plus de détails sur les équivalences entre les opérations en termes de matrices/vecteur-colonnes/tenseurs et les opérations en termes de composantes et d'indices sont disponibles à la section 17.4.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

4.9 Résumé du chapitre

Vecteur vitesse et dérivée d'un vecteur position (base b non-inertielle)

$$\underbrace{\vec{v}_B = {}^b\frac{d}{dt}\vec{r}_{B/A} + \vec{w}_{b/a} \times \vec{r}_{B/A}}_{\text{Relation vectorielle}} \quad \Leftrightarrow \quad \underbrace{\boldsymbol{v}_B^a = {}^aR^b\left[\dot{\boldsymbol{r}}_{B/A}^b + (\boldsymbol{w}_{b/a}^b)^\times \boldsymbol{r}_{B/A}^b\right]}_{\text{Relation matricielle des composantes}} \tag{4.93}$$

Vecteur accélération et dérivée seconde d'un vecteur position

$$\vec{a}_B = {}^b\frac{d^2}{dt^2}\vec{r}_{B/A} + {}^b\frac{d}{dt}\vec{w}_{b/a} \times \vec{r}_{B/A} + \underbrace{2\vec{w}_{b/a} \times {}^b\frac{d^2}{dt^2}\vec{r}_{B/A}}_{\text{Coriolis}} + \underbrace{\vec{w}_{b/a} \times \vec{w}_{b/a} \times \vec{r}_{B/A}}_{\text{Centrifuge}} \tag{4.94}$$

$$\boldsymbol{a}_B^a = {}^aR^b\left[\ddot{\boldsymbol{r}}_{B/A}^b + (\dot{\boldsymbol{w}}_{b/a}^b)^\times \boldsymbol{r}_{B/A}^b + \underbrace{2(\boldsymbol{w}_{b/a}^b)^\times \dot{\boldsymbol{r}}_{B/A}^b}_{\text{Coriolis}} + \underbrace{(\boldsymbol{w}_{b/a}^b)^\times (\boldsymbol{w}_{b/a}^b)^\times \boldsymbol{r}_{B/A}^b}_{\text{Centrifuge}}\right] \tag{4.95}$$

Dérivée temporelle d'une matrice de rotation :

$$\dot{R} = \frac{d}{dt}R = (\boldsymbol{w})^x\, R \tag{4.96}$$

$${}^a\dot{R}^b = (\boldsymbol{w}_{b/a}^a)^\times\, {}^aR^b = {}^aR^b\,(\boldsymbol{w}_{b/a}^b)^\times \tag{4.97}$$

Relations différentielles entre l'espace des joints et de la tâche :

$$\boldsymbol{r} = f(\boldsymbol{q}) \tag{4.98}$$

$$\dot{\boldsymbol{r}} = J(\boldsymbol{q})\,\dot{\boldsymbol{q}} \tag{4.99}$$

$$\ddot{\boldsymbol{r}} = J(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + \dot{J}(\boldsymbol{q}, \dot{\boldsymbol{q}})\,\dot{\boldsymbol{q}} \tag{4.100}$$

La matrice Jacobienne :

$$J(\boldsymbol{q}) = \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{q}} = \begin{bmatrix} \frac{\partial r_1}{\partial q_1} & \cdots & \frac{\partial r_1}{\partial q_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_m}{\partial q_1} & \cdots & \frac{\partial r_m}{\partial q_n} \end{bmatrix}_{m\times n} \tag{4.101}$$

Chapitre 5

Statique des robots manipulateurs

5.1 Introduction

Ce chapitre présente des méthodes pour déterminer les relations entre les divers forces agissant sur un robot manipulateur lorsque le système est à l'équilibre. La matrice jacobienne J, la même matrice de la relation de cinématique différentielle de l'équation 4.29, permet de transformer une force cartésienne $\boldsymbol{f}_E$, que l'effecteur du robot applique sur l'environnement, en couples (ou force pour un joint prismatique) équivalents $\boldsymbol{\tau}$ dans le domaine des joints :

Définition 5.1 Transformation efforts articulaires ↔ efforts à l'effecteur:

$$\boldsymbol{\tau} = J(\boldsymbol{q})^T \, \boldsymbol{f}_E \tag{5.1}$$

Cette relation permet de calculer les liens entre les forces externes et les forces produites par les actionneurs sans avoir à déterminer toutes les forces internes du systèmes, ce qui particulièrement utile pour les robots manipulateurs et sera exploité par les méthodes de ce chapitre.

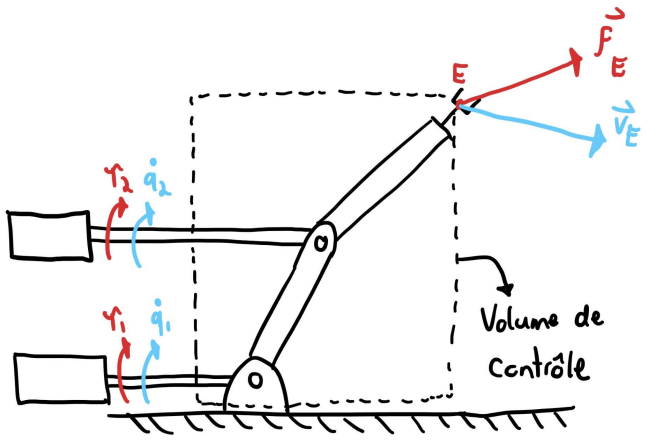


FIGURE 5.1 – Bilan de puissance mécanique pour un robot manipulateur

Capsule vidéo
Statique des robots manipulateurs
https://youtu.be/klAfYgPYi0k

5.2 Les vecteurs forces

Une force est une quantité vectorielle avec une amplitude et une direction. Comme pour les notions de cinématique, nous utiliserons une notation qui distingue le vecteur-géométrique $\vec{f}$ et le vecteur-colonne de composantes $\boldsymbol{f}^a$ dans une base a. De plus, une force appliquée sur un corps C à un point A sera notée :

$$\vec{f}_{C_A} \tag{5.2}$$

Le principe d'action/réaction indique que si une force est appliquée sur un corps $C1$ il y a nécessairement une réaction égale et opposée sur un corps $C2$. Une force agissant sur un corps $C1$ par un corps $C2$ sera notée :

$$\vec{f}_{C1/C2} \tag{5.3}$$

et le principe d'action/réaction nous donne donc la propriété suivante :

$$\vec{f}_{C1/C2} = -\vec{f}_{C2/C1} \tag{5.4}$$

Cette notation sera utilisée lorsqu'il est question de forces d'interactions entre deux corps car le signe source de confusion lorsqu'on ne distingue pas la force appliquée *sur* à la force appliqué *par* un objet.

Dans ce chapitre, nous définirons un vecteur $\vec{f}_{R_E}$ comme un force appliquée sur le robot à un point E (typiquement l'effecteur) par l'environnement, et $\vec{f}_E$ comme son inverse, i.e. la force que le robot applique sur l'environnement :

$$\vec{f}_{R_E} = \vec{f}_{Robot/Environment} \tag{5.5}$$

$$\vec{f}_E = \vec{f}_{Environment/Robot} = -\vec{f}_{R_E} \tag{5.6}$$

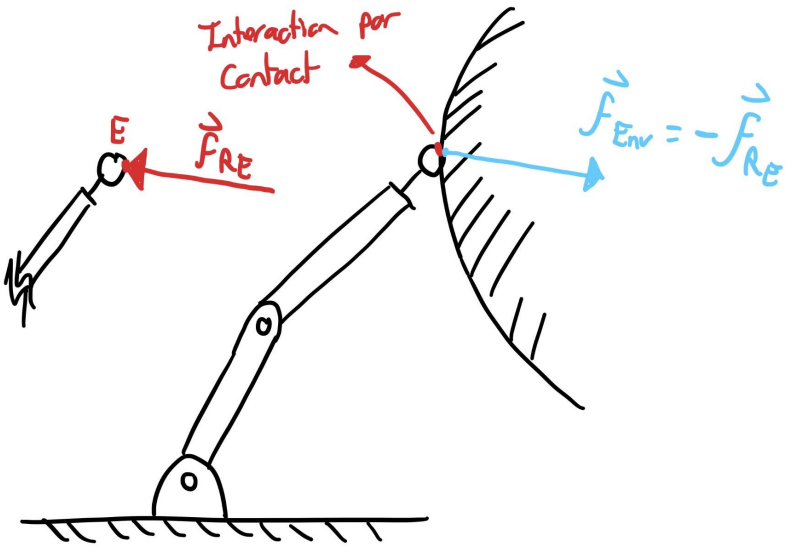


FIGURE 5.2 – Action-réaction pour une force de contact

5.3 Relation entre les forces aux joints et les forces à l'effecteur

La relation entre les forces/couples dans l'espace des joints $\boldsymbol{\tau}$ et un vecteur de forces externes $\vec{f}_{R_E}$ appliquée sur l'effecteur d'un robot peut être décrite par la relation suivante :

$$\boldsymbol{\tau} = J(\boldsymbol{q})^T \boldsymbol{f}_E = -J(\boldsymbol{q})^T \boldsymbol{f}_{R_E} \tag{5.7}$$

où la matrice Jacobienne J est la même matrice de la relation de cinématique différentielle de l'équation 4.29. Comme illustré à la Figure 5.1, les variables de forces/couples doivent correspondent aux même dégrées de liberté que les variables vitesses associées à la relation de cinématique différentielle.

Démonstration. La relation statique qui implique le Jacobien (eq. (5.7)) peut être déterminée à partir d'un bilan de puissance. Si on applique la 1ère loi de la thermodynamique au volume de contrôle indiqué à la Figure 5.1 pour calculer le bilan d'énergie :

$$\frac{dE}{dt} = P_{in} - P_{out} \tag{5.8}$$

Les forces internes inertielles et dissipatrices sont ici négligés pour trouver une relation valide dans des conditions quasi-statique. De plus considérons d'abord un cas sans force conservatrice (voir section 5.4 pour ce cas) Dans ces conditions, il n'y a aucune accumulation d'énergie interne : les seules entrée/sorties de puissance dans le système sont le travail mécanique fait par les actionneurs sur le robot, et le travail mécanique fait par le robot sur l'environnement. L'égalité du travail mécanique en entrée et sortie peut être convertie en relation matricielle :

$$P_{in} = P_{out} \tag{5.9}$$

$$\dot{q}_1\tau_1 + \dot{q}_2\tau_2 + \ldots = \vec{v} \cdot \vec{f}_E \tag{5.10}$$

$$\begin{bmatrix} \dot{q}_1 & \dot{q}_2 & \ldots \end{bmatrix} \begin{bmatrix} \tau_1 \\ \tau_2 \\ \vdots \end{bmatrix} = \dot{\boldsymbol{r}}^T \boldsymbol{f}_E \tag{5.11}$$

$$\dot{\boldsymbol{q}}^T \boldsymbol{\tau} = \dot{\boldsymbol{r}}^T \boldsymbol{f}_E \tag{5.12}$$

$$\tag{5.13}$$

Ensuite, si on substitut les composantes du vecteur vitesse de l'effecteur $\dot{\boldsymbol{r}}$ par l'équation de cinématique différentielle directe (eq. (4.29)), on obtient :

$$\dot{\boldsymbol{q}}^T \boldsymbol{\tau} = \left(J(\boldsymbol{q})\,\dot{\boldsymbol{q}}\right)^T \boldsymbol{f}_E \tag{5.14}$$

$$\dot{\boldsymbol{q}}^T \boldsymbol{\tau} = \dot{\boldsymbol{q}}^T J(\boldsymbol{q})^T \boldsymbol{f}_E \tag{5.15}$$

$$\dot{\boldsymbol{q}}^T \Big(\boldsymbol{\tau}\Big) = \dot{\boldsymbol{q}}^T \Big(J(\boldsymbol{q})^T \boldsymbol{f}_E\Big) \quad \Rightarrow \quad \boldsymbol{\tau} = J\left(\boldsymbol{q}\right)^T \boldsymbol{f}_E \tag{5.16}$$

Donc les deux termes qui multiplient (produit intérieur) le vecteur colonne $\dot{\boldsymbol{q}}$ doivent nécessairement être égaux, se qui nous donne l'équation qui relie les forces dans l'espace des joints aux forces équivalentes dans l'espace de la tâche. □

5.3.1 Flux de puissance

Du point de vue du flux de puissance dans le système robotique, de l'énergie électrique vers le travail mécanique fait par le robot, la Jacobien peut-être vue comme une matrice de ratios de transformation. Par exemple, les Figures 5.3 et 5.4 illustrent quelques unes des transformations de puissance dans un système robotique et les paramètres associées. Dans chacun des domaines la puissance est caractérisée par deux variables, une de flux (courant, vitesse, débit, etc.) et une de d'effort (tension, couple, force, pression, etc.). La puissance est le produit de ces deux variables, ou le produit intérieur des vecteurs-colonnes de ces variables dans le cas de systèmes multi-variables. Le flux de puissance peut être transformé par des dispositifs qu'on appelle des transformateurs. Un transformateur amplifie la variable de flux et réduit la variable d'effort, ou vice-versa. Par exemple, les transformateurs électriques qui réduise la tension dans un réseau électrique, les transmissions mécaniques avec ratios de réduction, etc. Lorsqu'un dispositif transfert la puissance sous une forme d'énergie différente, on les appelle des transducteurs, par exemple un moteur électrique. La puissance en entrée et en sortie des transformateurs est conservée, si l'effort est amplifié d'un facteur T le flux est réduit d'un facteur T.

Capsule vidéo
Flux de puissance pour un robot manipulateur
https://youtu.be/qyrFzkjPhHY

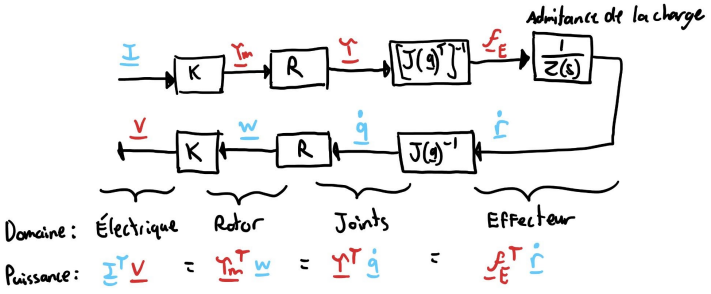


FIGURE 5.3 – Flux de puissance : les variables de flux et d'effort dans les différents domaines et espaces du robot.

Les Figures 5.3 et 5.4 illustrent les trois transformations de puissances principales dans un bras robotique typique. La puissance électrique est transformée en puissance mécanique des l'arbre des moteurs par les moteurs électriques, ensuite celle-ci est transformée en puissance mécanique dans les joints du robot par les transmissions, et finalement la puissance mécanique dans les joints est transformé en puissance mécanique à l'effecteur du robot. La première transformation est caractérisée par les constantes moteurs k_m :

$$\tau = k_m I \tag{5.17}$$
$$v = k_m w \tag{5.18}$$

la deuxième transformation est caractérisée par les ratios de réductions des transmissions R :

$$\tau = R\tau_m \tag{5.19}$$
$$w = R\dot{q} \tag{5.20}$$

et la dernière transformation est caractérisée par la matrice Jacobienne du robot J :

$$\boldsymbol{\tau} = J^T \boldsymbol{f}_E \tag{5.21}$$
$$\dot{\boldsymbol{r}} = J\dot{\boldsymbol{q}} \tag{5.22}$$

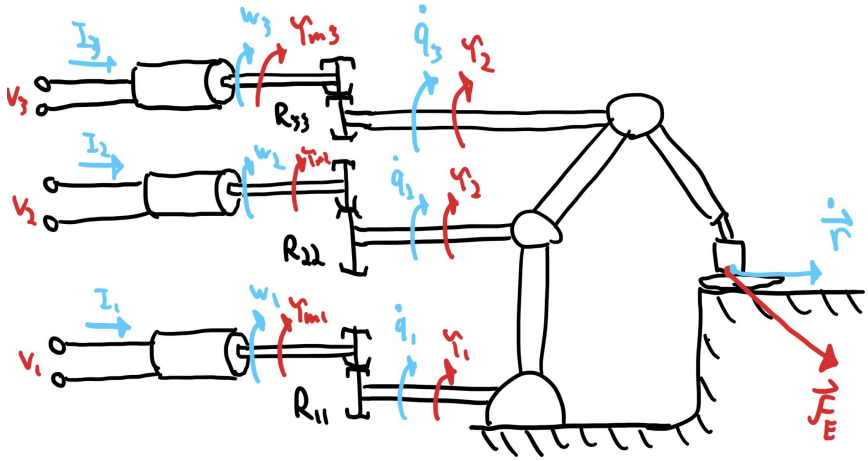


FIGURE 5.4 – Flux de puissance : de l'alimentation électrique vers le travail sur la charge

Exemple 5.1 Relation statique pour un robot simple à un DDL:

La Figure 5.5 illustre un la matrice Jacobienne d'un robot simple à 1 DDL, pour lequel l'espace de la tâche est simplement la position horizontale x de l'effecteur. Pour ce système à une entrée et une sortie, la matrice Jacobienne est de dimension 1×1, donc un scalaire qui ici correspond à la hauteur de l'effecteur du robot. La vitesse $\dot{x}$ pour une vitesse angulaire $\dot{\theta}$ donnée est fonction de cette hauteur. Finalement, le lien entre une force horizontale à l'effecteur et le couple équivalent au joint est aussi fonction de cette même hauteur.

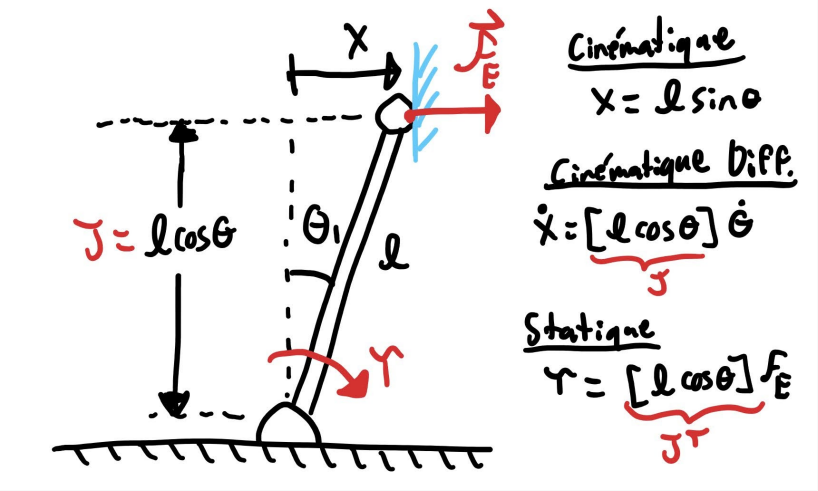


FIGURE 5.5 – Exemple de la relation statique pour un manipulateur à 1 DDL

5.3.2 Relation statique sur une singularité

Lorsqu'un robot manipulateur est sur une singularité, la transposée de la matrice Jacobienne est caractérisée par une amplification d'un facteur zéro pour certaines direction spécifique de vecteur force externe. Cela signifie que aucun couple dans l'espace des joints est nécessaire pour résister à une force externe dans cette direction. De façon générale si le un robot est proche d'une singularité, très peu de couple aux joint est nécessaire pour résister à certaines directions de forces externes. La marche humaine est particulièrement efficace car nos jambes sont maintenues dans des configurations proches de singularité qui permette de résister à la charge gravitationnelle avec peu d'effort musculaire.

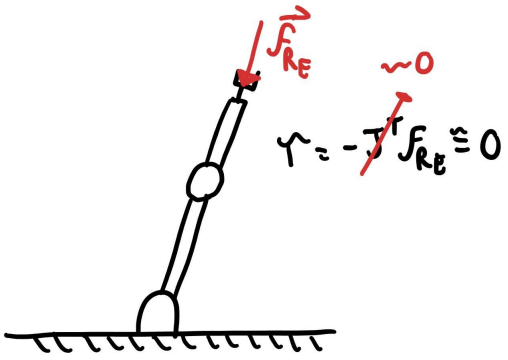


FIGURE 5.6 – Relation statique sur une singularité

5.4 Relation statique incluant les forces conservatrices

Si la bilan d'énergie effectué à la section 5.3 est fait en incluant l'énergie potentielle, un terme de forces conservatrices $\boldsymbol{g}$ doit être ajouté à l'équation de la balance des forces dans le systèmes. On nommera ici ce vecteur $\boldsymbol{g}$ car pour les robots manipulateur, c'est généralement un vecteur de forces gravitationnelles.

Définition 5.2 Relation statique avec forces conservatrices:

$$\boldsymbol{\tau} = J(\boldsymbol{q})^T \boldsymbol{f}_E + g(\boldsymbol{q}) \qquad \text{avec :} \qquad \underbrace{\boldsymbol{g}(\boldsymbol{q}) = \frac{\partial V(\boldsymbol{q})}{\partial \boldsymbol{q}^T}}_{\text{Gradient de l'énergie potentielle}} \tag{5.23}$$

Note :

Le vecteur colonne par lequel on prend les dérivés partielles de V est ici noté avec une transposition pour respecter la convention (voir section 17.4) pour obtenir pour $\boldsymbol{g}$ un vecteur-colonne $n \times 1$ plutôt que un vecteur-rangé $1 \times n$.

Démonstration. Le nouveau terme provient du fait que la variation de l'énergie interne du système n'est pas nulle, car l'énergie potentielle du système varie selon le travail effectué par les forces conservatrices :

$$E = V(\boldsymbol{q}) \quad \Rightarrow \quad \frac{dE}{dt} = \frac{dV}{dt} = \frac{\partial V}{\partial \boldsymbol{q}} \frac{d\boldsymbol{q}}{dt} = \frac{\partial V}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} \tag{5.24}$$

On obient alors :

$$\frac{dE}{dt} = P_{in} - P_{out} \tag{5.25}$$

$$\frac{\partial V}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} = \dot{\boldsymbol{q}}^T \boldsymbol{\tau} - \dot{\boldsymbol{q}}^T J(\boldsymbol{q})^T \boldsymbol{f}_E \tag{5.26}$$

$$\dot{\boldsymbol{q}}^T \left(\frac{\partial V}{\partial \boldsymbol{q}^T} \right) = \dot{\boldsymbol{q}}^T \left(\boldsymbol{\tau} - J(\boldsymbol{q})^T \boldsymbol{f}_E \right) \tag{5.27}$$

Donc pour que la relation de puissance soit respecté, les termes qui multiplie le vecteur-colonne de vitesses doivent être égaux, ce qui donne la relation entre les forces de l'équation (5.23). □

Capsule vidéo
Relation statique d'un robot manipulateur incluant la gravité
https://youtu.be/RcARRDVzrm8

5.5 Relation de la compliance aux joints et à l'effecteur

Lorsque les robots sont contrôlé en position par des servo-moteurs à chaque joint, chacun de ces moteurs et sa boucle d'asservissement en position peut être représenté comme une rigidité dans l'espace des joints, i.e. une variation du couple produit par le joint pour un déplacement de ce joint par rapport à la configuration que le contrôleur tente de maintenir :

$$\delta\tau_i = k_i \delta q_i \quad \Rightarrow \quad \delta\boldsymbol{\tau} = K_q\, \delta\boldsymbol{q} \tag{5.28}$$

où K_q est une matrice de rigidité dans l'espace des joints. Il est ensuite intéressant de déterminer la rigidité d'un robot, comme illustré à la Figure 5.7, lorsqu'il subit une force externe en fonction de la rigidité angulaire de ses servo-moteurs. La variation de position de l'effecteur peut être reliée aux déplacement angulaire dans

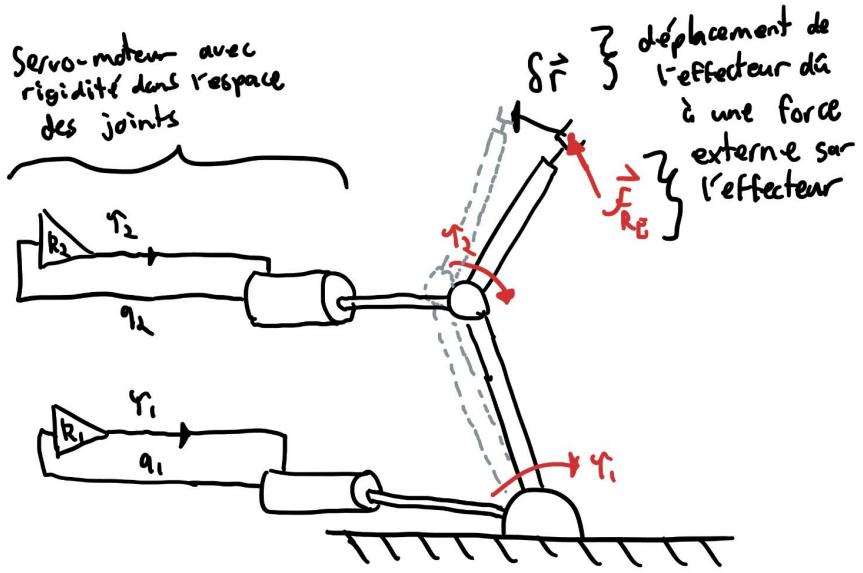


FIGURE 5.7 – Compliance d'un robot à une force externe

l'espace des joints, et la force externe peut être transformée en couples équivalents à chacun des joints (pour un équilibre statique). En manipulant les équations on obtient :

$$\delta\boldsymbol{r} = J\delta\boldsymbol{q} = JK_q^{-1}\boldsymbol{\tau} = \underbrace{\left[JK_q^{-1}J^T\right]}_{C_r} \boldsymbol{f}_{R_E} \tag{5.29}$$

où C_r est une matrice de compliance à l'effecteur. La rigidité apparente à l'effecteur, due à un rigidité dans l'espace des joints, est donc donnée par une relation qui relie la variation de la position de l'effecteur à un force externe appliquée sur l'effecteur :

$$\boldsymbol{f}_E = -\boldsymbol{f}_{R_E} = -K_r\delta\boldsymbol{r} \quad \text{avec } K_r = C_r^{-1} = \left[JK_q^{-1}J^T\right]^{-1} \tag{5.30}$$

Comme illustré à la Figure 5.8, la matrice de rigidité dans l'espace des joints est généralement diagonale (les moteurs sont indépendents). Il est à noter que la matrice de rigidité à l'effecteur est dépendante de la configuration nominale du robot (i.e. J est une fonction de $\boldsymbol{q}$ en général). Finalement, il est à noter que la matrice C de compliance est singulière lorsque le Jacobien est singulier, donc sur une singularité un robot manipulateur a une ou des directions avec aucune compliance, i.e. infiniment rigide, comme illustré à la Figure 5.9.

Capsule vidéo
Compliance et rigidité des robots manipulateurs
https://youtu.be/U3fhFlkVoEg

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

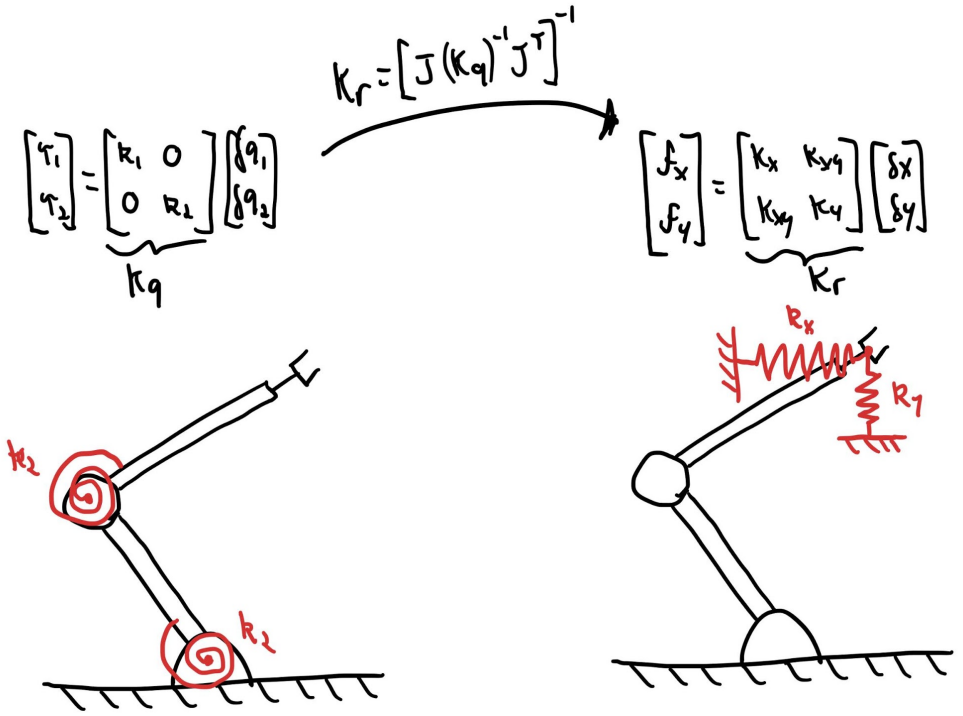


FIGURE 5.8 – Matrice de rigidité équivalente d'un robot dans l'espace de l'effecteur

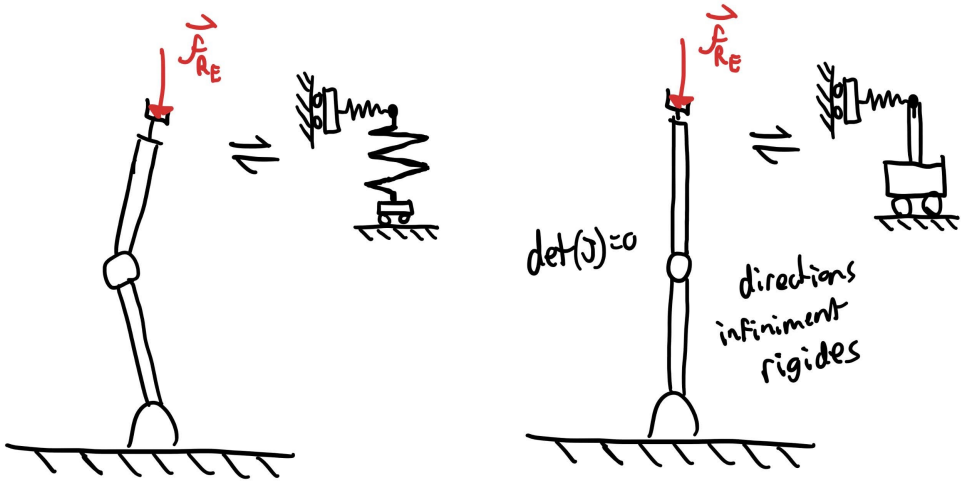


FIGURE 5.9 – Effet des configuration singulière sur la rigidité d'un robot manipulateur

5.6 Manipulabilité en force d'un robot manipulateur

L'ellipsoïde de force d'un robot manipulateur est l'enveloppe des vecteurs forces possibles à l'effecteur pour des vecteurs d'entrée τ unitaires (typiquement des couples moteurs), voir **??**. L'enveloppe de forces possibles à l'effecteur, résultant d'un vecteur de couples des joints unitaire, peut être calculer en débutant avec une équation qui contraint la norme du vecteur de couple des joint et en substituant $\boldsymbol{\tau}$ par $J^T \boldsymbol{f}_E$ dans l'équation :

$$1 = \boldsymbol{\tau}^T \boldsymbol{\tau} \tag{5.31}$$

$$1 = (J^T \boldsymbol{f}_E)^T J^T \boldsymbol{f}_E \tag{5.32}$$

$$1 = \boldsymbol{f}_E^T (J J^T) \boldsymbol{f}_E \tag{5.33}$$

$$1 = \boldsymbol{f}_E^T A \boldsymbol{f}_E \qquad \text{avec} \qquad A = J J^T \tag{5.34}$$

On voit donc que c'est la même matrice JJ^T qui caractérise l'ellipsoïde de manipulabilité en force que celle qui caractérise l'ellispoïde de vitesse (voir section 4.7, mais ici c'est l'inverse de la matrice qui apparaît. Les axes principaux sont donc les mêmes mais les amplitudes sont inversés par rapport à l'ellispoïde. Par exemple, si le vecteur de vitesse possible à l'effecteur est faible dans une certaine direction, les forces externes possible à soutenir dans cette même direction vont être grandes. De plus, sur une singularité, il y a des directions pour lesquelles le vecteur vitesse possible est nul, ce qui va correspondre à des direction pour lesquelles le vecteur force possible va tendre vers l'infini, ce qui correspond à une direction pour laquelle une force externe va être reprise par la structure sans effort requis par les actionneurs.

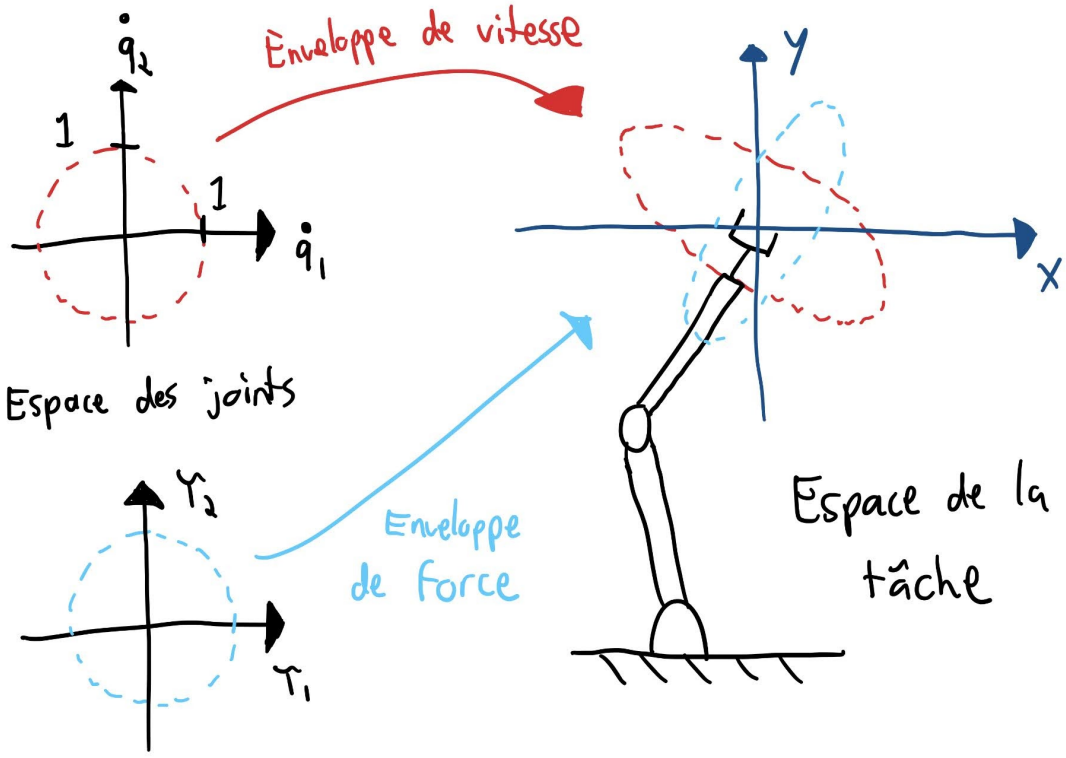


FIGURE 5.10 – Ellispoïde de manipulabilité pour la force et la vitesse

Exemple 5.2 Exemples d'ellispoïdes de manipulabilité pour un robot 2DDL:

La Figure 5.11 illustre un robot 2DDL planaire dans quelques configurations. Lorsque le robot est dans une configuration avec le coude partiellement plié les capacités du robot sont le plus isotropique.

Lorsque le coude est déplié le robot est capable de supporter des grande forces dans la direction radiale et l'inverse lorsque le coude est complètement plié.

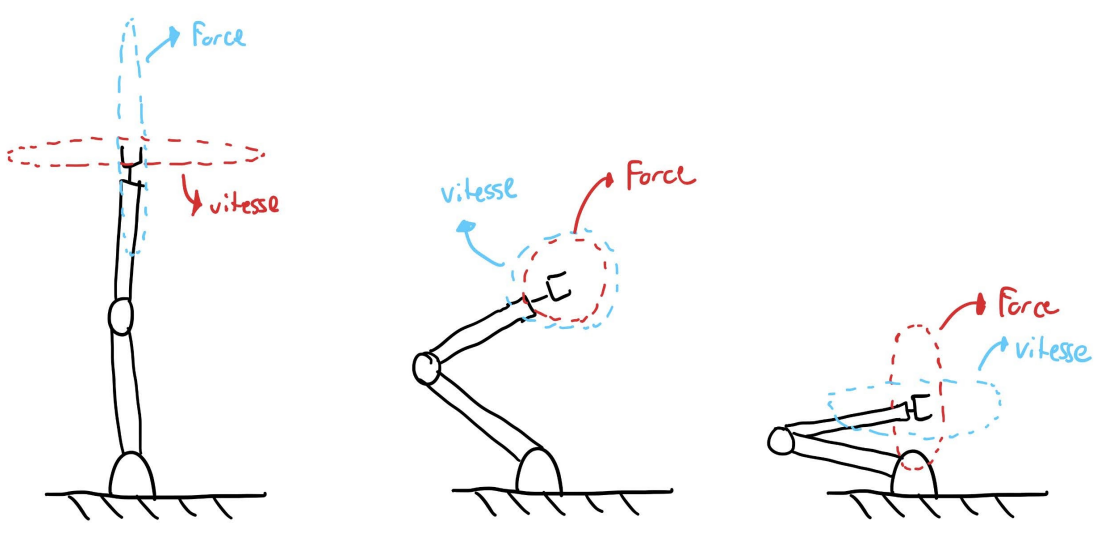


FIGURE 5.11 – Exemples d'ellispoïde de manipulabilité pour divers configration

Notre expérience humaine peut nous aider à avoir une intuition à savoir comment des propriétés peuvent être exploitées. Comme exercice de pensée je vous suggère de réfléchir à divers tâches du quotidien, la configuration de nos bras et jambe naturellement utilisé et les requis en force/vitesse : Comment configurez vous votre coude pour pousser contre une lourde porte ? Quelle est la configuration naturelle de votre bras pour écrire ? Comment configurez vous votre bras pour lancer une balle le plus loin possible ? Est-ce qu'il serait intéressant de marcher les genous pliés ?

5.7 Résumé du chapitre

Force externes à l'effecteur :

$$\boldsymbol{f}_{R_E} = -\boldsymbol{f}_E \tag{5.35}$$
$$\boldsymbol{f}_{R_E} = \text{Force appliquée sur l'effecteur du robot par l'environnement} \tag{5.36}$$
$$\boldsymbol{f}_E = \text{Force appliquée sur l'environnement par le robot} \tag{5.37}$$

Relation force joints-effecteur :

$$\boldsymbol{\tau} = J(\boldsymbol{q})^T \ \boldsymbol{f}_E \tag{5.38}$$

Équilibre statique :

$$\boldsymbol{\tau} = J(\boldsymbol{q})^T \ \boldsymbol{f}_E + g(\boldsymbol{q}) \qquad \text{avec : } g(\boldsymbol{q}) = \frac{\partial V(\boldsymbol{q})}{\partial \boldsymbol{q}} \tag{5.39}$$

Rigidité apparente à l'effecteur :

$$K_r = C_r^{-1} = \left[J K_q^{-1} J^T \right]^{-1} \tag{5.40}$$

Chapitre 6

Dynamique des robots manipulateurs

6.1 Introduction

Ce chapitre traite de la relation entre la l'évolution temporelle de la position d'un robot et les forces appliquées sur celui-ci. Notre outil mathématique principal pour décrire cette relation est un système d'équations différentielles qui relie des variables d'accélération du robot aux autres variables qui influence les forces internes et externes impliqués. Obtenir cette relation est utile lorsqu'on désire prédire des capacités dynamiques d'un système, simuler numériquement l'évolution temporelle d'un système dans des conditions connues ou bien synthétiser des lois de commandes.



Capsule vidéo
Dynamique des robots manipulateurs
`https://youtu.be/6z3grrVNBj4`

6.2 Structure des équations

La forme la plus générique (pour un système continu) des équations différentielles représentant l'évolution d'un système dynamique dans le temps est les équations d'états :

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}) \tag{6.1}$$

où $\boldsymbol{x}$ est un vecteur d'état, i.e. la mémoire ou les niveaux d'énergie du système, et $\boldsymbol{u}$ est un vecteur des entrées du système. C'est la représentation qui est utilisée par les simulateurs numériques pour calculer des trajectoires, pour générer des diagrammes de phases, et par plusieurs méthodes numérique pour générer des lois de commandes ou des trajectoires.

Lorsque le système dynamique à représenter est un système purement mécanique où les entrée sont des forces, comme c'est souvent le cas en robotique, il est possible d'utiliser une représentation plus spécifique avec une structure d'ordre 2 qui décrit la relation reliant les dérivées secondes des coordonnées (les accélérations) aux autres forces impliquées :

$$\underbrace{H(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}}_{m\,\vec{a}} = \underbrace{\boldsymbol{f}}_{\vec{f}} \tag{6.2}$$

où $\boldsymbol{q}$ est un vecteur de coordonnées généralisée (des positions linéaires ou angulaires qui caractérise la configuration du système mécanique), H est une matrice d'inertie, C est la matrice de coriolis et $\boldsymbol{f}$ un vecteur des forces impliquée dans le système.

Il est souvent utile de décomposer $\boldsymbol{f}$ ainsi pour représenter spécifiquement les types de forces impliquées

dans le système :

$$\underbrace{\underbrace{H(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}}_{\text{effets inertiels}}}_{m\,\vec{a}} + \underbrace{\underbrace{d(\dot{\boldsymbol{q}},\boldsymbol{q})}_{\text{forces dissipatrices}} + \underbrace{\boldsymbol{g}(\boldsymbol{q})}_{\text{forces conservatrices}} = \underbrace{B(\boldsymbol{q})\boldsymbol{u}}_{\text{effet des actionneurs}} + \underbrace{J^T(\boldsymbol{q})\boldsymbol{f}_{R_E}}_{\text{force externes sur l'effecteur}}}_{\vec{f}} \tag{6.3}$$

Cette équation est parfois appelé l'équation des manipulateurs et cette représentation est très utile pour analyser le comportement de système robotique et synthétiser des lois de commandes.

6.3 Variables, nomenclature et dimensions

Le tableau ci-dessous présente les variables qui sont utilisée dans ce chapitre.

TABLE 6.1 – Nomenclature pour la dynamique des manipulateurs

Dimensions			
n	:	nombre de DDL et de coordonnées généralisées	
m	:	nombre d'actionneur	
c	:	nombre de contraintes de contact	
o	:	nombre de coordonnée de l'espace de la tâche	
Scalaires			
T	:	Énergie cinétique	
V	:	Énergie potentielle	
Vecteurs			
$\boldsymbol{u}$	:	Forces/couples des actionneurs	m
$\boldsymbol{\tau}$	:	Forces des actionneurs transformées en coordonnées généralisées	n
$\boldsymbol{q}$	:	Coordonnées généralisée / espace des joints	n
$\boldsymbol{d}$	:	Somme de force dissipatrices dans l'espace des joints	n
$\boldsymbol{g}$	:	Somme des force conservatrices dans l'espace des joints	n
$\boldsymbol{h}$	:	Sommes des forces internes dans l'espace des joints	n
$\boldsymbol{\phi}_C$	:	Fonction de contraintes pour un point de contact C	c
$\boldsymbol{f}_{R_C}$	:	Forces cartésienne de contact appliquée au point C	c
$\boldsymbol{f}_{R_E}$	:	Forces cartésienne externe appliquée sur l'effecteur du robot	o
$\boldsymbol{r}$	:	Position de l'effecteur / coordonnées de l'espace de la tâche	o
Matrices			
H	:	Matrice d'inertie	n x n
C	:	Matrice de coriolis	n x n
B	:	Matrice des actionneurs	n x m
J	:	Matrice jacobienne de l'effecteur / espace de la tâche	o x n
J_C	:	Matrice jacobienne pour les coordonnées contraintes du point de contact C	c x n
J_i	:	Matrice jacobienne pour un point arbitraire i sur le robot	3 x n

6.4 Équation des manipulateurs

L'équation des manipulateur caractérise la dynamique d'un système robotique décrite dans l'espace des coordonnées des joints :

Définition 6.1 Équation des manipulateurs:

$$H(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + d(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = B(\boldsymbol{q})\boldsymbol{u} + J^T(\boldsymbol{q})\boldsymbol{f}_{R_E} \tag{6.4}$$

où $\boldsymbol{q}$ est le vecteur des forces généralisée, H est la matrice d'inertie, C est la matrice des forces de Coriolis/centrifuge, $\boldsymbol{d}$ est un vecteur de force dissipatrices, $\boldsymbol{g}$ un vecteur de forces conservatrices (typiquement la gravité), B est une matrice qui transforme le vecteur des forces des actionneurs $\boldsymbol{u}$ (les entrées du système) en forces généralisée $\boldsymbol{\tau}$, et J^T la transposée de la matrice jacobienne associée au point d'application E d'une force externe $\boldsymbol{f}_{R_E}$.

Exercice de code
Exemple interactif pour l'équation des manipulateurs
`https://colab.research.google.com/drive/1sGmE68lUHv6Y_xy5GMA_k9Qda7SPH4kD?usp=sharing`

6.4.1 Forces externes multiples

L'équation (6.4) est pour une situation ou une seule force externe est appliquée sur le robot, sur l'effectuer i.e. sur le point du robot dont la position est décrit par l'équation de la cinématique directe $\boldsymbol{r} = f(\boldsymbol{q})$. Si plusieurs forces externes sont appliquées à divers points sur le robot, il serait possible de décrire cette situation avec une sommation :

$$H(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + d(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = B(\boldsymbol{q})\boldsymbol{u} + \sum_i J_i^T(\boldsymbol{q})\boldsymbol{f}_{R_i} \tag{6.5}$$

où $\boldsymbol{f}_{R_i}$ est une force externe appliquée sur le robot à un point i, et J_i est une matrice jacobienne associé au mouvement du point i :

$$J_i(\boldsymbol{q}) = \frac{\partial \boldsymbol{r}_i(\boldsymbol{q})}{\partial \boldsymbol{q}} \tag{6.6}$$

où $\boldsymbol{r}_i(\boldsymbol{q})$ est la position du point i exprimée comme une fonction des coordonnées $\boldsymbol{q}$, i.e. la fonction de cinématique directe mais pour un point particulier.

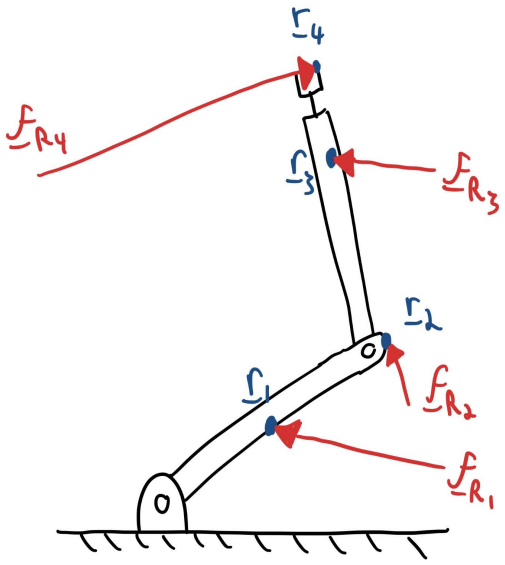


FIGURE 6.1 – Robot avec plusieurs forces externes

6.4.2 Matrice des actionneurs

La matrice $B(q)$ est une matrice de transformation qui relie le vecteur $\boldsymbol{u}$, représentants les entrées du système, et les forces généralisées $\boldsymbol{\tau}$ associées dans le systèmes de coordonnées $\boldsymbol{q}$ utilisé pour décrire les équations dans l'espace de la tache :

$$\boldsymbol{\tau} = B(\boldsymbol{q})\boldsymbol{u} \tag{6.7}$$

Note :

Dans le chapitre sur la statique (5), les divers équations et analyses étaient présentés en assumant qu'on nos actionneurs permettent de commander le vecteur $\boldsymbol{\tau}$ directement. Ici dans ce chapitre nous incluons une matrice de transformation B pour avoir une représentation plus englobante.

La Figure 6.2 illustre qualitativement divers architecture d'actionneurs qui mène à différentes structures pour B. Le cas le plus simple est lorsque les entrées τ sont des forces colocalisées avec les coordonnées généralisées $\boldsymbol{q}$, par exemple si $\boldsymbol{u}$ représente des couples net aux joint d'un système alors la matrice B est la matrice identité car les entrée $\boldsymbol{u}$ sont directement des forces généralisée aux joints $\boldsymbol{\tau}$. Si on désire que les entrée $\boldsymbol{u}$ soit directement les couples des moteurs électriques d'un robot alors la matrice B va représenter les ratios de réduction des actionneurs. Ensuite, si les actionneurs ne sont pas colocalisés avec les coordonnées généralisée, alors la matrice B fait le pont entre les forces aux actionneurs et les forces généralisées qu'ils génèrent. Finalement, si il y a moins d'actionneurs que de DDL alors la matrice B ne sera pas carrée.

La notion de puissance est utile pour définir cette matrice : le produit scalaire entre le vecteur vitesse des coordonnées généralisée $\dot{\boldsymbol{q}}$ et les forces généralisées associées aux actionneurs $B\boldsymbol{\tau}$ correspond à la puissance entrante fournit par les actionneurs :

$$P_{actionneur} = \dot{\boldsymbol{q}}^T\boldsymbol{\tau} = \dot{\boldsymbol{q}}^T B(\boldsymbol{q})\boldsymbol{u} \tag{6.8}$$

6.4.3 Forces conservatrices

Le vecteur de force conservatrice est défini comme le gradient de l'énergie potentiel par rapport au vecteur de coordonnées généralisée :

$$\boldsymbol{g}(\boldsymbol{q}) = \frac{\partial V(\boldsymbol{q})}{\partial \boldsymbol{q}^T} \tag{6.9}$$

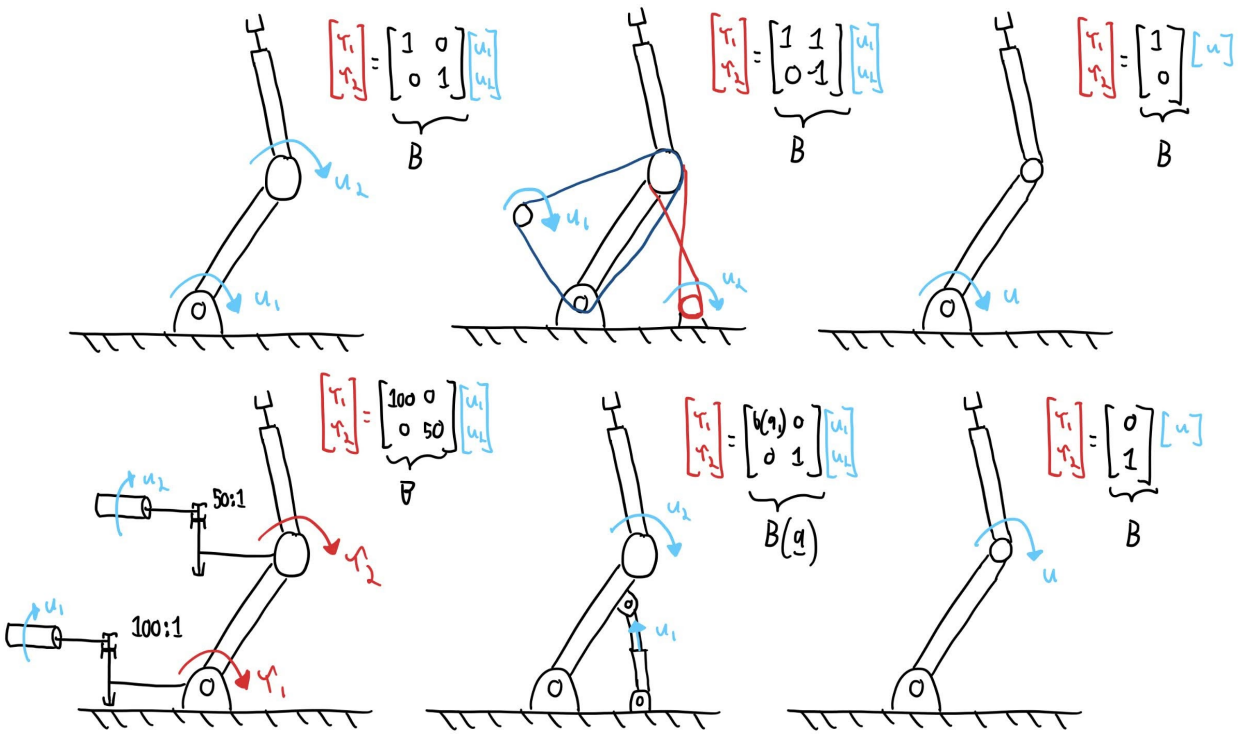


FIGURE 6.2 – Matrice B des actionneurs

de sorte que le taux de variation de l'énergie potentiel soit donnée par

$$\dot{V} = \frac{\partial V}{\partial \boldsymbol{q}} \dot{\boldsymbol{q}} = \boldsymbol{g}^T \dot{\boldsymbol{q}} \tag{6.10}$$

voir section 5.4 pour le détail.

Gravité : Dans ce chapitre ce vecteur est noté $\boldsymbol{g}$ car typiquement la seule énergie potentielle significative est l'énergie gravitationnelle et les forces conservatrices sont les forces gravitationnelles. L'énergie potentielle gravitationnelle peut être calculé comme la sommation de l'énergie potentielle de chacun des liens rigides du robot :

$$V_g(\boldsymbol{q}) = \sum_i m_i g h_i(\boldsymbol{q}) \tag{6.11}$$

où m_i est la masse du lien i, g la constante gravitationnelle et $h_i(\boldsymbol{q})$ est la hauteur du centre de gravité (c.g.) du lien i exprimée en fonction des coordonnées généralisées, comme illustré à la Figure 6.3. Il est à noter qu'on doit généralement introduire des paramètres géométriques additionnel pour décrire la position des c.g., ici notés avec l'indice c.

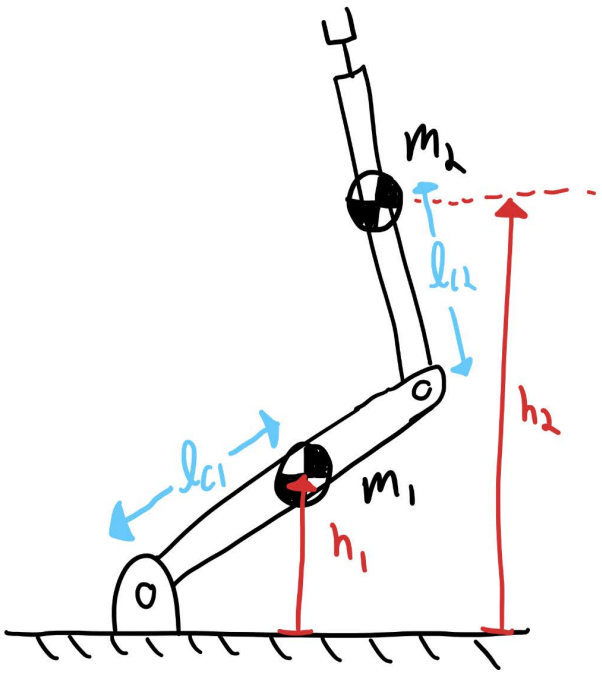


FIGURE 6.3 – La hauteur des centres de gravité des liens rigides

6.4.4 Forces dissipatrices

Les forces dissipatrices représentent les phénomènes qui transforment l'énergie du mouvement mécaniques en pertes thermiques et phénomènes irréversibles. Pour être incluse dans l'équation des manipulateurs, ces force doivent être exprimée comme des forces associées aux coordonnées généralisée de sorte que le puissance dissipée soit égale au produit scalaire entre le vecteur de ces forces et le vecteur vitesse :

$$\dot{Q}_{out} = \dot{\boldsymbol{q}}^T \boldsymbol{d} \tag{6.12}$$

Pour les robots manipulateurs, la principale source de dissipation vient des joints et des actionneurs, ce qui amène généralement à faire l'hypothèse que la force de dissipation d_i est seulement un fonction de la vitesse du joint associé $\dot{q}_i$:

$$d_i = f_d(\dot{q}_i) \approx b_s \operatorname{sgn}(\dot{q}_i) + b_v \dot{q}_i + \frac{1}{2}\rho C_d A \operatorname{sgn}(\dot{q}_i)\dot{q}_i^2 \tag{6.13}$$

Modéliser les phénomènes complexes de friction est une science imparfaite, c'est généralement l'aspect du modèle dynamique d'un robot qui est le moins précis et qui limite la fidélité du modèle. Selon l'objectif de modélisation il va y avoir un compromis à faire entre simplicité et fidélité. Dans un contexte de conception, analyse et/ou développement de loi de commande, on se contente normalement d'approximer la friction comme une combinaison de friction sèche, friction visceuse et friction quadratique, voir Figure 6.4, mais c'est une grosse approximation de phénomène microscopique de contact complexes. Comme illustré à la Figure 6.5, le comportement observé réel serait typiquement plus complexe et caractérisé par de l'hystérésis.

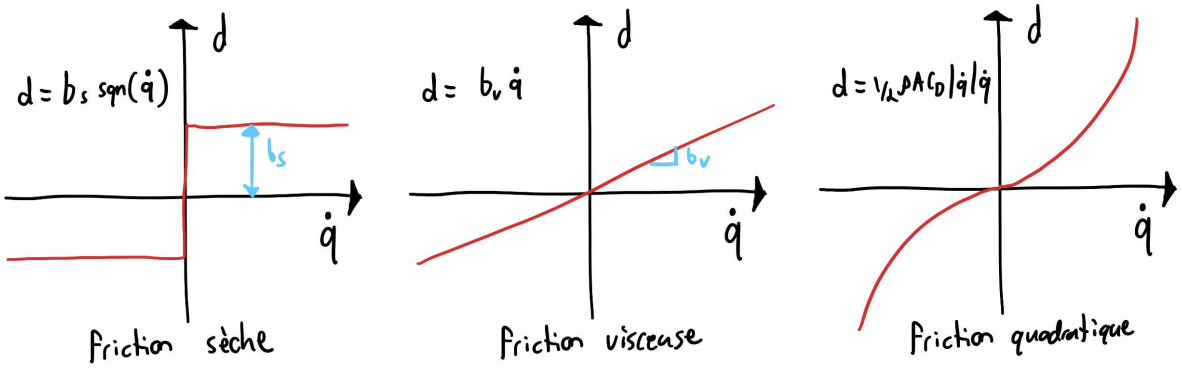


FIGURE 6.4 – Modèles simples de friction

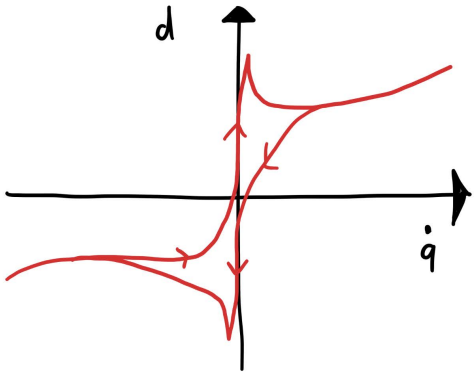


FIGURE 6.5 – Friction : comportement réel

6.4.5 Effets inertiels

Les termes $H(\boldsymbol{q})\ddot{\boldsymbol{q}}$ et $C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$ dans l'équation des manipulateur sont l'équivalent du terme ma dans l'équation $f = ma$ alors que tout les autres termes sont des forces. Ici le terme ma prend cette forme particulière car il est exprimé dans les coordonnées généralisée du système relié à l'espace des joints. Le terme $H(\boldsymbol{q})\ddot{\boldsymbol{q}}$ représente la "force" inertielle nécessaire pour obtenir une accélération $\ddot{\boldsymbol{q}}$ à un configuration $\boldsymbol{q}$, alors que le terme $C(\boldsymbol{q},\dot{\boldsymbol{q}})$ représente une "force" inertielle nécessaire pour maintenir une certaine vitesse $\dot{\boldsymbol{q}}$ à la configuration $\boldsymbol{q}$, comme l'effet centrifuge ou de coriolis. Le second terme existe seulement lorsque la matrice H varie dans le temps, lorsque de momentum peut varier sans avoir une variation de vitesse dans l'espace des joints.

Energie cinétique :

L'énergie cinétique T du système est directement reliée à la matrice d'inertie H et la vitesse des joints $\dot{\boldsymbol{q}}$. L'équivalent multi-dimension de l'équation de l'énergie cinémique scalaire $\frac{1}{2}mv^2$ est donné par :

$$T(\boldsymbol{q},\dot{\boldsymbol{q}}) = \frac{1}{2}\dot{\boldsymbol{q}}^T H(\boldsymbol{q})\dot{\boldsymbol{q}} = \frac{1}{2}\sum_i \sum_j H_{ij}\dot{q}_i\dot{q}_j \tag{6.14}$$

Propriété de la matrice H :

Il est a noté que la matrice d'inertie est une matrice n x n (n est le nombre de DDL), toujours symétrique et définie positive (ce qui veux dire que l'équation ci-dessus ne peut jamais prendre de

valeur négative peut importe le vecteur vitesse $\dot{\boldsymbol{q}}$). Cela implique que l'inverse H^{-1} va toujours exister, une propriété qui sera utile pour la suite !

La matrice C, souvent appelé la matrice de coriolis, n'est pas une propriété indépendante du système, elle est en fait reliée à la variation temporelle de la matrice H :

$$\dot{H} = C + C^T \tag{6.15}$$

On peut en fait aussi définir directement la matrice C connaissant l'expression de la matrice H en utilisant la relation suivante :

$$C_{ij} = \sum_k \Gamma_{ijk}\dot{q}_k \quad \text{avec} \quad \Gamma_{ijk} = \frac{1}{2}\left(\frac{\partial H_{ij}}{\partial q_k} + \frac{\partial H_{ik}}{\partial q_j} - \frac{\partial H_{jk}}{\partial q_i}\right) \tag{6.16}$$

Si on définie $\boldsymbol{c}$ comme le vecteur résultant du terme $C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}$, on peut aussi obtenir la relation sous cette forme :

$$c_i = \sum_j \sum_k \Gamma_{ijk}\dot{q}_j\dot{q}_k \tag{6.17}$$

$$\boldsymbol{c} = C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \dot{\boldsymbol{q}}^T\Gamma(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{6.18}$$

Démonstration. À venir ! ! voir photo ipad alex 26 janvier 2023 □

6.4.6 Forme compacte des équations des manipulateurs

Parfois pour allégé la manipulation des équations, une version abrégée où $\boldsymbol{h}$ est parfois utilisé pour regrouper toute les forces internes (qui sont des fonctions des états du systèmes) :

$$\boldsymbol{h} = C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + d(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) \tag{6.19}$$

ce qui simplifie, pour un cas sans forces externes, l'équation des manipulateurs à :

$$H\ddot{\boldsymbol{q}} + \boldsymbol{h} = \boldsymbol{\tau} \tag{6.20}$$

6.4.7 Systèmes de coordonnées additionnels

TODO : Uniformiser le nom des variables pour cette section.

Parfois il peut être utile de travailler avec des systèmes de coordonnées supplémentaire à l'espace des joints. La Figure 6.6 montre quelques espaces supplémentaires qui peuvent être utilisés.

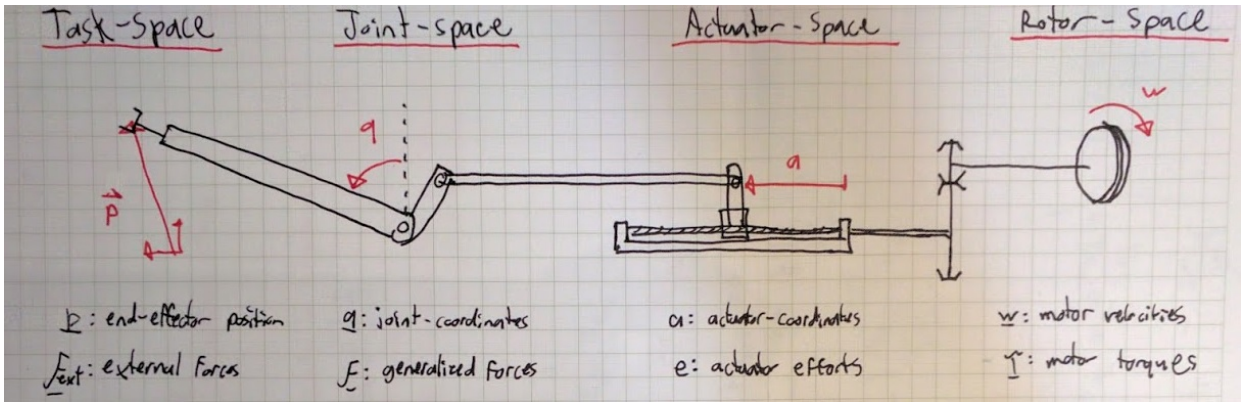


FIGURE 6.6 – Coordinate systems

Si les relations cinématiques entre les coordonnées de ces espaces sont donné par :

$$\dot{\boldsymbol{r}} = J_e(\boldsymbol{q})\dot{\boldsymbol{q}} \quad \text{de l'espace des joints vers l'espace de la tâche} \tag{6.21}$$

$$\dot{\boldsymbol{a}} = J_a(\boldsymbol{q})\dot{\boldsymbol{q}} \quad \text{de l'espace des joints vers l'espace des actionneurs} \tag{6.22}$$

$$\boldsymbol{w} = R\dot{\boldsymbol{a}} \quad \text{de l'espace des actionneurs vers l'espace des moteurs} \tag{6.23}$$

On pourrait utiliser ces relations pour inclure directement des forces dans l'espace des actionneurs $\boldsymbol{f}_a$, et définir $\boldsymbol{\tau}$ comme les couples moteurs plutôt celui rapporté au joint ainsi :

$$H\ddot{\boldsymbol{q}} + C\dot{\boldsymbol{q}} + \boldsymbol{d} + \boldsymbol{g} = \underbrace{J_a^T(\boldsymbol{q})R^T}_{B(\boldsymbol{q})}\boldsymbol{u} + J_e^T(\boldsymbol{q})\boldsymbol{f}_e + J_a^T(\boldsymbol{q})\boldsymbol{f}_a \tag{6.24}$$

6.5 Conservation de l'énergie

Si on reprend le bilan d'énergie avec le volume de contrôle de la figure 5.1, mais cette fois çi en incluant aussi l'énergie cinétique dans l'énergie interne E et les pertes thermiques dues aux forces dissipatrices $\dot{Q}$, on trouve avec la première loi de la thermodynamique :

$$\frac{dE}{dt} = P_{in} - P_{out} \tag{6.25}$$

$$\dot{T} + \dot{V} = \sum \tau_i \dot{q}_i - \sum f_i \dot{r}_i - \dot{Q} \tag{6.26}$$

$$\frac{d}{dt}\left(\frac{1}{2}\dot{\boldsymbol{q}}^T H \dot{\boldsymbol{q}}\right) + \frac{\partial V}{\partial \boldsymbol{q}^T}\dot{\boldsymbol{q}} = \boldsymbol{\tau}^T \dot{\boldsymbol{q}} - \boldsymbol{f}_E^T J \dot{\boldsymbol{q}} - \boldsymbol{d}^T \dot{\boldsymbol{q}} \tag{6.27}$$

$$\dot{\boldsymbol{q}}^T H \ddot{\boldsymbol{q}} + \dot{\boldsymbol{q}}^T \frac{1}{2}\dot{H}\dot{\boldsymbol{q}} + \dot{\boldsymbol{q}}^T \boldsymbol{g} = \dot{\boldsymbol{q}}^T B \boldsymbol{u} + \dot{\boldsymbol{q}}^T J^T \boldsymbol{f}_{R_E} - \dot{\boldsymbol{q}}^T \boldsymbol{d} \tag{6.28}$$

On peut donc voir toute l'expression comme un produit scalaire entre le vecteur et les termes restant qui peuvent être simplifier si on substitue par notre définition de l'équation des manipulateurs. Les seules termes qui ne s'annulent pas sont $\dot{H}$ et C, ce qui mène à une relation entre les deux :

$$\dot{\boldsymbol{q}}^T \left[H\ddot{\boldsymbol{q}} + \frac{1}{2}\dot{H}\dot{\boldsymbol{q}} + \boldsymbol{g} + \boldsymbol{d} - \boldsymbol{\tau} - J^T \boldsymbol{f}_E\right] = 0 \tag{6.29}$$

$$\dot{\boldsymbol{q}}^T \left[\frac{1}{2}\dot{H} - C\right]\dot{\boldsymbol{q}} = 0 \tag{6.30}$$

Cette expression va toujours être égale à zéro si $\dot{H} = C + C^T$, une condition qui correspond donc d'un certain sens à une condition pour la conservation de l'énergie pour le système.

Matrice anti-symétrique

Une matrice est dit anti-symétrique, *skew-symetric* en anglais, si ...

6.6 Dynamique inverse

L'équation des manipulateurs détermine des forces généralisée $\boldsymbol{\tau}$ pour une configuration $\boldsymbol{q}$, une vitesse $\dot{\boldsymbol{q}}$ et une accélération $\ddot{\boldsymbol{q}}$. La dynamique inverse représente la même équation mais lorsqu'on isole le vecteur accélération comme une fonction de toute les forces internes et externes, c'est en fait le seul normal de la causalité. En version compacte on peut retrouver :

$$\ddot{\boldsymbol{q}} = H^{-1}\left(\boldsymbol{\tau} - \boldsymbol{h}\right) \tag{6.31}$$

ou en version longue :

$$\ddot{\boldsymbol{q}} = H^{-1}(\boldsymbol{q})\left(B(\boldsymbol{q})\boldsymbol{u} + J^T(\boldsymbol{q})\boldsymbol{f}_{R_E} - C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - d(\boldsymbol{q},\dot{\boldsymbol{q}}) - \boldsymbol{g}(\boldsymbol{q})\right) \tag{6.32}$$

Comme la matrice d'inertie H est symétrique et définie positive, sont inverse existe toujours et donc cette équation a toujours une solution.

6.7 Systèmes sous-actionnés vs. complètement actionnés

Une caractéristique importante pour déterminer les capabilités physique d'un robot manipulateur est de déterminer si le système est pleinement ou sous-actionné. Un système pleinement actionné a la capacité (théorique) d'imposer un vecteur d'accélération $\ddot{\boldsymbol{q}}$ arbitraire si on suppose que le vecteur d'entrée $\boldsymbol{u}$ peut prendre des valeurs arbitraires (pas de maximum ou minimum). La condition mathématique pour être pleinement actionné est que le rang (rangé si la matrice n'est pas carrée) de B doit être égal à n, le nombre de DDL. On dira alors que le système est sous-actionné si

$$rang\left(B(\boldsymbol{q})\right) < n = dim(\boldsymbol{q}) \tag{6.33}$$

6.8 Manipulabilité dynamique

À la section 4.7, la notion de manipulabilité définissait une enveloppe de vitesse possible à l'effecteur. Il est aussi possible d'analyser l'enveloppe d'accélérations cartésiennes possibles à l'effecteur pour un vecteur de couple des actionneurs unitaire.

Détails à venir !

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

6.9 Dérivation des équations avec la méthode de Lagrange

Pour des systèmes robotiques et mécanismes articulés relativement simpleS, la méthode de Lagrange est un outil bien adapté pour calculé rapidement analytiquement les équations du mouvement. Pour utiliser cette méthode, il suffit de déterminer les expressions pour l'énergie cinétique et potentielle en fonction des coordonnées généralisée. :

$$\mathcal{L} = T - V = \underbrace{T(\boldsymbol{q}, \dot{\boldsymbol{q}})}_{\text{Énergie cinétique}} - \underbrace{V(\boldsymbol{q})}_{\text{Énergie potentielle}} \tag{6.34}$$

Ensuite il suffit d'appliquer une "recette de cuisine" qui consiste a effectuer des dérivées pour obtenir les équations de la dynamique. Chaque ligne i de l'équation des manipulateurs peut être déterminée avec l'expression suivante :

$$\forall i \qquad \frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}_i}\right) - \frac{\partial \mathcal{L}}{\partial q_i} = \sum \tau_i \tag{6.35}$$

qui est aussi équivalente à :

$$\forall i \qquad \underbrace{\frac{d}{dt}(\frac{\partial T}{\partial \dot{q}_i}) - \frac{\partial T}{\partial q_i}}_{\text{Effets inertiels}} + \underbrace{\frac{\partial V}{\partial q_i}}_{\text{Force conservatrices}} = \underbrace{\sum \tau_i}_{\text{Forces dissipatrices, externes et actionneurs}} \tag{6.36}$$

On voit donc que cette méthode est surtout intéressante pour déterminer les termes inertiels et les forces conservatives, est ne guide pas vraiment la détermination des autres forces ni leur expression comme des forces généralisées. L'avantage de cette méthode est qu'il est possible d'ignorer les forces de contraintes dans la démarche, versus la méthode classique qui demanderait de débuter par des diagrammes de corps libres (DCL), appliqué $f = ma$ sur tout les axes et faire un processus d'élimination des forces de contrainte.

Capsule vidéo
Méthode de Lagrange pour déterminer les équations dynamiques
`https://youtu.be/AUcs0L85liM`

Note :

Il est a noté que la méthode de Lagrange n'est pas un type de modèle mais bien une méthodologie, i.e. une recette de cuisine pour déterminer les équations dynamiques d'un système de corps rigides. En utilisant la méthode classique (DCL + lois de Newtons) ou n'importe quelle autre méthode, si on part des mêmes hypothèses de modélisation on va trouver exactement les mêmes équations.

6.10 Équations dans l'espace de la tâche

En dérivant la relation de cinématique différentielle il est possible d'obtenir des relations entre les vitesses et accélérations :

$$\dot{\boldsymbol{r}} = J\dot{\boldsymbol{q}} \tag{6.37}$$

$$\ddot{\boldsymbol{r}} = J\ddot{\boldsymbol{q}} + \dot{J}\dot{\boldsymbol{q}} \tag{6.38}$$

Ensuite si on utilise ces équations pour remplacer les variables de l'espace des joints par ceux dans l'espace de la tâche dans l'équation des manipulateurs :

$$H\ddot{\boldsymbol{q}} + \boldsymbol{h} = \boldsymbol{\tau} + J^T \boldsymbol{f}_{R_E} \tag{6.39}$$

$$HJ^{-1}(\ddot{\boldsymbol{r}} - \dot{J}\dot{\boldsymbol{q}}) + \boldsymbol{h} = \boldsymbol{\tau} + J^T \boldsymbol{f}_{R_E} \tag{6.40}$$

et qu'on multiplie ensuite par la droite par l'inverse du jacobien transposé ($J^{-T} = (J^T)^{-1}$), nous obtenons :

$$J^{-T}HJ^{-1}\ddot{\boldsymbol{r}} - J^{-T}HJ^{-1}\dot{J}\dot{\boldsymbol{q}} + J^{-T}\boldsymbol{h} = J^{-T}B\boldsymbol{u} + J^{-T}J^T\boldsymbol{f}_{R_E} \tag{6.41}$$

$$\underbrace{J^{-T}HJ^{-1}}_{H^r}\ddot{\boldsymbol{r}} + \underbrace{J^{-T}\boldsymbol{h} - J^{-T}HJ^{-1}\dot{J}J^{-1}\dot{\boldsymbol{r}}}_{\boldsymbol{h}^r} = \underbrace{J^{-T}B}_{B^r}\boldsymbol{u} + \boldsymbol{f}_{R_E} \tag{6.42}$$

Nous pouvons ensuite réorganiser les termes pour obtenir une nouvelle équation, qui a la même forme originale que l'équation des manipulateurs dans l'espace des joints, mais avec tout les termes par rapport aux coordonnées de la tâche :

$$H^r\ddot{\boldsymbol{r}} + \boldsymbol{h}^r = B^r\boldsymbol{u} + \boldsymbol{f}_{R_E} \tag{6.43}$$

où

$$H^r = J^{-T}HJ^{-1} \tag{6.44}$$

$$\boldsymbol{h}^r = J^{-T}\boldsymbol{h} - H^r\dot{J}J^{-1}\dot{\boldsymbol{r}} \tag{6.45}$$

$$B^r = J^{-T}B \tag{6.46}$$

Cette équation représente la même dynamique mais exprimée avec les coordonnées de l'espace de la tâche directement. Il est à noter que cette transformation de coordonnées n'est pas possible sur une singularité. Si l'espace de la tâche représente des coordonnées cartésienne de l'outil du robot, alors la matrice H^r représente l'inertie ressentie lorsqu'on pousse sur l'effecteur. Le terme H^r_{11} est la masse ressentie dans la direction de l'axe 1.

6.11 Manipulateur en contact avec l'environnement

Cette section présente les outils pour obtenir les équations du mouvement lorsqu'un système robotique est en contact avec l'environnement, ce qui contraint sont mouvement.

6.11.1 Contraintes cinématique

Si un robot manipulateur est en contact avec un objet fixe, il pert certains DDL. Dans le cas de contraintes bilatérales, la contrainte peut être décrite par une fonction :

$$\boldsymbol{\phi}(\boldsymbol{q}) = 0 \tag{6.47}$$

Si on dérive cette fonction par rapport au temps, il est possible d'obtenir des conditions pour la vitesse et l'accélération du système :

$$\frac{d\boldsymbol{\phi}(\boldsymbol{q})}{dt} = J_C(\boldsymbol{q})\dot{\boldsymbol{q}} = 0 \tag{6.48}$$

$$\frac{d^2\boldsymbol{\phi}(\boldsymbol{q})}{dt^2} = J_C(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{J}_C(\boldsymbol{q})\dot{\boldsymbol{q}} = 0 \tag{6.49}$$

où J_C est le jacobien des contraintes :

$$J_C(\boldsymbol{q}) = \frac{d\boldsymbol{\phi}(\boldsymbol{q})}{d\boldsymbol{q}} \tag{6.50}$$

6.11.2 Forces de contraintes

Le jacobien des contraintes peut être utilisé pour transformer des forces de contact cartésiennes $\boldsymbol{f}_C$ en force généralisée au joint dans l'équation des manipulateurs :

$$H\ddot{\boldsymbol{q}} + \boldsymbol{h} = B\boldsymbol{\tau} + J_C(\boldsymbol{q})^T\boldsymbol{f}_C \tag{6.51}$$

Si on isole $\ddot{\boldsymbol{q}}$ dans l'équation (6.51) pour ensuite substituer dans l'équation (6.49), on retrouve une expression des forces de contraintes qui dépend des états actuel du système et des forces/couples aux actionneurs :

$$\boldsymbol{f}_C = \left(J_C H^{-1} J_C^T\right)^{-1}\left(J_C H^{-1}[\boldsymbol{h} - B\boldsymbol{\tau}] - \dot{J}_C(\boldsymbol{q})\dot{\boldsymbol{q}}\right) \tag{6.52}$$

Alternativement il est possible, pour trouver l'accélération $\ddot{\boldsymbol{q}}$ et les forces de contact $\boldsymbol{f}_C$ simultanément, de résoudre le système d'équation suivant :

$$\begin{bmatrix} H & -J_C^T \\ J_C & 0 \end{bmatrix}\begin{bmatrix} \ddot{\boldsymbol{q}} \\ \boldsymbol{f}_C \end{bmatrix} = \begin{bmatrix} B\boldsymbol{\tau} - \boldsymbol{h} \\ -\dot{J}_C\dot{\boldsymbol{q}} \end{bmatrix} \tag{6.53}$$

6.11.3 Impulsion lors d'un impact

Quand le robot rentre en contact avec un objet fixe très rigide (frapper le sol par exemple), des forces de contact impulsives vont agir sur le système. Si on intègre l'équation (6.51) sur une période infinitésimal de temps dt donne :

$$\int (H\ddot{\boldsymbol{q}} + \boldsymbol{h})dt = \int (B\boldsymbol{\tau} + J_C(\boldsymbol{q})^T\boldsymbol{f}_C)dt \tag{6.54}$$

$$H\dot{\boldsymbol{q}}^+ - H\dot{\boldsymbol{q}}^- = J_C(\boldsymbol{q})^T\int \boldsymbol{f}_C dt \tag{6.55}$$

où la contribution des forces non-impulsive est négligée. Si on projette ces équations sur les coordonnées contraintes en multipliant par $J_C H^{-1}$ on trouve :

$$J_C\dot{\boldsymbol{q}}^+ - J_C\dot{\boldsymbol{q}}^- = J_C H^{-1} J_C^T\int \boldsymbol{f}_C dt \tag{6.56}$$

Si on fait l'hypothèse d'une collision complètement inélastique (sans rebond), la contrainte est respectée à l'instant t^+ qui suit immédiatement l'impact. Dons comme $J_C\dot{\boldsymbol{q}}^+ = 0$, il est ensuite possible de résoudre pour obtenir l'impulsion due au contact dans ces conditions :

$$\int \boldsymbol{f}_C dt = -\left(J_C H^{-1} J_C^T\right)^{-1} J_C \dot{\boldsymbol{q}}^- \tag{6.57}$$

et pour la vitesse des joints qui va suivre immédiatement l'impact :

$$\dot{\boldsymbol{q}}^+ = -\Big[I - H^{-1} J_C^T \left(J_C H^{-1} J_C^T\right)^{-1} J_C\Big]\dot{\boldsymbol{q}}^- \tag{6.58}$$

ou pour la variation de vitesse durant l'impulsion :

$$\Delta\dot{\boldsymbol{q}} = \Big[H^{-1} J_C^T \left(J_C H^{-1} J_C^T\right)^{-1} J_C\Big]\dot{\boldsymbol{q}}^- \tag{6.59}$$

On peut alternativement résoudre le système de $n + c$ équations suivant pour obtenir ces résultats simultanément :

$$\begin{bmatrix} H & -J_C^T \\ J_C & 0 \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}^+ \\ \int \boldsymbol{f}_C dt \end{bmatrix} = \begin{bmatrix} H\dot{\boldsymbol{q}}^- \\ 0 \end{bmatrix} \tag{6.60}$$

6.12 Dynamique des actionneurs et effet des ratios de réduction

Jusqu'à maintenant les équations étaient développées en supposant que l'entrée sur le système robotique était une source de force pure. La plupart des robots sont actionnés par des moteurs électriques, pour lesquels le couple appliqué par le moteur est relativement proportionnel au courant qui circule dans le moteur qu'il est possible d'asservir avec de l'électronique de puissance adapté. Toutefois, l'inertie et la friction interne de ces moteurs va avoir un gros impact sur la dynamique des robots lorsque de grands ratios de réductions sont utilisés comme c'est le cas pour les manipulateurs industriels.

Pour illustré ce phénomène, débutons avec un modèle simple de dynamique de moteur qui aurait une inertie I, de la friction visceuse b, un couple transmis net τ_{net} et comme entrée un couple électromagnétique τ_{mag}. La dynamique serait donnée par :

$$I\dot{w} + bw = \tau_{mag} - \tau_{net} \tag{6.61}$$

Si un ratio de réduction R est utilisé entre le moteur et le joint du robot, alors le couplage entre le bras robotique et le moteur est donné par ces deux équations :

$$\tau = R\tau_{net} \tag{6.62}$$

$$w = R\dot{q} \tag{6.63}$$

détails à venir !

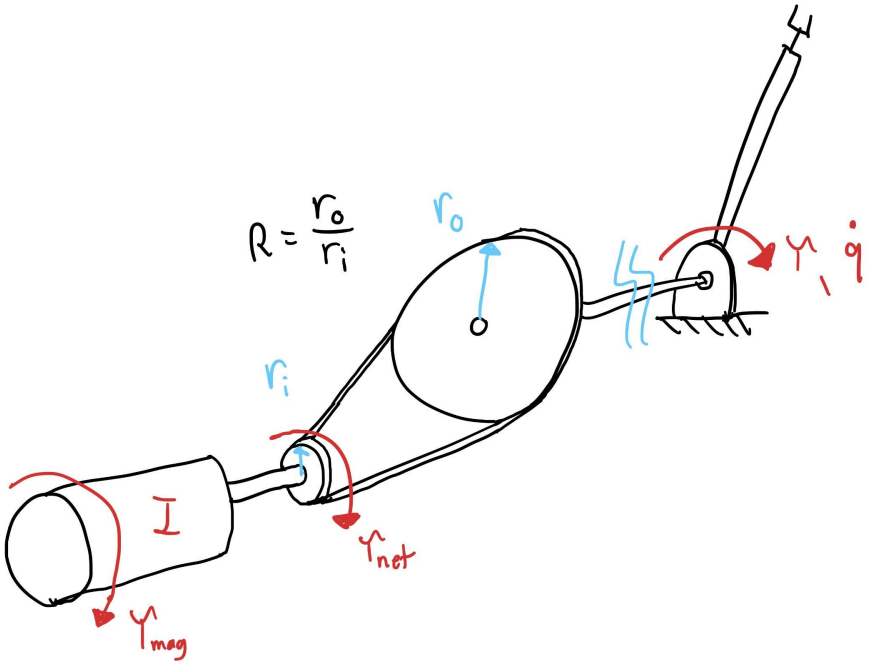


FIGURE 6.7 – Dynamique d'un moteur électrique couplé avec un joint robotique

Chapitre 7

Introduction à la commande des systèmes robotisés

La science de *la commande* traite du développement d'algorithmes qui déterminent les actions qu'un système robotisé doit prendre en fonction de la situation pour atteindre un objectif (par exemple une position cible). Ces algorithmes sont généralement implémentés sur un micro-contrôleur, qui choisit les actions des actionneurs (par exemple le couple des moteurs électriques). Mathématiquement, le problème consiste à choisir les termes qui représentent les actions possible dans un système d'équations différentielles qui représentent le comportement du robot, afin d'obtenir le comportement désiré.

Lors de la conception des lois de commandes, plusieurs aspects (illustrés qualitativement à la Figure 7.1) peuvent être analysés selon le contexte :

La stabilité Lorsqu'une loi de commande est proposée, un critère essentiel dans la plupart des situations est la stabilité du système. La stabilité caractérise si les états du systèmes converge vers les états désirés ou bien si ils divergent.

La faisabilité Dans certain contextes, principalement ceux ou les actions possibles sont limités (ex. couple maximum des moteurs), on s'intéresse tout simplement à savoir si il existe une séquence d'actions possibles qui peuvent permettre au système d'atteindre l'objectif.

Le temps de réponse La stabilité et la faisabilité détermine si le système pourra atteindre l'objectif. Ensuite, on cherche généralement à atteindre l'objectif le plus rapidement possible ce qui est caractérisé par le temps de réponse. Un critère relié est la bande passante, la rapidité du système exprimée dans le domaine fréquentiel.

L'optimalité Il y a généralement un compromis à faire entre la vitesse (le temps de réponse) et l'utilisation agressive des actions possibles du système qui ont généralement un coût énergétique. Un objectif commun lors de la conception d'un asservissement est d'optimiser la performance, ce qui est généralement formulé comme la minimisation d'un coût qui combine les dépenses énergétiques et le temps passé loin de l'objectif.

La robustesse Finalement, dans plusieurs situations il faut considérer l'incertitude dans le système lors de la conception des lois de commande. L'incertitude peut être due à des perturbations externes inconnues (ex. : rafales de vent pour un avion) ou bien à des paramètres incertains du système robotique (ex. : poids de la charge transportée pour un bras manipulateur). L'analyse de robustesse consiste à vérifier si les lois de commandes vont fonctionner dans divers scénarios possibles. Des méthodes existe pour garantir le fonctionnement d'un contrôleur malgré un certain niveau d'incertitude (commande robuste ou commande adaptative).

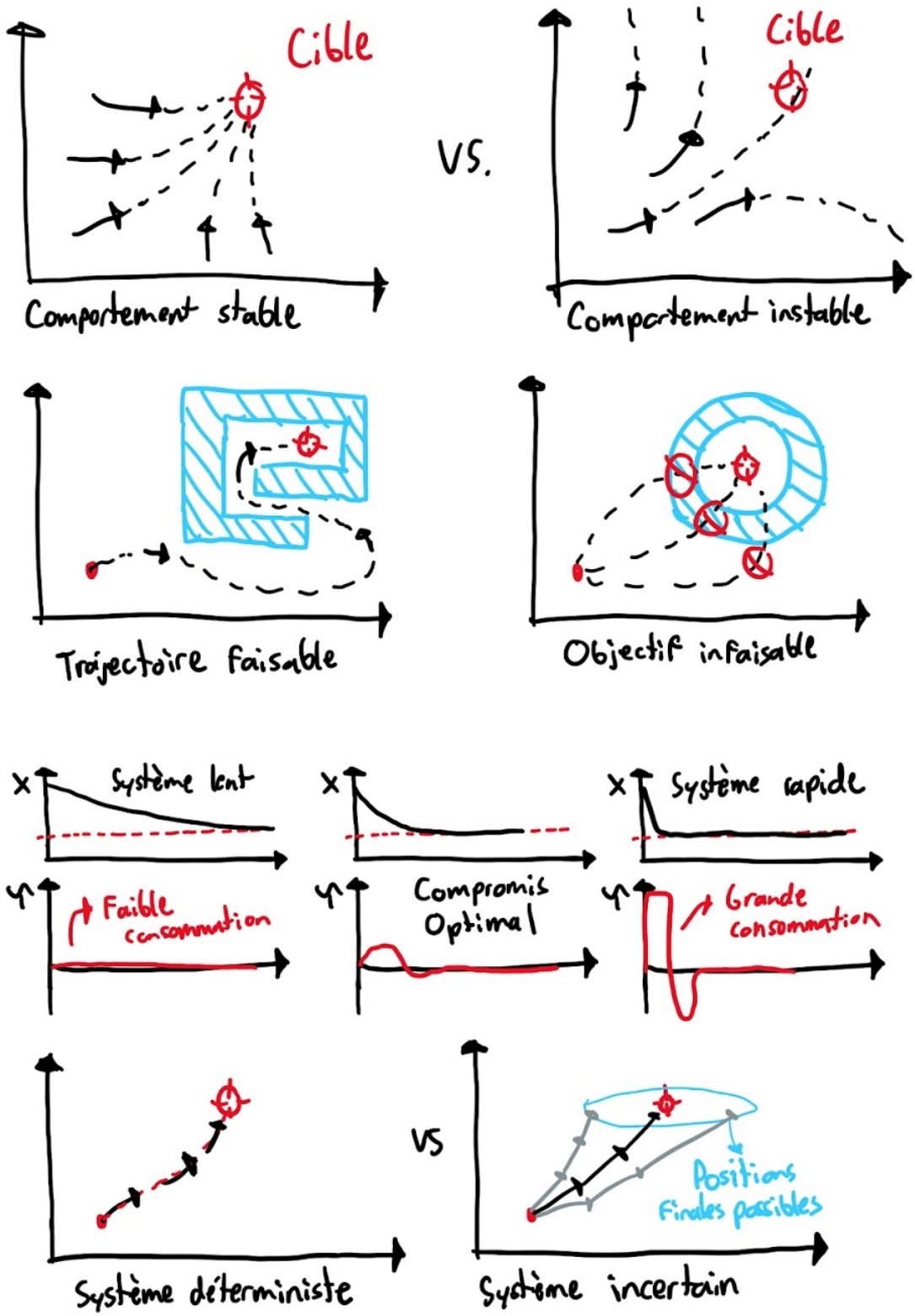


FIGURE 7.1 – Différents aspects qui peuvent être considéré lors de la conception de loi de commandes

7.1 Architectures

Cette section introduits aux grandes catégories d'architecture de commande pour les systèmes robotisées.

7.1.1 Boucle ouverte vs. boucle fermée

La méthode de commande la plus simple est appelée boucle ouverte, voir Figure 7.2. Par exemple, on détermine une séquence d'action pour les actionneurs (plan) qui va théoriquement faire bouger un système robotisé d'un point A au point B (objectif). Dans ce cas, la loi de commande est seulement une fonction du temps qui envoi les consignes pré-déterminés aux actionneurs.

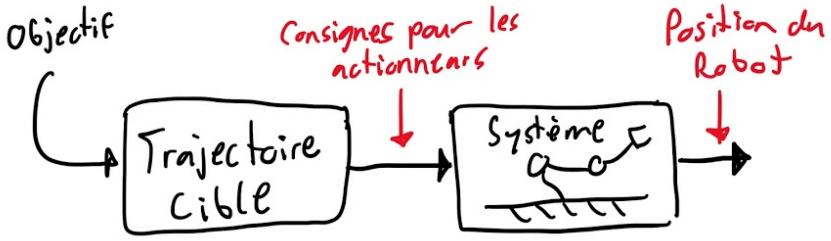


FIGURE 7.2 – Boucle ouverte

En général, cette méthode que l'on pourrait qualifier "d'aveugle" fonctionne seulement lorsque les systèmes ont très peu d'incertitude. Par exemple, une machine outil qui lit un code-G et traduit directement les instructions en consignes pour ses moteurs pas à pas est une approche purement boucle ouverte. Par contre, on peut facilement s'imaginer que cette approche ne fonctionnerait pas pour une voiture autonome qui doit aller d'une ville à une autre. Un voiture automne qui utiliserait un contrôleur basé sur une séquence d'angle de volant pré-déterminée divergerait rapidement de la trajectoire désirée du aux incertitudes.

La plupart des contrôleurs utilisent une rétroaction continue basé sur des capteurs pour vérifier si le système évolue de façon approprié. C'est ce qu'on appelle l'approche boucle fermée, illustré à la Figure 7.3. Dans ce cas, la loi de commande est une généralement fonction d'une erreur qui résulte de la comparaison de la position cible et la position réelle mesurée par des capteurs.

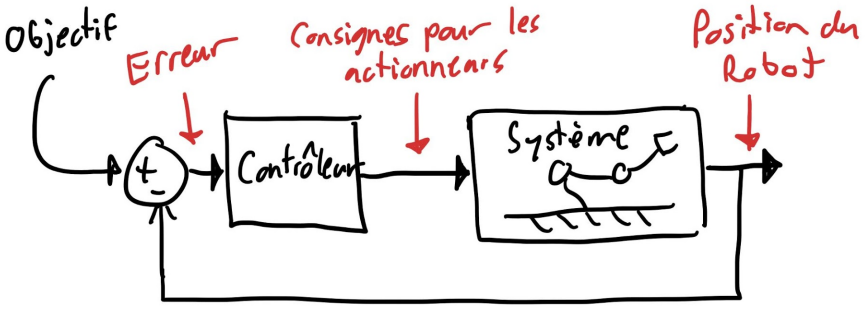


FIGURE 7.3 – Boucle fermée

Le principe de base d'une telle loi de commande peut se résumer à : si le système est trop à gauche pousse vers la droite et si le système est trop à droite pousse vers la gauche. Cette approche fonctionne bien pour des systèmes simples. Par exemple, pour contrôler la position d'un vérin électrique une loi de commande de type "pousse dans le sens inverse de l'erreur" est suffisante. Toutefois pour une voiture autonome, si la direction et la vitesse à prendre serait basée sur l'erreur totale, par exemple 150 km NW par rapport à la ville que l'on désire aller, la voiture tenterais de traverser des champs en ligne droite ! Pour les systèmes plus compliqués, particulièrement avec des limites comme des obstacles, des saturations d'actionneurs, etc, le contrôleur n'est pas simplement une fonction de l'erreur.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Pour plusieurs systèmes robotiques, une loi de commande hybride est utilisée comme illustré à la Figure 7.4. Un plan est déterminé comme avec l'approche en boucle ouverte, toutefois un contrôleur en boucle fermée qui agit sur l'erreur par rapport à la trajectoire cible est ensuite utilisé. La loi de commande dans ce cas, ne dépend pas seulement de la position actuel et de la position désirée, mais aussi du temps car elle compare la position actuelle à celle ou le robot doit théorique être rendu à ce moment.

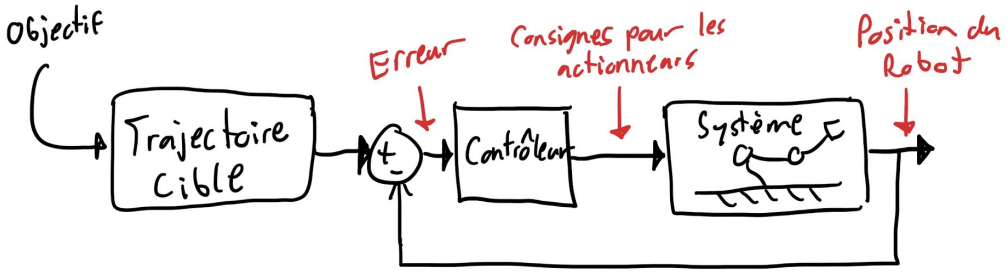


FIGURE 7.4 – Contrôleur de suivi de trajectoire

7.1.2 Boucles imbriquées

La plupart des systèmes robotiques n'ont pas un seul gros contrôleur qui contrôle toutes les actions des actionneurs de façon centralisé, mais plutôt plusieurs sous-systèmes avec chacun des contrôleur locaux. Par exemple, lorsqu'un système robotisé utilise des servo-moteurs, l'électronique des moteurs inclus déjà un contrôleur qui effectue un asservissement en position ou en vitesse.

Une architecture qui est souvent utilisée pour les robots manipulateur est illustrée à la Figure 7.5. Chaque joint du robot du robot a sont propre contrôleur en vitesse qui recoit une consigne d'un contrôleur central. Les contrôleurs locaux comparent la vitesse du joint désirée à une mesure de vitesse et détermine un couple moteur selon l'erreur en vitesse. Le contrôleur central lui compare la position cible à la position mesurée du robot et détermine le mouvement (consignes de vitesse) que les joints devraient faire pour corriger la position.

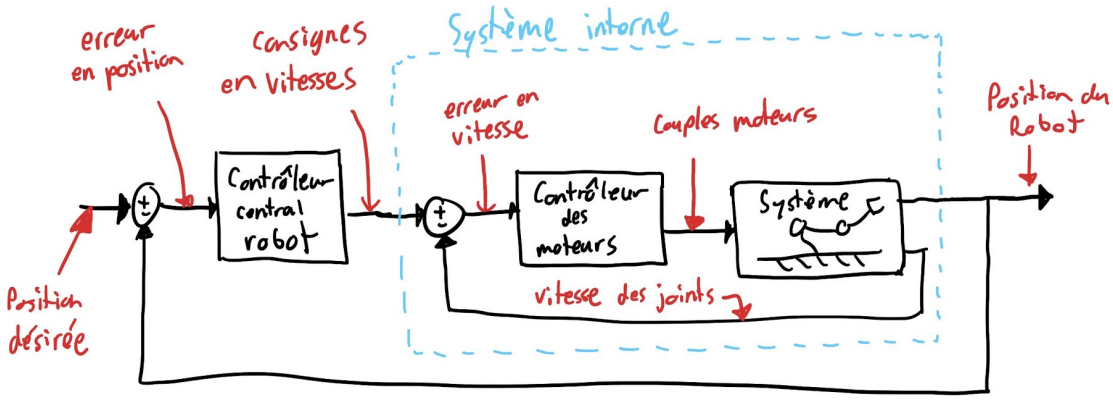


FIGURE 7.5 – Boucles imbriquées

Cette approche a comme avantage de découpler le problème en deux problèmes plus simples et rendre plus modulaires les loi de commande qui sont développées. Par exemple, le concepteur du contrôleur des moteurs pourrait effectuer sont travail sans ce soucier du modèle cinématique compliqué du bras manipulateur. Inversement, le concepteur du contrôleur central du robot pourrait travailler avec un modèle purement cinématique du robot (plutôt que les équations dynamiques qui relient les couples moteurs aux accélérations), en assumant que les boucles en vitesse des moteurs fonctionnent et sont rapide. Dans ces conditions, comme

illustré à la Figure 7.6, le concepteur du contrôleur central aurait un problème classique avec une seule boucle fermée.

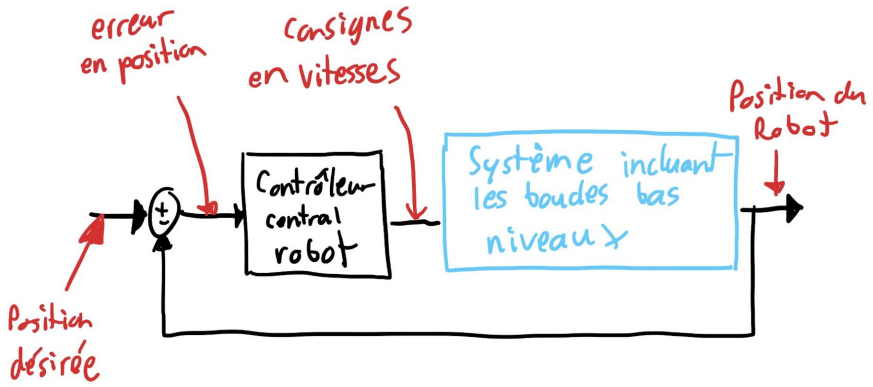


FIGURE 7.6 – Boucle fermée du contrôleur central

Il est donc important de noter que le modèle utilisé pour la conception du ou des contrôleurs d'un système robotisé doit être adapté au contexte et à l'architecture utilisée.

7.1.3 Synthétisation d'un plan hors-ligne ou en-ligne

Un autre aspect pour les méthodes de commandes avancées qui utilise beaucoup de calculs numériques, est qu'est-ce qui est calculé d'avance vs. en temps réel durant l'exécution du contrôleur. Une architecture souvent utilisée avec les algorithmes de planification/optimisation de trajectoire (ex. RRT) est illustrée à la Figure 7.7. Les calculs lourds de rechercher et/ou optimisation sont effectués d'avance, et lors de l'exécution la loi de commande utilise seulement la trajectoire cible qui a été synthétisée hors-ligne.

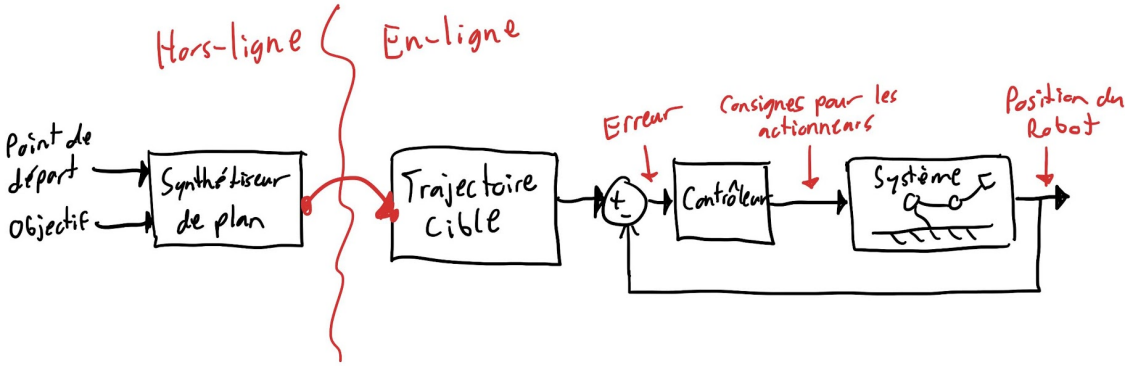


FIGURE 7.7 – Générateur de trajectoire hors-ligne

De façon similaire, les méthodes comme *LQR* ou bien *value-iteration* effectuent des calculs généralement hors-ligne pour synthétiser une loi de commande sous forme de carte $\boldsymbol{u} = c(\boldsymbol{x})$ de quelles actions $\boldsymbol{u}$ prendre en fonction des états mesurés $\boldsymbol{x}$ du robot. C'est donc ici une fonction qui est synthétisée hors-ligne et utilisée en-ligne, voir Figure 7.8. La fonction synthétisée peut-être analytique, par exemple pour la méthode *LQR* avec un format $\boldsymbol{u} = K\boldsymbol{x}$, ou bien numérique, comme par exemple pour un contrôleur synthétisé avec la méthode *value-iteration*.

Ensuite, certain type de contrôleurs de systèmes robotiques vont avoir une certaine forme de synthétisation de trajectoire ou de contrôleur mais en ligne, voir un exemple à la Figure 7.9. Typiquement, du au lourd calcul numérique qui doivent être fait en ligne, les architectures vont utiliser des boucles imbriquée avec une boucle interne plus rapide et une boucle externe plus lente qui met à jour la trajectoire cible. L'approche de

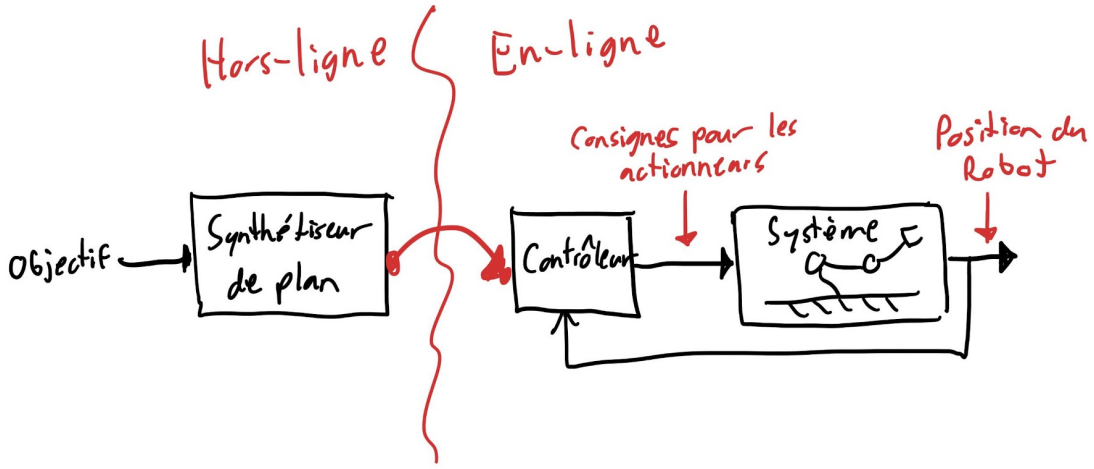


FIGURE 7.8 – Génération de contrôleur hors-ligne

commande prédictive (MPC pour *Model Predictive Control* en anglais), est un exemple ou on optimise des trajectoire cible constamment en ligne dans la boucle.

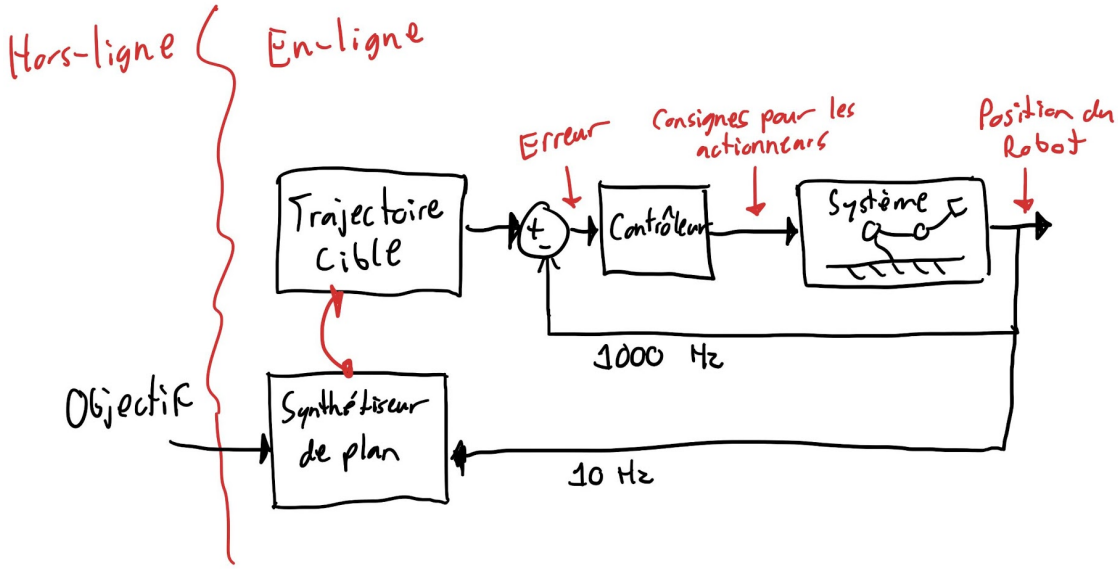


FIGURE 7.9 – Génération de trajectoire en ligne

Chapitre 8

Commande des robots manipulateurs I : quasi-statique

Les objectifs en termes de mouvement désiré d'un robot sont généralement plus naturellement spécifiés en termes de variables dans l'espace de la tâche d'un robot, alors que à bas niveau les consignes des actionneurs sont reliés à des variables dans l'espace des joints. Ce chapitre présente des méthodes de commande qui permettent de calculer les consignes pour les actionneurs basé sur des objectifs directement spécifié dans l'espace de la tâche, malgré la relation géométrique hautement non-linéaire. Les méthodes ici présenté utilisent grandement les notions de cinématique différentiel et de statique présentés dans les sections 4.5 et 5.

Les méthodes présentés dans ce chapitre négligent les effets dynamiques (inerties, frottement, etc.) et considèrent juste un comportement simplifié des robots : la relation statique non-linéaire entre le mouvement/force des actionneurs et le mouvement/force à l'effecteur. Ces méthodes sont généralement performante lorsque les mouvements du robot sont relativement lents, c'est pourquoi ils sont ici regroupé sous la caractéristique *quasi-statique*. Ensuite, selon la nature des actionneurs des systèmes robotiques, différentes variantes peuvent être utilisée :

Actionneurs commandés en vitesse : Pour les méthodes de commande du mouvement de l'effecteur présentées aux sections 8.1, 8.2 et 8.5, il est considéré que le robot a des asservissements bas-niveau en vitesse à chacun des joints. Ces méthodes calculent les consignes en vitesse à envoyer aux joints pour contrôler le mouvement de l'effecteur. Ces méthodes fonctionnent bien dans des situations ou le suivi de consigne en vitesse des joints est très performant, c'est généralement le cas des manipulateurs industriels qui ont de très grand ratios de réduction.

Actionneurs commandés en force : Pour les méthodes de commande présentées aux section 8.3 et 8.4, on considère seulement la relation géométrique entre les forces des actionneurs et ceux à l'effecteur. Les deux méthodes calculent des forces à appliquer au niveau des actionneurs/joint, pour contrôler la force au niveau de l'effecteur. Ces méthodes fonctionnent donc bien pour des systèmes robotisées à basse impédance (peu d'inertie, peu d'effets dissipatifs, transmission réversibles, etc.), comme les systèmes haptiques et certain robot collaboratif où les forces des actionneurs sont pratiquement proportionnelles au courant dans les moteurs du à des très petits ratio de transmissions.

8.1 Commande en vitesse de l'effecteur

Si un robot a des actionneurs contrôlés en vitesse à bas niveau, contrôler la vitesse de l'effecteur se résume à mettre en oeuvre la relation de cinématique différentielle inverse (voir section 4.5). Comme illustré par un schéma bloc à la Figure 8.1, les consignes en vitesse pour les actionneurs sont déterminées en multipliant le vecteur-colonne de vitesses désirés dans l'espace des tâches, par l'inverse du Jacobien qui relie l'espace des joints à l'espace des tâches du systèmes. Le Jacobien est généralement dépendent de la position des joints du robot ce qui nécessite une boucle de rétroaction basé sur les capteurs de position des actionneurs pour effectuer le calcul de J en continu basé sur la position actuelle des joints. Finalement, comme mis en évidence à la Figure 8.1, cette méthode s'intègre comme une boucle haut-niveau pour coordonner les différents joints d'un système robotisé où chacun des actionneurs est asservis en vitesse, souvent avec des asservissements bas-niveaux implémentées directement dans l'électronique de contrôle des moteurs.

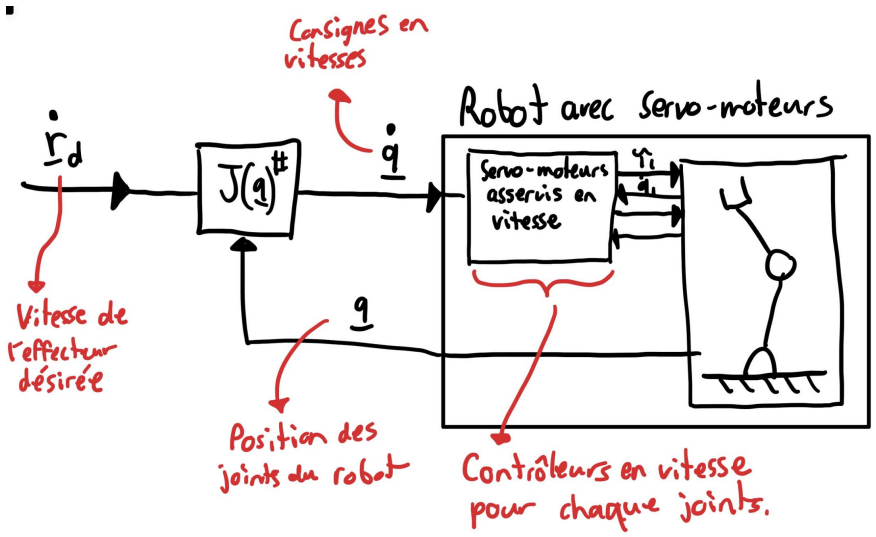


FIGURE 8.1 – Commande de la vitesse de l'effecteur d'un robot : schéma bloc

Si le nombre de joints n est égale au nombre de DDL de l'espace de la tâche m, alors la matrice Jacobienne est carrée et peut être inversée (sauf sur les singularités). Si le nombre de joint n est supérieur au nombre de DDL de l'espace de la tâche m, alors on peut utiliser une matrice pseudo-inverse droite $J^{\#} = J^T(JJ^T)^{-1}$ (voir section 18.8). La loi de commande en équation peut donc être exprimée comme :

$$\dot{\boldsymbol{q}} = \begin{cases} J(\boldsymbol{q})^{-1}\dot{\boldsymbol{r}}_d & \text{if } n = m \\ \\ J(\boldsymbol{q})^{\#}\,\dot{\boldsymbol{r}}_d & \text{if } n > m \end{cases} \tag{8.1}$$

En substituant la loi de commande dans la relation de cinématique différentielle, on confirme que la vitesse de l'effecteur sera exactement la vitesse désirée :

$$\dot{\boldsymbol{r}} = J(\boldsymbol{q})\,\dot{\boldsymbol{q}} \tag{8.2}$$
$$\dot{\boldsymbol{r}} = J(\boldsymbol{q})\,J(\boldsymbol{q})^{\#}\,\dot{\boldsymbol{r}}_d \tag{8.3}$$
$$\dot{\boldsymbol{r}} = JJ^T(JJ^T)^{-1}\,\dot{\boldsymbol{r}}_d \tag{8.4}$$
$$\dot{\boldsymbol{r}} = \dot{\boldsymbol{r}}_{\boldsymbol{d}} \tag{8.5}$$

sous les hypothèses que : 1) le Jacobien utilisé par le contrôleur est exacte, 2) le Jacobien est inversible (i.e. le robot n'est pas sur une singularité) et 3) la vitesse des joints est parfaitement asservis par les boucles bas-niveaux.

8.2 Commande en position de l'effecteur

Pour commander la position de l'effecteur d'un robot, tenter de trouver une solution directement au problème de cinématique inverse (voir section 3.9) n'est généralement pas la méthode la plus appropriée car la cinématique directe des manipulateurs est hautement non-linéaire. La méthode ici présentée utilise plutôt la méthode de commande en vitesse de l'effecteur (section 8.1) pour indirectement résoudre la cinématique inverse du robot. Le principe ce résume au fait que pour contrôler la position de l'effecteur, il suffit de diriger le vecteur vitesse de l'effecteur vers la position cible. La méthode est illustrée graphiquement à la Figure 8.2 : 1) Un vecteur d'erreur $\boldsymbol{r}_e$ est calculé en comparant la position désirée $\boldsymbol{r}_d$ à la position actuelle $\boldsymbol{r}$. 2) Le vecteur d'erreur $\boldsymbol{r}_e$ est multiplié par un paramètre scalaire λ pour déterminer la vitesse cible instantanée pour l'effecteur notée $\dot{\boldsymbol{r}}_r$, qui pointe en direction de la cible. 3) L'inverse du Jacobien est utilisé pour convertir la vitesse instantanée désirée de l'effecteur en consignes de vitesse pour les joints.

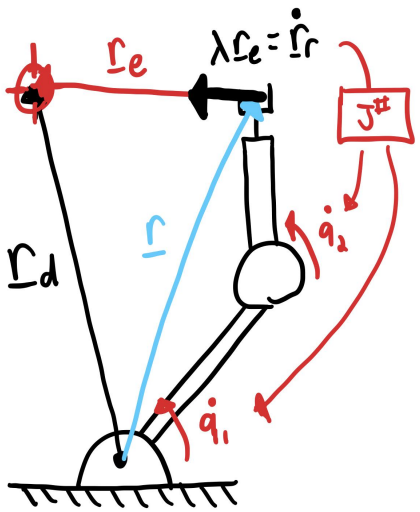


FIGURE 8.2 – Commande de la trajectoire de l'effecteur d'un robot : interprétation géométrique

Capsule vidéo
Commande du mouvement de l'effecteur d'un robot manipulateur
https://youtu.be/Qo60ySYaMqg

Formellement, la loi de commande est exprimée par l'expression mathématique :

$$\dot{\boldsymbol{q}} = \begin{cases} J(\boldsymbol{q})^{-1} \lambda \underbrace{\left(\boldsymbol{r}_d - \underbrace{f(\boldsymbol{q})}_{\boldsymbol{r}} \right)}_{\boldsymbol{r}_e} & \text{if } n = m \\ J(\boldsymbol{q})^{\#} \lambda \underbrace{\left(\boldsymbol{r}_d - \underbrace{f(\boldsymbol{q})}_{\boldsymbol{r}} \right)}_{\boldsymbol{r}_e} & \text{if } n > m \end{cases} \tag{8.6}$$

ou λ est un paramètre scalaire de gain du contrôleur, J est le Jacobien, la matrice $n \times m$, qui relie l'espace des joints à l'espace de la tâche du manipulateur, et $f(\boldsymbol{q})$ la fonction de cinématique directe du manipulateur. La Figure 8.3 illustre cette méthode de commande sous la forme d'un schéma bloc.

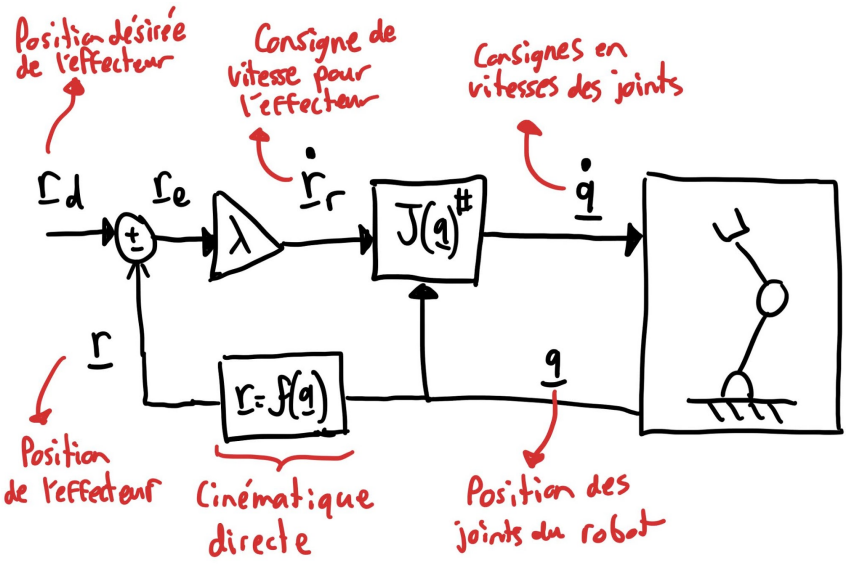


FIGURE 8.3 – Commande de la position de l'effecteur d'un robot : schéma bloc

Exercice de code
Commande d'un manipulateur en position
https://colab.research.google.com/drive/1M3O_nD8iLbRSvZzyUpBn0Mo8KUclrG26?usp=sharing

8.2.1 Suivi de trajectoire

Lorsque le robot doit suivre une position cible de l'effecteur qui varie dans le temps, il est préférable de calculer la dérivée temporelle de la trajectoire et d'utiliser cette information directement dans la loi de commande comme indiqué dans l'équation suivante :

$$\dot{\boldsymbol{q}} = \begin{cases} J(\boldsymbol{q})^{-1} \left[\dot{\boldsymbol{r}}_d + \lambda \left(\boldsymbol{r}_d - \underbrace{f(\boldsymbol{q})}_{\boldsymbol{r}} \right) \right] & \text{if } n = m \\ J(\boldsymbol{q})^{\#} \left[\dot{\boldsymbol{r}}_d + \lambda \left(\boldsymbol{r}_d - \underbrace{f(\boldsymbol{q})}_{\boldsymbol{r}} \right) \right] & \text{if } n > m \end{cases} \tag{8.7}$$

En terme de schéma bloc, la vitesse de la trajectoire doit être utilisée comme illustré à la Figure 8.4, ce qu'on appel un *feedforward* en anglais, pour garantir la convergence sur la trajectoire.

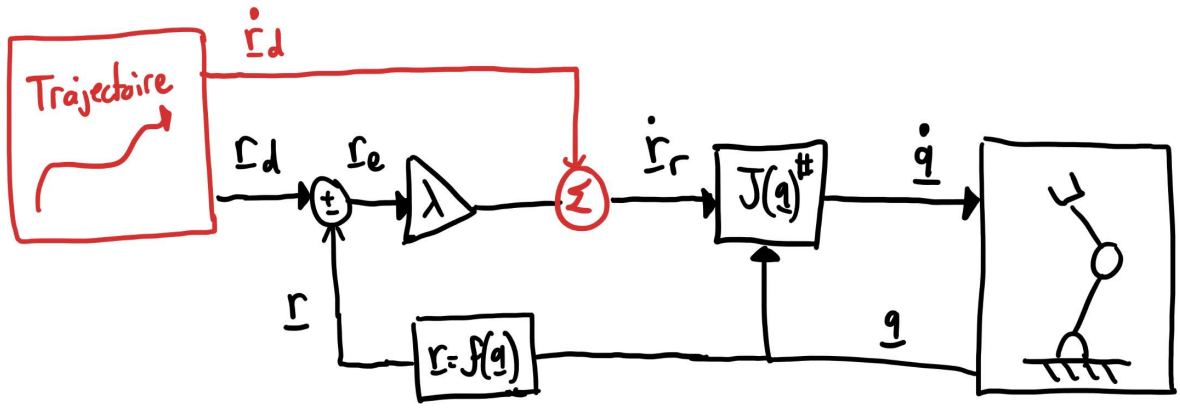


FIGURE 8.4 – Commande de la trajectoire de l'effecteur d'un robot : schéma bloc

8.2.2 Convergence

Les méthodes de commande en position de l'effecteur convergent sous certaines conditions, i.e. l'erreur (sur la position de l'effecteur) converge vers zéro lorsque le temps tend vers l'infini :

$$\lim_{t\to\infty} \boldsymbol{r}_e(t) = 0 \tag{8.8}$$

Démonstration. L'erreur est une fonction de la position désirée et de la positon réelle du robot :

$$\boldsymbol{r}_e = \boldsymbol{r}_d - \boldsymbol{r} \tag{8.9}$$

Si on dérive cette équation dans le temps, on obtient une relation différentielle. On peut alors substituer le modèle de cinématique différentiel et les lois de commandes pour obtenir :

$$\dot{\boldsymbol{r}}_e = \dot{\boldsymbol{r}}_d - \dot{\boldsymbol{r}} \tag{8.10}$$
$$\dot{\boldsymbol{r}}_e = \dot{\boldsymbol{r}}_d - J(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{8.11}$$
$$\dot{\boldsymbol{r}}_e = \dot{\boldsymbol{r}}_d - \underbrace{J(\boldsymbol{q})J(\boldsymbol{q})^{\#}}_{1}\dot{\boldsymbol{r}}_r \tag{8.12}$$
$$\dot{\boldsymbol{r}}_e = \dot{\boldsymbol{r}}_d - \underbrace{J(\boldsymbol{q})J(\boldsymbol{q})^{\#}}_{1}(\dot{\boldsymbol{r}}_d + \lambda\,\boldsymbol{r}_e) \tag{8.13}$$
$$\dot{\boldsymbol{r}}_e = -\lambda\,\boldsymbol{r}_e \tag{8.14}$$

La dynamique de l'erreur est une équation différentielle d'ordre 1. Si la constante de temps est positive $\lambda > 0$ (ici directement déterminée par le gain de notre contrôleur), l'erreur converge exponentiellement vers zéro :

$$\dot{\boldsymbol{r}}_e = -\lambda\,\boldsymbol{r}_e \qquad \Rightarrow \qquad \boldsymbol{r}_e(t) = \boldsymbol{r}_e(t=0)\,e^{-\lambda t} \qquad \Rightarrow \qquad \boldsymbol{r}_e(t=\infty) = 0 \tag{8.15}$$

□

Notez ici que cette analyse assume un contrôle parfait et instantané de la vitesse des moteurs. Une autre limite est que la convergence cesse si le robot passe sur une singularité, i.e. mathématiquement l'inverse (ou pseudo-inverse) du Jacobien n'excisera pas. Finalement, comme démontré dans la démarche, pour garantir la convergence exponentielle sur une trajectoire (position cible qui varie dans le temps), le terme de *feedforward* illustré à la Figure 8.4 est nécessaire.

8.2.3 Utilisation de l'espace nul pour un objectif secondaire

Comme il a été vu à la section 4.6.2, lorsque le nombre de DDL du robot $n = dim(\boldsymbol{q})$ est supérieur à celui de l'espace de la tâche qu'on désire contrôler $m = dim(\boldsymbol{r})$, il y a plusieurs solutions $\dot{\boldsymbol{q}}$ pour lesquels la vitesse cible pour l'effecteur est parfaitement atteinte. Lorsque la consigne de vitesse de l'effecteur $\dot{\boldsymbol{r}}_r$ est fixée par la loi de commande qui la fait converger vers $\boldsymbol{r}_d$, l'ensemble des solutions pour $\dot{\boldsymbol{q}}$ peuvent être décrite par l'équation suivante :

$$\dot{\boldsymbol{q}} = J^{\#} \underbrace{\left[\dot{\boldsymbol{r}}_d + \lambda \left(\boldsymbol{r}_d - \underbrace{f(\boldsymbol{q})}_{\boldsymbol{r}} \right) \right]}_{\dot{\boldsymbol{r}}_r} + \left[I - J^{\#} J \right] \boldsymbol{v} \tag{8.16}$$

où le vecteur-colonne $\boldsymbol{v}$ contient les variables libres. On peut donc considérer le vecteur $\boldsymbol{v}$ comme une entrée de contrôle secondaire, qu'il est possible d'utiliser pour faire autre chose avec le robot sans affecter la convergence de l'effecteur. Par exemple, les bras humains : on peut les utiliser pour attraper un objet avec notre main (objectif principal), tout en utilisant notre bras pour tenir un cahier contre notre corps (objectif secondaire). On peut aussi utiliser l'espace nul pour tenter de garder la configuration du robot le plus loin possible des configurations singulières. La Figure 8.5 illustre graphiquement l'effet de divers options de vecteur $\boldsymbol{v}$ sur les mouvements internes des joints d'un robot manipulateur redondant.

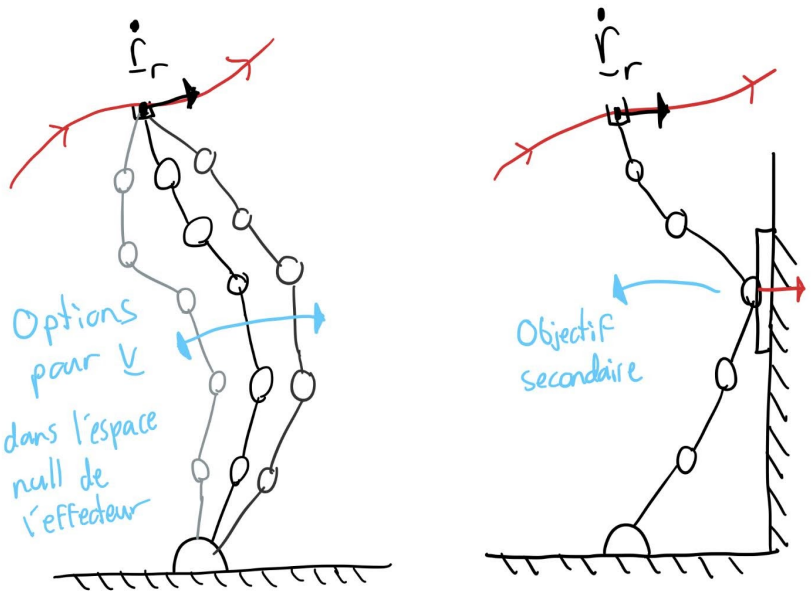


FIGURE 8.5 – Utilisation de l'espace nul d'un robot redondant pour effectuer un objectif secondaire

Capsule vidéo
Commande de l'effecteur d'un robot redondant
https://youtu.be/R0T4shGPBKo

De façon général, on peut formuler l'objectif secondaire comme une fonction potentielle scalaire G qui dépend de la configuration (un scalaire qui est le plus grand possible le plus on est loin de l'objectif). La commande secondaire $\boldsymbol{v}$ peut alors être déterminée par le gradient de cette fonction évalué à la configuration actuelle du robot :

$$G(\boldsymbol{q}) \;\Rightarrow\; \boldsymbol{v} = \frac{\partial G}{\partial \boldsymbol{q}} \tag{8.17}$$

Si l'objectif secondaire est une position cible $\boldsymbol{q}_d$ dans l'espace des joints. La fonction objectif G pourrait être la norme de l'erreur, et on obtiendrait :

$$G(\boldsymbol{q}) = -\frac{1}{2}\lambda \boldsymbol{q}_e^T \boldsymbol{q}_e \quad \text{avec :} \quad \boldsymbol{q}_e = \boldsymbol{q}_d - \boldsymbol{q} \tag{8.18}$$

$$\boldsymbol{v} = \frac{\partial G}{\partial_{\boldsymbol{q}}} = \lambda(\boldsymbol{q}_d - \boldsymbol{q}) \tag{8.19}$$

où λ est un paramètre de gain de convergence pour cette boucle de commande secondaire avec une cible de configuration $\boldsymbol{q}_d$ dans l'espace des joints. Notez que la boucle principale a toujours la priorité absolue avec cette formulation, la convergence de l'objectif principal est garantie mais pas celle de l'objectif secondaire.



Exercice de code
Espace nul d'un robot redondant
`https://colab.research.google.com/drive/16ACenFOLOHVNeReqJTbkATAB3281iVbp?usp=sharing`

8.2.4 Formulation avec régulation de la norme du vecteur vitesse

Dans certaines situation, par exemple lorsque le robot passe proche d'une singularité, les lois de commande présentées dans les sections précédentes peuvent demander des très grande vitesses irréalistes aux joints. Plutôt que d'inverser directement la matrice Jacobienne, il est possible de plutôt trouver une solution au système d'équation suivant qui pénalise aussi la norme du vecteur $\dot{\boldsymbol{q}}$:

$$\begin{bmatrix} \dot{\boldsymbol{r}}_r \\ \mathbf{0} \end{bmatrix}_{(m+n)\times 1} = \begin{bmatrix} J(\boldsymbol{q}) \\ \lambda I \end{bmatrix}_{(m+n)\times n} \begin{bmatrix} \dot{\boldsymbol{q}} \end{bmatrix}_{n\times 1} \tag{8.20}$$

où λ est un paramètre de poids sur la pénalité d'utiliser des grandes vitesses de joints. Ce système a donc $m+n$ équations et n variables, il est donc sur-contraint. Une méthode standard est d'utiliser la solution des moindres-carrés (voir section 18.7) pour trouver une solution non-exacte pour $\dot{\boldsymbol{q}}$ qui minimise la norme de l'erreur au carré, ce qui correspond ici à minimiser :

$$\|\boldsymbol{e}\|^2 = \|\dot{\boldsymbol{r}}_r - J\dot{\boldsymbol{q}}\|^2 + \lambda^2 \|\dot{\boldsymbol{q}}\|^2 \tag{8.21}$$

La solution des moindres carrés a une solution explicite qui correspond à

$$\hat{\dot{\boldsymbol{q}}} = \underset{\dot{\boldsymbol{q}}}{\operatorname{argmin}} \|\boldsymbol{e}\|^2 \tag{8.22}$$

$$\hat{\dot{\boldsymbol{q}}} = \left(J^T J + \lambda^2 I\right)^{-1} J^T \dot{\boldsymbol{r}}_r \tag{8.23}$$

La convergence de cette méthode n'est toutefois pas garantie, le robot peut dériver de la trajectoire désirée surtout si λ est trop grand. Une méthode pour palier à ce problème est d'ajuster dynamiquement λ pour lui assigner de grandes valeurs seulement proche des configurations problématiques, i.e. proche des singularités. Dans la littérature, la méthode présentée dans cette section est appelée *damped least-square*.



Capsule vidéo
Commande en position avec régulation de vitesse
`https://youtu.be/n3G-07cpQTQ`

8.3 Commande en force de l'effecteur

Pour certaines tâches, ce n'est pas la position de l'outil que l'on désire contrôler, mais plutôt la force qu'il applique sur l'environnement. Si un systèmes robotisé a des actionneurs qui sont contrôlable en force ou couple, il suffit d'une transformation géométrique avec le Jacobien pour calculer les consignes en force des actionneurs basé sur une force désirée à l'effecteur, comme illustré à la Figure 8.6.

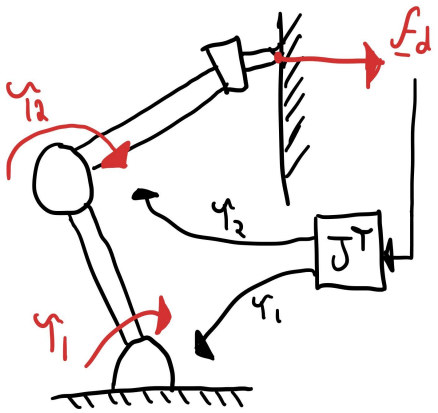


FIGURE 8.6 – Commande en force de l'effecteur d'un robot : interprétation géométrique

Capsule vidéo
Commande en force d'un robot manipulateur
`https://youtu.be/mDQInbWXcj4`

La Figure 8.7 montre cette méthode de commande avec un schéma bloc. À l'exception du calcul du Jacobien qui nécessite de connaître la position des joints actuelle, la méthode peut être peut être considérée comme une boucle ouverte. Par exemple dans le cas des robots à entraînement direct, i.e. utilisant des gros moteurs électrique qui n'ont pas de ratio de réduction, il suffit de traduire la force désirée en couples moteurs à l'aide du Jacobien et ensuite de traduire les couples moteurs en consigne de courant des les moteurs en divisant par la constante des moteurs. Selon le type d'actionneur, des boucles de rétroaction en force à bas niveau peuvent être utilisées, par exemple un asservissement de la pression dans un actionneur pneumatique. La loi de commande présentée ici spécifie les consignes en force pour les actionneurs.

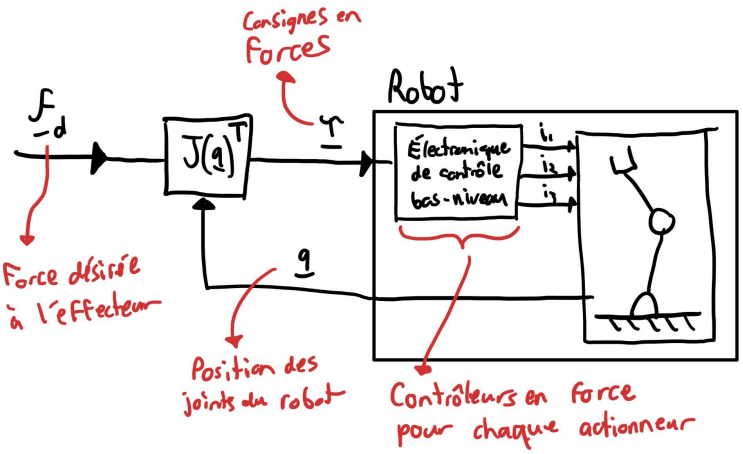


FIGURE 8.7 – Commande en force de l'effecteur d'un robot : schéma bloc

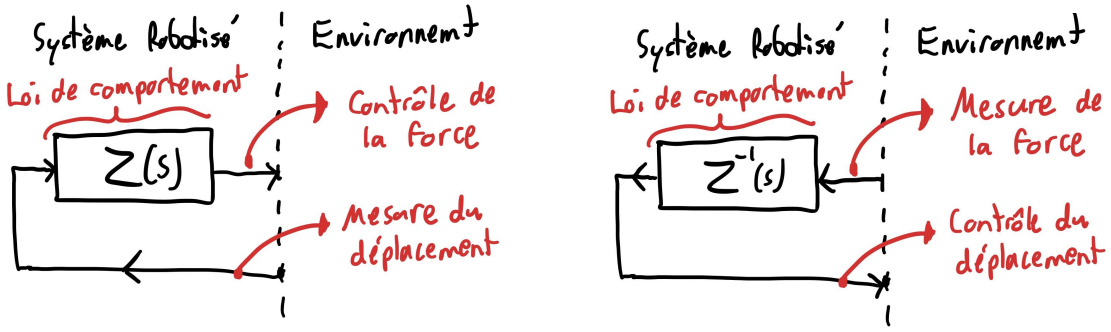
Il est à noter que pour produire une force, le principe action-réaction nous rappel qu'il faut avoir une résistance. Donc si l'effecteur n'est pas en contact avec un objet le comportement de cette loi de commande de donnera pas nécessairement le résultat attendu, les forces des actionneurs vont alors ce balancer avec la résistance interne du robot : forces dissipatrices et inertielles. Le robot va donc accélérer s'il n'y a pas de contact ou de résistance. Comme le mentionne le titre du chapitre, cette loi de commande **quasi-statique** fonctionne bien lorsque le robot est bloqué ou bien bouge très lentement, sinon les forces dissipatives et inertielles interne au robot vont influencer la force de contact.

8.4 Commande en impédance et en admittance

Jusqu'à maintenant, à la section 8.2 une technique a été discutée pour contrôler la position d'un robot et à la section 8.3 une technique a été discuté pour contrôler la force transmise par un robot. La dernière grande catégorie de type de commande est de contrôler la relation entre la positon et la force du robot. L'idée est d'imposer une loi de comportement qui relie la force au déplacement du système. Cette approche est particulièrement utile pour les situations ou un robot va entrer en contact avec un objet ou un humain. Un des inconvénients d'un contrôle en position pur est que si le robot rencontre un obstacle qui bloque son chemin, le robot va soudainement appliquer une très grande force pour tenter de continuer, ce qui peut risquer de briser soit l'objet rencontré ou le robot lui-même. À l'opposé, le problème avec une approche de contrôle en force pur est que si le robot n'a pas d'obstacle pour créer une résistance et établir la force désirée, le robot va plutôt accélérer jusqu'à se qu'il atteindre la limite de son espace de travail. La commande en impédance ou en admittance est une approche hybride qui vise plutôt à imposer une relation entre la force et le déplacement. Par exemple, on pourrait désirer que le robot se déplace de façon proportionnelle à une force appliqué sur celui-ci, en imposant ce comportement le robot émulerait le comportement d'un ressort.

Capsule vidéo
Commande en impédance et admittance
https://youtu.be/h_3qngSmwPo

Deux approches permettent d'imposer une relation entre la force et le mouvement d'un système : la commande en impédance et en admittance. D'un point de vu théorique les approches sont pratiquement équivalentes, comme illustré à la Figure 8.8, la différence est la causalité. Un contrôleur en impédance mesure le déplacement et impose une force, et un contrôleur en admittance mesure la force pour imposer un déplacement. Dans un monde idéal ou les mesures et le contrôle de la force/déplacement serait parfait, pour la même loi de comportement les deux approches donneraient un résultat identique. Toutefois, en pratique il y a de très grandes différences d'implémentation et de performance entre ces approches.



(a) Commande en **impédance**

(b) Commande en **admittance**

FIGURE 8.8 – Deux façon d'imposer une loi de comportement qui relie la force au déplacement.

Les lois de comportements mécaniques sont souvent décrit avec des fonctions de transfert dans le domaine de Laplace pour une notation compacte. La fonction $Z(s)$ est utilisée pour décrire l'impédance : le signal

de force $f(s)$ divisé par le signal de déplacement $x(s)$ dans le domaine de Laplace. L'admittance $Y(s)$ est l'inverse de la fonction d'impédance.

$$\text{Impédance :} \quad Z(s) = \frac{f(s)}{x(s)} \qquad \text{Admittance :} \quad Y(s) = \frac{1}{Z(s)} = \frac{x(s)}{f(s)} \tag{8.24}$$

Note :

Dans certains ouvrage le concept d'impédance est définie comme la force divisée par la vitesse (contrairement à la position). La définition relative au déplacement sera toutefois utilisée ici car elle s'intègre plus naturellement au contexte de commande des robots.

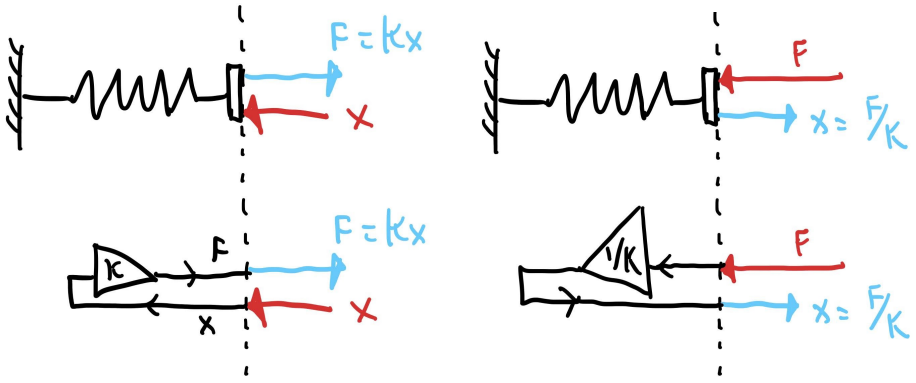
Un système mécanique avec une inertie m, un amortissement linéaire b et une rigidité k a une impédance complexe égale à :

$$Z(s) = ms^2 + bs + k \tag{8.25}$$

L'opération de multiplié le signal de déplacement par l'impédance $Z(s)$ est équivalent dans le domaine temporel à :

$$f(s) = Z(s)x(s) = \left(ms^2 + bs + k\right)x(s) \quad \Longleftrightarrow \quad f(t) = m\frac{d^2}{dt^2}x(t) + b\frac{d}{dt}x(t) + kx(t) \tag{8.26}$$

Exemple Une loi de comportement qui est souvent désirable d'implémenter est la loi de *Hooke*, i.e. un ressort. Comme illustré à la Figure 8.9, l'approche en impédance se résume à mesurer un signal de position x et le multiplier par une constante de rigidité k pour obtenir la force à appliquer. L'approche en admittance se résume à mesurer un signal de force f et le diviser par la constante de rigidité k pour obtenir le déplacement x à imposer. On utilise parfois la notion de compliance $C = 1/k$ qui est l'inverse de la rigidité.

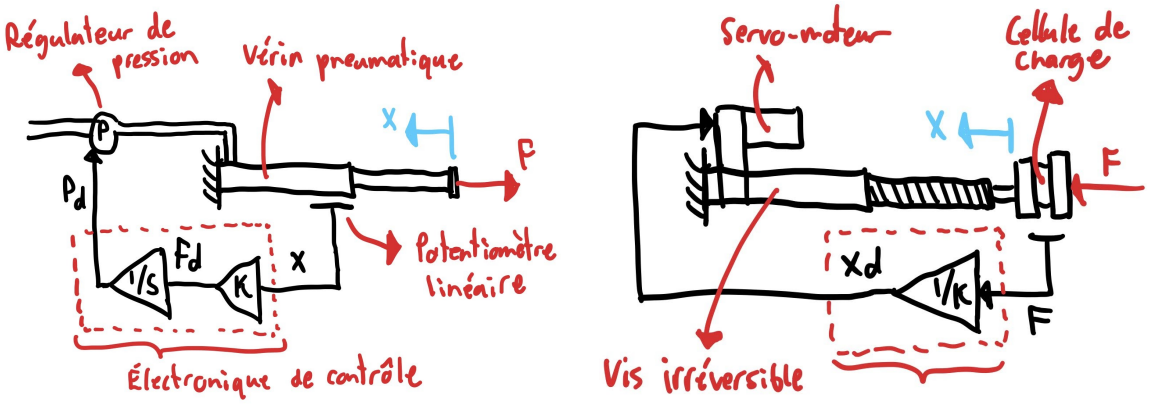


(a) Commande en **impédance** (b) Commande en **admittance**

FIGURE 8.9 – Imposition d'une loi de comportement $f = kx$ (ressort)

En pratique, se qui va guider le choix d'une approche ou d'une autre est principalement le type d'actionneur utilisé dans le système robotisé. L'approche par impédance est plus naturelle pour un système avec des actionneurs qui sont des sources de forces (moteur électrique sans réducteur, cylindre pneumatique, etc.), tandis que l'approche par admittance est plus naturelle pour les systèmes avec des actionneurs qui sont des sources de déplacement, i.e. plus facilement contrôlable en position ou vitesse comme la plupart des moteurs électriques jumelés à des grandes réductions utilisés dans les robots industriels. La Figure 8.10 illustre deux implémentations possibles pour émuler la loi de *Hooke*. Généralement, l'approche en impédance jumelée à des actionneurs à faible résistance hautement réversible (source de force pure) est idéale pour bien émuler des loi de comportement caractérisés par des petites impédances. La performance est toutefois limitée pour émuler des rigidités très élevée. Cette approche est généralement utilisée par les systèmes haptiques et certains systèmes robotiques collaboratif. À l'inverse, l'approche en admittance avec des actionneurs irréversibles (source

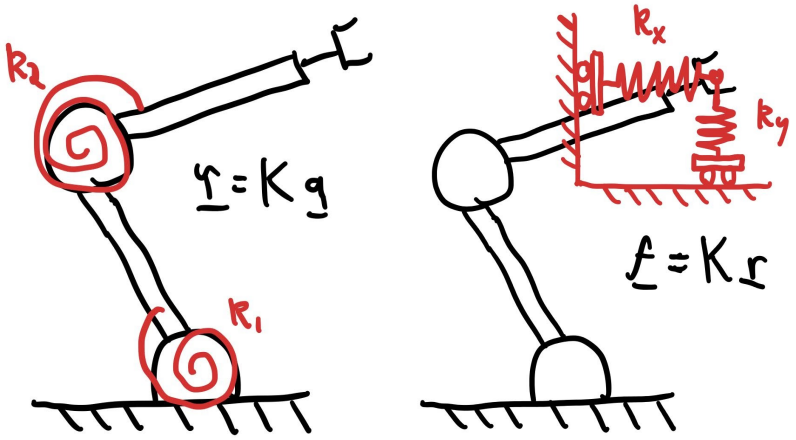
de position), est idéale pour émuler de très grandes impédances. La performance est toutefois limitée pour émuler un comportement très compliant (peu de résistance) car l'accélération et la vitesse des actionneurs sont limités en pratique. Cette approche est généralement utilisée par les robots collaboratifs industriels.



(a) Commande en **impédance** (b) Commande en **admittance**

FIGURE 8.10 – Deux exemples d'implémentation de commande pour une loi de comportement $f = kx$ (ressort). L'approche en impédance est adaptée à aux actionneurs qui sont des sources de force avec peu de résistance interne. L'approche en admittance est adaptée aux actionneurs qui sont irréversibles et donc des sources de déplacements peu influencées par la résistance externe.

Les sections suivantes présentent des approches de commande haut-niveau, i.e. la coordination spatiale de plusieurs joints pour obtenir des lois de comportement, soit exprimées dans l'espace des joints ou de la tâche comme illustré à la Figure C.8. Dans ces sections il est considéré que les actionneurs sont soit des sources parfaites de force ou des sources parfaites de déplacement. La performance dynamique des boucles en forces/impédance/admittance implique une analyse de la dynamique bas-niveau des actionneurs qui est plutôt traité au chapitre **??**.



(a) Dans l'espace des joints (b) Dans l'espace de la tâche

FIGURE 8.11 – Commande de la rigidité/compliance d'un robot

8.4.1 Commande en impédance aux joints

Pour des robots avec des actionneurs contrôlés en force ou couple, la commande en impédance aux joints se résume à relié le déplacement mesuré aux joints aux efforts à chaque joint. Dans le cas le plus simple, comme illustré à la Figure C.8a, on peut commander des couples moteurs proportionnel aux déplacement

angulaire des joints, ce qui est équivalent à émuler des ressorts angulaires sur chaque joint. De façon général on pourrait émuler des ressorts-amortisseurs sur chaque joint avec la loi de commande :

$$\boldsymbol{\tau} = K\boldsymbol{q}_e + B\dot{\boldsymbol{q}}_e \tag{8.27}$$

ou $\boldsymbol{q}_e = \boldsymbol{q}_d - \boldsymbol{q}$ est une erreur par rapport à la configuration de référence ou les ressorts sont "au repos". Les matrices K et B sont respectivement une matrice de coefficients de rigidité et une matrice de coefficients d'amortissement. Les matrice sont diagonales pour un comportement de chaque joint indépendant. Dans ce cas diagonal, la loi de commande est équivalente à plusieurs boucle d'asservissement indépendantes de type Proportionnel-Dérivé sur la position des joints. La loi de commande est illustrée sous forme de schéma bloc à la Figure 8.12. Comme ici on programme des lois de comportement, il est possible de programmer

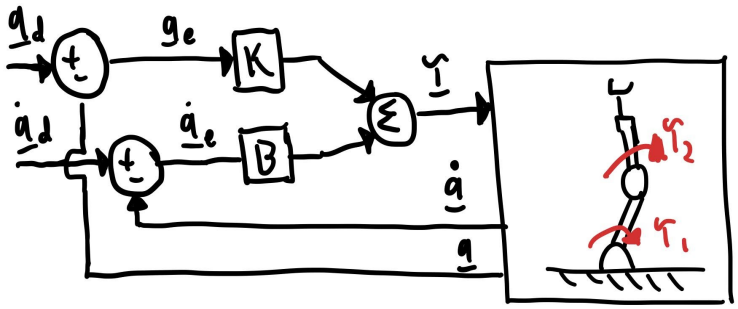


FIGURE 8.12 – Commande de l'impédance au joints d'un robot : schéma bloc

des impédances non-linéaires arbitraires. Par exemple, pour les systèmes haptiques un problème courant est d'émuler un obstacle rigide, ce qui est appeler mur virtuel dans la littérature. L'impédance cible dans ce cas est discontinue, aucune résistance pour une certaine plage de déplacement et une très grand rigidité lorsque la position atteint la position du mur virtuel.

Compensation de friction : Les systèmes robotiques vont normalement avoir de la dissipation naturelle présente dans les joints (roulements, engrenage, etc.). Si la friction naturelle est significative et plus grande que l'amortissement qu'on désire émuler, les coefficients de friction dans B pourraient être négatif pour appliquer une force opposée à la friction naturelle. La matrice B pourrait être définie par :

$$B = B_d - B_{sys} \tag{8.28}$$

ou B_d contient les coefficients d'amortissement que l'on désire émuler et B_{sys} contient les coefficients d'amortissement naturel du robot. Il est toutefois risqué en pratique de tenter de totalement compenser la friction dans les joints d'un robot, car si la compensation est légèrement plus forte que la friction naturelle le système se retrouve instable. De plus le comportement de la friction autour de zéro vitesse est généralement très non-linéaire et avec de l'hystérésis, donc dur à modéliser avec précision.

Note sur l'émulation de l'inertie :

On pourrait être tenté de rajouter un terme de type $M\ddot{\boldsymbol{q}}_e$ à l'équation (8.27) pour rajouter l'option d'émuler un effet inertiel sur chaque joint. Toutefois il y a un problème de causalité à une telle opération. L'accélération n'est pas un état du système mais résulte de l'application de forces sur le système incluant $\boldsymbol{\tau}$. Donc les couples dépendraient de l'accélération mesurée, qui dépend des couples appliquées par les actionneurs, qui dépendraient de l'accélération mesurée, qui dépend ... etc. On se retrouve avec une boucle algébrique analogue au paradoxe de la poule et de l'oeuf (lequel vient en premier ?). En pratique, l'accélération mesurée serait celle d'un instant passé dû au délai de calcul dans le contrôleur et aux filtres dans les capteurs, et la rétroaction de la mesure de l'accélération risque de déstabiliser le système.

8.4.2 Commande en impédance de l'effecteur

Pour la commande en impédance de l'effecteur d'un robot, c'est la relation force-déplacement à l'effecteur qu'on impose au robot :

$$\boldsymbol{f}_e = K\boldsymbol{r}_e + B\dot{\boldsymbol{r}}_e \tag{8.29}$$

ou $\boldsymbol{r}_e = \boldsymbol{r}_d - \boldsymbol{r}$ est le vecteur position qui de la position désirée de l'effecteur du robot par rapport à la position actuelle. On peut interpréter l'effet de la loi de commande comme un ressort-amortisseur virtuel qui relit l'effecteur du robot à sa position cible comme illustré à la Figure 8.13. Le ressort-amortisseur virtuel

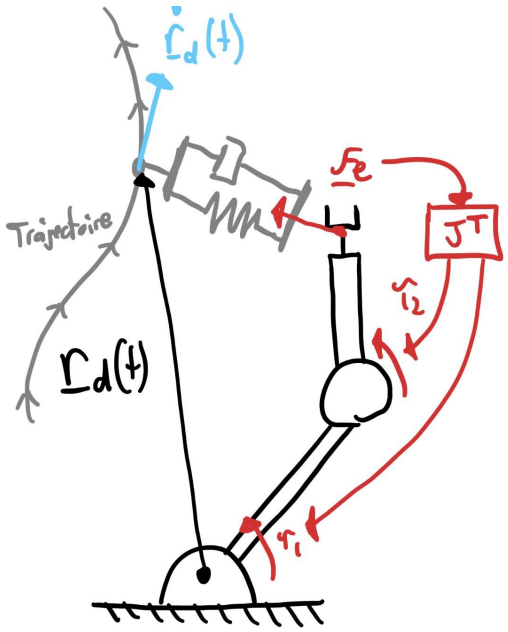


FIGURE 8.13 – Commande de l'impédance de l'effecteur d'un robot : interprétation géométrique

produit une force virtuelle $\boldsymbol{f}_e$ désirée à l'effecteur qui est traduite en commande de couple grâce à la relation statique d'un manipulateur :

$$\boldsymbol{\tau} = J(\boldsymbol{q})^T \boldsymbol{f}_e \tag{8.30}$$

Capsule vidéo
Commande en impédance d'un robot manipulateur
https://youtu.be/Rl2kDwsYS9E

La Figure 8.14 illustre cette loi de commande avec un schéma bloc. La première étape du contrôleur est de calculer la position et la vitesse de l'effecteur à partir des capteurs qui mesurent la position et la vitesse des joints. Il faut donc ici utiliser la cinématique directe et la cinématique-différentielle directe pour calculer la position et la vitesse de l'effecteur. Ensuite la position et la vitesse de l'effecteur sont comparées à la position et la vitesse désirée (qui peut varier dans le temps, i.e. une trajectoire). L'erreur en termes de position et vitesse est convertie en force cartésienne basé sur la loi de comportement désirée (l'impédance è l'effecteur). Finalement cette force cartésienne est convertie en couple aux joints grâce au Jacobien du robot manipulateur. On peut aussi ajouter une compensation de gravité, voir la section 8.4.4. L'équation de la loi

de commande est égale à :

$$\boldsymbol{\tau} = J(\boldsymbol{q})^T \boldsymbol{f}_e \tag{8.31}$$
$$\boldsymbol{\tau} = J(\boldsymbol{q})^T \left[K \boldsymbol{r}_e + B \dot{\boldsymbol{r}}_e\right] \tag{8.32}$$
$$\boldsymbol{\tau} = J(\boldsymbol{q})^T \left[K \left(\boldsymbol{r}_d - \boldsymbol{r}\right) + B \left(\dot{\boldsymbol{r}}_d - \dot{\boldsymbol{r}}\right)\right] \tag{8.33}$$
$$\boldsymbol{\tau} = J(\boldsymbol{q})^T \left[K \left(\boldsymbol{r}_d - f(\boldsymbol{q})\right) + B \left(\dot{\boldsymbol{r}}_d - J(\boldsymbol{q})\dot{\boldsymbol{q}}\right)\right] \tag{8.34}$$

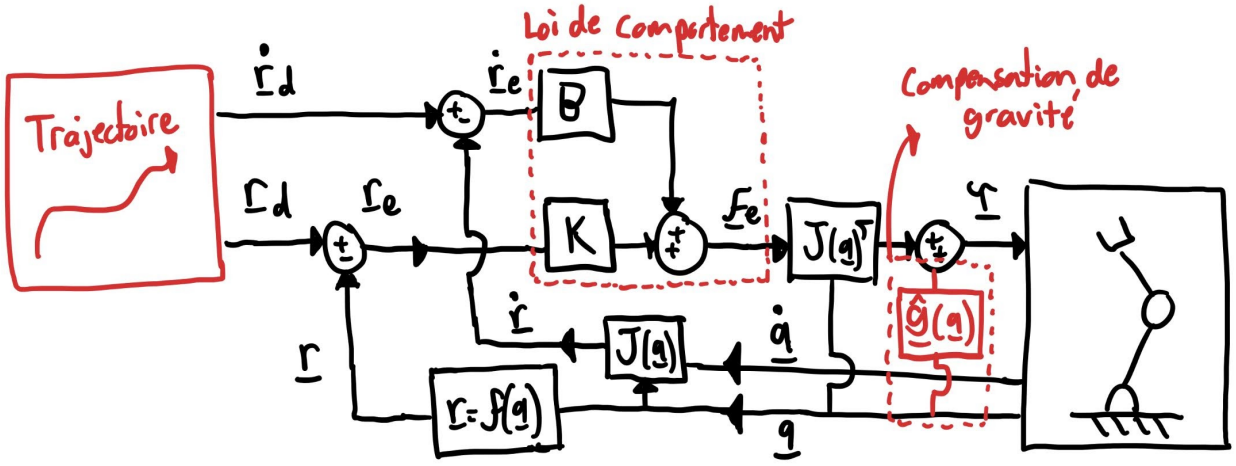


FIGURE 8.14 – Commande de l'impédance de l'effecteur d'un robot : schéma bloc

Exercice de code
Simulation d'un robot planaire contrôlé en impédance
https://colab.research.google.com/drive/1EM3hNEwz2aiBqx9GDHoM7QGuPF-Afv1Q?usp=sharing

8.4.3 Équivalence entre une impédance définie aux joints vs. définie à l'effecteur

Il y a une certaine équivalence locale entre une loi de commande en impédance à l'effecteur et une loi de commande en impédance dans l'espace des joints. Pour un cas simplifié ou la cible $\boldsymbol{q}_d$ et $\boldsymbol{r}_d$ sont égale à zéro, les lois de commandes sont données par :

$$\text{Joints} : \boldsymbol{\tau} = -K_q \boldsymbol{q} - B_q \dot{\boldsymbol{q}} \qquad \text{Effecteur} : \boldsymbol{\tau} = -J(\boldsymbol{q})^T \left[K_e \boldsymbol{r} + B_e \dot{\boldsymbol{r}}\right] \tag{8.35}$$

où on a rajouté les indices q ou e aux matrices pour spécifier l'espace, joint ou effecteur, dans lesquelles elles sont définies. Ensuite si on distribue et utilise la relation de cinématique différentielle, on observe que la matrice d'amortissement B_e à l'effecteur a exactement le même effet qu'une matrice d'amortissement dans l'espace des joints égale à $B_q = J^T B_e J$:

$$\boldsymbol{\tau} = -J(\boldsymbol{q})^T K_e \boldsymbol{r} - J(\boldsymbol{q})^T B_e \dot{\boldsymbol{r}} \tag{8.36}$$
$$\boldsymbol{\tau} = -J(\boldsymbol{q})^T K_e \boldsymbol{r} - \underbrace{\left[J(\boldsymbol{q})^T B_e J(\boldsymbol{q})\right]}_{B_q} \dot{\boldsymbol{q}} \tag{8.37}$$

Pour des petites variations autour de la position d'équilibre des ressorts virtuels (celle pour laquelle la force nette est nulle), on peut aussi retrouver une équivalence similaire pour la rigidité $K_q = J^T K_e J$:

$$\delta\boldsymbol{\tau} = -\underbrace{\left[J(\boldsymbol{q})^T K_e J(\boldsymbol{q})\right]}_{K_q} \delta\boldsymbol{q} - \underbrace{\left[J(\boldsymbol{q})^T B_e J(\boldsymbol{q})\right]}_{B_q} \delta\dot{\boldsymbol{q}} \tag{8.38}$$

Il est à bien noter que ces équivalences sont locales seulement due aux effets non-linéaires.

8.4.4 Convergence des lois de commande en impédance

Pour les robots avec des actionneurs contrôlés en force, une bonne façon de contrôler la position de l'effecteur est d'utiliser l'approche en impédance à l'effecteur avec une matrice K très rigide. C'est l'équivalent de prendre un gros ressort très rigide, attacher une extrémité sur la position cible et l'autre sur l'effecteur du robot. Le robot va être attiré vers la position cible. D'un point de vu énergétique, le robot va converger vers le point ou l'énergie potentielle du système est minimum, ce qui est la position cible si le ressort est la seule source d'énergie potentielle. Pour que le point de convergence soit stable il faut aussi qu'il y ait une certaine forme de dissipation dans le système (qui peut venir des coefficients du ressort virtuel dans B ou bien de phénomènes physique dans les joints du robot) sinon le robot oscillerait autour du point d'énergie potentiel minimum.

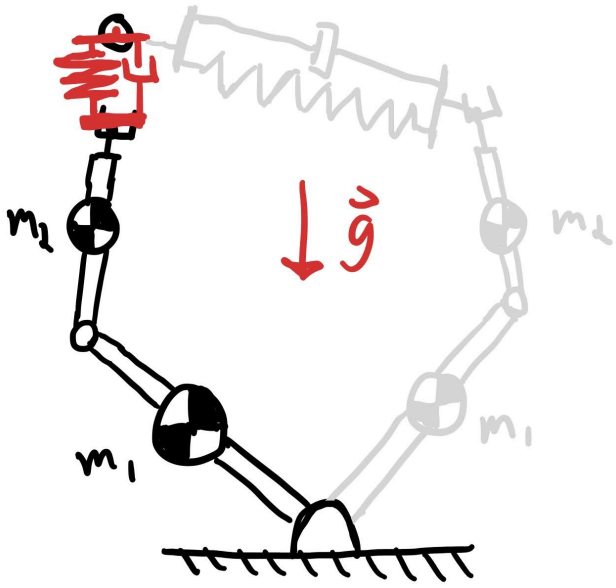


FIGURE 8.15 – Convergence d'un contrôleur en impédance à l'effecteur

Comme illustré à la Figure 8.15, si considère d'abord le robot comme seulement une chaîne de corps rigide avec leurs propriétés inertielles, on peut bien visualiser qu'un ressort à l'effecteur va attirer le robot sur la cible. Les propriétés inertielles du bras et la géométrie non-linéaire vont influencer la trajectoire, mais dans ce cas le point d'équilibre dépend seulement des forces conservatrice. Si le ressort virtuel de la loi d'impédance est le seul élément qui produit une force conservatrice, alors le point d'équilibre sera la configuration ou l'effecteur est exactement sur la cible. Si le robot est influencé par des forces gravitationnelles, le point d'équilibre sera le point minimum de l'énergie potentielle totale. Le robot aura donc une erreur finale avec une amplitude qui dépend de la force relative du ressort virtuel et de la gravité. Comme illustré à la Figure 8.14, il est toutefois possible d'inclure une **compensation de gravité $\hat{\boldsymbol{g}}$** à une loi de commande en impédance pour éliminer cette erreur finale :

$$\boldsymbol{\tau} = J(\boldsymbol{q})^T \left[K\boldsymbol{r}_e + B\dot{\boldsymbol{r}}_e\right] + \hat{\boldsymbol{g}}(\boldsymbol{q}) \tag{8.39}$$

où $\hat{\boldsymbol{g}}$ est l'estimation par le contrôleur (basé sur un modèle) du vecteur des forces gravitationnelles dans l'espace des joints. Finalement, un autre phénomène courant qui pourrait causer une erreur final serait s'il y a de la friction sèche dans les joints du robot. Le robot pourrait rester pris à un point ou la force commandée est égale à la friction sèche dans le système.

Cette analyse qualitative basée sur des principes énergétiques peut être formalisé avec une analyse de la stabilité avec la méthode de *Lyapunov*. Si on considère les équations génériques de la dynamique du robot manipulateur avec des actionneurs contrôlés en forces :

$$\underbrace{H\ddot{\boldsymbol{q}} + C\dot{\boldsymbol{q}}}_{\text{Forces inertielles}} + \underbrace{\boldsymbol{d}}_{\text{Forces dissipatives}} + \underbrace{\boldsymbol{g}}_{\text{Forces conservatrices}} = \boldsymbol{\tau} \tag{8.40}$$

avec comme loi de commande, une impédance à l'effecteur incluant une compensation de gravité :

$$\boldsymbol{\tau} = J^T \left[K\boldsymbol{r}_e + B\dot{\boldsymbol{r}}_e \right] + \hat{\boldsymbol{g}} \tag{8.41}$$

En utilisant comme fonction candidate de *Lyapunov* (voir détails au chapitre 20) l'énergie mécanique réelle plus l'énergie potentielle virtuelle dans le ressort du contrôleur :

$$V = \underbrace{1/2\, \dot{\boldsymbol{q}}^T H \dot{\boldsymbol{q}}}_{\text{Énergie cinétique réelle du robot}} + \underbrace{1/2\, \boldsymbol{r}_e^T K \boldsymbol{r}_e}_{\text{Énergie potentielle virtuelle du ressort}} \tag{8.42}$$

Il est possible de démontré que cette fonction est toujours décroissante dans le temps sous certaines conditions :

$$\dot{V} = \frac{d}{dt} \left[1/2\, \dot{\boldsymbol{q}}^T H \dot{\boldsymbol{q}} + 1/2\, \boldsymbol{r}_e^T K \boldsymbol{r}_e \right] \tag{8.43}$$

$$\dot{V} = \dot{\boldsymbol{q}}^T \left[H\ddot{\boldsymbol{q}} + 2\dot{H}\dot{\boldsymbol{q}} \right] + \boldsymbol{r}_e^T K \dot{\boldsymbol{r}}_e \tag{8.44}$$

$$\dot{V} = \dot{\boldsymbol{q}}^T \left[\boldsymbol{\tau} - \boldsymbol{g} - \boldsymbol{d} \right] - \dot{\boldsymbol{r}}^T K \boldsymbol{r}_e \qquad \text{pour une position désirée fixe : } \dot{\boldsymbol{r}}_e = \dot{\boldsymbol{r}}_d - \dot{\boldsymbol{r}} = -\dot{\boldsymbol{r}} \tag{8.45}$$

$$\dot{V} = \dot{\boldsymbol{q}}^T \left[\boldsymbol{\tau} - \boldsymbol{g} - \boldsymbol{d} \right] - \dot{\boldsymbol{q}}^T J^T K \boldsymbol{r}_e \tag{8.46}$$

$$\dot{V} = \dot{\boldsymbol{q}}^T \left[\boldsymbol{\tau} - \boldsymbol{g} - \boldsymbol{d} \right] - \dot{\boldsymbol{q}}^T J^T K \boldsymbol{r}_e \tag{8.47}$$

$$\dot{V} = \dot{\boldsymbol{q}}^T \left[\boldsymbol{\tau} - \boldsymbol{g} - \boldsymbol{d} - J^T K \boldsymbol{r}_e \right] \tag{8.48}$$

$$\dot{V} = \dot{\boldsymbol{q}}^T \left[J^T \left[K\boldsymbol{r}_e + B\dot{\boldsymbol{r}}_e \right] + \hat{\boldsymbol{g}} - \boldsymbol{g} - \boldsymbol{d} - J^T K \boldsymbol{r}_e \right] \tag{8.49}$$

$$\dot{V} = \dot{\boldsymbol{q}}^T \left[J^T K\boldsymbol{r}_e - J^T B\dot{\boldsymbol{r}} + \hat{\boldsymbol{g}} - \boldsymbol{g} - \boldsymbol{d} - J^T K \boldsymbol{r}_e \right] \tag{8.50}$$

$$\dot{V} = \dot{\boldsymbol{q}}^T \left[-J^T B J\dot{\boldsymbol{q}} + \hat{\boldsymbol{g}} - \boldsymbol{g} - \boldsymbol{d} \right] \tag{8.51}$$

$$\dot{V} = \dot{\boldsymbol{q}}^T \left[-J^T B J\dot{\boldsymbol{q}} - \boldsymbol{d} \right] \qquad \text{si la compensation de gravité est parfaite} \tag{8.52}$$

$$\dot{V} = \dot{\boldsymbol{q}}^T \left[-J^T B J\dot{\boldsymbol{q}} - D\dot{\boldsymbol{q}} \right] \qquad \text{si on considère la friction dans les joints du robot comme linéaire} \tag{8.53}$$

$$\dot{V} = -\dot{\boldsymbol{q}}^T \left[J^T B J + D \right] \dot{\boldsymbol{q}} \tag{8.54}$$

$$\dot{V} < 0 \quad \forall \dot{\boldsymbol{q}} \neq \boldsymbol{0} \quad \text{si } \left[J^T B J + D \right] > 0 \qquad \Rightarrow \qquad \lim_{t\to\infty} \boldsymbol{r}_e(t) = \boldsymbol{0} \quad \& \quad \lim_{t\to\infty} \dot{\boldsymbol{q}}(t) = \boldsymbol{0} \tag{8.55}$$

Ce qui démontre que le système va nécessairement converger vers le point minimum de la fonction V, ce qui correspond à $\boldsymbol{q} = \boldsymbol{0}$ et $\boldsymbol{r}_e = \boldsymbol{0}$, avec comme condition que les termes d'amortissement total du système (contrôleur + friction naturelle) sont positifs. Cette analyse peut aussi être faites pour une impédance dans le domaine des joints, et une combinaison des deux, pour obtenir des conclusions équivalentes.

8.5 Commande en admittance

Pour un contrôle en admittance, dans l'espace des joints ou de la tâche, des capteurs mesurent des forces et le déplacement du robot est contrôlé à bas niveau, soit en position ou en vitesse. Si les actionneurs sont asservis en position, la relation générique est :

$$q_i = \frac{1}{Z_i(s)} \tau_i \tag{8.56}$$

où q_i est une consigne de position pour le joint i, Z_i est une impédance désirée pour le joint i et τ_i est la force mesurée au joint i. Si les actionneurs sont asservis en vitesse à bas niveau, la relation générique est :

$$\dot{q}_i = \frac{s}{Z_i(s)} \tau_i \tag{8.57}$$

Lorsqu'on implémente une loi de commande il est préférable d'éviter de dériver des signaux, donc certaines combinaison de type de consigne bas-niveau (position vs. vitesse) avec un type d'impédance sont préférable.

Par exemple, pour émuler une force proportionnelle à une vitesse $Z_i(s) = b_i s$ avec des actionneurs contrôlés en vitesse, l'équation (8.57) se réduit à

$$\dot{q}_i = \frac{1}{b}\tau_i \tag{8.58}$$

donc simplement une relation algébrique entre les signaux. Avec des actionneurs contrôlés en position l'équation (8.56) se réduit à

$$q_i = \frac{1}{bs}\tau_i = \frac{1}{b}\int \tau_i \tag{8.59}$$

ce qui implique d'intégrer le signal de force. Par contre, si on désire émuler un système masse-ressort-amortisseur c'est l'approche avec des actionneurs en position qui est préférable. Avec des actionneurs contrôlés en position l'équation (8.56) se réduit dans ce cas à

$$\dot{q}_i = \frac{1}{ms^2 + bs + k}\tau_i \tag{8.60}$$

La fonction de transfert a deux pôles donc l'implémentation inclurait deux intégrales. Avec des actionneurs contrôlés en vitesse l'équation (8.57) se réduit dans ce cas à

$$\dot{q}_i = \frac{s}{ms^2 + bs + k}\tau_i \tag{8.61}$$

une fonction de transfert qui a en plus un zéro, donc demanderait de dériver un signal en plus des deux intégrales. Pour l'implémentation d'une loi de commande, l'idéal lorsque possible est une relation algébrique. Si on doit avoir une relation différentielle il est préférable d'avoir seulement des intégrales. Les opérations de dérivations sont à éviter.

8.5.1 Commande en admittance aux joints

Pour un contrôle en admittance dans l'espace des joints, des capteurs de forces vont mesurée la force actuelle à chaque joint et commander un déplacement à l'actionneur relié. Les lois en admittance peuvent aussi être exprimées en format matricielle pour décrire le comportement de tous les joints en une seule équation. La Figure 8.16 présente un contrôleur en admittance de type amortisseurs avec comme comportement cible la relation :

$$B\dot{\boldsymbol{q}} = \boldsymbol{\tau} \tag{8.62}$$

pour un robot contrôlé en vitesse à bas niveau dans l'espace des joints, avec des capteurs qui mesure le couple de chaque joint.

La Figure 8.17 présente le schéma bloc d'un contrôleur en admittance avec comme comportement cible la relation :

$$M\ddot{\boldsymbol{q}} + B\dot{\boldsymbol{q}} + K\boldsymbol{q} = \boldsymbol{\tau} \tag{8.63}$$

pour un robot contrôlé en position à bas niveau dans l'espace des joints, avec des capteurs qui mesure le couple de chaque joint. Pour ce cas il y a deux intégrale dans la loi de commande. On peut interpréter le bloc "loi de comportement" à la Figure 8.17 qui est tout simplement la simulation d'un système masse-ressort-amortisseur qui reçoit en entrée une force réelle lue par un capteur. Le résultat de la simulation est ensuite envoyer à un contrôleur en déplacement. Le robot va donc émuler le système virtuel simulé.

Capsule vidéo
Commande en admittance d'un robot manipulateur
https://youtu.be/SP5bISWmkT0

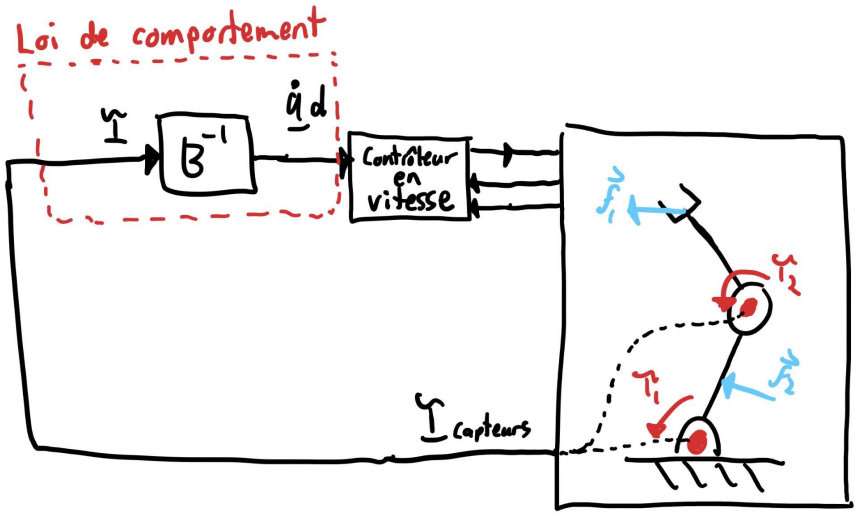


FIGURE 8.16 – Commande de l'admittance au joints d'un robot : schéma bloc

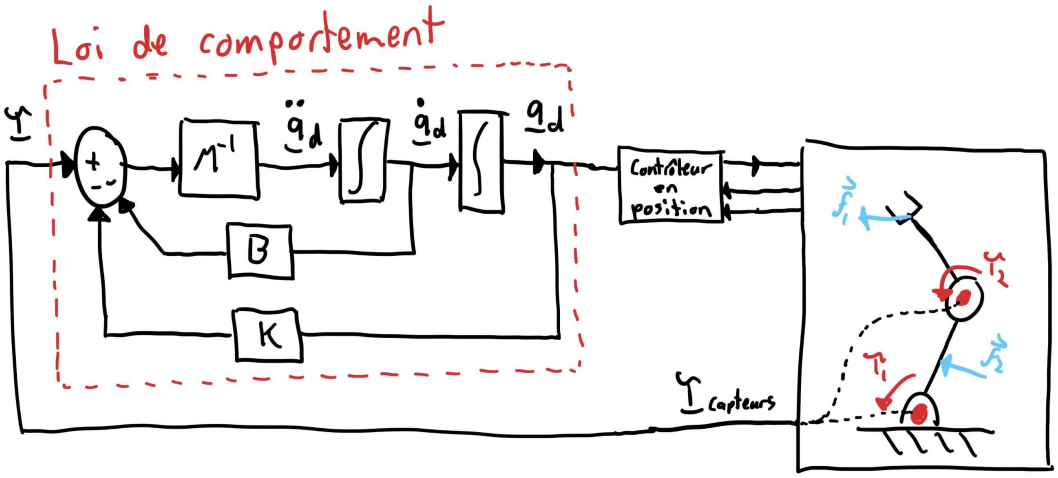


FIGURE 8.17 – Commande de l'admittance au joints d'un robot : schéma bloc

8.5.2 Commande en admittance de l'effecteur

L'approche en admittance dans l'espace de l'effecteur est basée sur une mesure de la force à l'effecteur (généralement avec une cellule de charge) et le contrôle du déplacement de l'effecteur à bas niveau. La Figure 8.18 présente un schéma bloc d'une loi de commande en admittance pour imposer un comportement :

$$B\dot{\boldsymbol{r}} = \boldsymbol{f}_e \tag{8.64}$$

Une chose à noter est que généralement la cellule de charge va mesurer les composantes du vecteur force $\vec{f}_e$ dans une base vectorielle mobile t attachée à l'outil. Il faut donc faire un changement de base avec la matrice de rotation ${}^wR^t$ pour avoir les composantes dans la base vectorielle fixe w dans laquelle le reste des autres vecteurs sont exprimés. La matrice de rotation ${}^wR^t$ doit donc être obtenue basé sur la positon des joints actuels en utilisant la relation de cinématique directe pour l'orientation. La loi de commande de la Figure 8.18 est utilisée pour le mode démonstration (*teach mode*) des robots manipulateur, i.e. le mode ou un opérateur peut déplacer le robot manuellement dans l'espace. Avec cette loi de commande, un humain peut pousser sur l'effecteur du robot et le robot se déplacera alors dans le sens de la poussée avec une vitesse proportionnelle à l'amplitude de la poussée. Si il n'y a aucune poussée le robot reste en place.

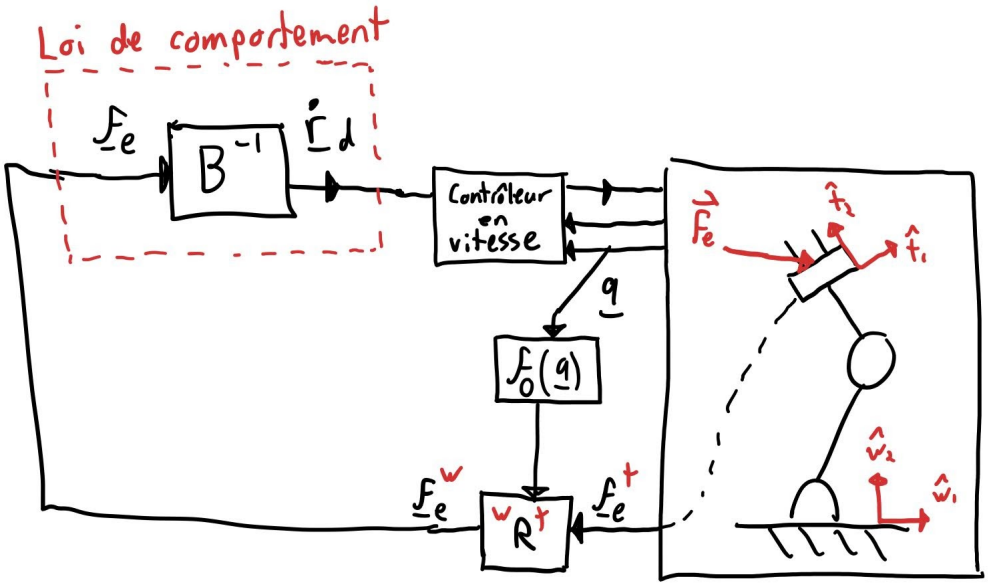


FIGURE 8.18 – Commande de l'admittance à l'effecteur d'un robot avec une loi de comportement de type amortisseur : schéma bloc

La Figure 8.18 montre le schéma bloc d'une loi de commande plus générique pour une loi de comportement égale à une masse-ressort-amortisseur à l'effecteur :

$$M\ddot{\boldsymbol{r}} + B\dot{\boldsymbol{r}} + K\boldsymbol{r} = \boldsymbol{f}_e \tag{8.65}$$

Il est à noté que les Figures 8.18 et 8.19 ne montre pas le détail de la boucle interne du contrôleur en vitesse ou position de l'effecteur. Les méthodes présentées aux sections 8.1 et 8.2 peuvent être utilisée. Une différence à noter pour l'approche en admittance basée sur l'utilisation d'une cellule de charge à l'effecteur du robot, par rapport aux autres approches présentés est que seul les forces qui sont appliquées à l'effecteur (en aval de la cellule de charge) vont influencer la position du robot. Si une force est appliqué sur les liens du robot en amont de la cellule de charge, le contrôleur n'aura pas connaissance de cette force externe qui ne sera donc pas pris en compte dans le calcul du déplacement du robot basée sur la loi de comportement désirée.

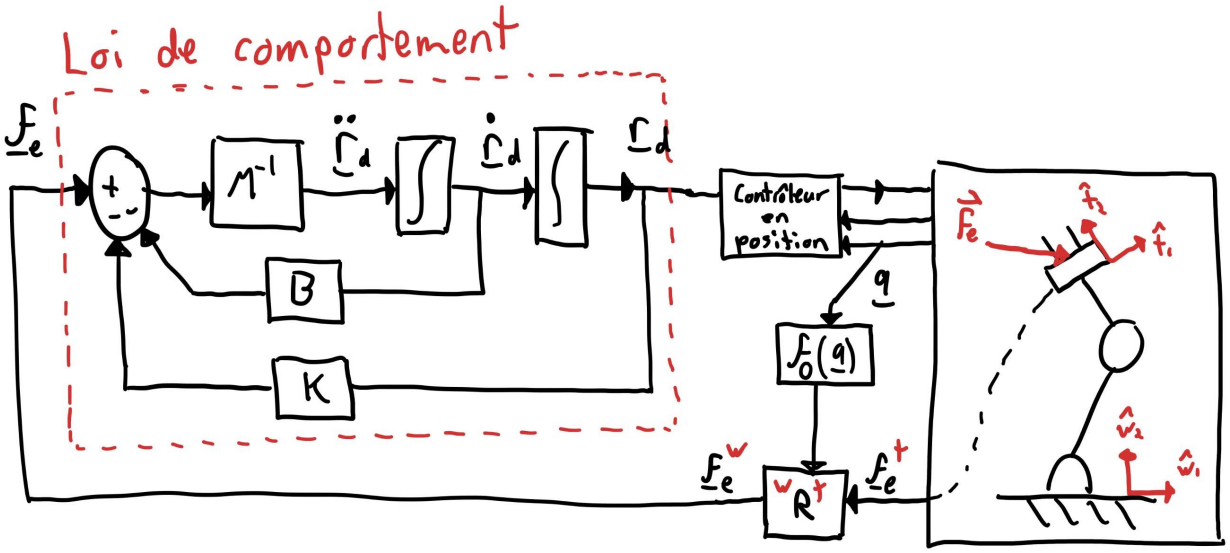


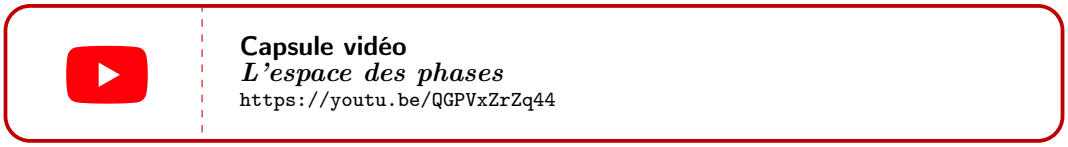
FIGURE 8.19 – Commande de l'admittance à l'effecteur d'un robot avec une loi de comportement de type masse-ressort-amortisseur : schéma bloc

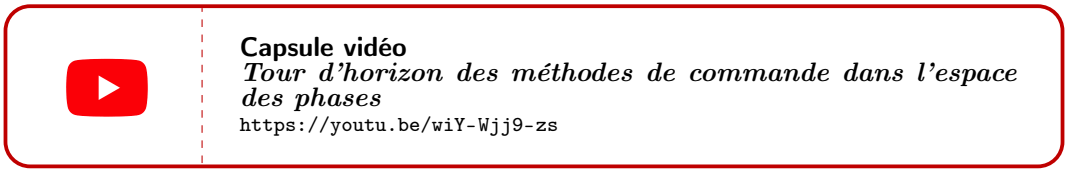
Chapitre 9

Commande des robots manipulateurs II : dynamique

Ce chapitre présente des techniques de commande qui s'applique pour les systèmes robotisés où les entrées du système sont des forces ou couples produites par les actionneurs. Dans ce contexte, le système peut être modéliser comme un système d'équations différentielles non-linéaire d'ordre deux et être représentés par l'équation des manipulateurs décrite au chapitre 6. Les vidéos suivants présentent une introduction, avec un point de vue graphique dans l'espace des phases, aux méthodes pour se contexte :







Au lien suivant, une amorce de code pour tester des lois de commande sur un robot manipulateur à 2 DDL :



9.1 Commande dé-localisée joint par joint

Dans certains situations, il est possible de traiter le problème de commande joint par joint avec des méthodes de commande classique comme des lois de commandes de type PID. Cette approche peut bien fonctionner surtout lorsque le système robotique a des actionneurs avec des grands ratios de réduction, ce qui minimise le couplage dynamique entre les divers DDL. Plus les effets inertiels et plus on désire suivre précisément des trajectoires dynamiques, moins ce type d'approche va être performante.

Si on utilise des lois de commandes indépendantes de type PID pour chaque joint, mais avec le terme intégral à zéro, alors l'approche de commande est équivalente à définir une matrice d'impédance diagonale dans le domaine des joints comme discuté à la section 8.4.1. Une telle approche va être stable mais avec comme principal inconvénient une erreur finale non-nulle, typiquement due à la gravité et/ou de la friction sèche. On a vu qu'il était possible d'ajouter un terme de compensation de gravité mais cela implique de centraliser les calculs et un bon modèle du système. Une alternative intéressante est d'utiliser l'effet intégrateur des PIDs sur chaque joint pour annuler les erreurs finales en position pour chaque joint du à ces effets. L'effet intégrateur va ajouter toutefois une dynamique supplémentaire qui peut créer des oscillations et déstabiliser le système. Il y a donc un compromis entre avec des grands gains intégrateurs pour réduire l'erreur finale rapidement vs. minimiser les oscillations et le risque d'instabilité. L'exercice suivant permet d'explorer ce compromis :

Exercice de code
Simulation d'un manipulateur avec des PIDs
`https://colab.research.google.com/drive/1qaCNY2ohQIbC6dV2ZW4KY6FDM9NhJzH2?usp=sharing`

Il serait possible de faire une analyse de stabilité locale en linéarisant la dynamique d'un robot autour du point d'opération et de faire le choix des gains PIDs avec les outils d'asservissement classique. L'analyse serait toutefois limité à décrire le comportement du robot très proche de se point d'opération. Si on utilise le robot sur une grande plage d'opération qui implique des grande non-linéarités, alors il est beaucoup plus difficile de garantir la stabilité du système et aussi tout simplement d'avoir des gains de PIDs qui vont bien fonctionner pour une grande plage de conditions initiales et positions cibles.

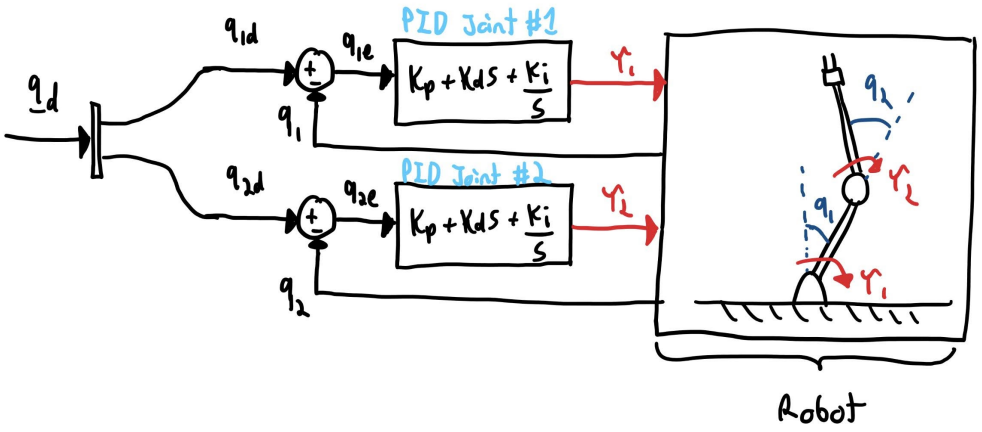


FIGURE 9.1 – Schéma bloc d'un robot avec de PIDs délocalisés

Un avantage non-négligeable pratico-pratique lors de l'implémentation d'une telle approche délocalisée, est que les boucles d'asservissements de chaque joint peuvent être implémentée directement dans l'électronique de puissance des moteurs puisse que les calculs sont indépendants. Typiquement, l'électronique de puissance permet d'avoir des délais de boucle très court, dans l'ordre des dizaines de kHz, alors un ordinateur qui centralise l'information de tous les joints implique généralement plus de délais et des boucles moins rapide.

9.2 Commande avec la méthode du couple calculé

La méthode du couple calculé consiste à utiliser les équations d'un modèle dynamique d'un système robotique pour déterminer les forces à appliquer pour obtenir une accélération cible. Il est ensuite possible de calculer cette accélération cible de sorte à converger vers la position ou trajectoire désirée.

Capsule vidéo
Méthode du couple calculé
`https://youtu.be/QuEhwAUxx5Y`

Exercice de code
Robot avec une loi de commande du couple calculé
`https://colab.research.google.com/drive/19SLbtnJHDen3-EUIyP1BSFVhGs4WSc5J?usp=sharing`

9.2.1 Commande de l'accélération des joints

Premièrement, si un système est pleinement actionné, il est possible d'imposé d'une accélération arbitraire dans l'espace des joints à un instant donné. Pour un système décrit par l'équation des manipulateurs :

$$H(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + d(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = B(\boldsymbol{q})\boldsymbol{u} \tag{9.1}$$

Définition 9.1 Couple calculé : loi de commande:

Le couple calculé est une loi de commande avec comme référence une accélération instantanée cible $\ddot{\boldsymbol{q}}_r$:

$$\boldsymbol{u} = B(\boldsymbol{q})^{-1}\left[H(\boldsymbol{q})\ddot{\boldsymbol{q}}_r + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + d(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q})\right] \tag{9.2}$$

qui assume ici que la matrice B est inversible, donc que le système est pleinement actionné.

Alors si on substitut la loi de commande dans l'équation de la dynamique on obtient :

$$H(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + d(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = B(\boldsymbol{q})B(\boldsymbol{q})^{-1}\left[H(\boldsymbol{q})\ddot{\boldsymbol{q}}_r + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + d(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q})\right] \tag{9.3}$$

$$H(\boldsymbol{q})\ddot{\boldsymbol{q}} = H(\boldsymbol{q})\ddot{\boldsymbol{q}}_r \tag{9.4}$$

$$\ddot{\boldsymbol{q}} = \ddot{\boldsymbol{q}}_r \tag{9.5}$$

on obtient comme résultat que l'accélération du système peut être directement commandée. Comme illustré à la figure 9.2, avec cette loi de commande, la relation entre la référence $\ddot{\boldsymbol{q}}_r$ et la variable à contrôler $\boldsymbol{q}$ est réduite à une relation linéaire, ici un double intégrateur. On réfère aussi parfois à cette méthode comme une boucle linéarisante.

9.2.2 Suivi de trajectoire avec le couple calculé

Lorsqu'on désire qu'un robot suivre une certaine trajectoire, il est possible de définir la référence d'accélération $\ddot{\boldsymbol{q}}_r$ comme une fonction de la trajectoire cible pour obtenir que le robot va converger sur la trajectoire. Si une trajectoire cible est définie comme une fonction du temps pour les coordonnées $\boldsymbol{q}$ et leur deux premières dérivées :

$$\text{Trajectoire désirée : } \ddot{\boldsymbol{q}}_d(t) \quad \dot{\boldsymbol{q}}_d(t) \quad \boldsymbol{q}_d(t) \tag{9.6}$$

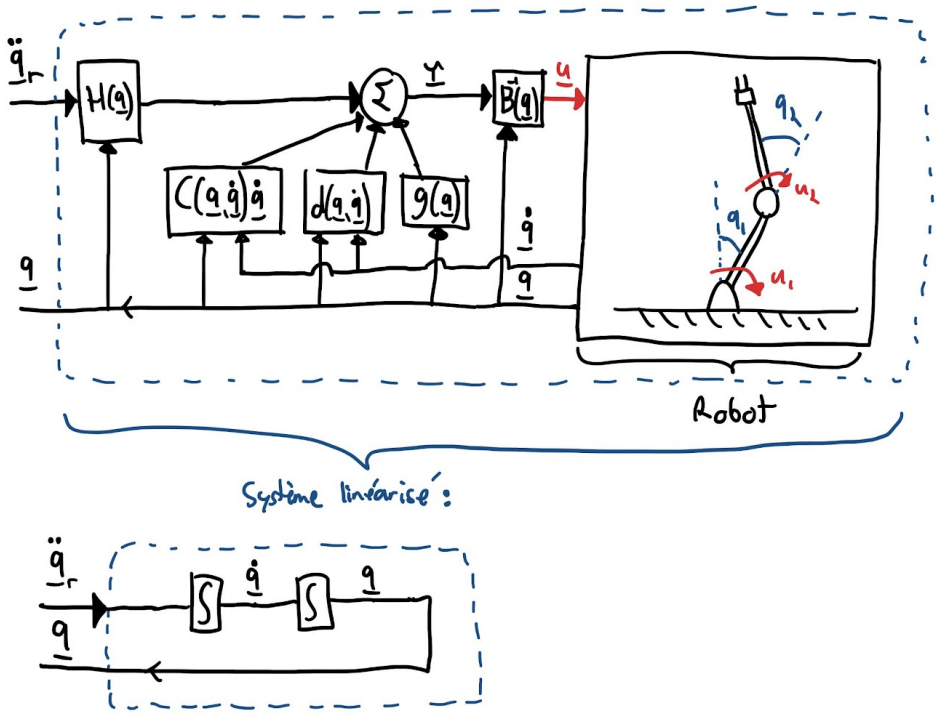


FIGURE 9.2 – Couple calculé et boucle linéarisante.

Définition 9.2 Couple calculé : accélération cible:

Pour un suivi de tracjectoire, la référence d'accélération utilisée par la loi de commande du couple calculé est définit par :

$$\ddot{\boldsymbol{q}}_r = \ddot{\boldsymbol{q}}_d + 2\zeta w \underbrace{(\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}})}_{\dot{\boldsymbol{q}}_e} + w^2 \underbrace{(\boldsymbol{q}_d - \boldsymbol{q})}_{\boldsymbol{q}_e} \tag{9.7}$$

Puisqu'on obtient $\ddot{\boldsymbol{q}} = \ddot{\boldsymbol{q}}_r$ avec la méthode du couple calculé, en combinant cette définition pour $\ddot{\boldsymbol{q}}_r$ on va obtenir :

$$\underbrace{(\ddot{\boldsymbol{q}}_d - \ddot{\boldsymbol{q}})}_{\ddot{\boldsymbol{q}}_e} + 2\zeta w \underbrace{(\dot{\boldsymbol{q}}_d - \dot{\boldsymbol{q}})}_{\dot{\boldsymbol{q}}_e} + w^2 \underbrace{(\boldsymbol{q}_d - \boldsymbol{q})}_{\boldsymbol{q}_e} = 0 \tag{9.8}$$

ce qui représente une équation de la dynamique pour l'erreur d'ordre 2 qui converge exponentiellement vers zéro si on choisi $w^2 > 0$ et $2\zeta w > 0$, ce qui implique que la trajectoire du robot va converger sur la trajectoire désirée.

On voit ici selon cette équation que ces paramètres dans la loi de commande ont été intégré de sorte qu'ils correspondent directement aux paramètres de fréquence naturelle et d'amortissement pour la forme standard d'une équation différentielle d'ordre deux. Ici ζ serait un paramètre de la loi de commande directement relié au dépassement et au temps d'établissement de l'erreur du système en boucle fermée et le w au temps de monté et la bande passante du système en boucle fermée. De façon plus général, dans l'équation (9.8) le terme scalaire w^2 pourrait être remplacé par une matrice K_p et le terme $2\zeta w$ par une matrice K_d. Plutôt qu'avoir alors une dynamique d'erreur identique pour chaque DDL, on pourrait paramétrer des comportements plus varié. Par exemple, on pourrait choisir des matrices diagonales et les termes indépendants sur leur diagonale correspondraient à des valeurs de w et ζ qui pourraient être choisi indépendemment pour chaque DDL du robot.

Donc pour résumé, en assumant que notre modèle dynamique est parfait et que nous avons aussi des mesures exacte pour $\boldsymbol{q}$ et $\dot{\boldsymbol{q}}$, il est possible d'imposer une accélération instantanée $\ddot{\boldsymbol{q}}_r$ arbitraire (si on ignore

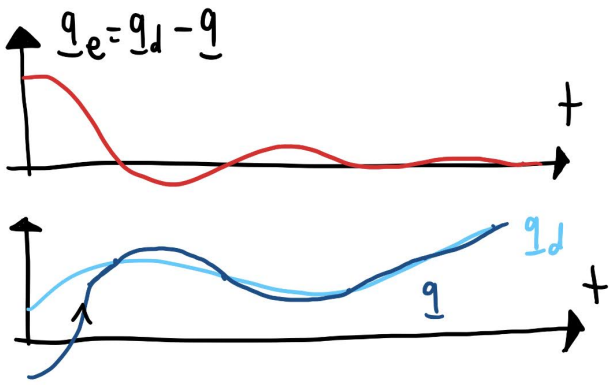


FIGURE 9.3 – Dynamique de l'erreur et convergence sur la trajectoire

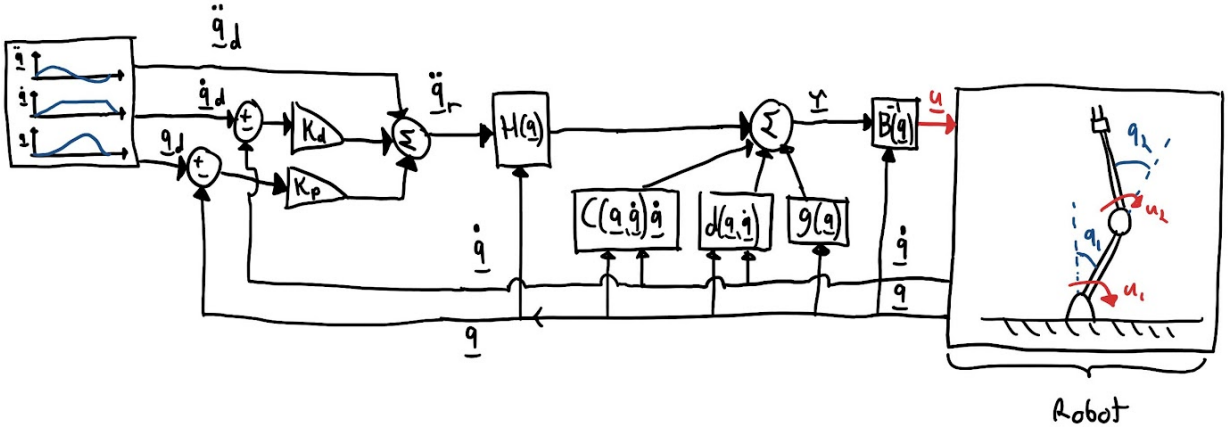


FIGURE 9.4 – Commande par couple calculé pour suivre une trajectoire.

les limites des actionneurs pour le moment). Ensuite, connaissant la trajectoire cible $\boldsymbol{q}_d(t)$ et ses dérivées d'ordre 1 et 2, il est possible d'imposer une accélération de sorte à avoir une erreur par rapport à la trajectoire $\boldsymbol{q}_e(t)$qui tend vers zéro.

Si on regarde comment l'erreur est relié à une force demandée par la loi de commande, on peut interprété cette relation comme une matrice de rigidité, de même que pour un la relation avec la dérivé de l'erreur qui peut être interprété comme une matrice d'amortissement. Les autres termes qui ne sont pas relié à l'erreur ou l'accélération de la trajectoire cible ont tous comme rôle d'annuler des forces internes. Donc si on décortique la loi de commande, on peut regrouper les termes ansi :

$$\boldsymbol{u} = B(\boldsymbol{q})^{-1}\left[H(\boldsymbol{q})\ddot{\boldsymbol{q}}_r + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + d(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q})\right] \tag{9.9}$$

$$\boldsymbol{u} = B(\boldsymbol{q})^{-1}\left[H(\boldsymbol{q})\left(\ddot{\boldsymbol{q}}_d + K_d\dot{\boldsymbol{q}}_e + K_p\boldsymbol{q}_e\right) + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + d(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q})\right] \tag{9.10}$$

$$\boldsymbol{u} = \underbrace{\left[B(\boldsymbol{q})^{-1}H(\boldsymbol{q})\right]\ \ddot{\boldsymbol{q}}_d}_{\text{Termes anticipatif}} + \underbrace{\left[B(\boldsymbol{q})^{-1}H(\boldsymbol{q})K_d\right]\ \dot{\boldsymbol{q}}_e + \left[B(\boldsymbol{q})^{-1}H(\boldsymbol{q})K_p\right]\ \boldsymbol{q}_e}_{\text{Termes réactifs}} + \underbrace{B(\boldsymbol{q})^{-1}\left[C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + d(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q})\right]}_{\text{Termes de compensation}} \tag{9.11}$$

et comparer à une loi de commande plus simple d'impédance dans le domaine des joints, présenté à la section 8.4, qui est en fait aussi équivalent à utiliser des lois de commande de type proportionnelle-dérivée indépendantes pour chacun des joints. Premièrement, en plus de termes réactifs la méthode du couple calculé inclus plusieurs termes de compensation des forces internes du système, ainsi qu'un terme anticipatif si elle est utilisée pour faire du suivi de trajectoire. Ensuite, les termes réactifs ont une structure similaire, des vecteurs-colonnes d'erreur multiplient des matrices de gains. Toutefois, on remarque que dans le cas du

couple calculé les matrices de gains K_p et K_d sont ajusté avec en étant multipliés par la matrice d'inertie $H(\boldsymbol{q})$ et la matrice d'actionnement $B(\boldsymbol{q})^{-1}$. Cet ajustement fait que nos gains dans les matrices K_p et K_d spécifient pas un lien linéaire avec un réaction en force, mais plutôt avec une réaction en accélération, pour laquelle on calcul la force nécessaire avec les matrice H et B.

9.2.3 Compensation de la friction

Dans les méthodes de commande comme celle du couple calculé, la loi de commande vise à compenser pour tous les termes dynamiques, incluant les forces dissipatives comme la friction qui sont difficiles à modéliser. Il faut toutefois prendre garde de de pas sur-estimé l'amplitude de des forces de friction dans une boucle de compensation car cela peut rendre le système instable. Les forces dissipatives aident naturellement à la stabilité d'un système, ils dissipent de l'énergie mécanique et donc forcent le système tranquillement vers un état ou l'énergie est minimum. Une compensation de friction annule donc l'effet dissipatif de cette force, mais le problème est surtout si on surestime de niveau de friction que notre contrôleur compense trop. Dans ce cas l'effet net de la vrai friction plus la compensation par le contrôleur est une injection d'énergie dans le système, la force net pousse dans le même sense que la vitesse, ce qui risque de rendre le système instable.

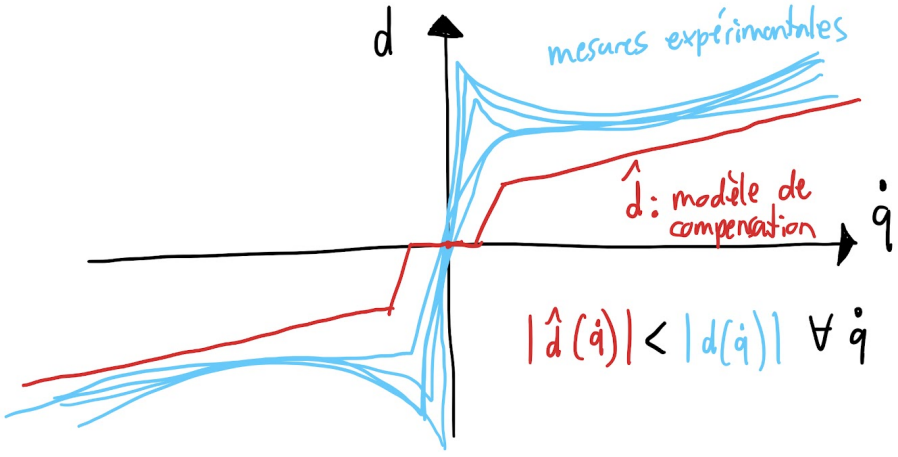


FIGURE 9.5 – Modèle de compensation de friction

Une approche pour éviter cette situation est de toujours avoir un modèle de compensation de friction qui sous-estime l'amplitude de la force de friction de façon conservateur, comme illustré à la figure 9.5. Mathématiquement, pour garantir que la compensation de friction ne déstabilise pas le système, on peut choisi un modèle de compensation de friction $\hat{d}$ avec une amplitude toujours inférieur à la vrai force de friction d pour tout les conditions. Lorsqu'on a un modèle de friction pour chaque joint i qui dépend juste de la vitesse de ce même joint, cette condition s'exprime par :

$$\left|\hat{d}_i(\dot{q}_i)\right| < |d_i(\dot{q}_i)| \quad \forall \dot{q}_i \quad \forall i \tag{9.12}$$

Sous-estimer la friction dans la loi de commande du couple calculé, ou bien ne pas inclure cette compensation, ne met pas typiquement pas en péril la convergence du système sur la cible. La dynamique de l'erreur va toutefois être affecté, les propriétés d'amortissements vont être affectés. Une approche de conception peut être de retirer le terme dérivé K_d de la loi de commande et le terme de compensation de friction, pour laisser la friction naturelle du système s'occuper de l'amortissement du système.

9.2.4 Suivi d'une trajectoire définie dans l'espace de la tâche

À venir !

9.2.5 Limites de la méthode

À venir !

9.2.6 Commande en impédance incluant les effets inertiels

TODO voir notes manuscriptes alex 3 juillet 2022

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

9.3 Commande robuste

La commande robuste est la science de gérer l'incertitude lors de la conception d'une loi de commande. Dans les sections précédentes, nous avions des lois de commande qui utilisaient divers propriétés du modèle cinématique et/ou dynamique du robot. Les démonstrations que ces lois de commande convergeaient assumaient alors que nos modèles utilisés dans les lois de commande étaient exacte. En pratique il y a toujours un certain degré d'incertitude, plus l'incertitude est grande plus il y a intérêt à considérer cet aspect explicitement avec des méthodes de commande robuste.

9.3.1 Approche pour modéliser l'incertitude

Dans cette section, la méthode de commande présentée est basée sur une modélisation de l'incertitude comme un vecteur de forces inconnues $\boldsymbol{w}$ qui s'additionnent aux autres termes de l'équation des manipulateurs :

$$H(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + d(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = B(\boldsymbol{q})\boldsymbol{u} + \boldsymbol{w} \tag{9.13}$$

Ensuite, en fonction de bornes de valeurs minimum et maximum que peut prendre le vecteur $\boldsymbol{w}$ il sera possible de synthétisé des loi de commandes qui vont garantir la convergence malgré cette incertitude.

Capsule vidéo
Grandes familles de méthodes de commande pour gérer l'incertitude
https://youtu.be/hbZBF-0EZEw

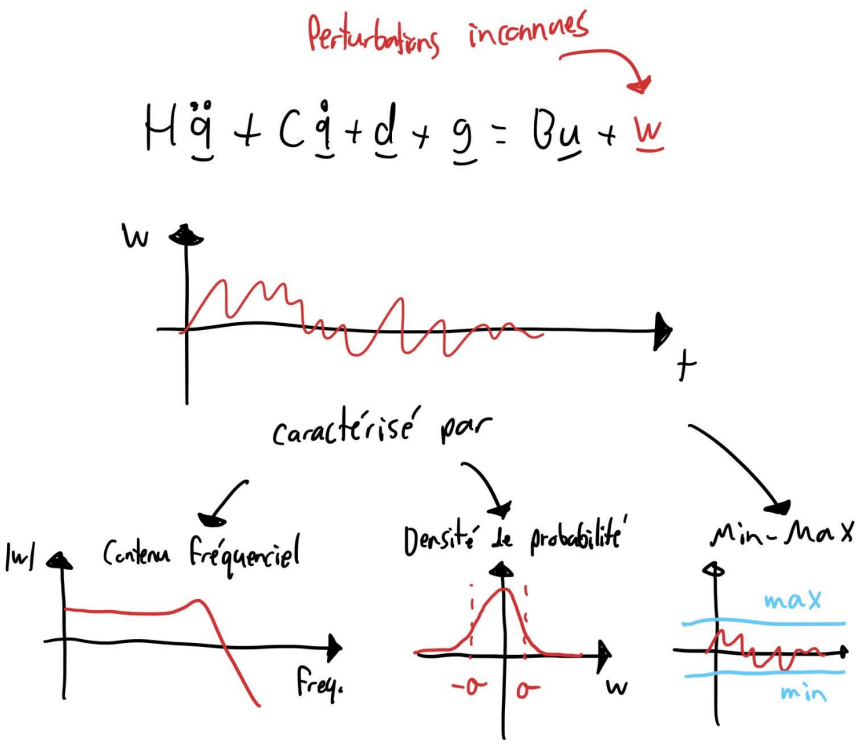


FIGURE 9.6 – Divers métriques pour caractériser l'incertitude

9.4 Méthode du mode glissant

La méthode du mode glissant est une approche de commande qui a comme principal attrait de pouvoir garantir la performance d'un système, malgré la présence de perturbations. En effet, si on connaît les valeurs maximums et mimimums qu'une perturbation peut prendre nous aurons une méthode pour sélectrionner des gains qui peuvent garantir la convergence.

Capsule vidéo
Commande avec le mode glissant
https://youtu.be/0Asg81SBjmk

Premièrement, la stratégie pour garantir la convergence malgré l'incertitude est basé sur le principe de "pousser" très fort sur système vers une trajectoire cible de sorte que peut importe la valeur prise par la perturbation w le système est tout de même forcé sur la trajectoire. Comme illustré à la figure 9.7, toutes les évolutions possible du système sont toujours dans la bonne direction malgré l'incertitude avec cette approche.

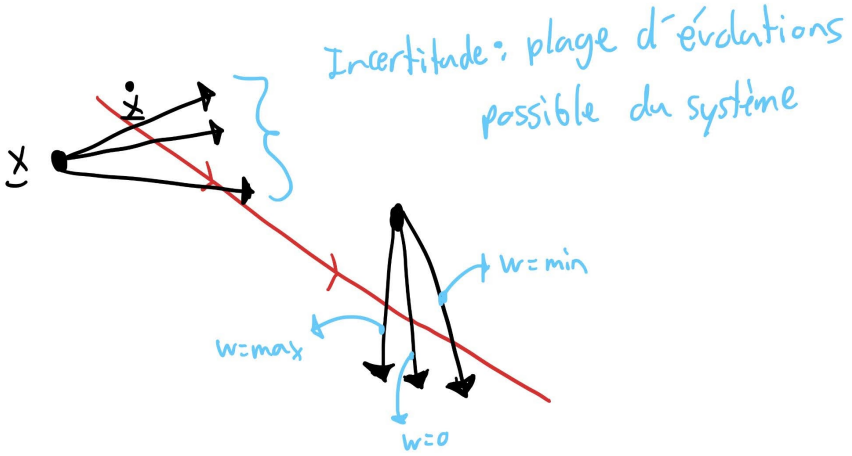


FIGURE 9.7 – Principe de fonctionnement pour la commande par surface glissante

Exercice de code
Robot avec une loi de commande du mode glissant
https://colab.research.google.com/drive/
1DQhoq-NbWHHI6RSf-KQXwoW7SLwfldgk?usp=sharing

Avec la méthode de la surface glissante, le système va être forcé sur une trajectoire définie par l'équation suivante :

$$\boldsymbol{s} = \dot{\boldsymbol{q}}_e + \lambda \boldsymbol{q}_e = \boldsymbol{0} \tag{9.14}$$

où les variables s sont des coordonnées pour chaque DDL du système, une combinaison de la position et de la vitesse (relatif à une trajectoire cible), choisi avec un paramètre λ de sorte que $s = 0$ correspond à une trajectoire qui converge exponentiellement vers $q_e = 0$. En effet,

$$\dot{\boldsymbol{q}}_e = -\lambda \boldsymbol{q}_e \quad \text{si} \quad \boldsymbol{s} = \boldsymbol{0} \tag{9.15}$$

une équation différentielle linéaire d'ordre un pour chaque DDL avec comme solution :

$$\boldsymbol{q}_e(t) = \boldsymbol{q}_e(0)\, e^{-\lambda t} \tag{9.16}$$

Comme illustré à la figure 9.8, les variables s sont donc des coordonnées pour lesquelles le sous-ensemble $s = 0$ implique une convergence exponentielle vers l'origine, c'est ce sous-ensemble qu'on appelle la surface glissante.

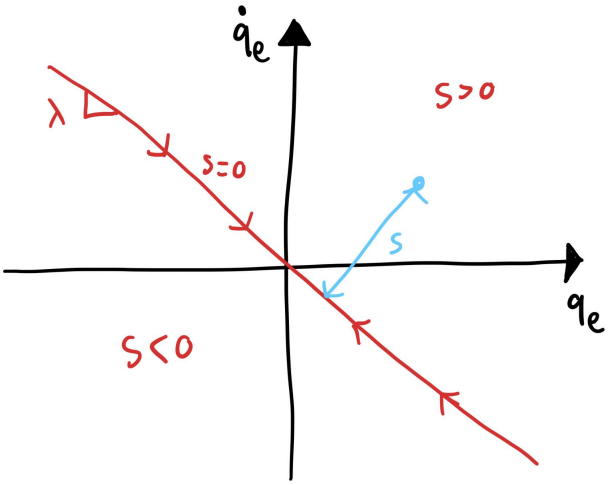


FIGURE 9.8 – Surface glissante

La loi de commande par surface glissante va avoir deux composantes :

$$\boldsymbol{u} = B(\boldsymbol{q})^{-1}\left[\underbrace{\hat{\boldsymbol{\tau}}}_{\text{Couple équivalent}} + \underbrace{k\,\operatorname{sgn}\boldsymbol{s}}_{\text{Couple discontinu}}\right] \tag{9.17}$$

Le premier terme, le couple équivalent, est un vecteur de couple calculé de façon continue qui compense plusieurs forces internes de façon similaire à la méthode du couple calculé. Le deuxième terme, est un vecteur de couple discontinus, ou chaque composant peux prendre deux valeur selon la valeur de la variable s associé à ce DDL. La figure 9.9 illustre le schéma bloc de cette loi de commande. Le couple équivalent $\hat{\boldsymbol{\tau}}$

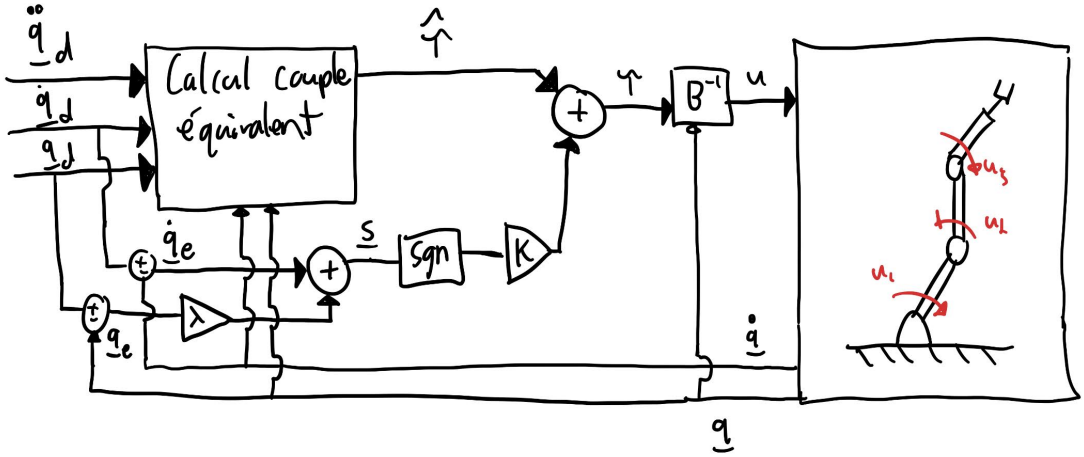


FIGURE 9.9 – Surface glissante

est calculé en fonction de la trajectoire désirée et l'état actuel du robot ainsi :

$$\hat{\boldsymbol{\tau}} = H\ddot{\boldsymbol{q}}_r + C\dot{\boldsymbol{q}}_r + \boldsymbol{d} + \boldsymbol{g} \tag{9.18}$$

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

avec :

$$\dot{\boldsymbol{q}}_r = \dot{\boldsymbol{q}}_d + \lambda \boldsymbol{q}_e \tag{9.19}$$
$$\ddot{\boldsymbol{q}}_r = \ddot{\boldsymbol{q}}_d + \lambda \dot{q}_e \tag{9.20}$$
$$\tag{9.21}$$

Le signal total de commande u calculé aura donc typiquement une allure continue lorsque le système est loin de la surface glissante $s = 0$ et le signal fera des saut rapide entre deux valeures de u lorsque le système est proche de la surface glissante, comme illustré aux figures 9.10 et 9.11. La fréquence des oscillations entre les valeurs de u va dépendre en pratique de la fréquence de la boucle de contrôle. Les oscillations vont aussi se refléter sur la trajectoire du système comme un va et vient d'un côté à l'autre de la surface glissante.

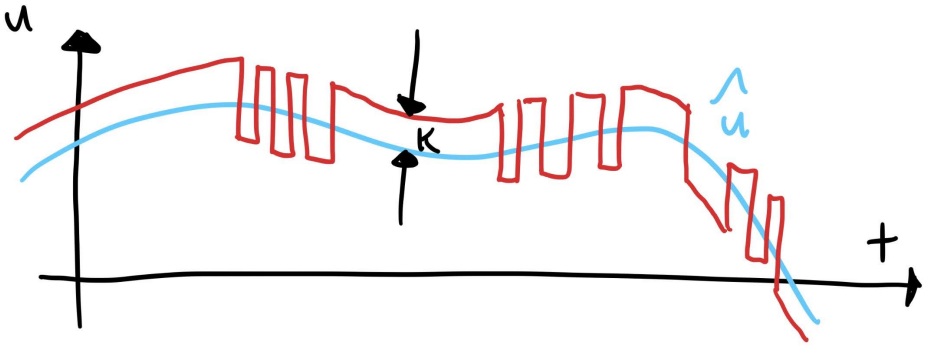


FIGURE 9.10 – Surface glissante

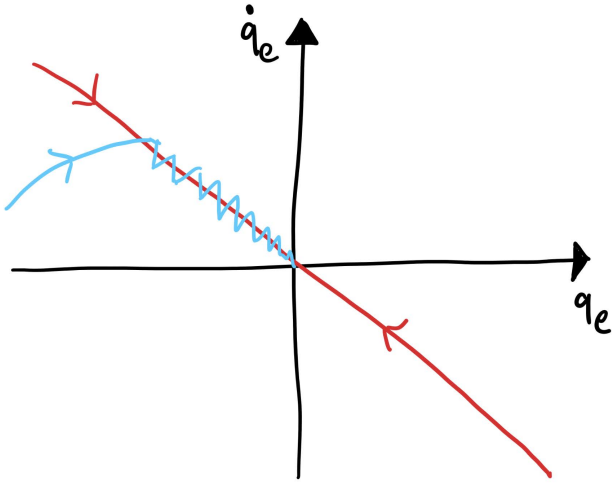


FIGURE 9.11 – Surface glissante

9.5 Commande adaptative

La commande dite adaptative, est une méthode de alternative pour gérer l'incertitude lorsque celle-ci est caractérisé par une erreur sur les paramètres dans l'équation dynamique. Cette approche s'applique bien pour les scénarios ou la principale source d'incertitude n'est pas des perturbations externes, mais plutôt une erreur d'estimation sur les paramètres cinématiques ou inertiels d'un système robotique. La stratégie d'une loi de commande adaptative peut se résumer en deux processus parallèles : **1) identification système :** on utilise les signaux d'entrée et de sortie du système pour effectuer une estimation en continue des paramètres du modèle, et **2) loi de commande :** on utilise l'approximation des paramètres dans une loi de commande de type couple calculé, ou autre approche basé sur un modèle du système. Cette approche est particulièrement adapté lorsqu'un système robotisé va avoir un chargement inconnu lors d'une opération par exemple.

Capsule vidéo
Exemple de loi de commande adaptative
`https://youtu.be/vmYjad6SOmU`

Apprentissage machine :

Le principe d'adaptation peut être vue comme un forme d'apprentissage machine en ligne. Avec une loi de commande adaptative, on chercher à "apprendre" automatiquement les bons paramètres d'une loi de commande (vecteur-colonne $\boldsymbol{a}$) à l'aide des données observées. La matrice P de gain d'adaptation peut être vue comme des paramètres de taux d'apprentissage.

9.5.1 Approche pour modéliser l'incertitude

Dans cette section, la méthode de commande présentée est basée sur une modélisation de l'incertitude ou on considère que des paramètres dans le modèle dynamique sont des approximations inexactes dans notre loi de commande $\hat{a}_i$. Par exemple, la masse d'un robot ou la longueur sont toujours approximé à un certain degrée lorsqu'on code une loi de commande. On va ici noter les paramètres exactes a_i, les approximations $\hat{a}_i$, et l'erreur sur un paramètre sera a_{ei}. Les paramètres seront regroupés des vecteur-colonnes :

$$\underbrace{\boldsymbol{a}_e}_{\text{erreur}} = \underbrace{\boldsymbol{a}}_{\text{réel}} - \underbrace{\hat{\boldsymbol{a}}}_{\text{approximation}} \tag{9.22}$$

Un atout qui sera exploité pour les systèmes dynamiques qui peuvent être représenté par l'équation des manipulateurs, est qu'on peut struturer l'équation de la dynamique comme une multiplication d'une matrice de termes connus Y, et du vecteur-colonnes qui regroupe tout les paramètres constants de l'équations.

$$H(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + d(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{g}(\boldsymbol{q}) = \boldsymbol{\tau} \tag{9.23}$$

$$Y(\boldsymbol{q},\dot{\boldsymbol{q}},\ddot{\boldsymbol{q}})\,\boldsymbol{a} = \boldsymbol{\tau} \tag{9.24}$$

Les termes connus dans la matrice Y vont être des fonctions des accélérations, vitesses et positions du robot à un instant donné. Par exemple, on pourrait avoir des termes tel que :

$$Y_{ii} \in \{\ddot{q}_i\ ,\ \dot{q}_i\ ,\ \dot{q}_i^2\ ,\ \dot{q}_i\dot{q}_j\ ,\ \cos q_i\ ,\ \sin q_i\dot{q}_i^2, ...\} \tag{9.25}$$

et dans le vecteur a des combinaisons de paramètres géométriques, inertiel et autres qui multiplient les termes de Y pour obtenir l'équation des manipulateurs :

$$a_i \in \{ml^2\ ,\ mgl\ ,\ b_i\ ,\ ...\} \tag{9.26}$$

Exemple 9.1 Séparation des paramètres inconnus pour un pendule simple:

Pour un pendule simple avec de la friction sèche, malgré les non-linéarités dans les équations il est possible de séparer les termes connus (mesurables) qui sont mesurable et les paramètres constants :

$$ml^2\ddot{q} + \mu \operatorname{sgn}\dot{q} + mgl\sin q = \tau \tag{9.27}$$

$$\underbrace{\begin{bmatrix} \ddot{q} & \operatorname{sgn}\dot{q} & \sin q \end{bmatrix}}_{Y} \underbrace{\begin{bmatrix} ml^2 \\ \mu \\ mgl \end{bmatrix}}_{\boldsymbol{a}} = \tau \tag{9.28}$$

Si on mesure l'angle q (et ses dérivés) on peut construire tout les termes de la matrice Y. Les paramètres constants sont impliqué linéairement par rapport aux termes de Y.

9.5.2 Loi de commande

Cette section présente une approche de commande adaptative générique pour un système pleinement actionné représenté par l'équation des manipulateurs. La loi de commande a une structure similaire aux lois de commandes présenté dans ce chapitre : un terme de compensation de dynamique et un terme supplémentaire pour garantir la convergence sur une cible. Ici la grande différence est que le terme de compensation dynamique est calculé avec l'approximation actuel des paramètres inconnus $\hat{\boldsymbol{a}}$:

$$\boldsymbol{\tau} = \underbrace{Y_r\hat{\boldsymbol{a}}}_{\text{compensation dynamique}} + \underbrace{K\boldsymbol{s}}_{\text{rétroaction de type PD}} \tag{9.29}$$

où K est une matrice de paramètres de gains.

Ensuite, l'approximation des paramètres inconnus va être continuellement mise à jour avec l'équation suivante qui dicte la dérivée temporelle de nos approximations comme une fonction de variables mesurables :

$$\dot{\hat{\boldsymbol{a}}} = -P^T Y_r^T \boldsymbol{s} \tag{9.30}$$

où P est une matrice inversible de gains d'adaptation. La loi de commande va ici avoir des états internes et une mémoire (la consigne $\boldsymbol{\tau}$ peut prendre différentes valeur pour un état donné du système physique selon l'historique). Le vecteur d'approximation des paramètres inconnus $\hat{\boldsymbol{a}}$ constitue des états internes qui vont être incrémenté continuellement avec un schéma d'intégration :

$$\hat{\boldsymbol{a}}(t) = \int_{t_0}^{t} \left(-P^T Y_r^T \boldsymbol{s}\right) dt + \hat{\boldsymbol{a}}(t_0) \tag{9.31}$$

Des variables intermédiaires on été utilisés dans les équations précédantes et sont définies si dessous :

$$\boldsymbol{q}_e = \boldsymbol{q}_d - \boldsymbol{q} \tag{9.32}$$

$$\dot{\boldsymbol{q}}_r = \dot{\boldsymbol{q}}_d + \lambda \boldsymbol{q}_e \tag{9.33}$$

$$\ddot{\boldsymbol{q}}_r = \ddot{\boldsymbol{q}}_d + \lambda \dot{\boldsymbol{q}}_e \tag{9.34}$$

$$\boldsymbol{s} = \dot{\boldsymbol{q}}_e + \lambda \boldsymbol{q}_e = \dot{\boldsymbol{q}}_r - \dot{\boldsymbol{q}} \tag{9.35}$$

Il est aussi à noter qu'il y a deux version de la matrice Y :

$$Y\boldsymbol{a} = H\ddot{\boldsymbol{q}} + C\dot{\boldsymbol{q}} + \boldsymbol{d} + \boldsymbol{g} = \boldsymbol{\tau} \tag{9.36}$$

$$Y_r\boldsymbol{a} = H\ddot{\boldsymbol{q}}_r + C\dot{\boldsymbol{q}}_r + \boldsymbol{d} + \boldsymbol{g} \tag{9.37}$$

Y dépend seulement des positions, vitesses et accélération, alors que Y_r n'utilise pas l'accélération réelle mais est plutôt basé sur l'erreur :

$$Y = Y(\ddot{\boldsymbol{q}}, \dot{\boldsymbol{q}}, \boldsymbol{q}) \tag{9.38}$$

$$Y_r = Y_r(\ddot{\boldsymbol{q}}_r, \dot{\boldsymbol{q}}_r, \dot{\boldsymbol{q}}, \boldsymbol{q}) \tag{9.39}$$



Exercice de code
Commande adaptative pour un pendule simple
https://colab.research.google.com/drive/1J8C-CvJdrhDU4q1294057Qopn3tnEBNi?usp=sharing

Démonstration. Il est possible de démontrer que la loi de commande proposée dans cette section, incluant la loi d'adaptation, va faire converger le système sur la trajectoire désirée pour n'importe quel valeur initial d'estimation des paramètres $\hat{\boldsymbol{a}}$.

Avec comme fonction de Lyapunov :

$$V = \frac{1}{2}\boldsymbol{s}^T H \boldsymbol{s} + \frac{1}{2}\hat{\boldsymbol{a}}^T P^{-1} \boldsymbol{a}_e \tag{9.40}$$

Il est possible de démontrer que la dérivée temporelle va être strictement négative sauf lorsque $\boldsymbol{s} = \boldsymbol{0}$:

$$\dot{V} = \boldsymbol{s}^T \left(H\dot{\boldsymbol{s}} + 2\dot{H}\boldsymbol{s} \right) + \dot{\hat{\boldsymbol{a}}}^T P^{-1} \boldsymbol{a}_e \tag{9.41}$$

$$\dot{V} = \boldsymbol{s}^T \left(H\ddot{\boldsymbol{q}}_r - \boldsymbol{\tau} + C\dot{\boldsymbol{q}} + \boldsymbol{d} + \boldsymbol{g} + 2\dot{H}\dot{\boldsymbol{q}}_r - 2\dot{H}\dot{\boldsymbol{q}} \right) + \dot{\hat{\boldsymbol{a}}}^T P^{-1} \boldsymbol{a}_e \tag{9.42}$$

$$\dot{V} = -\boldsymbol{s}^T K \boldsymbol{s} + \underbrace{\boldsymbol{s}^T Y_r \boldsymbol{a}_e + \dot{\hat{\boldsymbol{a}}}^T P^{-1} \boldsymbol{a}_e}_{0} \tag{9.43}$$

$$\dot{V} = -\boldsymbol{s}^T K \boldsymbol{s} < 0 \quad \forall \quad \boldsymbol{s} \neq \boldsymbol{0} \tag{9.44}$$

En utilisant le lemme de Barbalat (voir chapitre 20), il est possible de conclure que $\boldsymbol{s} \to \boldsymbol{0}$ ce qui implique qu'on va converger sur la trajectoire car $\boldsymbol{q}_e \to \boldsymbol{0}$. Il est toutefois important de noter que cela n'implique pas que $\boldsymbol{a}_e \to \boldsymbol{0}$. □

Exemple 9.2 Loi de commande et d'adaptation pour un pendule simple:

Si on reprend l'exemple du pendule simple décrit par l'équation suivante :

$$\underbrace{\begin{bmatrix} \ddot{q} & \operatorname{sgn}\dot{q} & \sin q \end{bmatrix}}_{Y} \underbrace{\begin{bmatrix} ml^2 \\ \mu \\ mgl \end{bmatrix}}_{\boldsymbol{a}} = \tau \tag{9.45}$$

La loi de commande serait alors :

$$\tau = Y_r \hat{\boldsymbol{a}} + Ks \tag{9.46}$$

$$\tau = \begin{bmatrix} \ddot{q}_r & \operatorname{sgn}\dot{q} & \sin q \end{bmatrix} \begin{bmatrix} \widehat{ml^2} \\ \widehat{\mu} \\ \widehat{mgl} \end{bmatrix} + K(\dot{q}_e + \lambda q_e) \tag{9.47}$$

$$\tau = \widehat{ml^2}(\ddot{q}_d + \lambda\dot{q}_e) + \widehat{\mu}\operatorname{sgn}\dot{q} + \widehat{mgl}\sin q + K(\dot{q}_e + \lambda q_e) \tag{9.48}$$

$$\tau = \widehat{ml^2}\ddot{q}_d + \widehat{\mu}\operatorname{sgn}\dot{q} + \widehat{mgl}\sin q + (\widehat{ml^2}\lambda + K)\dot{q}_e + K\lambda q_e \tag{9.49}$$

et la loi d'adaptation serait alors :

$$\dot{\hat{\boldsymbol{a}}} = -P^T Y_r^T s \tag{9.50}$$

$$\frac{d}{dt}\begin{bmatrix} \widehat{ml^2} \\ \widehat{\mu} \\ \widehat{mgl} \end{bmatrix} = -\begin{bmatrix} p_1 & 0 & 0 \\ 0 & p_2 & 0 \\ 0 & 0 & p_3 \end{bmatrix} \begin{bmatrix} \ddot{q}_r \\ \operatorname{sgn}\dot{q} \\ \sin q \end{bmatrix} (\dot{q}_e + \lambda q_e) \tag{9.51}$$

$$\frac{d}{dt}\begin{bmatrix} \widehat{ml^2} \\ \widehat{\mu} \\ \widehat{mgl} \end{bmatrix} = -\begin{bmatrix} p_1(\ddot{q}_d + \lambda\dot{q}_e) \\ p_2 \operatorname{sgn}\dot{q} \\ p_3 \sin q \end{bmatrix} (\dot{q}_e + \lambda q_e) \tag{9.52}$$

9.5.3 Estimation avec les moindres carrés

À venir !

9.5.4 Adaptation à des erreurs dans la matrice jacobienne

À venir !

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

9.6 Commande hybride en position et force

À venir !

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

9.7 Méthode d'analyse de la stabilité

L'analyse de la stabilité des lois de commande pour les robots manipulateurs repose typiquement sur la **Méthode Directe de Lyapunov**, voir détails au chapitre 20. L'objectif est généralement de prouver que l'erreur de suivi de trajectoire converge vers zéro, c'est-à-dire de démontrer la **stabilité asymptotique globale** de l'origine de la dynamique de l'erreur.

La stratégie typique pour analyser la convergence d'un robot avec une loi de commande non-linéaire, est de définir une fonction candidate de Lyapunov qui consiste en la vrai énergie cinétique du robot, plus une énergie virtuelle potentielle $E_{vir}(\mathbf{q}_e)$ associée à l'erreur en position :

$$V = \frac{1}{2}\dot{\mathbf{q}}^T H(\mathbf{q})\dot{\mathbf{q}} + E_{vir}(\mathbf{q}_e) \tag{9.53}$$

Par exemple, pour une loi de commande en impédance, on peut prendre l'énergie dans les ressorts virtuels. On calcul la dérivée temporelle de cette fonction :

$$\dot{V} = \dot{\mathbf{q}}^T H(\mathbf{q})\ddot{\mathbf{q}} + \frac{1}{2}\dot{\mathbf{q}}^T \dot{H}(\mathbf{q})\dot{\mathbf{q}} + \dot{E}_{vir} \tag{9.54}$$

$$\dot{V} = \dot{\mathbf{q}}^T H(\mathbf{q})\ddot{\mathbf{q}} + \frac{1}{2}\dot{\mathbf{q}}^T \dot{H}(\mathbf{q})\dot{\mathbf{q}} + \frac{\partial E_{vir}(\mathbf{q}_e)}{\partial \mathbf{q}}\dot{\mathbf{q}} \tag{9.55}$$

Si les équations du mouvement du robot sont :

$$H\ddot{\mathbf{q}} + C\dot{\mathbf{q}} + \mathbf{d} + \mathbf{g} = \tau \tag{9.56}$$

et la loi de commande est :

$$\tau = -\frac{\partial E_{vir}(\mathbf{q}_e)}{\partial \mathbf{q}^T} + \mathbf{g} \tag{9.57}$$

Donc ici on suppose que la loi de commande est un terme qui est le gradient de l'énergie virtuelle associée à l'erreur et une compensation de gravité. On peut substituer dans l'équation pour obtenir :

$$\dot{V} = \dot{\mathbf{q}}^T \left(-\frac{\partial E_{vir}}{\partial \mathbf{q}^T} + \mathbf{g} - C\dot{\mathbf{q}} - \mathbf{d} - \mathbf{g} \right) + \frac{1}{2}\dot{\mathbf{q}}^T \dot{H}(\mathbf{q})\dot{\mathbf{q}} + \dot{E}_{vir} \tag{9.58}$$

$$\dot{V} = \dot{\mathbf{q}}^T \left(-\frac{\partial E_{vir}}{\partial \mathbf{q}^T} + \mathbf{g} - C\dot{\mathbf{q}} - \mathbf{d} - \mathbf{g} + \frac{1}{2}\dot{H}(\mathbf{q})\dot{\mathbf{q}} + \frac{\partial E_{vir}}{\mathbf{q}^T} \right) \tag{9.59}$$

$$\dot{V} = \dot{\mathbf{q}}^T \left(-\frac{\partial E_{vir}}{\partial \mathbf{q}^T} + \mathbf{g} - \mathbf{d} - \mathbf{g} + \frac{\partial E_{vir}}{\mathbf{q}^T} \right) + \underbrace{\frac{1}{2}\dot{\mathbf{q}}^T \dot{H}\dot{\mathbf{q}} - \dot{\mathbf{q}}^T C\dot{\mathbf{q}}}_{0} \tag{9.60}$$

$$\dot{V} = -\dot{\mathbf{q}}^T \mathbf{d} \leq 0 \tag{9.61}$$

Ce qui permet de prouver une convergence asymptotique vers $\mathbf{q}_e = 0$

9.8 Commande optimale

Jusqu'à maintenant, les méthodes de commande présentées la section 8 et la section 9, étaient basé sur l'hypothèse qu'on pourrait résumé à dire que **les actionneurs du système étaient suffisamment nombreux, bien positionnés et puissants pour imposer une accélération arbitraire au système robotique**. Si cette condition n'est pas remplie, le problème de conception d'une loi de commande devient plus compliqué. Dans cette situation il faut gérer **une planification de l'évolution du système sur un horizon de temps futur** et une méthodologie appropriée pour accomplir cela est appelée la **commande optimale** où les effets long-termes des consignes choisies par une loi de commande sont encodés dans une fonction de coût. De plus, même si le robot à commander rencontre les conditions pour utiliser les techniques qui imposent un accélération, la commande optimale est aussi un outils pertinent à utiliser aussi si on désire utiliser avec parcimonie les actionneurs pour, par exemple, **accomplir une tâche en utilisant un minimum d'énergie**.

Note sur la séquence de lecture :

Comme les méthodes de commande optimale s'applique à une classe de systèmes plus large, incluant les véhicules, ce sujet est reporté à un chapitre ultérieur après ceux qui introduisent les véhicules.

Chapitre 10

Génération de trajectoires

10.1 Introduction

Dans les précédents chapitre ont présentés des lois de commandes qui calculent les commandes à envoyer aux actionneurs pour obtenir soit des positions, des vitesses, des accélérations ou des forces désirée. Certaines loi de commandes nécessitait seulement une position finale cible alors que pour d'autre nous devions spécifier une position, une vitesse et une accélération en tout temps sur une trajectoire cible. Plusieurs tâches en robotique nécessite donc une couche intermédiaire entre des spécifications à haut niveau pour une tâche et les références qui doivent être fournit à une boucle de commande à bas niveau, cette étape intermédiaire est appelée la planification de trajectoire.



Capsule vidéo
Introduction à la planification de trajectoires
`https://youtu.be/_SQumKYdFnM`

Pourquoi donner directement la position finale désirée à un contrôleur bas niveau en position n'est pas toujours approprié ? L'étape de planification de trajectoire est généralement essentiel lorsqu'un robot doit faire un mouvement de grande amplitude, i.e. lorsque la position cible est éloignée de la position actuelle. Les loi de commandes bas-niveau sont typiquement conçu pour garantir la convergence vers la position cible mais pas le détail du chemin pour s'y rendre ni la vitesse d'approche ; plus la cible est loin, plus il va y avoir de raison de mieux contrôler les étapes intermédiaire pour ce rendre à la cible. Voici une liste non-exhaustive de raisons pour mieux contrôler le trajet jusqu'à une cible :

1. **Obstacles et contraintes en position :** Parfois il y a des positions/configurations sur lesquelles on ne veux pas que le système robotique ce retrouve (collisions avec des obstacles, singularités, etc.). Les lois de commande simples vont typiquement allez directement droit vers la cible et possiblement la dépasser avant de se stabiliser sur celle-ci sans tenir compte de contraintes. Un planificateur de trajectoire peut déterminer un chemin à suivre pour atteindre la cible qui évite des obstacles et positions inadéquates.
2. **Contraintes en vitesse :** Les contrôleurs simples vont typiquement aller vers la cible avec une vitesse proportionnelle avec l'erreur initiale sans tenir compte des limites physiques. Si des saturations à bas niveau font que la vitesse réelle ne suit pas la vitesse commandée le comportement et la convergence de la loi de commande seront incertains. De plus, pour des raisons de sécurité il peut être nécessaire de limiter la vitesse d'un robot. Un planificateur de trajectoire peut déterminer un profil de vitesse à suivre qui respecte des contraintes.
3. **Faisabilité dynamique :** Finalement, surtout pour des robots dynamiques et/ou qui bougent rapidement, les lois de commande de base peuvent facilement se retrouver dans des situations ou ils demande des niveaux d'accélération ou de forces aux actionneurs qui se sont pas physiquement possible. Un planificateur peut permettre de déterminer une séquence d'action physiquement possible qui vont mener le robot jusqu'à la cible tout en respectant des contraintes.

10.2 Chemin et profil temporel

Système de coordonnées pour définir une trajectoire

On peut définir des trajectoires dans plusieurs systèmes de coordonnées. Toutefois, pour simplifier la notation, nous présenterons ici d'abord plusieurs concepts en considérant que le trajet est toujours définit dans l'espace des joints d'un robot. Nous verrons ensuite que ces concepts se généralisent pour d'autres systèmes de coordonnées.

Capsule vidéo
Chemin et profil temporel
https://youtu.be/p0NrGrDqFaU

Il est parfois utile d'utiliser le concept d'un chemin pour découpler le problème de planification de trajectoire en deux étapes : 1) trouver un chemin, une description géométrique d'une séquence de configuration $\boldsymbol{q}$ reliant le point de départ et la cible ; et ensuite 2) déterminer un profil temporel de vitesse sur ce chemin pour fixer une trajectoire temporelle.

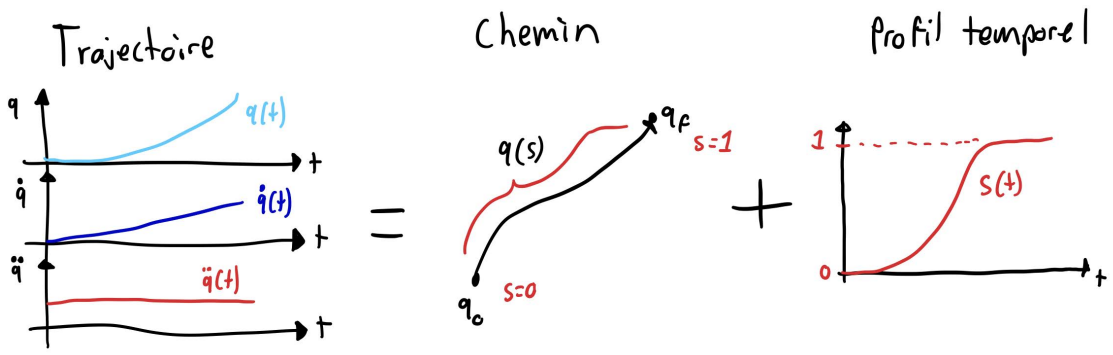


FIGURE 10.1 – Trajectoire (dans l'espace des joints), chemin et profil temporel

Trajectoire Une trajectoire est une fonction qui détermine la position d'un robot comme une fonction du temps $\boldsymbol{q}(t)$. Normalement il est désirable d'avoir des fonctions qui sont dérivables deux fois, i.e. que la dérivé temporelle de cette fonction ne soit pas discontinue, ce qui impliquerait des accélérations d'amplitude infinie. La trajectoire est donc ultimement les trois fonctions suivantes :

$$\text{Trajectoire :} \quad \boldsymbol{q}(t), \dot{\boldsymbol{q}}(t), \ddot{\boldsymbol{q}}(t) \quad \text{avec} \quad t \in [0, t_f] \tag{10.1}$$

Chemin Une chemin est une fonction qui détermine la position d'un robot comme une fonction continue d'une variable scalaire $s \in [0, 1]$, qui est égale à 0 au début du chemin et à 1 à la fin du chemin.

$$\text{Chemin :} \quad \boldsymbol{q}(s) \quad \text{avec} \quad s \in [0, 1] \tag{10.2}$$

Profil temporel Un profil temporel est une trajectoire pour une variable scalaire intermédiaire qui caractérise la progression du système sur un chemin :

$$\text{Profil temporel :} \quad s(t), \dot{s}(t), \ddot{s}(t) \quad \text{avec} \quad t \in [0, t_f] \tag{10.3}$$

Lorsqu'on construit une fonction de trajectoire basé sur un chemin et un profil temporel on obtient les

relations suivantes en appliquant une dérivée en chaîne :

$$\boldsymbol{q}(t) = \boldsymbol{q}(s(t)) \tag{10.4}$$

$$\dot{\boldsymbol{q}}(t) = \frac{\partial \boldsymbol{q}}{\partial s}\dot{s} \tag{10.5}$$

$$\ddot{\boldsymbol{q}}(t) = \frac{\partial \boldsymbol{q}}{\partial s}\ddot{s} + \frac{\partial^2 \boldsymbol{q}}{\partial s^2}\dot{s}^2 \tag{10.6}$$

10.2.1 Profils temporels

Cette section présente quelques types de profils temporels qui sont typiquement utilisée pour générer des trajectoires.

Profil temporel trapézoïdal

Une profil de type trapézoïdal, voir Figure 10.2, est utilisé fréquemment pour la génération de trajectoires simples pour plusieurs types de système asservis en position. Ce profil permet d'inclure directement une limite en vitesse v_{max} et une limite en accélération a_{max}, c'est le profile qui donne la trajectoire la plus rapide possible considérant ces deux contraintes. Il existe toutefois des profils temporels plus lisses sans discontinuités, voir les prochaines sections, ce qui peut être un avantage. Le profile trapézoïdal consiste en trois phases : une phase d'accélération avec $\ddot{s} = a_{max}$, une phase de croisière à vitesse constante avec $\dot{s} = v_{max}$ et une phase de décélération avec $\ddot{s} = -a_{max}$. Les profils de positions et d'accélérations associés sont illustrés à la Figure 10.3.

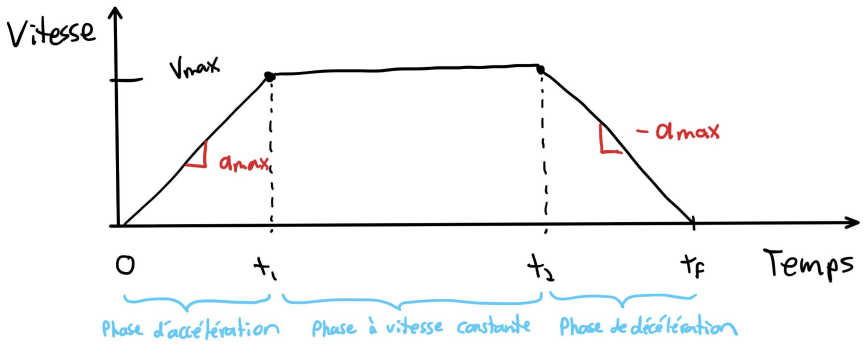


FIGURE 10.2 – Profil temporel de vitesse de type trapézoïdal

Si on utilise le concept de la variable s qui représente la fraction du chemin à parcourir (voir section 10.2), l'intégrale du profile de vitesse $\dot{s}(t)$ doit être égal à 1 (100% du chemin parcouru). Il reste donc seulement trois variable, donc seulement deux sont indépendantes, pour définir complètement le profile temporel de $s(t)$: la vitesse maximale v_{max}, l'accélération maximale a_{max} et la durée totale t_f du trajet. Il peut être intéressant de déterminer les contraintes v_{max} et a_{max} du profile basé sur les limites physiques d'un système et ainsi obtenir la trajectoire qui minimise t_f :

$$t_f = \frac{a_{max} + v_{max}^2}{a_{max} v_{max}} \tag{10.7}$$

Il est à noter que si $v_{max}^2/a_{max} > 1$ le système n'a pas le temps d'atteindre sa vitesse maximale de croisière et le profil est réduit à deux phases : accélération et décélération, voir Figure 10.4. Lorsque $v_{max}^2/a_{max} \leq 1$,

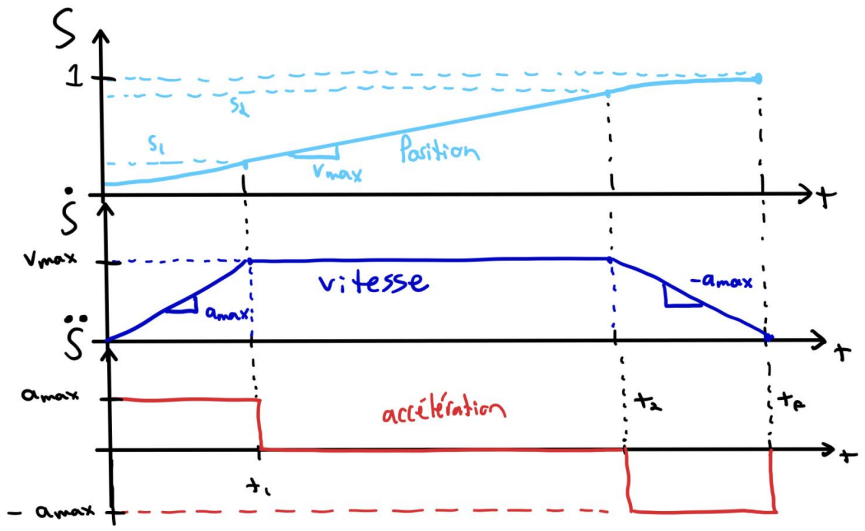


FIGURE 10.3 – Position, vitesse et accélération pour un profil temporel de type trapézoïdal

le profile est bien trapézoïdale et les équations de ce profile pour les trois segments temporels sont :

$$\text{si} \quad 0 \leq t \leq \frac{v_{max}}{a_{max}} \qquad \begin{cases} s(t) = \frac{1}{2} a_{max} t^2 \\ \dot{s}(t) = a_{max} t \\ \ddot{s}(t) = a_{max} \end{cases} \tag{10.8}$$

$$\text{si} \quad \frac{v_{max}}{a_{max}} < t \leq T - \frac{v_{max}}{a_{max}} \qquad \begin{cases} s(t) = v_{max} t - \frac{v_{max}^2}{2a} \\ \dot{s}(t) = v_{max} \\ \ddot{s}(t) = 0 \end{cases} \tag{10.9}$$

$$\text{si} \quad T - \frac{v_{max}}{a_{max}} < t \leq T \qquad \begin{cases} s(t) = \frac{2a_{max} v_{max} T - 2v_{max}^2 - a_{max}^2 (t-T)^2}{2a_{max}} \\ \dot{s}(t) = a(T-t) \\ \ddot{s}(t) = -a \end{cases} \tag{10.10}$$

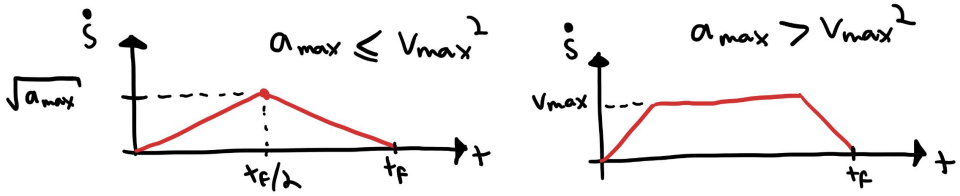


FIGURE 10.4 – Cas limite du profile trapézoïdale

Profil temporel polynomial d'ordre 3

Une alternative à une profil temporel de type trapézoïdale est d'utiliser une fonction polynomiale. Pour obtenir un profile de vitesse qui débute à l'arrêt et termine aussi à l'arrêt, l'ordre minimum de la fonction est de trois. On obtient alors un profile de position cubique, un profile de vitesse parabolique et un profile

d'accélération linéaire, avec comme équations :

$$s(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{10.11}$$
$$\dot{s}(t) = a_1 + 2a_2 t + 3a_3 t^2 \tag{10.12}$$
$$\ddot{s}(t) = 2a_2 + 6a_3 t \tag{10.13}$$

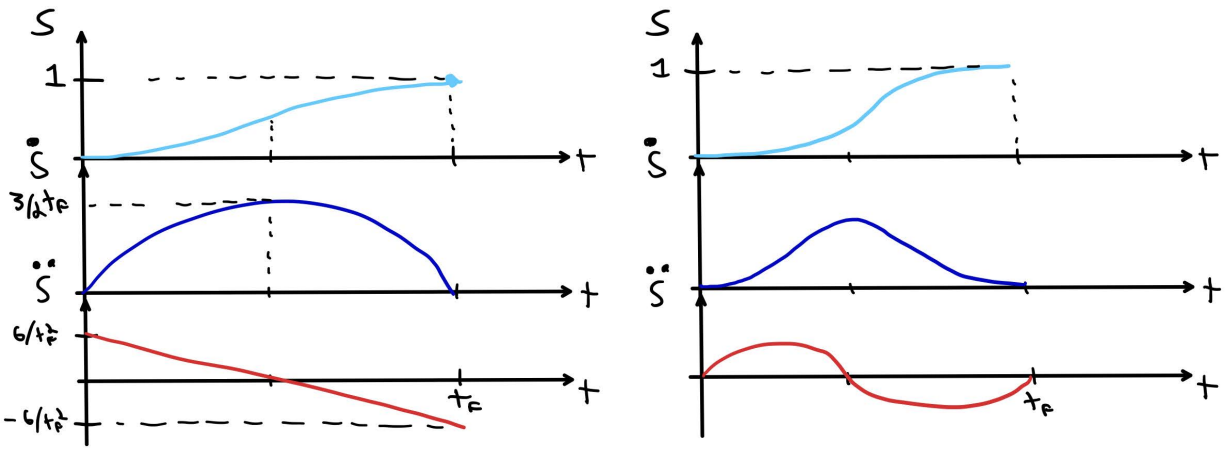
Considérant que nous voulons représenter un mouvement qui débute et termine à l'arrêt, dans un temps t_f, nous avons les 4 contraintes suivantes à imposer : la position initial $s(0) = 0$, la position finale $s(T) = 1$, la vitesse initiale $\dot{s}(t_0) = 0$ et la vitesse terminale $\dot{s}(t_f) = 0$. Il y a donc un seul paramètre libre, la durée du trajet t_f, et on peut trouver que :

$$a_0 = 0 \quad a_1 = 0 \quad a_2 = 3/t_f^2 \quad a_3 = -2/t_f^3 \tag{10.14}$$

ce qui donne comme équations de profile :

$$s(t) = \frac{3}{t_f^2} t^2 - \frac{2}{t_f^3} t^3 \tag{10.15}$$
$$\dot{s}(t) = \frac{6}{t_f^2} t - \frac{6}{t_f^3} t^2 \tag{10.16}$$
$$\ddot{s}(t) = \frac{6}{t_f^2} - \frac{12}{t_f^3} t \tag{10.17}$$

qui sont illustrée à la Figure 10.5a.



(a) Ordre 3 (b) Ordre 5

FIGURE 10.5 – Profiles temporels polynonimaux

Profil temporel polynomial d'ordre 5

Une option pour avoir des trajectoires plus lisses est d'utiliser une fonction polynomiale d'ordre 5. Avec 5 paramètres il est possible de rajouter deux contraintes pour avoir une accélération qui débute et termine à une valeur de zéro, ce qui permet d'avoir aucune discontinuité en terme d'accélération, voir Figure 10.5b pour un allure de ce type de profile. Il est à noter que puisque nous avons rajouter 2 paramètres et 2 contraintes, comme pour le profile d'ordre 3 le profil d'ordre 5 a seulement t_f comme paramètre libre.

Généralisation : Profil polynomial d'ordre n

Il est possible de généraliser l'approche polynomiale pour satisfaire un nombre arbitraire de conditions frontières. Soit un polynôme d'ordre N décrit par le vecteur de paramètres $\mathbf{p} \in \mathbb{R}^{N+1}$:

$$s(t) = p_0 + p_1 t + p_2 t^2 + \cdots + p_N t^N = \sum_{i=0}^{N} p_i t^i \tag{10.18}$$

Les dérivées successives (vitesse, accélération, jerk, snap, etc.) s'obtiennent linéairement par rapport aux paramètres :

$$\frac{d^{(k)} s(t)}{dt^{(k)}} = \sum_{i=k}^{N} \frac{i!}{(i-k)!} p_i t^{i-k} \tag{10.19}$$

Le problème de planification revient à trouver le vecteur $\mathbf{p}$ qui satisfait toutes les contraintes. Cela se formule comme un système linéaire $\mathbf{A}\mathbf{p} = \mathbf{b}$.

Construction de la matrice de contraintes A Chaque ligne de la matrice $\mathbf{A}$ et du vecteur $\mathbf{b}$ correspond à une contrainte imposée sur une dérivée k à un instant t. Dans le cas simple d'une trajectoire point-à-point (départ à $t = 0$, arrivée à $t = t_f$), la matrice est structurée en deux blocs :

$$\begin{bmatrix} \mathbf{A}(0) \\ \mathbf{A}(t_f) \end{bmatrix} \mathbf{p} = \begin{bmatrix} \mathbf{x}_0 \\ \mathbf{x}_f \end{bmatrix} \tag{10.20}$$

Les vecteurs $\mathbf{x}_0$ et $\mathbf{x}_f$ contiennent les valeurs désirées des M premières dérivées aux bornes (position, vitesse, accélération, etc.) :

$$\mathbf{x}_0 = \begin{bmatrix} s(0) \\ \dot{s}(0) \\ \ddot{s}(0) \\ \vdots \end{bmatrix}, \quad \mathbf{x}_f = \begin{bmatrix} s(t_f) \\ \dot{s}(t_f) \\ \ddot{s}(t_f) \\ \vdots \end{bmatrix} \tag{10.21}$$

La sous-matrice $\mathbf{A}(t)$, de dimension $(M+1) \times (N+1)$, relie les paramètres du polynôme aux dérivées à l'instant t. Ses éléments sont donnés par la relation de dérivation $A_{k,i} = \frac{i!}{(i-k)!} t^{i-k}$:

$$\mathbf{A}(t) = \begin{bmatrix} 1 & t & t^2 & t^3 & \cdots & t^N \\ 0 & 1 & 2t & 3t^2 & \cdots & N t^{N-1} \\ 0 & 0 & 2 & 6t & \cdots & N(N-1) t^{N-2} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \cdots & \frac{N!}{(N-M)!} t^{N-M} \end{bmatrix} \tag{10.22}$$

Si le nombre de contraintes N_c est exactement égal au nombre de paramètres ($N_c = N + 1$), la solution est unique et obtenue par inversion directe : $\mathbf{p} = \mathbf{A}^{-1}\mathbf{b}$.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

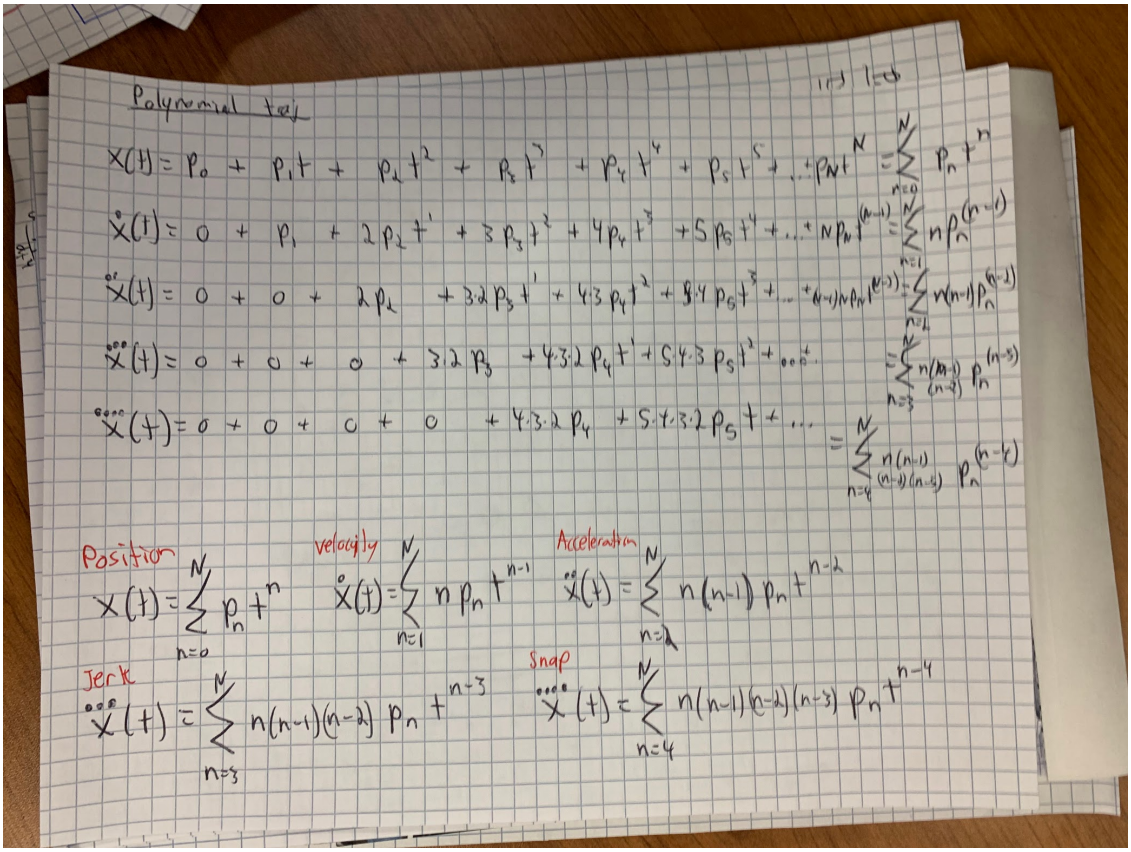


FIGURE 10.6 – Illustration d'une trajectoire polynomiale d'ordre n.

Optimisation (Système sous-contraint) Une approche puissante consiste à utiliser un polynôme d'ordre supérieur au minimum requis (par exemple, un ordre 12 pour seulement 6 conditions limites). Le système $\mathbf{Ap} = \mathbf{b}$ devient alors sous-contraint : il existe une infinité de solutions possibles. Pour choisir la "meilleure" trajectoire parmi cette infinité, on définit une fonction de coût J qui permet de définir des coûts associés à l'énergie des dérivées d'ordre supérieur. Le coût est défini comme une somme pondérée des intégrales du carré des dérivées (vitesse, accélération, jerk, snap, etc.) :

$$J(\mathbf{p}) = \sum_k w_k \int_0^{t_f} \left(\frac{d^{(k)}s(t)}{dt^{(k)}}\right)^2 dt = \mathbf{p}^T\mathbf{Q}\mathbf{p} \tag{10.23}$$

Où $\mathbf{Q}$ est une matrice hessienne symétrique définie positive calculée analytiquement en fonction de t_f.

Le problème devient alors un problème d'optimisation quadratique sous contraintes linéaires :

$$\min_{\mathbf{p}} \quad \frac{1}{2}\mathbf{p}^T\mathbf{Q}\mathbf{p} \tag{10.24}$$

$$\text{sujet à} \quad \mathbf{Ap} = \mathbf{b} \tag{10.25}$$

qui a une solution analytique exacte :

$$\mathbf{p}^* = \mathbf{Q}^{-1}\mathbf{A}^T(\mathbf{A}\mathbf{Q}^{-1}\mathbf{A}^T)^{-1}\mathbf{b} \tag{10.26}$$

Cette méthode permet de générer des trajectoires "Minimum Jerk" ou "Minimum Snap" très douces, idéales pour les drones ou les robots manipulateurs rapides. Voir implémentation ici : `https://github.com/SherbyRobotics/pyro/blob/master/pyro/planning/trajectorygeneration.py`

10.3 Planification cinématique pour les robots manipulateurs

Capsule vidéo
Generation de trajectoire pour les manipulateurs
https://youtu.be/_PvvpwGBu7E

10.3.1 Point à point dans l'espace des joints

Si la position désirée d'un robot est déjà connue dans l'espace de ses joints, l'approche la plus simples est de générer un chemin linéaire dans l'espace des joints (voir Figure 10.7) et ensuite d'ajuster le profil temporel en fonction des limites des actionneurs. L'objectif est donc de calculer des fonctions de trajectoires pour les joints en fonction d'une position actuelle, une position désirée et un temps de trajet :

$$\ddot{\boldsymbol{q}}(t), \dot{\boldsymbol{q}}(t), \boldsymbol{q}(t) = plan(\boldsymbol{q}_0, \boldsymbol{q}_f, t_f) \tag{10.27}$$

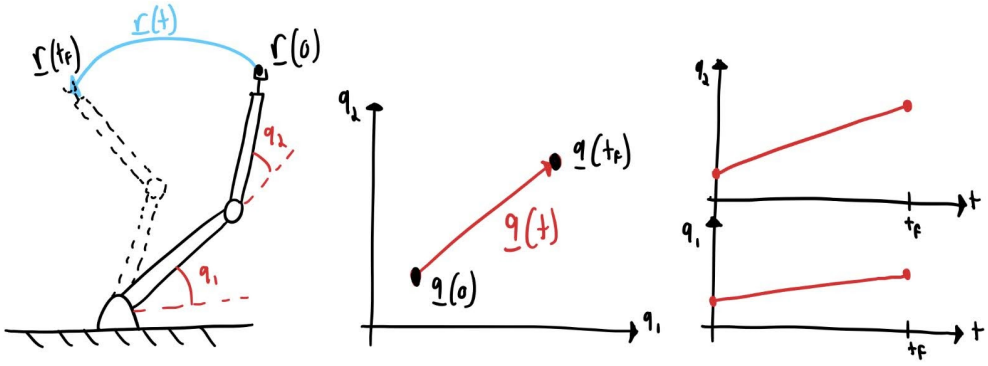


FIGURE 10.7 – Trajectoire en ligne droite dans l'espace de la tâche

Un chemin en ligne droite dans l'espace des joints peut être défini par l'équation :

$$\boldsymbol{q}(s) = \boldsymbol{q}_0 + s(\boldsymbol{q}_f - \boldsymbol{q}_0) \tag{10.28}$$

On obtient alors comme dérivés :

$$\frac{\partial \boldsymbol{q}}{\partial s} = (\boldsymbol{q}_f - \boldsymbol{q}_0) \tag{10.29}$$

$$\frac{\partial^2 \boldsymbol{q}}{\partial s^2} = 0 \tag{10.30}$$

et la relation entre les vitesses/accélérations des joints et le profil temporel est alors :

$$\dot{\boldsymbol{q}} = (\boldsymbol{q}_f - \boldsymbol{q}_0)\dot{s} \tag{10.31}$$

$$\ddot{\boldsymbol{q}} = (\boldsymbol{q}_f - \boldsymbol{q}_0)\ddot{s} \tag{10.32}$$

Donc par exemple si on choisi un profil $s(t)$ trapézoïdale on peut choisir les paramètres v_{max} et a_{max} pour qu'ils correspondent avec les limites physiques des joints :

$$v_{max}\,(q_i(t_f) - q_i(0)) \leq \max(\dot{q}_i) \quad \forall i \tag{10.33}$$

$$a_{max}\,(q_i(t_f) - q_i(0)) \leq \max(\ddot{q}_i) \quad \forall i \tag{10.34}$$

ce qui correspond en fait simplement a ajuster pour l'amplitude du déplacement.

Planifier directement dans l'espace des joints a comme avantage de pouvoir vérifier intégrer directement les contraintes liés aux actionneurs (angles maximums, vitesses et accélérations). L'inconvénient c'est que la trajectoire de l'effecteur du robot n'est directement contrôlée.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

10.3.2 Point à point dans l'espace de la tâche

Si la tâche se décrit mieux en termes de coordonnées cartésienne de l'effecteur par exemple, il est possible d'appliquer le même principe pour générer directement une trajectoire linéaire pour l'effecteur basé sur une position actuel, un position cible et un temps de trajet :

$$\ddot{\boldsymbol{r}}(t), \dot{\boldsymbol{r}}(t), \boldsymbol{r}(t) = plan(\boldsymbol{r}_0, \boldsymbol{r}_f, t_f) \tag{10.35}$$

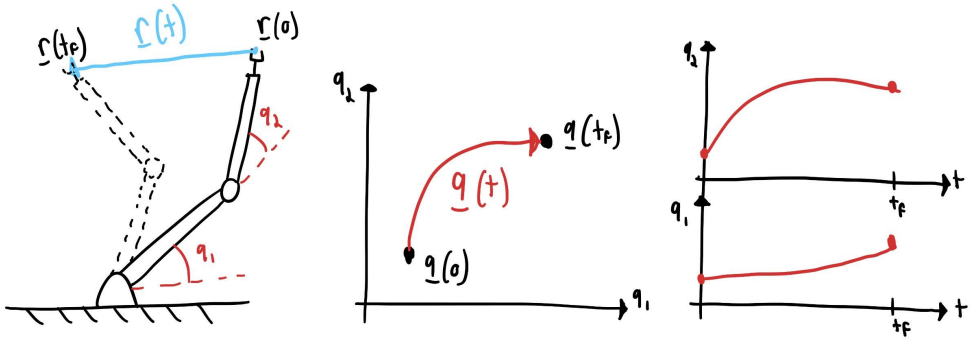


FIGURE 10.8 – Trajectoire en ligne droite dans l'espace de la tâche

Un chemin en ligne droite dans l'espace de la tâche peut être défini par l'équation :

$$\boldsymbol{r}(s) = \boldsymbol{r}_0 + s(\boldsymbol{r}_f - \boldsymbol{r}_0) \tag{10.36}$$

On obtient alors la relation suivante entre les vitesses/accélérations de l'effecteur et le profil temporel :

$$\dot{\boldsymbol{r}} = (\boldsymbol{r}_f - \boldsymbol{r}_0)\dot{s} \tag{10.37}$$

$$\ddot{\boldsymbol{r}} = (\boldsymbol{r}_f - \boldsymbol{r}_0)\ddot{s} \tag{10.38}$$

Certaines approches de commande peuvent utiliser directement une trajectoire cible pour l'effecteur, par exemple la loi de commande cinématique présentée à la section 8.2.1, ou la loi de commande en impédance présentée à la section 8.4.2. Alternativement, il est toujours possible de convertir la trajectoire de l'effecteur du robot dans l'espace des joints, avec un modèle cinématique (voir équation (4.87)), pour ensuite l'utilisée avec des lois de commande qui travaillent directement dans l'espace de joints.

Un inconvénient de planifier dans l'espace de la tâche est qu'on peut se retrouver dans des situations ou certains points intermédiaire du chemin sont hors de l'espace de travail du robot ou bien proche de singularité, même si le point de départ et la destination sont loin de configurations problématiques. Globalement, par rapport à l'approche d'envoyer directement la position cible dans une loi de commande, les méthodes point à point apportent seulement un meilleur contrôle du profile temporel.

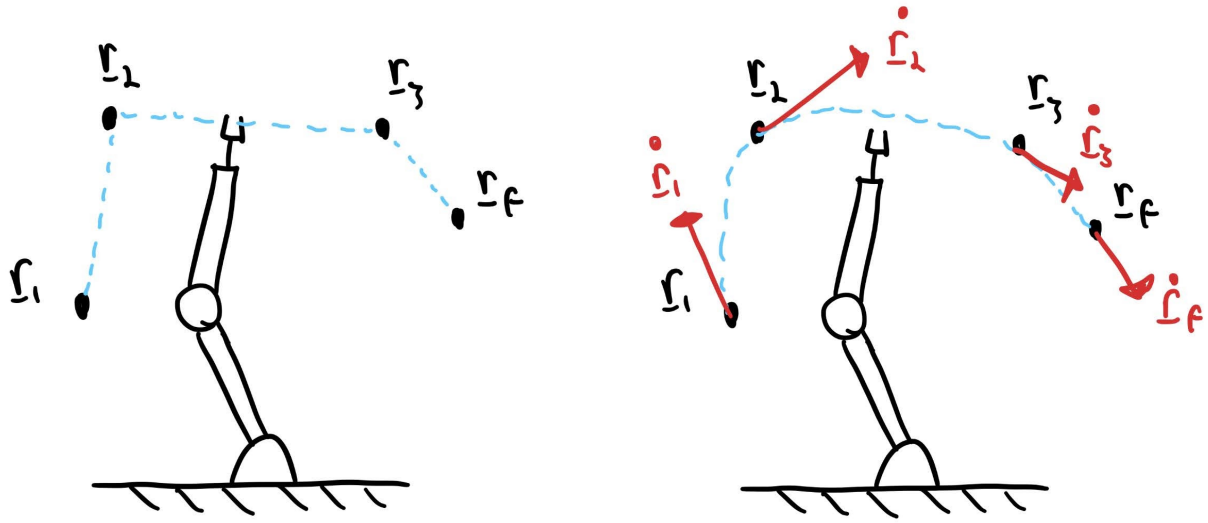
10.3.3 Multi-points dans l'espace de la tâche

Certaines tâches peuvent être décrites en termes de séquence de position à suivre, par exemple le code G pour des machines outils. Aussi, pour des problèmes de planification de chemin dans des environnements avec beaucoup d'obstacles, certains algorithmes de recherche vont retourner une séquence discrétisée de position à suivre pour le robot. Il y a donc plusieurs situations pour lesquels on doit générer une trajectoire basé sur un séquence de positions intermédiaires :

$$\ddot{\boldsymbol{r}}(t), \dot{\boldsymbol{r}}(t), \boldsymbol{r}(t) = plan(\boldsymbol{r}_0, \boldsymbol{r}_1, \boldsymbol{r}_2, ..., \boldsymbol{r}_f, t_1, t_2, ..., t_f) \tag{10.39}$$

Segments

L'approche la plus simpliste est d'utiliser la méthode point à point présentée à la section précédente pour chaque segment, voir Figure 10.9a. Toutefois, l'inconvénient majeur est que pour garantir qu'il n'y a aucune discontinuité dans les vitesses (ce qui mènerait à une accélération infinie), le robot doit s'immobiliser à tout les points où il y a des changements de directions. Ce n'est pas optimal ni en terme de consommation énergétique ni en terme de temps.



(a) Segments

(b) Interpolation

FIGURE 10.9 – Trajectoire multi-points dans l'espace de la tâche

Interpolation

Une alternative est d'utiliser des fonctions d'interpolation pour générer une trajectoire plus lisse. Une approche est d'utiliser des polynômes cubique pour chaque segment, avec des contraintes de continuité en position et vitesse à chaque point intermédiaire. Il est aussi possible de spécifier des vecteurs de vitesses désirées à chaque point intermédiaire, qui peuvent être inclus comme des contraintes pour déterminer les paramètres des polynômes. Il existe plusieurs méthodes d'interpolation qui peuvent être utilisées pour ce type de problème, une autre méthode populaire est les *B-spline*.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Deuxième partie

Les véhicules robotisés

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Chapitre 11

Modélisation des véhicules

Ce chapitre propose un survol des approches de modélisation couramment utilisés dans un contexte de commande et de planification.

11.1 Introduction : L'art de la simplification

Modéliser un véhicule, c'est l'art de mentir intelligemment. Un véhicule réel est un système infiniment complexe : déformation des pneus, turbulence de l'air, jeu dans les engrenages, vibrations du châssis. Pourtant, pour piloter un robot ou planifier un trajet, nous n'avons pas besoin de toute cette complexité. Ce chapitre explore comment nous passons de la réalité physique à des équations mathématiques exploitables par un ordinateur. L'objectif n'est pas de trouver le modèle parfait, mais le modèle suffisant pour une tâche à accomplir. Ici dans ce chapitre, l'accent est mis sur les modèles exploitables pour la synthèse de lois de commande ou la planification de trajectoire. Les approches de modélisation haute fidélité, telles que les éléments finis pour la déformation des pneus ou la mécanique des fluides numérique pour l'aérodynamisme complexe, généralement réservées à la validation en simulation, ne seront pas traitées ici.

11.1.1 Le spectre de la fidélité

Imaginez trois situations de conduite différentes. Dans chacune, votre cerveau utilise un modèle différent :

1. **Navigation (GPS) :** Vous planifiez un trajet de Montréal à Québec. À cette échelle, votre voiture est un simple **point** qui suit une ligne. La largeur de la voiture, son angle de braquage ou la pression des pneus n'ont aucune importance.
2. **Stationnement :** Vous vous garez dans une rue étroite. Ici, le modèle "point" ne suffit plus. Vous devez tenir compte de la **géométrie** : la voiture ne peut pas se déplacer latéralement (contrainte non-holonome), elle doit braquer et avancer.
3. **Conduite sur glace :** Vous prenez un virage serré sur une chaussée glissante. La géométrie ne suffit plus. Il faut comprendre la **physique** : l'inertie, la friction, le transfert de masse. Si vous tournez le volant, la voiture ne va pas nécessairement suivre la direction des roues.

En robotique, la trousse à outil comprend une hiérarchie de modèles (voir Tableau 11.1). Pour un algorithme de conduite autonome, il est parfois suffisant d'utiliser un modèle particule (x, y) pour la planification de chemin à haut niveau, alors qu'un modèle dynamique précis sera nécessaire pour calculer des forces de commande en temps réel. Cet **écosystème de modèles est donc une boîte à outils**, où chaque modèle est un instrument adapté à une tâche spécifique.

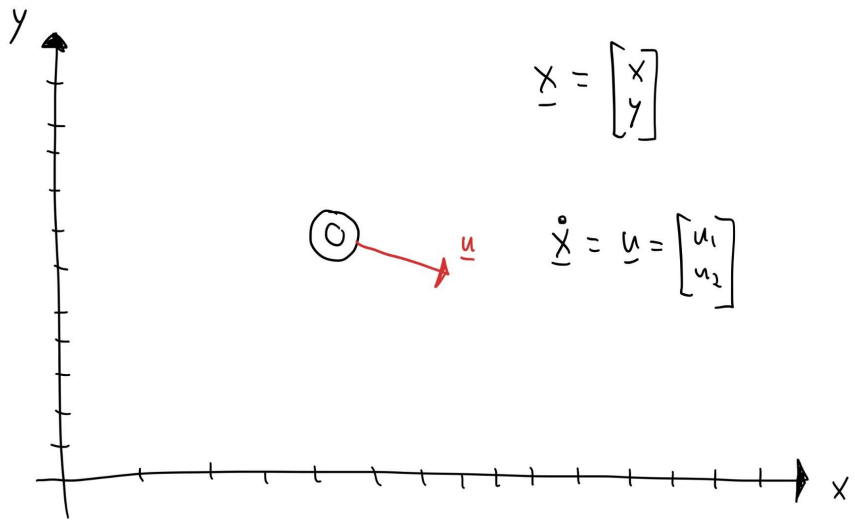


FIGURE 11.1 – Modèle particule cinématique 2D pour un véhicule holonomique. La configuration est donnée par la position soit $\mathbf{q} = [x, y]^T \in \mathbb{R}^2$, l'état est identique à la configuration, et les actions sont les vitesses linéaires $\mathbf{u} = [v_x, v_y]^T \in \mathbb{R}^2$.

11.2 Les trois ingrédients de base d'un modèle de véhicule

Quel que soit le véhicule (drone, voiture, sous-marin), nous devons répondre à trois questions fondamentales pour le contrôler.

11.2.1 La configuration q : "Où suis-je ?"

Définition 11.1 La configuration:

La configuration, notée $\mathbf{q}$, est l'ensemble minimal de variables nécessaires pour décrire la posture géométrique du véhicule à un instant figé. L'ensemble de toutes les configurations possibles forme l'espace de configuration, noté $\mathcal{C}$, dont la dimension correspond au nombre de degrés de liberté (DDL) du système.

La définition mathématique de $\mathbf{q}$ dépend du niveau d'abstraction choisi :

— **Pour une particule (Point 2D/3D) :** La configuration se résume à la position cartésienne, soit $\mathbf{q} = [x, y]^T \in \mathbb{R}^2$ ou $\mathbf{q} = [x, y, z]^T \in \mathbb{R}^3$.

— **Pour un corps rigide 2D (Robot mobile) :** Il est nécessaire d'ajouter l'orientation (le cap) à la position, ce qui donne $\mathbf{q} = [x, y, \psi]^T$. Mathématiquement, cet espace est le groupe spécial euclidien $SE(2)$.

— **Pour un corps rigide 3D (Drone/Avion) :** La configuration inclut la position et l'attitude complète (souvent représentée par des angles d'Euler ou des quaternions), appartenant alors au groupe $SE(3)$.

11.2.2 Les actions u : "Que puis-je faire ?"

Définition 11.2 Les actions:

Le vecteur d'action, noté $\mathbf{u}$, décrit les commandes que nous pouvons utiliser pour modifier le mouvement du véhicule. L'ensemble des entrées de commande admissibles est noté $\mathcal{U}$. Le vecteur $\mathbf{u} \in \mathcal{U}$ représente les variables d'entrée des équations du modèle utilisé.

La nature de ces commandes change radicalement selon que l'on considère le véhicule comme un objet géométrique ou un système physique :

- **Action Cinématique (Abstraite) :** On commande directement des vitesses (ex : "Avance à 1 m/s", "Tourne à 0.5 rad/s"). C'est l'hypothèse simplificatrice utilisée par la plupart des planificateurs de trajectoire. L'hypothèse sous-jacente est qu'il existe des contrôleurs de bas niveau suffisamment performants pour exécuter ces actions quasi-instantanément.
- **Action Dynamique (Réelle) :** On commande des efforts physiques (ex : "Couple moteur de 100 Nm", "Angle de braquage des roues", "Poussée des hélices"). C'est ce que le système de contrôle doit gérer en réalité à bas niveau.

11.2.3 L'équation d'évolution : "Comment je bouge ?"

C'est le cœur de la modélisation. C'est la fonction qui prédit l'évolution de la configuration en fonction des actions. On distingue deux grandes familles :

Modèles Cinématiques (Géométrie du mouvement) Ces modèles décrivent le mouvement possible sans s'occuper des forces. Ils répondent à la question : *"Si j'avance en tournant avec telle vitesse, comment vont évaluer mes coordonnées ?"* Ils sont typiquement utiles à basse vitesse. Lorsque les actions sont des vitesses, ces équations prennent la forme suivante, i.e. des équations différentielles de premier ordre :

Définition 11.3 Forme générique d'un modèle cinématique:

$$\dot{\mathbf{q}} = \mathbf{N}(\mathbf{q})\mathbf{u} \tag{11.1}$$

Modèles Dynamiques (Physique du mouvement) Ces modèles utilisent la seconde loi de Newton ($F = ma$). Ils répondent à la question : *"Si j'applique telles forces, comment vont évoluer mes coordonnées ?"*. Lorsque les actions sont des forces, ces équations prennent la forme d'équations différentielles d'ordre deux :

Définition 11.4 Forme générique d'un modèle dynamique (configuration):

$$\ddot{\mathbf{q}} = f_q(\dot{\mathbf{q}}, \mathbf{q}, \mathbf{u}) \tag{11.2}$$

Parfois, on s'intéresse seulement à l'évolution des vitesses d'un véhicule. On peut alors travailler avec des équations qui n'inclut que l'évolution des vitesses, sans prédire l'évolution de la configuration. Dans cette situation, on peut utiliser un vecteur de vitesses généralisées ν et des équations différentielles d'ordre un :

Définition 11.5 Forme générique d'un modèle dynamique (vitesses):

$$\dot{\nu} = f_v(\dot{\nu}, \mathbf{u}) \tag{11.3}$$

Lorsque aucune propriétés ou forces ne dépend d'une variable configuration, on l'appelle parfois une coordonnée ignorable. On peut alors ignorer l'équation qui prédit d'élolution de cette variable et simplifier le système d'équations. Un exemple est la position x d'un avion, il est possible de calculer l'évolution des vitesses de l'avion sans garder en mémoire l'évolution de sa position. Un contre-exemple est le tanguage d'un avion, l'angle d'attaque et donc la force de portance en dépende.

Formalisme d'état Pour un ordinateur, toutes ces notions sont unifiées sous la forme d'un système d'état.

Définition 11.6 Vecteur d'état x:

Le vecteur d'état, noté $\mathbf{x} \in \mathcal{X}$, comprend toutes les variable "mémoire" du système nécessaire pour prédire le futur. L'ensemble $\mathcal{X}$ est l'espace d'état, et ses dimensions sont appelées l'ordre d'un système. Si $\mathcal{X} = \mathbb{R}^n$, on dira que le modèle est d'ordre n, ce qui correspond au nombres de variable indépendantes pour décrire l'état du système.

— **Pour un modèle cinématique**, l'état est simplement la configuration : $\mathbf{x} = \mathbf{q}$.
— **Pour un modèle dynamique**, l'état doit généralement inclure la configuration et les vitesses (linéaire et angulaire) : $\mathbf{x} = [\mathbf{q}, \dot{\mathbf{q}}]^T$. Sans connaître la vitesse actuelle, on ne peut pas prédire la position future.

L'équation générale de tout véhicule s'écrit alors sous la forme canonique :

Définition 11.7 Équations du mouvement sous la forme de modèle d'état:

$$\dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{u}) \tag{11.4}$$

C'est la forme standard pour résoudre des équations différentielles, typiquement utilisé pour les simulations sur un ordinateur.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

11.2.4 Exemples de niveaux d'abstractions génériques pour véhicules

Le tableau suivant donne quelques exemple de niveaux d'abstractions de véhicules généralement utilisé dans un contexte de commande, planification et analyse. Les modèles de particule sont typiquement utilisés pour planifier des chemins en fonction d'une carte d'obstacles. On peut utiliser une particule dynamique si on veut planifier le détail d'un profil de vitesse et des rayons de courbures réalisables, par exemple si on essaye d'optimiser une ligne de course sur un circuit. Ensuite, les modèles corps rigide cinématiques sont typiquement utilisés pour synthétisé des lois de commande pour des régimes à basse vitesse. C'est aussi le niveau d'abstraction qui fait apparaître le cap d'un véhicule et qui serait adapté pour planifier un stationnement parallèle par exemple. Finalement, les modèles dynamique de corps rigide sont utile pour les analyses de stabilité et synthétiser des lois de commande pour les drones, avions, etc.

TABLE 11.1 – **Abstractions de véhicules génériques**

Caractér.	**Particule cinématique** (Ordre 1)	**Particule dynamique** (Ordre 2)	**Corps rigide cinématique 2D** (Ordre 1)	**Corps rigide dynamique 2D**	**Corps rigide dynamique 3D**
n	2	4	3	6	12
Modèle dynamique	$\dot{x} = v_x$ $\dot{y} = v_y$	$\ddot{x} = a_x$ $\ddot{y} = a_y$	$\dot{\mathbf{p}} = \mathbf{R}(\psi)\mathbf{v}_b$ $\dot{\psi} = r$	$m(\dot{u} - vr) = F_x$ $m(\dot{v} + ur) = F_y$ $I_z\dot{r} = M_z$	$m(\dot{\mathbf{v}} + \omega \times \mathbf{v}) = \mathbf{F}$ $\mathbf{I}\dot{\omega} + \omega \times \mathbf{I}\omega = \mathbf{M}$
Espace de config. q	$\mathbf{q} = [x, y]^T$	$\mathbf{q} = [x, y]^T$	$\mathbf{q} = [x, y, \psi]^T$	$\mathbf{q} = [x, y, \psi]^T$	$\mathbf{q} \in \mathbb{R}^6$ (Pos, Euler)
Espace d'état x	$\mathbf{x} = [x, y]^T$	$\mathbf{x} = [x, y, v_x, v_y]^T$	$\mathbf{x} = [x, y, \psi]^T$	$\mathbf{x} = [x, y, \psi, u, v, r]^T$	$\mathbf{x} \in \mathbb{R}^{12}$ (Pos, Euler, VitLin, VitAng)
Espace d'action u	$\mathbf{u} = [v_x, v_y]^T$	$\mathbf{u} = [a_x, a_y]^T$	$\mathbf{u} = [u, v, r]^T$ (Vit. Corps)	$\mathbf{u} = [F_x, F_y, M_z]^T$ (Torseur 2D)	$\mathbf{u} = [\mathbf{F}, \mathbf{M}]^T$ (Torseur 3D)
Contraintes d'état	Positions traversables $\mathbf{q} \in \mathcal{C}_{free}$	Positions traversables $\mathbf{q} \in \mathcal{C}_{free}$ Vitesses réalisables $\mathbf{u} \in \mathcal{V}_{feasible}$	$\mathbf{q} \in \mathcal{C}_{free} \subseteq \mathbb{R}^3$	$\mathbf{q} \in \mathcal{C}_{free} \subseteq \mathbb{R}^3$ $\mathbf{u} \in \mathcal{V}_{feasible} \subseteq \mathbb{R}^3$	$\mathbf{q} \in \mathcal{C}_{free} \subseteq \mathbb{R}^6$ $\mathbf{u} \in \mathcal{V}_{feasible} \subseteq \mathbb{R}^6$
Contraintes d'action	Vitesses réalisables $\mathbf{u} \in \mathcal{V}_{feasible}$	Accélérations réalisables $\mathbf{u} \in \mathcal{A}_{feasible}(x)$	$\mathbf{u} \in \mathcal{V}_{feasible} \subseteq \mathbb{R}^3$	$\mathbf{u} \in \mathcal{F}_{feasible}(x)$	$\mathbf{u} \in \mathcal{F}_{feasible}(x)$

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

11.3 Équations du mouvement dans le repère du corps

Il est fréquent en robotique d'observer deux formulations dynamiques qui semblent distinctes : celle utilisée pour les bras manipulateurs (Partie I de ce cours) et celle utilisée pour les véhicules (Partie II). Il s'agit en fait de la même équation fondamentale, exprimée dans des systèmes de coordonnées différents.

11.3.1 Coordonnées généralisées vs repère du corps

Équation dans l'espace de configuration Il est toujours possible d'écrire les équations de la dynamique d'un système mécanique en utilisant uniquement l'espace de configuration, tel que présenté pour les robots manipulateurs 6. Pour un manipulateur, on exprime généralement la dynamique dans l'espace des coordonnées généralisées $\mathbf{q}$ (angles des joints). L'équation prend la forme standard :

Définition 11.8 Dynamique d'un système mécanique avec coordonnées généralisées:

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}_q \tag{11.5}$$

Ici, la matrice d'inertie $\mathbf{H}(\mathbf{q})$ varie continuellement avec la configuration du robot (ex : lorsque le bras se déplie, l'inertie augmente). Il est théoriquement possible d'écrire la dynamique d'un véhicule sous cette forme, en utilisant les coordonnées généralisées appropriées (ex : position et orientation dans l'espace inertiel). Cependant, il est généralement plus pratique de formuler la dynamique des véhicules en utilisant des vitesses définies dans un repère attaché au corps du véhicule.

Repère du corps Pour un véhicule (drone, sous-marin), on préfère exprimer les vitesses $\boldsymbol{\nu}$ et les forces dans le repère attaché au corps, car les forces aérodynamiques/hydrodynamiques et les mesures des capteurs s'y expriment naturellement. L'équation devient :

Définition 11.9 Dynamique d'un système mécanique avec vitesses généralisées:

$$\mathbf{M}_b\dot{\boldsymbol{\nu}} + \mathbf{C}_b(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}_b(\mathbf{q}) = \boldsymbol{\tau}_b \tag{11.6}$$

$$\mathbf{J}(\mathbf{q})\dot{\mathbf{q}} = \boldsymbol{\nu} \tag{11.7}$$

Pour un corps rigide, $\mathbf{J}(\mathbf{q})$ est la matrice jacobienne (ou de transformation cinématique) faisant le lien entre le repère inertiel et le repère du corps. En utilisant le repère attaché au corps, la matrice d'inertie $\mathbf{M}_b$ du véhicule est typiquement constante. Toutefois, on note l'apparition de termes de Coriolis et centrifuges dans $\mathbf{C}_b(\boldsymbol{\nu})$ qui n'étaient pas présents sous cette forme dans les coordonnées généralisées. Ces termes apparaissent car le repère du corps est un référentiel non-inertiel (en rotation).

Propriété 11.1: Équivalence des formulations

Le lien entre ces deux formulations s'établit par les relations suivantes :

$$\mathbf{H}(\mathbf{q}) = \mathbf{J}^T(\mathbf{q})\mathbf{M}_b\mathbf{J}(\mathbf{q}) \tag{11.8}$$

$$\boldsymbol{\nu} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \tag{11.9}$$

$$\boldsymbol{\tau}_q = \mathbf{J}^T(\mathbf{q})\boldsymbol{\tau}_b \tag{11.10}$$

11.3.2 Forme vectorielle et matricielle

Les équations de la dynamique d'un corps rigide sont souvent présentées initialement sous forme vectorielle, séparant la translation et la rotation :

Définition 11.10 Équations de Netwon-Euler (vectorielles):

$$m(\dot{\vec{v}}_b + \vec{\omega}_b \times \vec{v}_b) = \vec{F}_b \tag{11.11}$$

$$\vec{\vec{I}}_b \cdot \dot{\vec{\omega}}_b + \vec{\omega}_b \times (\vec{\vec{I}}_b \cdot \vec{\omega}_b) = \vec{M}_b \tag{11.12}$$

Dans ces équations :
- m est la masse totale du véhicule.
- $\vec{\vec{I}}_b$ est le tenseur d'inertie du véhicule.
- $\vec{v}_b$ et $\vec{\omega}_b$ sont les vitesses linéaire et angulaire du corps rigide.
- $\vec{F}_b$ et $\vec{M}_b$ sont la force et le moment appliqués sur le corps.

Pour obtenir la forme matricielle unifiée, on définit les vecteurs de vitesses et d'effort généralisés avec les composantes des vecteurs, exprimées dans le repère tournant du corps : :

$$\boldsymbol{\nu} = \begin{bmatrix} \mathbf{v}_b \\ \boldsymbol{\omega}_b \end{bmatrix}, \quad \boldsymbol{\tau}_b = \begin{bmatrix} \mathbf{F}_b \\ \mathbf{M}_b \end{bmatrix} \tag{11.13}$$

En utilisant l'opérateur matriciel antisymétrique $S(\cdot)$ pour le produit vectoriel (tel que $\vec{a} \times \vec{b} = S(\mathbf{a})\mathbf{b}$), on peut réécrire les équations vectorielles de Newton-Euler en format matriciel :

Définition 11.11 Équations de Netwon-Euler (matricielles):

$$m\dot{\mathbf{v}}_b + mS(\boldsymbol{\omega}_b)\mathbf{v}_b = \mathbf{F}_b \tag{11.14}$$

$$\mathbf{I}_b\dot{\boldsymbol{\omega}}_b + S(\boldsymbol{\omega}_b)\mathbf{I}_b\boldsymbol{\omega}_b = \mathbf{M}_b \tag{11.15}$$

Ce système se regroupe finalement en une seule équation matricielle par blocs, correspondant à la forme (11.6) :

$$\underbrace{\begin{bmatrix} m\mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{I}_b \end{bmatrix}}_{\mathbf{M}_b} \begin{bmatrix} \dot{\mathbf{v}}_b \\ \dot{\boldsymbol{\omega}}_b \end{bmatrix} + \underbrace{\begin{bmatrix} mS(\boldsymbol{\omega}_b) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & -S(\mathbf{I}_b\boldsymbol{\omega}_b) \end{bmatrix}}_{\mathbf{C}_b(\boldsymbol{\nu})} \begin{bmatrix} \mathbf{v}_b \\ \boldsymbol{\omega}_b \end{bmatrix} = \begin{bmatrix} \mathbf{F}_b \\ \mathbf{M}_b \end{bmatrix} \tag{11.16}$$

11.3.3 Corps planaire (3 DDL)

Ce modèle s'applique aux véhicules dont le mouvement est restreint au plan. On définit les vitesses dans le repère du corps :
- u : Vitesse longitudinale (axe x).
- v : Vitesse latérale (axe y).
- r : Vitesse angulaire de lacet (rotation autour de z).

L'équation matricielle du mouvement $\mathbf{M}_b\dot{\boldsymbol{\nu}} + \mathbf{C}_b(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_b$ devient :

$$\begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I_z \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} 0 & -mr & 0 \\ mr & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u \\ v \\ r \end{bmatrix} = \begin{bmatrix} \sum F_x^b \\ \sum F_y^b \\ \sum M_z^b \end{bmatrix} \tag{11.17}$$

Les termes de droite représentent la somme de toutes les forces et moments externes (propulsion, aérodynamisme, contact pneu-sol, etc.) agissant sur le corps. En effectuant le produit matriciel, on obtient les

équations scalaires.

$$m(\dot{u} - rv) = \sum F_x \tag{11.18}$$

$$m(\dot{v} + ru) = \sum F_y \tag{11.19}$$

$$I_z\dot{r} = \sum M_z \tag{11.20}$$

11.3.4 Corps rigide (6 DDL)

Ce modèle général s'applique aux véhicules évoluant dans l'espace 3D (ex : drones, avions, sous-marins). On définit les vitesses dans le repère du corps :

— u, v, w : Vitesses linéaires (Surge, Sway, Heave).

— p, q, r : Vitesses angulaires (Roll, Pitch, Yaw rates).

Nous supposons ici que le repère du corps est aligné avec les axes principaux d'inertie, rendant la matrice d'inertie diagonale $\mathbf{I}_b = \text{diag}(I_x, I_y, I_z)$.

Équations matricielles

L'équation matricielle $\mathbf{M}_b\dot{\boldsymbol{\nu}} + \mathbf{C}_b(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\tau}_b$ s'écrit sous forme développée :

$$\begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_x & 0 & 0 \\ 0 & 0 & 0 & 0 & I_y & 0 \\ 0 & 0 & 0 & 0 & 0 & I_z \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} 0 & -mr & mq & 0 & 0 & 0 \\ mr & 0 & -mp & 0 & 0 & 0 \\ -mq & mp & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & I_z r & -I_y q \\ 0 & 0 & 0 & -I_z r & 0 & I_x p \\ 0 & 0 & 0 & I_y q & -I_x p & 0 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \end{bmatrix} = \begin{bmatrix} \sum F_x \\ \sum F_y \\ \sum F_z \\ \sum M_x \\ \sum M_y \\ \sum M_z \end{bmatrix} \tag{11.21}$$

Équations scalaires

Le système matriciel ci-dessus correspond aux 6 équations scalaires d'Euler-Newton. Les termes de droite représentent la somme des forces et moments externes appliqués au corps :

$$m(\dot{u} - rv + qw) = \sum F_x \tag{11.22}$$

$$m(\dot{v} - pw + ru) = \sum F_y \tag{11.23}$$

$$m(\dot{w} - qu + pv) = \sum F_z \tag{11.24}$$

$$I_x\dot{p} + (I_z - I_y)qr = \sum M_x \tag{11.25}$$

$$I_y\dot{q} + (I_x - I_z)rp = \sum M_y \tag{11.26}$$

$$I_z\dot{r} + (I_y - I_x)pq = \sum M_z \tag{11.27}$$

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

11.3.5 Exemple avec un drone planaire

Le drone est un cas d'école pertinent pour illustrer la différence entre les deux coordonnées de modélisation. Il s'agit d'un système sous-actionné possédant 3 degrés de liberté (x, y, θ) mais seulement deux entrées de commande : un poussée f_1 et ne poussée f_2.

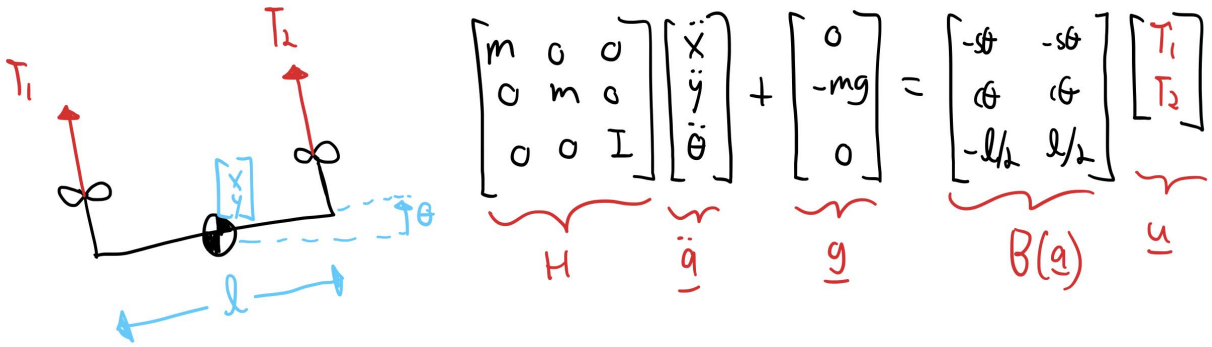


FIGURE 11.2 – Modèle simplifié : Drone planaire (3-DOF)

Approche A : Équations en coordonnées inertielles C'est l'approche la plus intuitive pour visualiser le mouvement. On écrit la seconde loi de Newton directement pour les coordonnées globales (x, y) :

$$\underbrace{\begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I \end{bmatrix}}_{\mathbf{H}(\mathbf{q})} \underbrace{\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\theta} \end{bmatrix}}_{\ddot{\mathbf{q}}} + \underbrace{\begin{bmatrix} 0 \\ mg \\ 0 \end{bmatrix}}_{\mathbf{g}(\mathbf{q})} = \underbrace{\begin{bmatrix} -\sin(\theta) & -\sin(\theta) \\ \cos(\theta) & \cos(\theta) \\ -L & L \end{bmatrix}}_{\mathbf{B}(\mathbf{q})} \underbrace{\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}}_{\mathbf{u}} \tag{11.28}$$

On remarque que la matrice d'inertie $\mathbf{H}$ est constante et diagonale (car exprimée dans le repère monde pour un corps symétrique simple), mais que la matrice d'actionneurs $\mathbf{B}(\mathbf{q})$ dépend fortement de l'orientation θ, ce qui rend le système non-linéaire.

Approche B : Équations dans le repère du corps Si on utilise les vitesses exprimées dans le repère corps, l'équation équivalente sera :

$$\underbrace{\begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I \end{bmatrix}}_{\mathbf{M}_b} \underbrace{\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{r} \end{bmatrix}}_{\dot{\nu}} + \underbrace{\begin{bmatrix} 0 & -mr & 0 \\ mr & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{\mathbf{C}_b(\boldsymbol{\nu})} \underbrace{\begin{bmatrix} u \\ v \\ r \end{bmatrix}}_{\nu} + \underbrace{\begin{bmatrix} mg\sin(\theta) \\ mg\cos(\theta) \\ 0 \end{bmatrix}}_{\mathbf{g}_b(\mathbf{q})} = \underbrace{\begin{bmatrix} 0 & 0 \\ 1 & 1 \\ -L & L \end{bmatrix}}_{\mathbf{B}_b} \underbrace{\begin{bmatrix} f_1 \\ f_2 \end{bmatrix}}_{\mathbf{u}} \tag{11.29}$$

$$\underbrace{\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix}}_{\dot{\mathbf{q}}} = \underbrace{\begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{J^{-1}} \underbrace{\begin{bmatrix} u \\ v \\ r \end{bmatrix}}_{\boldsymbol{\nu}} \tag{11.30}$$

Comparé à l'approche A, on note trois différences fondamentales :

1. La matrice d'allocation $\mathbf{B}_b$ est **constante** (elle ne dépend plus de θ).
2. La matrice de Coriolis $\mathbf{C}_b(\boldsymbol{\nu})$ apparaît explicitement pour modéliser les forces virtuelles ressenties dans le repère tournant.
3. La gravité $\mathbf{g}_b(\mathbf{q})$ est le seul terme qui dépend de l'orientation θ. Elle représente la projection du vecteur poids dans le repère tournant du drone.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

11.4 Nomenclature et Notation

Afin d'unifier la présentation des différents modèles (terrestres, marins, aériens) et de clarifier la distinction entre cinématique et dynamique, nous utiliserons la notation standard résumée dans le tableau 11.2.

TABLE 11.2 – **Nomenclature des variables et symboles mathématiques**

Symbole	Concept	Description
Espaces et Variables de Configuration		
$\mathbb{R}^n$	Espace Euclidien	Espace vectoriel réel de dimension n (ex : position pure).
$SE(n)$	Groupe Spécial Euclidien	Espace des mouvements rigides (Rotation + Translation).
$\mathbf{q}$	Configuration	Posture géométrique instantanée (ex : $\mathbf{q} = [x, y, \theta]^T$).
$\dot{\mathbf{q}}, \ddot{\mathbf{q}}$	Dérivées de config.	Vitesse et accélération généralisées dans l'espace de configuration (ex : $\dot{\mathbf{q}} = [\dot{x}, \dot{y}, \dot{\theta}]^T$).
Cinématique (Repère du Corps)		
$\mathbf{v}_b$	Vitesse Linéaire	Vitesse de translation du CG exprimée dans le repère du corps (ex : $[u, v, w]^T$).
$\boldsymbol{\omega}_b$	Vitesse Angulaire	Vitesse de rotation exprimée dans le repère du corps (ex : $[p, q, r]^T$).
$\boldsymbol{\nu}$	Vitesse Généralisée	Vecteur regroupant vitesses linéaires et angulaires du corps : $\boldsymbol{\nu} = [\mathbf{v}_b^T, \boldsymbol{\omega}_b^T]^T$.
$\dot{\boldsymbol{\nu}}$	Accélération du Corps	Dérivée temporelle des vitesses du corps.
$\mathbf{J}(\mathbf{q})$	Matrice Jacobienne	Matrice de transformation liant les vitesses du corps aux dérivées de configuration : $\mathbf{J}(\mathbf{q})\dot{\mathbf{q}} = \boldsymbol{\nu}$.
Dynamique et Inertie		
m	Masse	Masse totale du véhicule.
$\mathbf{I}_b$	Tenseur d'Inertie	Matrice 3×3 des moments d'inertie exprimée dans le repère du corps.
$\mathbf{M}_b$	Matrice d'Inertie (Corps)	Matrice de masse généralisée (6×6) constante dans le repère du corps (blocs $m\mathbf{I}_{3\times3}$ et $\mathbf{I}_b$).
$\mathbf{H}(\mathbf{q})$	Matrice d'Inertie (Config)	Matrice d'inertie exprimée dans l'espace de configuration (dépend de $\mathbf{q}$). Liée par $\mathbf{H} = \mathbf{N}^{-T}\mathbf{M}_b\mathbf{N}^{-1}$.
$\boldsymbol{\tau}$	Efforts Généralisés	Vecteur des forces et moments appliqués sur le corps. Ils peuvent être définit dans les coordonnées généralisée ou dans le repère du corps.

11.5 Véhicules Terrestres (Automobiles et Robots mobiles)

Cette section couvre les véhicules à roues ! Le tableau 11.3 présente une hiérarchie des modèles couramment utilisés pour les véhicules terrestres, allant des modèles les plus simples (particule) aux modèles les plus complexes (haute fidélité). Chaque niveau de modèle repose sur des hypothèses spécifiques et est adapté à des applications particulières, telles que la planification de trajectoire, le contrôle dynamique ou la simulation avancée.

TABLE 11.3 – **Hiérarchie de modèles pour véhicules à roues**

Type de modèle	**Hypothèses principales**	**Utilité principale**
Particule	Véhicule considéré comme un point (x, y)	Navigation globale (GPS)
Bicyclette Cinématique	Contrainte de roulement sans glissement (basse vitesse). Géométrie d'Ackermann.	Planification de trajectoire locale, suivi de chemin à basse vitesse.
Bicyclette Dynamique	Mouvement planaire (3 DDL). Glissement des pneus modélisé.	Analyse et commande de maneuvres dynamiques
Véhicule Complet 3D	Corps rigide 6 DDL. 4 roues distinctes. Prise en compte du transfert de charge (roulis/tangage) et des suspensions.	Analyse de stabilité, simulateur de conduites.
Haute Fidélité	Pneus et terrain déformables (FEA,DEM)	Simulation pour validations avancées.

11.5.1 Modèle Bicyclette Cinématique

Ces modèles reposent sur l'hypothèse fondamentale que les roues ne glissent pas (contrainte de roulement sans glissement). Ils sont valides à basse vitesse, lorsque les accélérations latérales sont négligeables (pas de dérapage).

Architecture Ackermann et Simplification Bicyclette

Pour qu'un véhicule à quatre roues puisse effectuer un virage sans qu'aucune roue ne dérape, les axes de rotation de toutes les roues doivent concourir en un point unique, appelé le **Centre Instantané de Rotation (CIR)**, voir Figure 11.3. Dans un véhicule classique, cela impose une contrainte géométrique appelée *condition d'Ackermann* : lors d'un virage, la roue intérieure doit braquer davantage que la roue extérieure, car elle parcourt un cercle de rayon plus petit.

Le **modèle bicyclette** est une simplification qui consiste à regrouper les deux roues avant en une seule roue centrale virtuelle et les deux roues arrière en une seule roue arrière centrale. Cette approximation est valide tant que le véhicule est symétrique et que le transfert de charge latéral reste modéré (ce qui est cohérent avec l'hypothèse de basse vitesse).

Équations cinématiques

Nous cherchons à établir la relation entre les entrées de commande (vitesse longitudinale u et angle de braquage δ) et la variation de la posture du robot $\dot{\mathbf{q}} = [\dot{x}, \dot{y}, \dot{\theta}]^T$.

Choix du point de référence : Pour ce modèle cinématique simple, il est courant de définir la position (x, y) du véhicule comme étant le **centre de l'essieu arrière**. Ce choix simplifie considérablement les

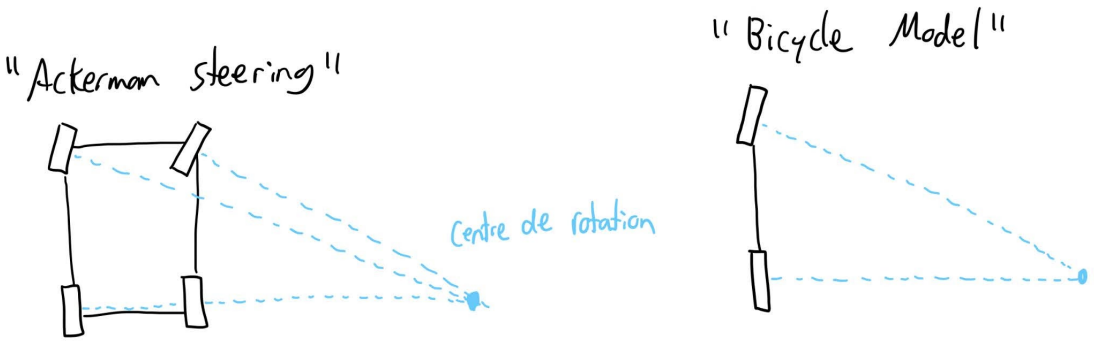


FIGURE 11.3 – Géométrie de direction d'Ackermann (gauche) et simplification bicyclette (droite). Notez que tous les axes convergent vers le CIR.

équations car la vitesse latérale y est nulle (contrainte non-holonome). Si l'on désire un point de référence différents, il faut inclure des termes supplémentaires.

Relation géométrique : En observant le triangle formé par le CIR, l'essieu arrière et l'essieu avant (voir Figure 11.4), on peut relier l'angle de braquage δ au rayon de courbure R et à l'empattement L (distance entre les essieux) :

$$\tan(\delta) = \frac{L}{R} \quad \Rightarrow \quad R = \frac{L}{\tan(\delta)} \tag{11.31}$$

Relation cinématique : Considérons que la vitesse longitudinale au centre de l'essieu arrière est u et que le véhicule possède une vitesse angulaire $\dot{\theta}$. Dans le repère du véhicule, le vecteur vitesse au niveau de l'essieu avant possède :

— Une composante longitudinale égale à u.
— Une composante latérale égale à $L\dot{\theta}$ (due à la rotation du corps rigide autour de l'essieu arrière).

Le vecteur vitesse de la roue avant est donc $(u, L\dot{\theta})$. La contrainte de non-glissement impose que l'angle de braquage δ soit aligné avec ce vecteur vitesse :

$$\tan(\delta) = \frac{L\dot{\theta}}{u} \quad \Rightarrow \quad \dot{\theta} = \frac{u}{L}\tan(\delta) \tag{11.32}$$

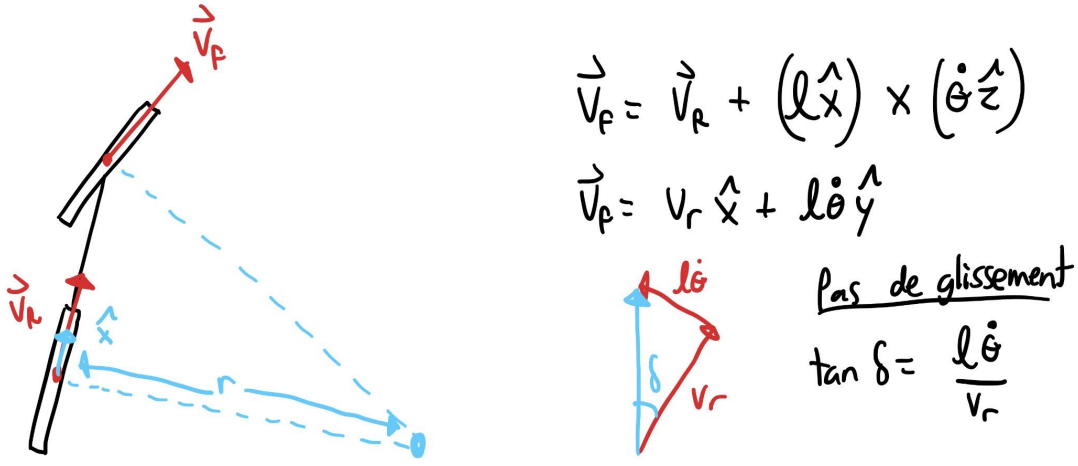


FIGURE 11.4 – Illustration des contraintes non-holonomes : la vitesse instantanée de chaque roue doit être alignée avec l'orientation des roues (pas de glissement latéral).

Modèle d'état : On peut maintenant formuler les équations différentielles liant les entrées de commande à la configuration. En utilisant le vecteur de configuration $\mathbf{q} = [x, y, \theta]^T$ (position et orientation dans le plan) et le vecteur de commande $\mathbf{u} = [V, \delta]^T$ (vitesse longitudinale et angle de braquage), on obtient les équations du mouvement suivantes :

Définition 11.12 Modèle bicyclette cinématique:

$$\dot{x} = V\cos(\theta) \tag{11.33}$$
$$\dot{y} = V\sin(\theta) \tag{11.34}$$
$$\dot{\theta} = \frac{V}{L}\tan(\delta) \tag{11.35}$$

Ces équations correspondent à un modèle d'état de la forme $\dot{\mathbf{q}} = f(\mathbf{q}, \mathbf{u})$, où le vecteur d'état est identique au vecteur de configuration, caractéristique des modèles purement cinématiques.

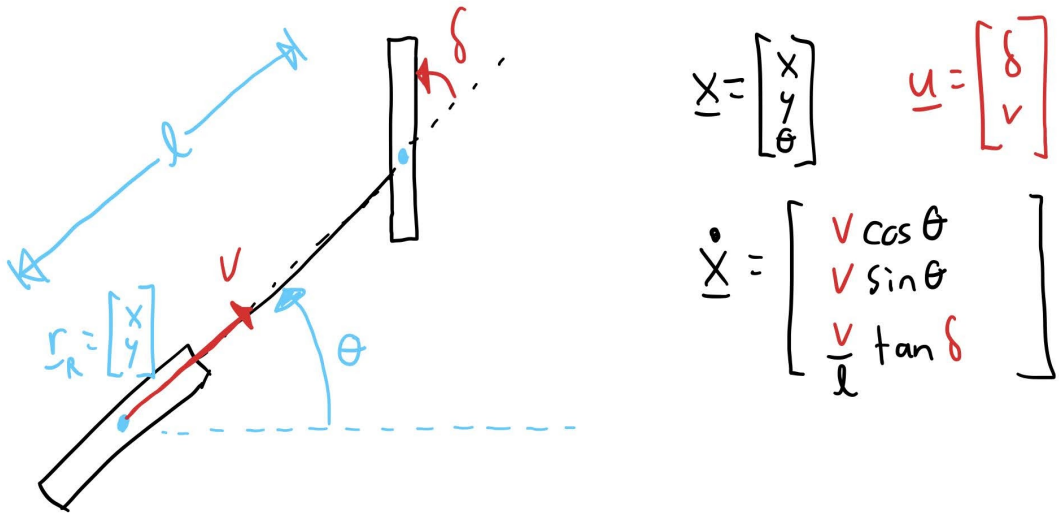


FIGURE 11.5 – Variables d'état du modèle bicyclette cinématique. L est l'empattement et θ représente l'orientation du véhicule.

11.5.2 Modèle Longitudinal

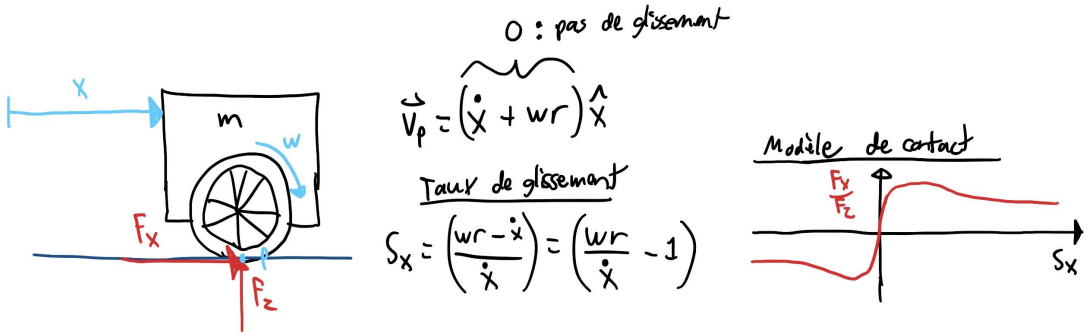


FIGURE 11.6 – Forces longitudinales s'appliquant au véhicule

11.5.3 Modèle Bicyclette Dynamique

Le modèle bicyclette dynamique prend en compte les forces, les masses et les inerties dans le plan avec 3 DDL. On réutilise les équations de corps rigide (voir section 11.3.3) avec les vitesses exprimées dans le repère du corps : u (longitudinale), v (latérale) et r (vitesse angulaire de lacet), et on ajoute les forces des pneus et une traînée aérodynamique :

Définition 11.13 Modèle bicyclette dynamique:

$$m(\dot{u} - rv) = (F_{x1}c_\delta - F_{y1}s_\delta) + F_{x2} - F_{aero} \tag{11.36}$$
$$m(\dot{v} + ru) = (F_{x1}s_\delta + F_{y1}c_\delta) + F_{y2} \tag{11.37}$$
$$I_z\dot{r} = a(F_{x1}s_\delta + F_{y1}c_\delta) - bF_{y2} \tag{11.38}$$

ou, tel qu'illustré à la figure 11.7 :
- Indice 1 : Roue avant (front), Indice 2 : Roue arrière (rear).
- a, b : Distances du centre de gravité (CG) aux essieux avant et arrière.
- c_δ, s_δ : Notation compacte pour $\cos(\delta)$ et $\sin(\delta)$, où δ est l'angle de braquage.
- F_{xi}, F_{yi} : Forces exprimées **dans le repère de la roue** (alignées avec le pneu).
- $F_{aero} = \frac{1}{2}\rho C_d A u|u|$: Traînée aérodynamique s'opposant au mouvement.

Exercice de code
Démonstration modèle bicyclette dynamique
`https://colab.research.google.com/drive/1NHcn2yDk9K5yCRhXo7X1MNb1Sp-bSHZh?usp=sharing`

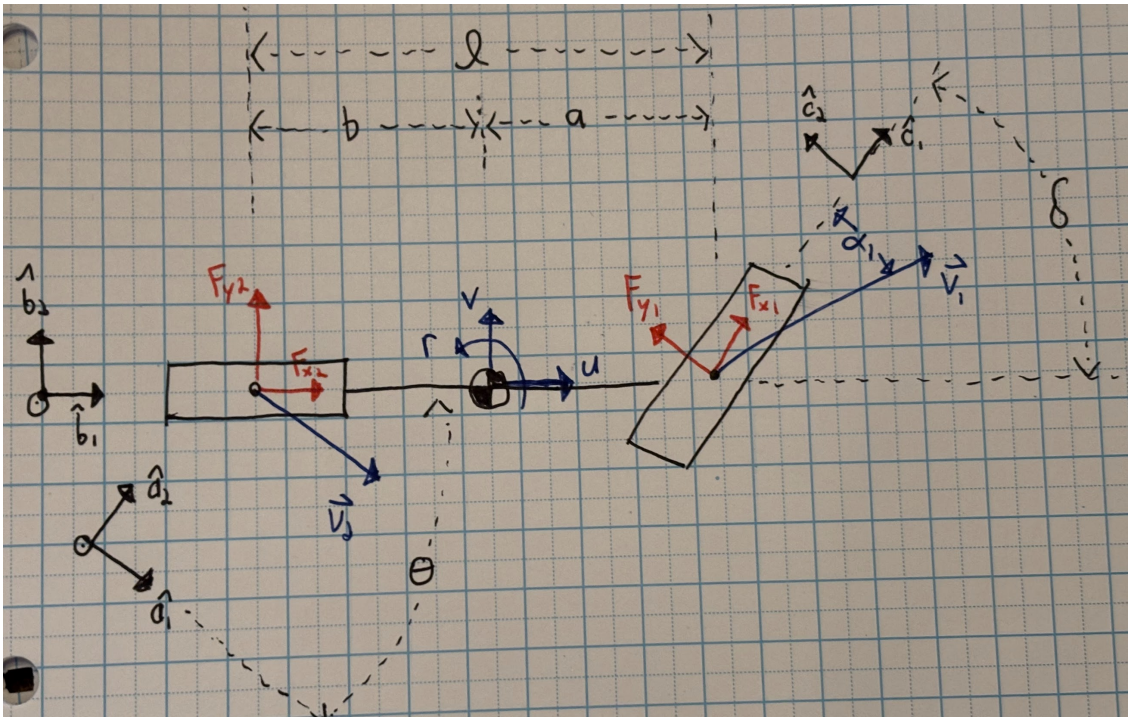


FIGURE 11.7 – Diagramme du corps libre. Les forces $F_{x,y}$ sont définies dans le repère de chaque roue.

Cinématique : Vitesses locales aux roues

Les modèles de pneus sont basés sur des vitesses relatives à la roue, nous devons d'abord exprimer la vitesse du centre de chaque roue *dans le repère de cette roue.*

Vitesses aux essieux (Repère Châssis) La cinématique du corps rigide nous donne la vitesse au point d'attache des essieux exprimée dans le repère du véhicule (b) :

$$\mathbf{v}_1^b = [u, \quad v + ar]^T \tag{11.39}$$

$$\mathbf{v}_2^b = [u, \quad v - br]^T \tag{11.40}$$

Projection dans le repère de la roue La roue avant ici n'a pas de rotation relative au corp, mais la roue avant est braquée de δ. On applique une rotation inverse pour exprimé les vitesses dans son repère local :

$$\mathbf{v}_1^c = [u\cos\delta + (v + ar)\sin\delta, \quad -u\sin\delta + (v + ar)\cos\delta]^T \tag{11.41}$$

Finalement les forces des pneus peuvent être calculés avec des fonctions de modèle de pneus, voir section 11.5.4. Les taux de glissements avant et arrière peuvent ici être calculé directement à partir des variables de vitesses ainsi :

$$\alpha_1 = -\arctan\left(\frac{v_{y1}^c}{|v_{x1}^c| + \epsilon}\right) \approx \delta - \arctan\left(\frac{v_{y1}^b}{|v_{x1}^b| + \epsilon}\right) \tag{11.42}$$

$$\alpha_2 = -\arctan\left(\frac{v_{y2}^b}{|v_{x2}^b| + \epsilon}\right) \tag{11.43}$$

$$\kappa_1 = \frac{R_1\omega_1 - v_{x1}^c}{|v_{x1}^c| + \epsilon} \tag{11.44}$$

$$\kappa_2 = \frac{R_2\omega_2 - v_{x2}^b}{|v_{x2}^b| + \epsilon} \tag{11.45}$$

11.5.4 Modélisation des forces d'intéractions pneus/route

La plupart des modèles de pneus ont la forme suivante : défini en terme de variable de taux de glissement et de la force normale :

Définition 11.14 Forme des modèles de pneus:

$$F_x = f(\alpha, \kappa, F_z) \qquad F_y = f(\alpha, \kappa, F_z) \tag{11.46}$$

La figure 11.5.4 illustre des courbes typiques pour ces fonctions.

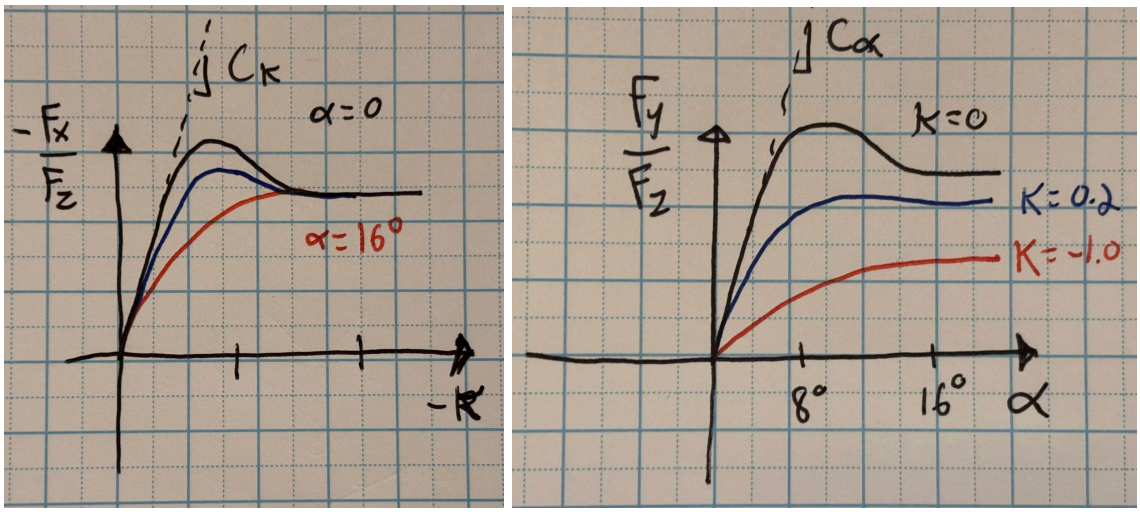


FIGURE 11.8 – Courbes typique de pneus

Définition des glissements

Deux variables de ratio de glissement sont typiquement utilisées comme intermédiaire pour calculer les forces. Les taux sont calculées à partir de la vitesse relative du centre d'une roue par rapport au sol, exprimée dans un repère local aligné avec la roue, et de sa vitesse de rotation, voir figure 11.5.4.

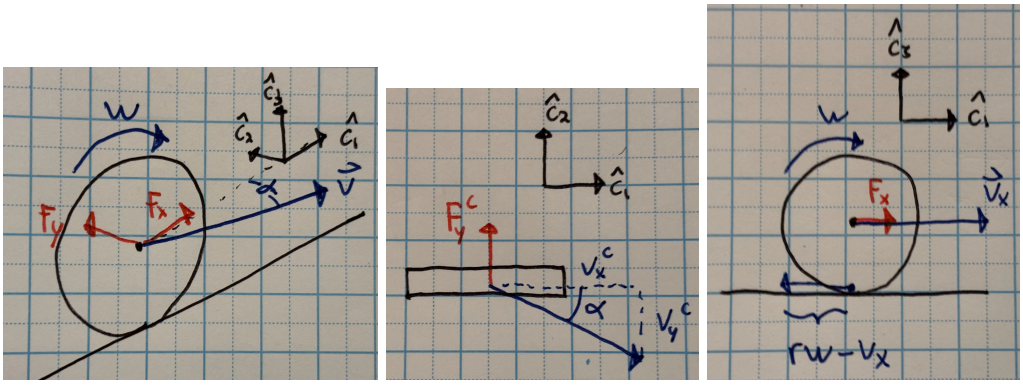


FIGURE 11.9 – Axes, vitesses et angles pour les modèles de pneus ($v_y < 0$)

Définition 11.15 Angle de dérive latéral (α):

Défini par le ratio entre vitesse latérale et longitudinale dans le repère de la roue. Le signe négatif est une convention pour assurer que la force latérale s'oppose au glissement.

$$\alpha = -\arctan\left(\frac{v_y}{|v_x| + \epsilon}\right) \tag{11.47}$$

Définition 11.16 Glissement longitudinal (κ):

Défini par la différence normalisée entre la vitesse tangentielle de la roue ($R\omega$) et la vitesse au sol (v_x^w).

$$\kappa = \frac{R\omega - v_x}{|v_x| + \epsilon} \tag{11.48}$$

Attention :

Les définitions exactes des taux de glissement peuvent varier selon le contexte et selon les ouvrages, ce n'est pas universel. De plus, nous avons inclus un *epsilon* au dénominateur dans les équations, ce qui est généralement inclus dans les méthodes numérique pour éviter les instabilités du modèle autour de vitesse nulles.

Modèle Linéaire

Lorsque le glissement est modéré, une modèle linéaire peut faire des bonnes prédictions :

Définition 11.17 Modèle de pneus linéaire:

$$F_x = C_k \kappa \tag{11.49}$$
$$F_y = C_\alpha \alpha \tag{11.50}$$

Les paramètres C sont souvent appelé les paramètres de rigidité des pneus, C_α est typiquement appelé *cornering stiffness* et est un paramètre important pour les caractéristiques de maneuvrabilité d'un véhicule. Pour le modèle linéaire de base, les forces x et y sont indépendantes, toutefois il est courant d'ajouter une saturation sur ces forces basée sur le **"Cercle de Friction"**, pour limiter la force totale du pneu à une limite supérieur disponible μF_z :

Modèle de Pacejka Surnommé dans le domaine ***Magic Formula***, ce modèle empirique capture le comportement non-linéaire et la saturation douce du pneu. La forme utilisée pour la force longitudinale (F_x) et latérale (F_y) est :

Définition 11.18 Formule magique de Pacejka (modèle de pneu non-linéaire):

$$F(x) = D\sin(C\arctan(Bx - E(Bx - \arctan(Bx)))) \tag{11.51}$$

Où x est le glissement (α ou κ). Les coefficients contrôlent la forme de la courbe :
- B (Stiffness) : Pente à l'origine (relié à C_α).
- C (Shape) : Forme de l'asymptote.
- D (Peak) : Valeur maximale (relié à μF_z).
- E (Curvature) : Courbure avant le pic.

11.5.5 Dynamique de Suspensions

Lorsque le véhicule accélère ou tourne, le poids se transfère, modifiant la traction disponible.

Modèle quart-de-véhicule

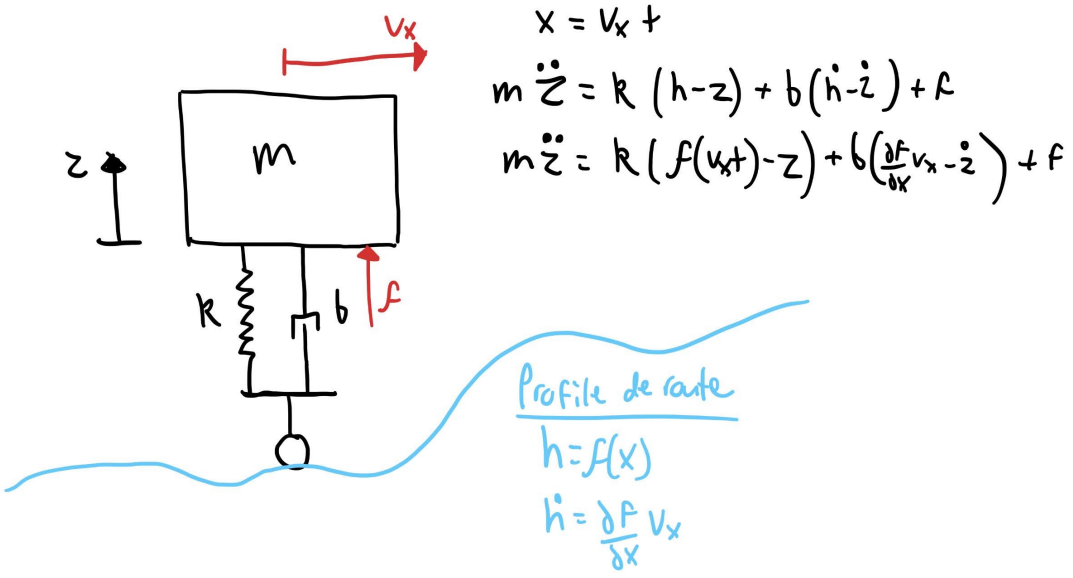


FIGURE 11.10 – Modèle de suspension 1-DDL (Quart de véhicule)

11.5.6 Interaction Pneu-Sol et terramecanique

Modèles de Sol Bekker-Wong

En robotique tout-terrain (agricole, planétaire), le sol se déforme sous la roue.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

11.6 Drones (Multirotors)

Modèle Planaire (2D)

Utile pour introduire les concepts sans la complexité de la 3D.

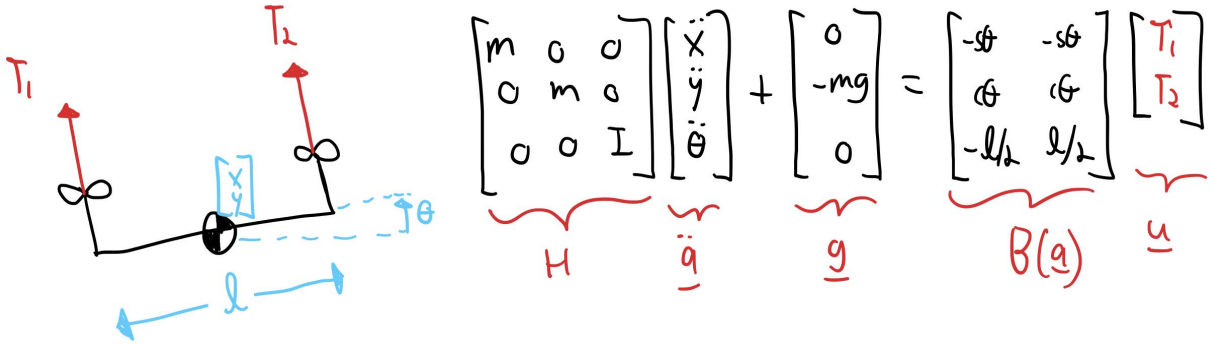


FIGURE 11.11 – Modèle simplifié : Drone planaire (3-DOF)

11.7 Avions (Ailes fixes)

11.7.1 Modèle d'avion 3DDL (Plan vertical)

Section en construction !

Cette section aborde la modélisation d'un aéronef à ailes fixes, dans le plan vertical (vue de côté). Le modèle à 3 DDL permet d'étudier la dynamique longitudinale :

— **Surge** (u) : Vitesse longitudinale (axe du fuselage).
— **Heave** (v) : Vitesse verticale (axe perpendiculaire aux ailes).
— **Pitch** (r) : Vitesse de tangage (rotation du nez).

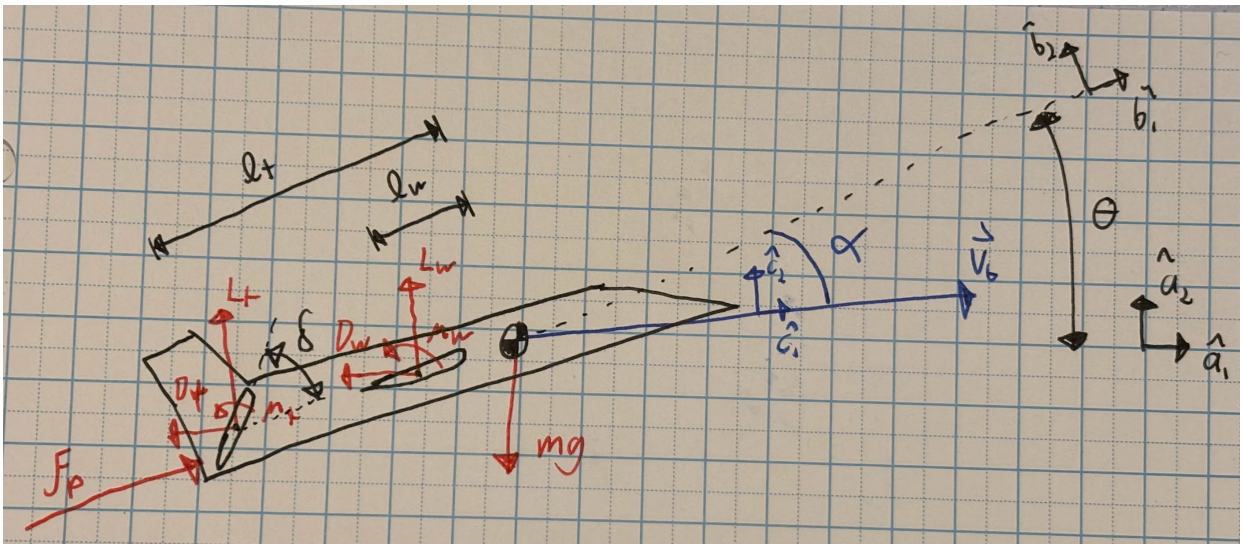


FIGURE 11.12 – Modèle simplifié : Avion 2D

Ce modèle capture les dynamiques essentielles de décrochage et de stabilité.

Exercice de code
Modèle avion planaire
https://colab.research.google.com/drive/15RBFDTUYSq0dFLBRd0VKLxJxXOONSYJI?usp=sharing

11.7.2 Équations du mouvement

On peut utiliser les équations génériques du corps rigide planaire, mais en incluant les forces aérodynamique et la gravité. Une particularité du modèle d'avion, est qu'une des actions est une position, l'angle de la gouverne, donc pas une source de force. Dans les équations, cela ce reflète comme un dépendance directe du vecteur de force aérodynamique à cette entrée

$$\mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{g}(\mathbf{q}) + \mathbf{f}_{aero}(\boldsymbol{\nu}, \delta) = \mathbf{B}\mathbf{f}_{\mathbf{p}} \tag{11.52}$$

où :

— $\boldsymbol{\nu} = [u, v, r]^T$ est le vecteur vitesse dans le repère du corps.
— $\mathbf{g}(\mathbf{q})$ est le vecteur gravité projeté dans le repère du corps (voir section 11.3.3).
— $\mathbf{f}_{aero}(\boldsymbol{\nu}, \delta)$ représente les forces de portance, traînée et moment.
— $\mathbf{B}\mathbf{f}_{\mathbf{p}}$ représente les forces des propulseurs.

Forces Aérodynamiques

Dans notre modèle simplifié, deux forces aérodynamiques s'appliquent sur le véhicule aux **centres de poussée** (CP) de l'aile et de la queue :

1. L'aile principale, situé à une distance l_w arrière du CG.
2. Queue, situé à une distance l_t arrière du CG.

Forces dans le repère vent La convention pour définir des forces aérodynamiques est de se placer dans un repère aligné avec le mouvement, la composante aligné avec la vitesse relative au fluide est appelée la trainée et la composante perpendidulaire est appelée la portance. L'air génère des forces que l'on approxime comme proportionnelles au carré de la vitesse relative absolue $V = \sqrt{u^2 + v^2}$, avec des coefficients dépendant de l'angle d'attaque α :

$$\alpha = \arctan\left(\frac{v}{u}\right) \tag{11.53}$$

Les forces aérodynamiques sont définies par les coefficients de portance (C_L), de traînée (C_D) et de moment (C_m) :

$$\mathbf{f}^c_{aero}(\alpha) = \begin{bmatrix} -\frac{1}{2}\rho S C_D(\alpha) V^2 \\ \frac{1}{2}\rho S C_L(\alpha) V^2 \\ \frac{1}{2}\rho S \bar{c} C_m(\alpha) V^2 \end{bmatrix} \tag{11.54}$$

Notez que la traînée (C_D) s'oppose au mouvement (signe négatif dans la direction $\hat{c}_1$ et la portance (C_L) agit perpendiculairement (axe $\hat{c}_2$).

Ici on utilise un modèle simplifié "tout régime" (incluant le vol normal et les acrobaties/décrochage), avec des courbes caractéristiques d'une aile (profil symétrique) par :

$$C_L(\alpha) = C_{L,max} \sin(2\alpha) \tag{11.55}$$

$$C_D(\alpha) = C_{D,0} + C_{D,max} \sin^2(\alpha) \tag{11.56}$$

$$C_m(\alpha) = -C_{m,max} \sin(\alpha)\cos(\alpha) \tag{11.57}$$

On projette ensuite ces forces dans le repère du corps à l'aide d'une matrice de rotation dépendant de l'angle d'attaque α. L'équation de transformation est :

$$\mathbf{f}^b_{aero} = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{f}^c_{aero} \tag{11.58}$$

Somme des forces et moments au CG Le force aérodynamique générale $\mathbf{f}_{aero}(\boldsymbol{\nu}, \delta)$ appliqué au centre de gravité est la somme des contributions de chaque surface.

TODO : détails à venir

11.7.3 Propulsion et Vecteur de Commande

Le vecteur de commande est $\mathbf{u} = [f_p, \delta_e]^T$. Ici nous avons une force (propulsion) et une configuration géométrique (angle δ_e).

1. **Poussée (f_p)** : Force générée par le moteur, appliquée généralement selon l'axe longitudinal x du corps. Elle entre dans l'équation via la matrice $\mathbf{B}$.
2. **Élévateur (δ)** : Comme vu précédemment, il n'apparaît pas dans la matrice $\mathbf{B}$ linéaire classique, mais modifie directement le vecteur $\mathbf{f}_{aero}$.

$$\mathbf{B}\mathbf{f}_p = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} f_p \tag{11.59}$$

11.8 Véhicules nautiques

11.8.1 Modèle de manœuvre basse-vitesse

Section en construction !

Cette section s'intéresse à la modélisation des navires et robots de surface. Bien qu'un navire évolue dans un espace 3D (6 DDL), pour les applications de positionnement dynamique ou de manœuvre de port, on utilise couramment un modèle planaire simplifié à 3 DDL :

— **Surge** (u) : Avancement longitudinal.
— **Sway** (v) : Dérive latérale.
— **Yaw** (r) : Lacet (cap).

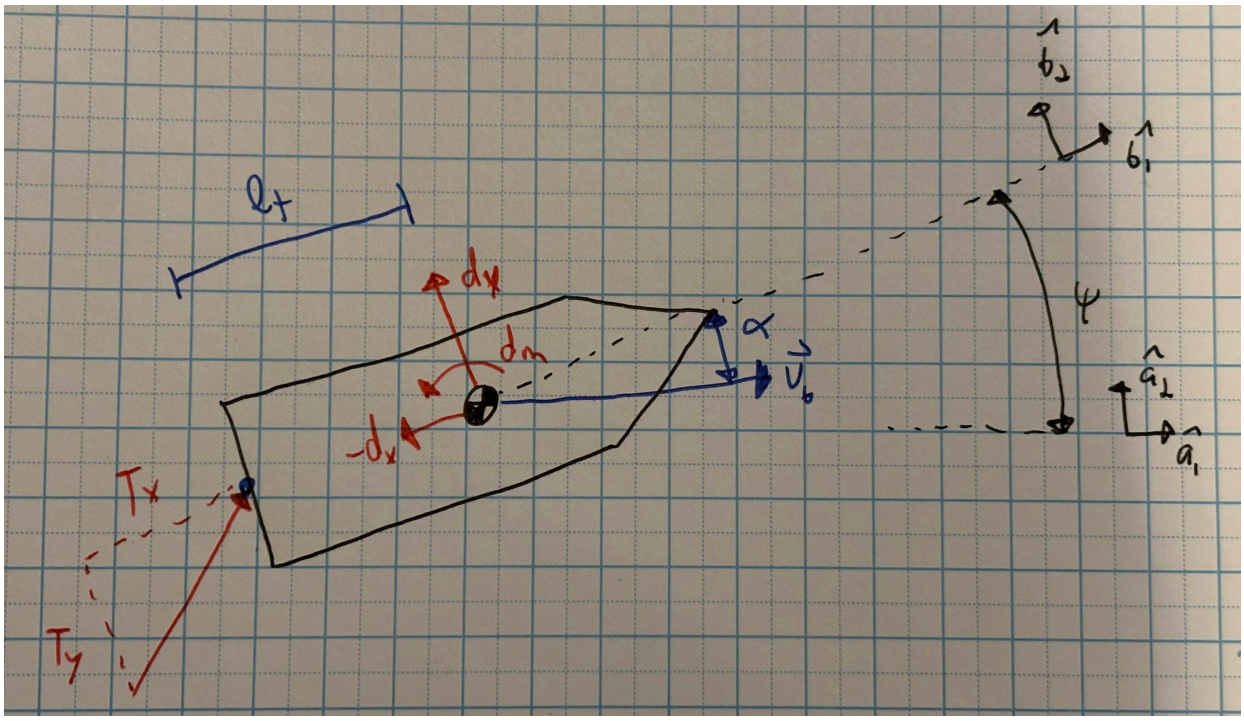


FIGURE 11.13 – Modèle simplifié : Bateau 2D

Le modèle présenté ici est basé sur le livre de Fossen [?], spécifiquement les modèles de manœuvre à basse vitesse.

Exercice de code
Modèle de véhicule nautique planaire basse vitesse
`LINK_TODO`

11.8.2 Équations du mouvement

L'équation du mouvement dans le repère du corps suit la structure standard Newton-Euler, mais adaptée pour inclure les effets hydrodynamiques :

$$\mathbf{M}\dot{\boldsymbol{\nu}} + \mathbf{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \mathbf{d}(\boldsymbol{\nu}) = \mathbf{B}\mathbf{u} \tag{11.60}$$

où :

— $\boldsymbol{\nu} = [u, v, r]^T$ est le vecteur vitesse dans le repère du corps.
— $\mathbf{M}$ est la matrice d'inertie (incluant théoriquement la masse ajoutée).

— $\mathbf{d}(\boldsymbol{\nu})$ est les forces hydrodynamiques (traînée).
— $\mathbf{Bu}$ représente les forces des propulseurs.

Forces Hydrodynamiques

L'eau oppose une résistance complexe au mouvement. Le modèle implémenté combine deux effets, un amortissement visqueux linéaire qui domine à très basse vitesse et des lois de portances/trainées quadratique, similaires aux modèles aérodynamique des avions. L'approche utilisée calcule les coefficients de traînée C_x, C_y, C_m en fonction de l'angle d'incidence avec l'eau :

$$\alpha = -\arctan\left(\frac{v}{u}\right) \tag{11.61}$$

Les forces sont ensuite calculées selon la pression dynamique :

$$\mathbf{d}_{quad}(\boldsymbol{\nu}) = \begin{bmatrix} -\frac{1}{2}\rho A_{front} C_x(\alpha) V^2 \\ -\frac{1}{2}\rho A_{lat} C_y(\alpha) V^2 \\ -\frac{1}{2}\rho A_{lat} L_{oa} C_m(\alpha) V^2 \end{bmatrix} \tag{11.62}$$

où $V^2 = u^2 + v^2$ et les C_x, C_y, C_m sont des courbes de coefficients fonction de l'angle avec le courant. Dans un modèle simplifié (basé sur [**?**]), ces coefficients sont approximés par des fonctions trigonométriques :

$$C_x(\alpha) = -C_{x,max} \cos(\alpha)|\cos(\alpha)| \tag{11.63}$$
$$C_y(\alpha) = C_{y,max} \sin(\alpha)|\sin(\alpha)| \tag{11.64}$$
$$C_m(\alpha) = C_{m,max} \sin(2\alpha) \tag{11.65}$$

11.8.3 Propulsion

Si la propulsion est un propulseur orientable situé à l'arrière du bateau, la force de proulsion est un vecteur de force 2D $\mathbf{u} = [T_x, T_y]^T$ appliqué à une distance l_t derrière le centre de gravité (par exemple, un moteur hors-bord ou un safran orientable).

La relation entre la commande et les efforts généralisés sur le corps est donnée par la matrice d'allocation $\mathbf{B}$:

$$\boldsymbol{\tau} = \mathbf{Bu} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & -l_t \end{bmatrix} \begin{bmatrix} T_x \\ T_y \end{bmatrix} = \begin{bmatrix} T_x \\ T_y \\ -l_t T_y \end{bmatrix} \tag{11.66}$$

On remarque qu'une force latérale T_y génère à la fois une force de dérive et un moment de lacet, créant un couplage fort typique des navires.

Chapitre 12

Véhicule : Analyse et stabilité

12.1 Linéarisation

12.2 Stabilité statique

12.2.1 Avion

12.2.2 Voiture

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Chapitre 13

Introduction à la commande optimale

Vue d'ensemble du chapitre

CONTEXTE ET MOTIVATION

Jusqu'à présent, nous avons abordé la **génération de trajectoires** et la **planification de mouvement** pour résoudre l'aspect géométrique du problème : « comment aller d'un point A à un point B », mais sans vérifier si une séquence d'actions permette d'exécuter cette séquence. Du côté de la commande, plusieurs lois de commande ont été proposées qui donnent des garanties de convergence vers une position cible, mais typiquement sans tenir compte des contraintes, ni de l'efficacité de la solution.
La commande optimale nous offre le cadre mathématique pour passer du simple mouvement admissible géométriquement au mouvement dynamique « idéal » (temps minimum, consommation d'énergie minimale, etc.). Ce chapitre introduit les fondements théoriques nécessaires pour traiter le système sous sa forme d'état générique $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t)$.

OBJECTIFS D'APPRENTISSAGE

À la fin de ce chapitre, vous serez capable de :
- Formuler mathématiquement un problème de commande optimale.
- Identifier les critères de performance (coûts) appropriés pour différentes tâches robotiques.

PRÉREQUIS ET CONCEPTS CLÉS

- Espace d'état et représentations vectorielles.



Capsule vidéo
Introduction à la commande optimale
`https://youtu.be/3x6Vg-RRZ50`

13.1 Formalisation mathématique

La plupart des méthodes de commande optimale sont conçues pour travailler avec la forme la plus générale des équations différentielles, permettant de traiter une vaste gamme de systèmes robotiques.

Capsule vidéo
Exemple de loi de commande optimale pour un double intégrateur
https://youtu.be/wKjEAXFvXlQ

Définition 13.1 Le problème de commande optimale:

Le problème de commande optimale est défini par une dynamique :

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}, t) \tag{13.1}$$

des contraintes sur les états et les actions (limites d'actionnement, butées) :

$$\boldsymbol{x} \in \mathcal{X}_{ok} \tag{13.2}$$
$$\boldsymbol{u} \in \mathcal{U}_{ok} \tag{13.3}$$

un objectif caractérisé par une fonction de coût J à minimiser :

$$J = \int_{t_0}^{t_f} g(\boldsymbol{x}, \boldsymbol{u}) dt + h(\boldsymbol{x}(t_f)) \tag{13.4}$$

et une (ou une distribution de) conditions initiales $\boldsymbol{x}_0$.

Dans cette formulation, $g(\boldsymbol{x}, \boldsymbol{u})$ représente le coût instantané (ex : consommation d'énergie à chaque instant) et $h(\boldsymbol{x}_f)$ représente le coût terminal (ex : précision de l'arrivée au point cible).

13.2 Théorie de la commande optimale

L'analyse théorique repose sur deux piliers : l'un traitant le problème de manière globale (HJB) et l'autre de manière locale (PMP).

13.2.1 L'équation de Hamilton-Jacobi-Bellman (HJB)

L'approche par programmation dynamique utilise le principe d'optimalité de Bellman pour définir la fonction de valeur optimale $J^*(\boldsymbol{x}, t) = \min_{\boldsymbol{u}} J$.

Définition 13.2 Équation de Hamilton-Jacobi-Bellman:

La fonction de valeur J^* doit satisfaire l'équation aux dérivées partielles :

$$-\frac{\partial J^*}{\partial t} = \min_{\boldsymbol{u} \in \mathcal{U}_{ok}} \left[g(\boldsymbol{x}, \boldsymbol{u}) + \left(\frac{\partial J^*}{\partial \boldsymbol{x}} \right)^T f(\boldsymbol{x}, \boldsymbol{u}, t) \right] \tag{13.5}$$

avec la condition terminale $J^*(\boldsymbol{x}, t_f) = h(\boldsymbol{x}(t_f))$.

13.2.2 Le Principe du Maximum de Pontryagin (PMP)

Le PMP définit les conditions nécessaires pour qu'une trajectoire soit optimale en introduisant un vecteur de variables adjointes $\boldsymbol{\lambda}$ (co-états).

Définition 13.3 Le Hamiltonien et le PMP:

On définit le Hamiltonien H comme :

$$H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, t) = g(\boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\lambda}^T f(\boldsymbol{x}, \boldsymbol{u}, t) \tag{13.6}$$

Une trajectoire optimale doit satisfaire :

1. **Équation du co-état :** $\dot{\boldsymbol{\lambda}} = -\frac{\partial H}{\partial \boldsymbol{x}}$
2. **Équation d'état :** $\dot{\boldsymbol{x}} = \frac{\partial H}{\partial \boldsymbol{\lambda}}$
3. **Stationnarité :** $\boldsymbol{u}^* = \arg\min_{\boldsymbol{u} \in \mathcal{U}_{ok}} H(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}, t)$

13.3 Grandes familles de méthodes

Pour résoudre un problème de commande optimale, le choix fondamental réside dans la définition de l'**espace de décision**. On distingue deux philosophies majeures selon que l'on cherche une solution globale (une loi de commande) ou une solution locale (une trajectoire spécifique).

13.3.1 Approches basées sur la loi de commande

Ces méthodes visent à trouver une loi de commande optimale π^*, capable de définir l'action optimale pour n'importe quel état $\boldsymbol{x}$ rencontré.

Définition 13.4 Programmation dynamique et apprentissage par renforcement:

On cherche une politique $\boldsymbol{u} = \pi(\boldsymbol{x}, t)$ qui minimise le coût J pour tout l'espace d'état $\mathcal{X}_{ok}$:

$$\pi(\boldsymbol{x}, t) = \arg \min_{\boldsymbol{u} \in \mathcal{U}_{ok}} J^*(\boldsymbol{x}, t) \tag{13.7}$$

En apprentissage par renforcement (RL), cette politique est apprise par essai-erreur, tandis qu'en programmation dynamique, elle est calculée avec des méthodes numériques itératives en utilisant les équations. Ces méthodes sont très génériques et potentiellement très puissantes, mais en pratique le problème est souvent *intractable* (le temps de calcul est trop long) et on doit chercher des solutions très approximées. Les techniques d'apprentissage par renforcement ont connu de grands succès récents, mais c'est typiquement un défi de les faire fonctionner et ce n'est pas la norme en date d'aujourd'hui (2026). Le sujet ne sera pas abordé en détail dans ces notes, il est le sujet de notes dédié au sujet disponible ici : `https://www.alexandregirard.com/teaching/dp/PDF/DP_Notes.pdf`

Alternativement, si on peut linéariser la dynamique du système, on peut utiliser une technique très pratique appelée la synthèse LQR, couramment utilisée en commande.

Définition 13.5 Régulateur Quadratique Linéaire (LQR):

Cas particulier où la dynamique est linéaire et le coût quadratique :

$$f(\boldsymbol{x}, \boldsymbol{u}, t) = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \tag{13.8}$$

$$g(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x} + \boldsymbol{u}^T \boldsymbol{R} \boldsymbol{u} \tag{13.9}$$

$$h(\boldsymbol{x}(t_f)) = \boldsymbol{x}(t_f)^T \boldsymbol{Q}_f \boldsymbol{x}(t_f) \tag{13.10}$$

La solution se trouve analytiquement et est une politique linéaire optimale de la forme $\boldsymbol{u} = -\boldsymbol{K}\boldsymbol{x}$.

13.3.2 Approches basées sur la trajectoire

Plutôt que de chercher une solution universelle, ces méthodes se concentrent sur la recherche d'une trajectoire $\{\boldsymbol{x}^*(t), \boldsymbol{u}^*(t)\}$ partant d'une condition initiale spécifique $\boldsymbol{x}_0$.

Définition 13.6 Optimisation de trajectoire (Méthodes directes):

On discrétise le problème continu en N segments de temps pour transformer le problème en programmation non-linéaire (NLP) :

$$\begin{aligned} \min_{\boldsymbol{u}_0,\dots,\boldsymbol{u}_{N-1}} \quad & \sum_{k=0}^{N-1} g_d(\boldsymbol{x}_k, \boldsymbol{u}_k)\Delta t + h(\boldsymbol{x}_N) \\ \text{s.c.} \quad & \boldsymbol{x}_{k+1} = f_d(\boldsymbol{x}_k, \boldsymbol{u}_k) \\ & \boldsymbol{x}_k \in \mathcal{X}_{ok}, \quad \boldsymbol{u}_k \in \mathcal{U}_{ok} \end{aligned}$$

Cette technique est abordée en détail au chapitre 16.
Il existe une autre famille de techniques moins courantes :

Définition 13.7 Méthodes indirectes:

On cherche à satisfaire les conditions nécessaires d'optimalité analytiques le long d'une trajectoire unique. Cela revient à résoudre un problème aux limites basé sur le principe du maximum de Pontryagin.

13.3.3 Hybrides

En pratique, les approches basées sur une loi de commande et celles basées sur la trajectoire ne sont pas mutuellement exclusives. Elles sont souvent combinées pour bénéficier à la fois de la précision de la planification et de la robustesse des boucles fermées.

Définition 13.8 Optimisation de trajectoire + LQR:

Cette approche hybride sépare la planification de la régulation en deux étapes distinctes :

1. **Planification (Hors-ligne/basse fréquence) :** On utilise l'optimisation de trajectoire pour trouver une trajectoire nominale optimale $\{\boldsymbol{x}^*(t), \boldsymbol{u}^*(t)\}$.
2. **Régulation (Temps réel) :** On linéarise la dynamique autour de cette trajectoire pour concevoir un régulateur LQR variant dans le temps. La loi de commande devient :

$$\boldsymbol{u}(t) = \boldsymbol{u}^*(t) - \boldsymbol{K}(t)(\boldsymbol{x}(t) - \boldsymbol{x}^*(t))$$

Le gain $\boldsymbol{K}(t)$ permet de rejeter les perturbations et de maintenir le robot dans un « tube » de stabilité autour du plan initial.

Alternativement, une autre stratégie pour donner un mécanisme de rétroaction est de constamment remesurer l'état actuel et ré-optimiser une trajectoire à partir de ce nouveau point :

Définition 13.9 Commande prédictive (MPC):

La commande prédictive (*Model Predictive Control* ou MPC) consiste à résoudre un problème d'optimisation de trajectoire en temps réel sur un horizon de temps glissant :

— À chaque instant t, on résout une optimisation sur un horizon court à partir de l'état actuel mesuré $\boldsymbol{x}(t)$.
— On n'applique que la première commande $\boldsymbol{u}_0$ de la séquence calculée.
— Au pas de temps suivant, on décale l'horizon et on recommence l'optimisation avec la nouvelle mesure de l'état.

Bien que le MPC repose mathématiquement sur l'optimisation de trajectoire, le fait de ré-optimiser à chaque pas de temps à partir de l'état réel transforme cette méthode en une **politique de rétroaction** implicite $\boldsymbol{u} = \pi(\boldsymbol{x})$. C'est l'un des outils les plus puissants pour gérer simultanément la dynamique et les contraintes $\mathcal{X}_{ok}$ et $\mathcal{U}_{ok}$.

Chapitre 14

Commande de véhicules

14.1 Lois cinématiques

14.1.1 Poursuite

14.1.2 Stanley

14.1.3 Ligne de vue

14.2 Lois dynamiques

14.2.1 Boucles en cascades

14.2.2 LQR

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Chapitre 15

Planification cinématique

15.1 Espace configuration

15.2 Fonction guidage

15.3 RRT

15.4 RRT*

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Chapitre 16

Optimisation de trajectoires

Chapitre en construction!!

Sources externes utiles :
https://arxiv.org/pdf/1707.00284
https://underactuated.mit.edu/trajopt.html

16.1 Introduction et contexte

Trouver une loi de commande optimale pour un système non linéaire avec des contraintes est typiquement impossible sans faire des approximations. Toutefois, résoudre le problème plus simple de trouver une trajectoire optimale à partir d'un seul état est possible numériquement avec des algorithmes d'optimisation, typiquement un programme quadratique ou non linéaire (voir chapitre 21). Contrairement à la planification cinématique qui se limite souvent à l'espace des configurations $\boldsymbol{q}(t)$, le cadre ici est générique et s'applique à la forme d'état générale $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$.



Exercice de code
Démo d'introduction à l'optimisation de trajectoires
https://colab.research.google.com/drive/1yq2GHAkvO6fTF2W-tRbACDBa9_scec2k?usp=sharing

16.2 Formalisation mathématique

Pour l'optimisation de trajectoire, le but général est de trouver une fonction temporelle pour les états et les actions $\{\boldsymbol{x}^*(t), \boldsymbol{u}^*(t)\}$, i.e. une trajectoire, qui minimise un coût tout en satisfaisant des contraintes.

Définition 16.1 Le problème d'optimisation de trajectoire:

On cherche à déterminer les trajectoires optimales de l'état $\boldsymbol{x}(t)$ et de la commande $\boldsymbol{u}(t)$ qui minimisent la fonctionnelle de coût :

$$\min_{\boldsymbol{x}(t),\boldsymbol{u}(t)} J = \int_{t_0}^{t_f} g(\boldsymbol{x}(t), \boldsymbol{u}(t), t)dt + h(\boldsymbol{x}(t_f)) \tag{16.1}$$

Sujet aux contraintes d'égalité suivantes :

$$\dot{\boldsymbol{x}}(t) - f(\boldsymbol{x}(t), \boldsymbol{u}(t), t) = \mathbf{0} \quad \text{(Dynamique du système)} \tag{16.2}$$

$$\boldsymbol{x}(t_0) - \boldsymbol{x}_{init} = \mathbf{0} \quad \text{(État initial imposé)} \tag{16.3}$$

aux contraintes d'inégalité suivantes :

$$\boldsymbol{x}_{min} \leq \boldsymbol{x}(t) \leq \boldsymbol{x}_{max} \quad \text{(Enveloppe d'état)} \tag{16.4}$$
$$\boldsymbol{u}_{min} \leq \boldsymbol{u}(t) \leq \boldsymbol{u}_{max} \quad \text{(Enveloppe des actions)} \tag{16.5}$$

et possiblement des contraintes d'inégalité supplémentaires qui illustrent des limites opérationnelles plus complexes, comme des obstacles.

Ce problème peut se traduire en un programme mathématique, voir section 16.3, qui est typiquement non convexe et non linéaire. Donc sans garantie de converger vers une solution, de plus typiquement si le programme converge, c'est vers un optimum local.

Parfois, on désire seulement trouver une solution qui fonctionne, certains algorithmes vont plutôt tenter de résoudre un problème de recherche de trajectoire :

Définition 16.2 Le problème de recherche de trajectoire:

On cherche à déterminer une trajectoire réalisable $\boldsymbol{x}(t)$ et une commande $\boldsymbol{u}(t)$ qui débutent à un état initial et terminent sur un état cible :

$$\text{trouver} \quad \boldsymbol{x}(t), \boldsymbol{u}(t) \tag{16.6}$$

Sujet aux contraintes d'égalité suivantes :

$$\dot{\boldsymbol{x}}(t) - f(\boldsymbol{x}(t), \boldsymbol{u}(t), t) = \boldsymbol{0} \quad \text{(Dynamique du système)} \tag{16.7}$$
$$\boldsymbol{x}(t_0) - \boldsymbol{x}_{init} = \boldsymbol{0} \quad \text{(État initial imposé)} \tag{16.8}$$
$$\boldsymbol{x}(t_f) - \boldsymbol{x}_{cible} = \boldsymbol{0} \quad \text{(État cible atteint)} \tag{16.9}$$

aux contraintes d'inégalité suivantes :

$$\boldsymbol{x}_{min} \leq \boldsymbol{x}(t) \leq \boldsymbol{x}_{max} \quad \text{(Enveloppe d'état)} \tag{16.10}$$
$$\boldsymbol{u}_{min} \leq \boldsymbol{u}(t) \leq \boldsymbol{u}_{max} \quad \text{(Enveloppe des actions)} \tag{16.11}$$

et possiblement des contraintes d'inégalité supplémentaires qui illustrent des limites opérationnelles plus complexes, comme l'évitement d'obstacles.

Alternativement, bien que le problème d'optimisation de trajectoire général soit non linéaire et non convexe, il existe une classe de problèmes fondamentaux bénéficiant d'une structure mathématique privilégiée : les **Programmes Quadratiques (QP)**. Ces problèmes sont caractérisés par une solution unique globale et des algorithmes de résolution extrêmement performants. Il est possible de retrouver cette forme si on utilise une approximation linéaire pour la dynamique et une fonction de coût quadratique :

Définition 16.3 Problème d'optimisation de trajectoire Linéaire-Quadratique:

On cherche à déterminer les trajectoires optimales $\boldsymbol{x}(t)$ et $\boldsymbol{u}(t)$ qui minimisent une fonction de coût **quadratique** :

$$\min_{\boldsymbol{x}(t), \boldsymbol{u}(t)} J = \frac{1}{2}\boldsymbol{x}(t_f)^T \boldsymbol{Q}_f \boldsymbol{x}(t_f) + \int_{t_0}^{t_f} \left(\frac{1}{2}\boldsymbol{x}(t)^T \boldsymbol{Q}\boldsymbol{x}(t) + \frac{1}{2}\boldsymbol{u}(t)^T \boldsymbol{R}\boldsymbol{u}(t) \right) dt \tag{16.12}$$

Sujet à une dynamique **linéaire** (contraintes d'égalité) :

$$\dot{\boldsymbol{x}}(t) - \boldsymbol{A}\boldsymbol{x}(t) - \boldsymbol{B}\boldsymbol{u}(t) = \boldsymbol{0} \tag{16.13}$$
$$\boldsymbol{x}(t_0) - \boldsymbol{x}_{init} = \boldsymbol{0} \tag{16.14}$$

et à des contraintes d'**inégalité linéaires** :

$$\boldsymbol{x}_{min} \leq \boldsymbol{x}(t) \leq \boldsymbol{x}_{max} \tag{16.15}$$
$$\boldsymbol{u}_{min} \leq \boldsymbol{u}(t) \leq \boldsymbol{u}_{max} \tag{16.16}$$

qui peuvent inclure des contraintes polyédriques additionnelles ($\boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}\boldsymbol{u} \leq \boldsymbol{e}$).

Cette approximation du vrai problème d'optimisation non linéaire va mener à des programmes quadratiques qui ont le grand avantage de garantir une convergence en temps fini vers le minimum global.

16.3 Méthodes de transcription en programme mathématique

La transcription convertit le problème d'optimisation de trajectoire (général **??** ou linéaire-quadratique **??**) en un programme mathématique avec un nombre de variables fini, sous un format standard de programme mathématique (voir chapitre 21). Avec ces méthodes, on peut généralement espérer trouver un minimum local de trajectoire optimale, avec des algorithmes basés sur une descente du gradient. Il est donc à noter que la performance de ces méthodes est très sensible à un point de départ, une estimation initiale de la solution sur laquelle débutera la descente du gradient.

La première étape est de discrétiser le temps en n intervalles et les fonctions continues $\boldsymbol{x}(t)$ et $\boldsymbol{u}(t)$ par une séquence de points à ces instants précis :

$$t \Rightarrow t_0, \ldots, t_i, \ldots, t_{n-1}, t_n \tag{16.17}$$
$$\boldsymbol{x}(t) \Rightarrow \boldsymbol{x}_0, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_{n-1}, \boldsymbol{x}_n \tag{16.18}$$
$$\boldsymbol{u}(t) \Rightarrow \boldsymbol{u}_0, \ldots, \boldsymbol{u}_i, \ldots, \boldsymbol{u}_{n-1} \tag{16.19}$$

avec comme notation $\boldsymbol{x}_i = \boldsymbol{x}(t_i)$ et $\boldsymbol{u}_i = \boldsymbol{u}(t_i)$.

16.3.1 Tir simple (*Direct Shooting*)

Dans cette approche, seules les entrées de commande sont optimisées. L'état est une fonction implicite calculée par simulation.

Définition 16.4 Transcription par Tir Simple:

La transcription par tir simple consiste à formuler le problème d'optimisation avec comme variables de décision n points d'une trajectoire d'action $\boldsymbol{u}$ discrétisée, i.e. $\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{n-1}$. On cherche donc la séquence d'actions qui minimise la fonction de coût, qui est ici une version approximée en temps discret :

$$\min_{\boldsymbol{u}_0,\ldots,\boldsymbol{u}_{n-1}} \quad J = h(\boldsymbol{x}_n) + \sum_{i=0}^{n-1} g(\boldsymbol{x}_i, \boldsymbol{u}_i)\Delta t$$
$$\text{sujet à}$$
$$\boldsymbol{u}_{min} \leq \boldsymbol{u}_i \leq \boldsymbol{u}_{max}$$

avec la trajectoire (les $\boldsymbol{x}_i$) obtenue par intégration aux temps discrétisés :

$$\boldsymbol{x}_0 = \boldsymbol{x}_{init} \quad \text{et} \quad \boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \int_{t_i}^{t_{i+1}} f(\boldsymbol{x}, \boldsymbol{u})dt, \quad \forall i \in \{0, \ldots, n-1\}$$

Le schéma d'intégration peut être un simulateur *boîte noire*, on doit juste pouvoir obtenir la séquence des états en réponse à une condition initiale et une séquence d'actions.

16.3.2 Transcription directe (*Direct Transcription*)

Contrairement aux méthodes de tir, avec la transcription directe, les états $\boldsymbol{x}_i$ sont ici des variables de décision indépendantes pour le solveur, et la dynamique est imposée via des contraintes d'égalité algébriques :

Définition 16.5 Transcription Directe:

La transcription directe consiste à utiliser comme variables de décision à la fois l'état $\boldsymbol{x}$ et l'action $\boldsymbol{u}$ aux n points de la trajectoire.

$$\begin{aligned} \min_{\substack{\boldsymbol{x}_0,\dots,\boldsymbol{x}_n \\ \boldsymbol{u}_0,\dots,\boldsymbol{u}_{n-1}}} \quad & J = h(\boldsymbol{x}_n) + \sum_{i=0}^{n-1} g(\boldsymbol{x}_i, \boldsymbol{u}_i)\Delta t \\ \text{sujet à} \quad & \\ & \boldsymbol{x}_{min} \leq \boldsymbol{x}_i \leq \boldsymbol{x}_{max} \\ & \boldsymbol{u}_{min} \leq \boldsymbol{u}_i \leq \boldsymbol{u}_{max} \end{aligned}$$

avec la dynamique imposée comme contrainte d'égalité :

$$\begin{aligned} & \boldsymbol{x}_0 = \boldsymbol{x}_{init} \\ & \boldsymbol{x}_{i+1} - f_d(\boldsymbol{x}_i, \boldsymbol{u}_i) = \boldsymbol{0}, \quad \forall i \in \{0, \dots, n-1\} \end{aligned}$$

Le schéma d'intégration f_d est typiquement une approximation discrète comme la méthode d'Euler :

$$f_d(\boldsymbol{x}_i, \boldsymbol{u}_i) = \boldsymbol{x}_i + \Delta t f(\boldsymbol{x}_i, \boldsymbol{u}_i)$$

Si on approxime la dynamique linéaire et utilise un coût quadratique, le programme est quadratique :

Définition 16.6 Transcription Directe Linéaire-Quadratique:

Ici l'objectif est une fonction quadratique et les contraintes sont linéaires. On cherche à minimiser :

$$\begin{aligned} \min_{\substack{\boldsymbol{x}_0,\dots,\boldsymbol{x}_n \\ \boldsymbol{u}_0,\dots,\boldsymbol{u}_{n-1}}} \quad & J = \boldsymbol{x}_n^T \boldsymbol{Q}_f \boldsymbol{x}_n + \sum_{i=0}^{n-1} (\boldsymbol{x}_i^T \boldsymbol{Q} \boldsymbol{x}_i + \boldsymbol{u}_i^T \boldsymbol{R} \boldsymbol{u}_i)\Delta t \\ \text{sujet à} \quad & \\ & \boldsymbol{x}_{i+1} = \boldsymbol{A}_i \boldsymbol{x}_i + \boldsymbol{B}_i \boldsymbol{u}_i, \quad \forall i \in \{0, \dots, n-1\} \\ & \boldsymbol{x}_{min} \leq \boldsymbol{x}_i \leq \boldsymbol{x}_{max} \\ & \boldsymbol{u}_{min} \leq \boldsymbol{u}_i \leq \boldsymbol{u}_{max} \end{aligned}$$

Pour ce type de problème, si les matrices de poids $\boldsymbol{Q}$ et $\boldsymbol{R}$ sont semi-définies positives, le problème est **convexe**. Cela garantit que toute solution locale trouvée par le solveur est l'**optimum global**.

16.3.3 Collocation directe (*Direct Collocation*)

Cette méthode est similaire à la transcription directe, toutefois les contraintes d'égalité qui imposent la dynamique sont plutôt imposées à des points de collocation, qui sont calculés en supposant que les trajectoires sont des polynômes d'un certain degré. Ces méthodes visent donc à être plus précises pour un certain niveau de discrétisation, similairement à un schéma d'intégration Runge-Kutta vs utiliser l'intégration Euler. Typiquement, les états sont approximés par des polynômes d'ordre deux et les actions par un polynôme d'ordre un.

Exercice de code
Exemple d'optimisation par collocation directe pour un pendule
https://colab.research.google.com/drive/1NcyB1aoFiM9ok7KiMIhSDx3bpfB-YWKG?usp=sharing

Exercice de code
Exemple d'optimisation par collocation directe pour un système chariot-pendule
https://colab.research.google.com/drive/1Wyc5j3oXn_JD0XJmCWfXaDlZ5lHdUHpJ?usp=sharing

Définition 16.7 Collocation Directe:

La collocation directe consiste à utiliser comme variables de décision à la fois l'état $\boldsymbol{x}$ et l'action $\boldsymbol{u}$ aux n points de la trajectoire :

$$\begin{aligned} \min_{\substack{\boldsymbol{x}_0,\dots,\boldsymbol{x}_n \\ \boldsymbol{u}_0,\dots,\boldsymbol{u}_{n-1}}} \quad & J = h(\boldsymbol{x}_n) + \sum_{i=0}^{n-1} g(\boldsymbol{x}_i, \boldsymbol{u}_i)\Delta t \\ \text{sujet à} \quad & \\ & \boldsymbol{x}_{min} \leq \boldsymbol{x}_i \leq \boldsymbol{x}_{max} \\ & \boldsymbol{u}_{min} \leq \boldsymbol{u}_i \leq \boldsymbol{u}_{max} \end{aligned}$$

avec la dynamique imposée comme contrainte d'égalité à des points de collocation :

$$\begin{aligned} & \boldsymbol{x}_0 = \boldsymbol{x}_{init} \\ & \dot{\boldsymbol{x}}_{c,i} = f(\boldsymbol{x}_{c,i}, \boldsymbol{u}_{c,i}), \quad \forall i \in \{0, \dots, n-1\} \end{aligned}$$

L'état interpolé au point milieu $\boldsymbol{x}_{c,i}$ et sa dérivée $\dot{\boldsymbol{x}}_{c,i}$ sont calculés à partir des états et des actions aux nœuds i et $i+1$. Pour la méthode Hermite-Simpson, on a :

$$\begin{aligned} \boldsymbol{u}_{c,i} &= \frac{\boldsymbol{u}_i + \boldsymbol{u}_{i+1}}{2} \\ \boldsymbol{x}_{c,i} &= \frac{1}{2}(\boldsymbol{x}_i + \boldsymbol{x}_{i+1}) + \frac{\Delta t}{8}\left(f(\boldsymbol{x}_i, \boldsymbol{u}_i) - f(\boldsymbol{x}_{i+1}, \boldsymbol{u}_{i+1})\right) \\ \dot{\boldsymbol{x}}_{c,i} &= \frac{3}{2\Delta t}(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i) - \frac{1}{4}\left(f(\boldsymbol{x}_i, \boldsymbol{u}_i) + f(\boldsymbol{x}_{i+1}, \boldsymbol{u}_{i+1})\right) \end{aligned}$$

Il est à noter que la méthode de transcription directe peut être interprétée comme une méthode de collocation directe, avec comme point de collocation $\boldsymbol{x}_{c,i} = \boldsymbol{x}_i$ et une approximation d'ordre zéro (*zero-order hold*) pour les actions et d'ordre un pour les états :

$$\begin{aligned} \boldsymbol{u}_{c,i} &= \boldsymbol{u}_i \\ \boldsymbol{x}_{c,i} &= \boldsymbol{x}_i \\ \dot{\boldsymbol{x}}_{c,i} &= \frac{1}{\Delta t}(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i) \end{aligned}$$

16.3.4 Tir multiple (*multiple-shooting*)

Définition 16.8 Transcription par Tir Multiple:

La transcription par tir multiple divise l'horizon de temps en m segments de simulation, tout en conservant une discrétisation fine de l'action sur n points ($n > m$).
Les variables de décision incluent :

1. **Actions ($n-1$) :** La séquence complète $\boldsymbol{u}_0, \ldots, \boldsymbol{u}_{n-1}$.
2. **États initiaux sur les segments ($m-1$) :** Les conditions initiales des segments intermédiaires $\boldsymbol{x}_{s,1}, \ldots, \boldsymbol{x}_{s,m-1}$ (l'état initial $\boldsymbol{x}_{s,0} = \boldsymbol{x}_0$ étant fixé).

Le problème d'optimisation se formule ainsi :

$$\min_{\substack{\boldsymbol{u}_0,\ldots,\boldsymbol{u}_{n-1} \\ \boldsymbol{x}_{s,1},\ldots,\boldsymbol{x}_{s,m-1}}} \quad J = h(\boldsymbol{x}_n) + \sum_{i=0}^{n-1} g(\boldsymbol{x}_i, \boldsymbol{u}_i)\Delta t$$

sujet à

$$\boldsymbol{x}_{s,j} + \int_{t_j}^{t_{j+1}} f(\boldsymbol{x}, \boldsymbol{u})dt - \boldsymbol{x}_{s,j+1} = \boldsymbol{0}, \quad \forall j \in \{0, \ldots, m-2\}$$

$$\boldsymbol{u}_{min} \leq \boldsymbol{u}_i \leq \boldsymbol{u}_{max}$$

Les $m-1$ **contraintes d'égalité** (contraintes de défaut) assurent la continuité de la trajectoire globale.

16.3.5 Notes sur les définitions précédentes

1. Il est possible d'inclure les temps t_i comme des variables de décision dans l'optimisation. Ce qui peut être utile si on ne connaît pas la durée prévue d'une manœuvre.
2. Pour alléger la notation dans les définitions, la dépendance au temps n'est pas explicitement notée, mais la fonction de coût, la dynamique et les contraintes peuvent tous dépendre explicitement du temps sans problème.
3. Pour alléger les définitions, le cas le plus commun et simple d'avoir seulement des contraintes min/max est noté, mais des contraintes plus compliquées peuvent être incluses.

16.4 Méthodes de recherche globale

16.4.1 RRT dans l'espace d'état (Espace Dynamique)

L'algorithme de RRT, souvent utilisé pour la planification purement géométrique, peut aussi être utilisé pour faire une recherche directement dans l'espace d'état $\mathcal{X}$, une approche souvent appelée **RRT Kinodynamique**.

Définition 16.9 RRT dans l'espace d'état:

Le RRT dynamique construit un arbre de trajectoires réalisables en propageant la dynamique du système. Contrairement au cas géométrique, les nœuds sont reliés par des segments de trajectoires qui satisfont $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$.
L'algorithme suit les étapes suivantes :

1. **Échantillonnage :** Tirer un état cible aléatoire $\boldsymbol{x}_{rand} \in \mathcal{X}_{ok}$.
2. **Sélection :** Trouver le nœud $\boldsymbol{x}_{near}$ de l'arbre le plus « proche » de $\boldsymbol{x}_{rand}$, selon une métrique de distance.
3. **Propagation avec pilotage :** Trouver une commande $\boldsymbol{u} \in \mathcal{U}_{ok}$ qui, appliquée pendant un temps Δt, mène de $\boldsymbol{x}_{near}$ vers un nouvel état $\boldsymbol{x}_{new}$ en direction de $\boldsymbol{x}_{rand}$:

$$\boldsymbol{x}_{new} = \boldsymbol{x}_{near} + \int_{t}^{t+\Delta t} f(\boldsymbol{x}, \boldsymbol{u}) dt \tag{16.20}$$

4. **Validation :** Si le segment $[\boldsymbol{x}_{near}, \boldsymbol{x}_{new}]$ respecte les contraintes d'état (obstacles) et d'action, $\boldsymbol{x}_{new}$ est ajouté à l'arbre.

La fonction de pilotage

Dans le RRT géométrique, connecter deux points se fait via une ligne droite. Dans l'espace dynamique, connecter exactement $\boldsymbol{x}_{near}$ à $\boldsymbol{x}_{rand}$ est un problème de recherche de trajectoire locale. Il y a deux familles de méthodes :

— **Échantillonnage d'action :** On teste plusieurs commandes aléatoires à partir de $\boldsymbol{x}_{near}$ et on conserve celle qui minimise la distance à $\boldsymbol{x}_{rand}$.
— **Utilisation de solveurs locaux :** Pour les systèmes linéaires, on peut résoudre analytiquement. Sinon typiquement on peut essayer de résoudre un problème d'optimisation local simplifié.

La métrique de distance

Contrairement à un problème géométrique, la distance euclidienne $||\boldsymbol{x}_{rand} - \boldsymbol{x}_{near}||$ est souvent une mauvaise mesure de proximité dans l'espace d'état. Par exemple, pour un véhicule ayant une vitesse initiale élevée, un point situé derrière lui est géométriquement proche mais "très loin" en termes de dynamique, car il nécessite une manœuvre complexe pour être atteint. La meilleure métrique serait donc d'utiliser une fonction de coût et de calculer le coût minimum possible entre ces deux points, ce qui revient à utiliser un solveur local d'optimisation de trajectoire. En pratique, il y a un compromis à faire entre avoir une mauvaise métrique de distance qui est rapide à calculer, et une bonne métrique de distance qui est très longue à calculer.

Note

Le RRT dynamique est particulièrement efficace pour trouver une première solution admissible dans des environnements complexes. Cette solution peut ensuite servir d'**estimation initiale** (*initial guess*) pour un algorithme d'optimisation (comme la collocation directe) afin de lisser la trajectoire et minimiser un coût de performance.



Exercice de code
Demo RRT + optimisation de trajectoire
`https://colab.research.google.com/drive/1yq2GHAkvO6fTF2W-tRbACDBa9_scec2k?usp=sharing`

16.5 Planéité différentielle (*Differential Flatness*)

La planéité différentielle est un outil de planification puissant qui permet d'éviter l'intégration numérique de la dynamique.

Définition 16.10 Planéité différentielle:

Un système est différentiellement plat s'il existe un ensemble de sorties "plates" $\boldsymbol{y}$ telles que tous les états $\boldsymbol{x}$ et toutes les commandes $\boldsymbol{u}$ peuvent être exprimés uniquement en fonction de $\boldsymbol{y}$ et de ses dérivées temporelles.

Pour un système plat (ex : drone quadrotor), on peut planifier une trajectoire fluide (spline) dans l'espace cartésien (Chapitre 6). La "planéité" garantit qu'il existe une commande $\boldsymbol{u}(t)$ réalisable pour suivre exactement cette courbe sans avoir à résoudre un NLP complexe.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Troisième partie

Boîte à outils mathématique

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Chapitre 17

Calcul vectoriel

Chapitre en construction !

17.1 Les Vecteurs

Un vecteur-géométrique est une quantité physique représentée par **une amplitude et une direction**. Lorsqu'une base vectorielle est spécifiée, un vecteur peut alors être représenté par des composantes scalaires selon chaque vecteur unitaire de la base. Il est important de distinguer la notion de vecteur-géométrique $\vec{v}$ et de ses composantes regroupées dans un vecteur colonne $\boldsymbol{v}$, surtout lorsque plusieurs bases sont utilisés.

La notation suivant est utilisée dans ces notes :

$$\textbf{Vecteur-géométrique :} \quad \vec{v} = v_1\,\hat{a}_1 + v_2\,\hat{a}_2 + v_3\,\hat{a}_3 \tag{17.1}$$

$$\textbf{Vecteur-colonne des composantes :} \quad \boldsymbol{v}^a = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \tag{17.2}$$

où l'exposant a est utilisé pour spécifier la base vectorielle associée aux composantes scalaires. Selon les domaines, le terme vecteur réfère soit à la quantité avec une amplitude et une direction (physique, géométrie, etc) soit à une colonne de scalaire (algèbre linéaire, programmation, etc). Comme tous ces domaines sont utilisés en robotique on fera toujours la distinction

17.2 Les bases vectorielles



Capsule vidéo
Les bases vectorielles et composantes d'un vecteur
`https://youtu.be/pZdoFz5PpKU`

Une base vectorielle correspond à trois vecteurs unitaires (en 3D) orthogonaux qui déterminent l'orientation de trois axes dans l'espace et forment une base qui permet de décrire n'importe quel vecteur $\vec{v}$ sous la forme :

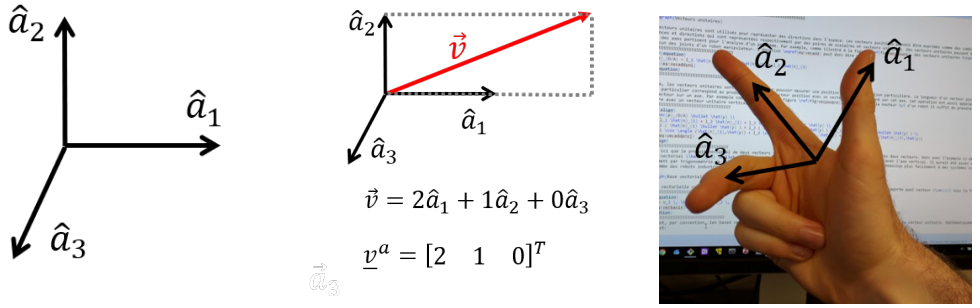
$$\vec{v} = v_1^a\,\hat{a}_1 + v_2^a\,\hat{a}_2 + v_3^a\,\hat{a}_3 \tag{17.3}$$

On appellera *base vectorielle a*, une base formée par l'ensemble des vecteurs unitaires $\{\hat{a}_1, \hat{a}_2, \hat{a}_3\}$.

Mathématiquement, les critères de dimensions unitaires et d'orthogonalités sont donnés par les équations :

$$\hat{a}_1 \cdot \hat{a}_1 = 1 \qquad \hat{a}_2 \cdot \hat{a}_2 = 1 \qquad \hat{a}_3 \cdot \hat{a}_3 = 1 \tag{17.4}$$

$$\hat{a}_1 \cdot \hat{a}_2 = 0 \qquad \hat{a}_2 \cdot \hat{a}_3 = 0 \qquad \hat{a}_3 \cdot \hat{a}_1 = 0 \tag{17.5}$$



(a) Base a (b) Exemple (c) Convention de la main droite

FIGURE 17.1 – Les bases vectorielles

Normalement, par convention, les bases vectorielles suivent la règle de main droite (voir figure 17.1c) pour spécifier la direction du troisième vecteur unitaire. La convention de la main droite permet aussi de spécifier des relations en termes de produits vectoriels entre les vecteurs unitaires :

$$\hat{a}_1 = \hat{a}_2 \times \hat{a}_3 \qquad \hat{a}_2 = \hat{a}_3 \times \hat{a}_1 \qquad \hat{a}_3 = \hat{a}_1 \times \hat{a}_2 \tag{17.6}$$

Les bases vectorielles permettent de faire des opérations et combiner des vecteurs en les exprimant avec une base commune de vecteurs unitaires. Dans la littérature, la notation $(\hat{i}, \hat{j}, \hat{k})$ ou $(\hat{x}, \hat{y}, \hat{z})$ est parfois utilisée. Dans ces notes, on utilisera des lettres (a,b,c,...) pour identifier des bases, ainsi que des indices (1,2,3) pour spécifier les trois axes (voir figure 17.1a).

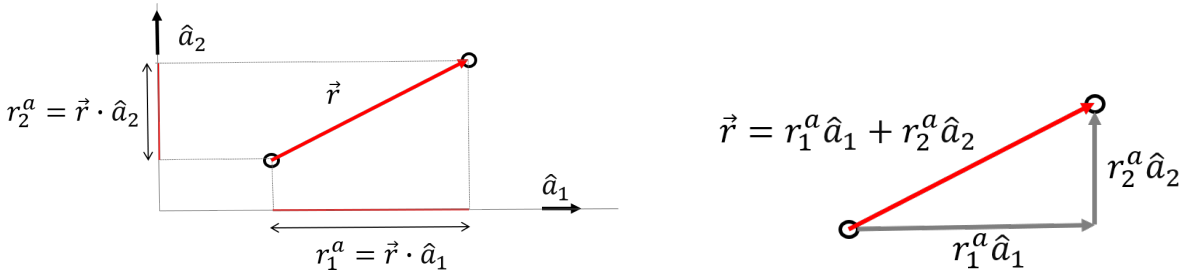
17.2.1 Relation entre un vecteur et ses composantes dans une base

Comme illustré à la figure 17.2a, il est possible de calculer les composantes d'un vecteur exprimé dans une base par un produit scalaire du vecteur géométrique avec chacun des vecteurs unitaires de la base :

$$v_i^a = \vec{v} \cdot \hat{a}_i \quad \Rightarrow \quad \boldsymbol{v}^a = \begin{bmatrix} \vec{v} \cdot \hat{a}_1 \\ \vec{v} \cdot \hat{a}_2 \\ \vec{v} \cdot \hat{a}_3 \end{bmatrix} \tag{17.7}$$

Inversement, comme illustré à la figure 17.2b, le vecteur position géométrique peut être reconstruit en effectuant la somme des composantes multipliées avec leurs vecteurs unitaires respectifs :

$$\vec{r} = \sum_i v_i^a \hat{a}_i = v_1^a \hat{a}_1 + v_2^a \hat{a}_2 + v_3^a \hat{a}_3 \tag{17.8}$$



(a) Projection sur une base vectorielle (b) Décomposition selon les directions de la base

FIGURE 17.2 – Relations entre le vecteur géométrique de position et les composantes du vecteur-colonne

17.2.2 Transfert d'une équation vectorielle vers une équation matricielle

Lorsque les composantes de plusieurs vecteurs sont exprimées avec la même base, il est possible de faire des opérations directement avec les vecteur-colonnes de composantes. Cela permet de traiter simultanément les calculs de plusieurs axes, et aussi de faire des calculs numériques efficaces en utilisant les outils de l'algèbre linéaire. Par exemple, les additions et soustractions de vecteurs géométriques peuvent être calculées directement en termes des vecteur-colonnes dans une base :

$$\vec{u} = \vec{v} + \vec{w} \quad \Rightarrow \quad \boldsymbol{u}^a = \boldsymbol{v}^a + \boldsymbol{w}^a \tag{17.9}$$

Ce transfert d'une seule équation vectorielle à une équation matricielle (équivalent à un système de trois équations scalaires) correspond à une projection de l'équation vectorielle sur chacun des axes de la base :

$$\vec{u} = \vec{v} + \vec{w} \quad \Rightarrow \quad \begin{bmatrix} (\vec{u} = \vec{v} + \vec{w}) \cdot \hat{a}_1 \\ (\vec{u} = \vec{v} + \vec{w}) \cdot \hat{a}_2 \\ (\vec{u} = \vec{v} + \vec{w}) \cdot \hat{a}_3 \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} u_1^a = v_1^a + w_1^a \\ u_2^a = v_2^a + w_2^a \\ u_3^a = v_3^a + w_3^a \end{bmatrix} \quad \Rightarrow \quad \boldsymbol{u}^a = \boldsymbol{v}^a + \boldsymbol{w}^a \tag{17.10}$$

Les équations vectorielles avec des vecteurs de position peuvent donc être substituées par des équations matricielles équivalentes avec les vecteur-colonnes, **à condition que tous les vecteur-colonnes soient exprimés dans une base commune**.

17.3 Opérations vectorielles avec les vecteur-colonnes

Les opérations vectorielles ont des équivalents en termes d'opérations matricielle avec les vecteur-colonnes qui sont très utiles pour les calculs numériques. Cette section introduit les opérations principales utiles pour la cinématique.

Capsule vidéo
Opérations vectorielles avec les composantes de vecteurs
https://youtu.be/7brvShmayAk

17.3.1 Produit scalaire / produit intérieur

Un produit scalaire peut être calculé directement par une opération matricielle avec les vecteur-colonnes appelée produit intérieur (*inner product*) :

$$\vec{u} \cdot \vec{v} = \boldsymbol{u}^T \boldsymbol{v} = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = u_1 v_1 + u_2 v_2 + u_3 v_3 \tag{17.11}$$

si $\boldsymbol{u}$ et $\boldsymbol{v}$ sont les composantes exprimées dans une base vectorielle commune, par exemple :

$$\vec{u} \cdot \vec{v} = (\boldsymbol{u}^a)^T \boldsymbol{v}^a = (\boldsymbol{u}^b)^T \boldsymbol{v}^b \tag{17.12}$$

mais attention car :

$$\vec{u} \cdot \vec{v} \neq (\boldsymbol{u}^a)^T \boldsymbol{v}^b \tag{17.13}$$

$$\vec{u} \cdot \vec{v} \neq (\boldsymbol{u}^b)^T \boldsymbol{v}^a \tag{17.14}$$

Il est à noter que l'ordre de multiplication ne change pas le résultat pour le produit scalaire :

$$\vec{u} \cdot \vec{v} = \vec{v} \cdot \vec{u} = \boldsymbol{u}^T \boldsymbol{v} = \boldsymbol{v}^T \boldsymbol{u} \tag{17.15}$$

17.3.2 Produit vectoriel

Un produit vectoriel (*cross product*) peut être calculé directement par une opération matricielle définie avec les vecteur-colonnes :

$$\vec{u} = \vec{v} \times \vec{w} \quad \Rightarrow \quad \boldsymbol{u} = \boldsymbol{v}^{\times}\boldsymbol{w} \quad \Rightarrow \quad \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} \tag{17.16}$$

L'ordre de multiplication est important pour le produit vectoriel, il change la direction du vecteur résultant :

$$\vec{u} = \vec{v} \times \vec{w} = -\vec{w} \times \vec{v} \quad \Rightarrow \quad \boldsymbol{u} = \boldsymbol{v}^{\times}\boldsymbol{w} = -\boldsymbol{w}^{\times}\boldsymbol{v} \tag{17.17}$$

Vecteurs unitaires d'une base vectorielle : Il est possible de calculer les composantes d'un vecteur unitaire qui complète une base vectorielle avec une opération de produit vectorielle, connaissant les composantes de deux vecteurs unitaires orthogonaux :

$$\hat{b}_3 = \hat{b}_1 \times \hat{b}_2 \quad \Rightarrow \quad \boldsymbol{b}_3 = \boldsymbol{b}_1^{\times}\boldsymbol{b}_2 \tag{17.18}$$

Note : Pour alléger la présentation, les exposants qui spécifient les bases vectorielles associées aux vecteur-colonnes sont parfois omis dans les équations, on considère alors que tous les vecteur-colonnes impliqués dans l'équation sont exprimés dans une base vectorielle commune.

17.3.3 Produit tensoriel / produit extérieur

Un produit tensoriel (*outer-product*) peut être calculé directement par une opération matricielle définie avec les vecteur-colonnes :

$$\vec{\vec{T}} = \vec{v}\vec{w} \quad \Rightarrow \quad T = \boldsymbol{v}\,\boldsymbol{w}^T = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \begin{bmatrix} w_1 & w_2 & w_3 \end{bmatrix} = \begin{bmatrix} v_1w_1 & v_1w_2 & v_1w_3 \\ v_2w_1 & v_2w_2 & v_2w_3 \\ v_3w_1 & v_3w_2 & v_3w_3 \end{bmatrix} \tag{17.19}$$

où le matrice T correspond aux composantes du tenseur $\vec{\vec{T}}$, avec comme correspondance si toutes les composantes sont relatives à une base a :

$$\vec{\vec{T}} = T_{11}\hat{a}_1\hat{a}_1 + T_{12}\hat{a}_1\hat{a}_2 + \ldots + T_{23}\hat{a}_2\hat{a}_3 + T_{33}\hat{a}_3\hat{a}_3 \tag{17.20}$$

17.3.4 Changement de base

Les composantes d'un vecteur géométrique $\vec{v}$ dans une base a peuvent être directement reliées aux composantes du même vecteur géométrique $\vec{v}$ dans une autre base b, par une équation matricielle qui implique les vecteur-colonnes et une matrice 3×3 appelée une *matrice de rotation*. On va noter une matrice ${}^bR^a$ lorsqu'une multiplication par la gauche de cette matrice avec un vecteur colonne $\boldsymbol{v}^a$ donne le vecteur-colonne $\boldsymbol{v}^b$:

$$\boldsymbol{v}^b = {}^bR^a\boldsymbol{v}^a \quad \Rightarrow \quad \begin{bmatrix} v_1^b \\ v_2^b \\ v_3^b \end{bmatrix} = \begin{bmatrix} {}^bR^a_{11} & {}^bR^a_{12} & {}^bR^a_{13} \\ {}^bR^a_{21} & {}^bR^a_{22} & {}^bR^a_{23} \\ {}^bR^a_{31} & {}^bR^a_{32} & {}^bR^a_{33} \end{bmatrix} \begin{bmatrix} v_1^a \\ v_2^a \\ v_3^a \end{bmatrix} \tag{17.21}$$

Les propriétés et le calcul des matrices de rotation seront traités en détails à la section 3.5.

17.3.5 Invariants

Le vecteur géométrique $\vec{v}$ est une quantité constante peu importe la base vectorielle utilisée :

$$\vec{v} = \sum_i v_i^a\hat{a}_i = \sum_i v_i^b\hat{b}_i = \sum_i v_i^c\hat{c}_i \tag{17.22}$$

mais le vecteur-colonne dépend de la base vectorielle. Les vecteur-colonnes sont en générale différents sauf dans le cas particulier de bases vectorielles coïncidentes :

$$\boldsymbol{v}^a \neq \boldsymbol{v}^b \qquad \text{sauf dans le cas particulier :} \quad \hat{a}_1 = \hat{b}_1 \ , \ \hat{a}_2 = \hat{b}_2 \text{ et } \hat{a}_3 = \hat{b}_3 \tag{17.23}$$

La longueur du vecteur-géométrique étant une constante peut importe la base vectorielle utilisée, il est possible d'établir la relation suivante entre les vecteur-colonnes utilisant différentes bases :

$$\|\vec{v}\|^2 = \vec{v} \cdot \vec{v} = (\boldsymbol{v}^a)^T \boldsymbol{v}^a = (\boldsymbol{v}^b)^T \boldsymbol{v}^b = (\boldsymbol{v}^c)^T \boldsymbol{v}^c \tag{17.24}$$

La norme des vecteur-colonnes est donc invariante par rapport au choix de la base.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

17.4 Summary of vector operations in terms of components

Vector and Tensors	Matrix, Row and Columns	Components and index
Definitions		
Vector $\vec{v}$ and Tensor $\vec{\vec{T}}$	Column $\underline{c}$, Row $\underline{r}$ and Matrix M	Scalar components x_i and x_{ij}
$\vec{v} = x_1\vec{i} + x_2\vec{j} + x_3\vec{k}$	$\underline{c} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$ $\underline{r} = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}$	$x_i \quad i \in \{1,2,3\}$
$\vec{\vec{T}} = x_{11}\vec{i}\,\vec{i} + x_{12}\vec{i}\,\vec{j} + ... + x_{33}\vec{k}\,\vec{k}$	$M = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix}$	$x_{ij} \quad i \in \{1,2,3\}\ j \in \{1,2,3\}$
Dot or inner product		
$z = \vec{x} \cdot \vec{y}$	$z = \underline{x}^T\underline{y} = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$	$z = \sum_{ij} x_i y_j \delta_{ij} = \sum_i x_i y_i$
$\vec{z} = \vec{\vec{A}} \cdot \vec{x}$	$\underline{z} = A\underline{x} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$	$z_i = \sum_{jk} a_{ij} x_k \delta_{jk} = \sum_j a_{ij} x_j$
$\vec{z} = \vec{x} \cdot \vec{\vec{A}}$	$\underline{z} = \underline{x}^T A$	$z_j = \sum_{ik} x_k a_{ij} \delta_{ki} = \sum_i x_i a_{ij}$
Dyadic or outer product		
$\vec{\vec{Z}} = \vec{x}\,\vec{y}$	$Z = \underline{x}\,\underline{y}^T = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \begin{bmatrix} y_1 & y_2 & y_3 \end{bmatrix}$	$z_{ij} = x_i\, y_j$
Cross product		
$\vec{z} = \vec{x} \times \vec{y}$	$\underline{z} = \underline{x}^{\times}\underline{y} = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$	$z_i = \sum_{jk} \epsilon_{ijk} x_j y_k$

17.4.1 Index notation special symbols

Kronecker delta

$$\delta_{ij} = \begin{cases} 1 & \text{if} \quad i = j \\ 0 & \text{if} \quad i \neq j \end{cases} \tag{17.25}$$

Levi-Civita permutation symbol

$$\epsilon_{ijk} = \begin{cases} +1 & \text{if} \quad ijk = (123), (231) \text{ or } (312) \\ -1 & \text{if} \quad ijk = (321), (213) \text{ or } (132) \\ 0 & \text{if an index is repeated} \end{cases} \tag{17.26}$$

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

17.5 Column-vector and matrix differentiation

Note that the shape of vector/matrix resulting from a multi-axis differentiation is a question of layout convention. The numerator layout convention is used here. Identities presented need to be transposed if using a denominator layout instead. There is no ambiguity with the index representation.

Scalar by Scalar

For a scalar function $y = f(x)$:

$$z = \frac{\partial y}{\partial x} \tag{17.27}$$

Vector by Scalar

For the derivation of a vector function $\underline{y} = f(x)$ where $\underline{y} \in \mathbb{R}^n$ with respect to a scalar x, by convention if the numerator $\underline{y}$ is a $n \times 1$ column vector, the result is a $n \times 1$ column vector too :

$$\underline{z} = \frac{\partial \underline{y}}{\partial x} = \begin{bmatrix} \frac{\partial y_1}{\partial x} \\ \vdots \\ \frac{\partial y_n}{\partial x} \end{bmatrix} \quad \Leftrightarrow \quad z_i = \frac{\partial y_i}{\partial x} \tag{17.28}$$

Scalar by Vector

For the gradient of a multi-inputs scalar function $y = f(\underline{x})$ where $\underline{x} \in \mathbb{R}^n$, by convention if the denominator $\underline{x}$ is a $n \times 1$ column vector, the result is a $1 \times n$ row vector :

$$\underline{z} = \frac{\partial y}{\partial \underline{x}} = \begin{bmatrix} \frac{\partial y}{\partial x_1} & \dots & \frac{\partial y}{\partial x_n} \end{bmatrix} \quad \Leftrightarrow \quad z_i = \frac{\partial y}{\partial x_i} \tag{17.29}$$

TABLE 17.1 – Scalar by a vector : Identities

Scalar $y = f(\underline{x})$ expression	Gradient Vector $\frac{\partial y}{\partial \underline{x}}$	Notes
$\underline{a}^T \underline{x} = \underline{x}^T \underline{a}$	$\underline{a}^T$	If $\underline{a}$ is not a function of $\underline{x}$
$\underline{x}^T \underline{x}$	$2\,\underline{x}^T$	
$\underline{x}^T A \underline{x}$	$\underline{x}^T (A + A^T)$	If A is not a function of $\underline{x}$
$\underline{x}^T A \underline{x}$	$2\,\underline{x}^T A$	If A is symmetric and not a function of $\underline{x}$

Vector by Vector

For the gradient of a multi-inputs vector function $\underline{y} = f(\underline{x})$ where $\underline{x} \in \mathbb{R}^n$ and $\underline{y} \in \mathbb{R}^m$, by convention if the numerator $\underline{y}$ is a $m \times 1$ column vector and the denominator a $n \times 1$ column vector, the result is a $m \times n$ matrix, often called the Jacobian matrix :

$$Z = \frac{\partial \underline{y}}{\partial \underline{\underline{x}}} = \begin{bmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \cdots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n} \end{bmatrix} \quad \Leftrightarrow \quad z_{ij} = \frac{\partial y_i}{\partial x_j} \tag{17.30}$$

TABLE 17.2 – Vector by a vector : Identities

Vector $\underline{y} = f(\underline{x})$ expression	Jacobian Matrix $\frac{\partial \underline{y}}{\partial \underline{\underline{x}}}$	Notes
$\underline{x}$	I	Identity Matrix
$A\underline{x}$	A	If A is not a function of $\underline{x}$
$\underline{x}^T A$	A^T	If A is not a function of $\underline{x}$

Matrix by Scalar

For the gradient of a matrix function $\underline{\underline{A}} = f(x)$ where x is a scalar and A is a $n \times m$ matrix, the result is also a $n \times m$ matrix :

$$Z = \frac{\partial A}{\partial x} = \begin{bmatrix} \frac{\partial a_{11}}{\partial x} & \cdots & \frac{\partial a_{1m}}{\partial x} \\ \vdots & \cdots & \vdots \\ \frac{\partial a_{n1}}{\partial x} & \cdots & \frac{\partial a_{nm}}{\partial x} \end{bmatrix} \quad \Leftrightarrow \quad z_{ij} = \frac{\partial a_{ij}}{\partial x} \tag{17.31}$$

TABLE 17.3 – Matrix by scalar : Identities

Matrix A expression	Differential Matrix $\frac{\partial A}{\partial x}$	Notes
M^{-1}	$-M^{-1}\frac{\partial M}{\partial x}M^{-1}$	

Scalar by Matrix

For the gradient of a multi-inputs scalar function $y = f(A)$ where A is an $n \times m$ matrix, by convention the result is a $m \times n$ matrix :

$$Z = \frac{\partial y}{\partial A} = \begin{bmatrix} \frac{\partial y}{\partial a_{11}} & \cdots & \frac{\partial y}{\partial a_{1n}} \\ \vdots & \cdots & \vdots \\ \frac{\partial y}{\partial a_{m1}} & \cdots & \frac{\partial y}{\partial a_{mn}} \end{bmatrix} \quad \Leftrightarrow \quad z_{ji} = \frac{\partial y}{\partial a_{ij}} \tag{17.32}$$

TABLE 17.4 – Scalar by Matrix : Identities

Scalar y expression	Differential Matrix $\frac{\partial y}{\partial A}$	Notes
$\underline{a}^T A \underline{b}$	$\underline{b}\,\underline{a}^T$	$\underline{a}$ and $\underline{b}$ are not function of A
$\underline{a}^T A^T A \underline{b}$	$\underline{a}\,\underline{b}^T A^T + \underline{b}\,\underline{a}^T A^T$	$\underline{a}$ and $\underline{b}$ are not function of A

Matrix by Vector

For the gradient of a matrix function $\underline{A} = f(\boldsymbol{x})$ where $\boldsymbol{x}$ is a $p \times 1$ column-vector and A is a $n \times m$ matrix, the result is a $n \times m \times p$ tensor :

$$T = \frac{\partial A}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial a_{11}}{\partial \boldsymbol{x}} & \cdots & \frac{\partial a_{1m}}{\partial \boldsymbol{x}} \\ \vdots & \cdots & \vdots \\ \frac{\partial a_{n1}}{\partial \boldsymbol{x}} & \cdots & \frac{\partial a_{nm}}{\partial \boldsymbol{x}} \end{bmatrix} \quad \Leftrightarrow \quad z_{ijk} = \frac{\partial a_{ij}}{\partial x_k} \tag{17.33}$$

Exemple 17.1 Gradient of Quadratic Function:

Finding the gradient $\frac{\partial f}{\partial \underline{x}}$ of the quadratic function $f = \underline{x}^T A \underline{x}$. First converting to index notations :

$$f = \sum_{ij} x_i A_{ij} x_j \tag{17.34}$$

Then differentiating per x_k components :

$$\frac{\partial f}{\partial x_k} = \sum_{ij} \frac{\partial}{\partial x_k} (x_i A_{ij} x_j) \tag{17.35}$$

$$\frac{\partial f}{\partial x_k} = \sum_{ij} \left(\underbrace{\frac{\partial x_i}{\partial x_k}}_{\delta_{ik}} A_{ij} x_j + x_i \underbrace{\frac{\partial A_{ij}}{\partial x_k}}_{0} x_j + x_i A_{ij} \underbrace{\frac{\partial x_j}{\partial x_k}}_{\delta_{jk}} \right) \tag{17.36}$$

$$\frac{\partial f}{\partial x_k} = \sum_{ij} \delta_{ik} A_{ij} x_j + \sum_{ij} \delta_{jk} x_i A_{ij} \tag{17.37}$$

$$\frac{\partial f}{\partial x_k} = \sum_{j} A_{kj} x_j + \sum_{i} x_i A_{ik} \tag{17.38}$$

$$\frac{\partial f}{\partial x_k} = \sum_{i} (A_{ki} x_i + x_i A_{ik}) \tag{17.39}$$

$$\tag{17.40}$$

Which correspond in matrix form, with the numerator layout convention, to :

$$\frac{\partial f}{\partial \underline{x}} = \underline{x}^T (A + A^T) \tag{17.41}$$

Exemple 17.2 Least square solution derivation in vector form:

Minimizing the error norm $\|\underline{e}\|^2$ of the estimation problem $\underline{e} = A\underline{\hat{x}} - \underline{b}$ by selecting the parameters $\underline{\hat{x}}$. If the gradient of the error norm with respect to the parameter vector is set to zero :

$$\underline{0} = \frac{\partial \|\underline{e}\|^2}{\partial \underline{\hat{x}}} = \frac{\partial (\underline{e}^T \underline{e})}{\partial \underline{\hat{x}}} = 2\,\underline{e}^T \frac{\partial \underline{e}}{\partial \underline{\hat{x}}} = 2\,(A\underline{\hat{x}} - \underline{b})^T A \tag{17.42}$$

For which the solution is

$$\underline{\hat{x}} = \left[A^T A\right]^{-1} A^T \underline{b} \tag{17.43}$$

Chapitre 18

Algèbre linéaire

L'algèbre linéaire, c'est une collection d'outils mathématiques pour traiter des opérations multi-variables en parallèle, avec un niveau d'abstraction supérieur. Par exemple, l'opération $\boldsymbol{y} = A\boldsymbol{x}$ peut être interprétée comme une simple opération d'une matrice par un vecteur-colonne, par un humain qui maitrise ces concepts, ou un GPU qui a des processeur spécialisés pour effectuer des opérations matricielles ; et ce même si on parle d'une matrice de 1000x1000 et d'un vecteur-colonne de 1000 variables. Sans ce niveau d'abstraction, l'humain ou l'ordinateur qui travaillerait avec la même opération aurait à réfléchir à milles équations qui consiste en milles multiplications et milles additions. L'algèbre linéaire permet de traiter ces opérations en une seule instruction.

Calcul sur un ordinateur

À bas niveau, un ordinateur effectue des opérations de sommation sur les divers composantes pour faire une opération matricielle. Toutefois, plusieurs langages de programmation, comme *Python* (avec *Numpy* et *PyTorch*) et *Matlab*, donnent accès à des fonctions pour demander des opérations matricielles à haut-niveau, avec un code source très optimisé en terme d'utilisation de la mémoire de l'ordinateur. Il est donc beaucoup plus rapide d'utiliser de telles fonctions que de coder ces opérations de sommation manuellement. De plus, les cartes graphiques des ordinateurs ont des circuits spécialement conçu pour faire des opérations spécialisées typiques d'algèbre linéaire. Pour faire des calculs avec des très grandes matrices et long vecteurs, il peut y avoir des gains de temps très significatifs en utilisant les capacités des cartes graphiques.



Exercice de code
Démonstration de vitesse de calcul
`https://colab.research.google.com/drive/1YFA7BPbYki6FH78xEY8wf9cxrCaDYvcZ?usp=sharing`



Exercice de code
Tutoriel Numpy
`https://colab.research.google.com/drive/18GMWAG_1C6TFbjEt9FLiSKTDl-CT1WIy?usp=sharing`

18.1 Opérations matricielles de base

Toutes les opérations matricielles peuvent être écrite comme une équation avec des sommations et des indices.

18.1.1 Définition de l'opération $\boldsymbol{y} = A\boldsymbol{x}$

L'opération $\boldsymbol{y} = A\boldsymbol{x}$ est un outil mathématique pour faire m opérations linéaires sur n variables x_i dans le vecteur-colonne $\boldsymbol{x}$.

Propriété 18.1: Opérations $\boldsymbol{y} = A\boldsymbol{x}$ en indices

L'opération matricielle $\boldsymbol{y} = A\boldsymbol{x}$ peut être écrite comme :

$$\boldsymbol{y} = A\boldsymbol{x} \quad \Leftrightarrow \quad y_j = \sum_{i=1}^{m} A_{ji}x_i = a_{1j}x_1 + a_{2j}x_2 + \ldots + a_{mj}x_m \tag{18.1}$$

18.1.2 Additions de matrices $C = A + B$

Définition 18.1 Addition de matrices:

Une matrice additionnée à une autre matrice est équivalent en termes de composantes à additionner les termes individuellements :

$$C = A + B \quad \Leftrightarrow \quad C_{ij} = A_{ij} + B_{ij} \tag{18.2}$$

18.1.3 Matrice multipliée par un scalaire $B = cA$

Définition 18.2 Multiplication par un scalaire:

Une matrice multipliée par un scalaire $c \in \mathbb{R}$ est équivalent en termes de composantes à multiplier chaque composante par le scalaire :

$$B = cA \quad \Leftrightarrow \quad B_{ij} = cA_{ij} \tag{18.3}$$

18.1.4 Multiplication de matrices $C = AB$

Définition 18.3 $C = AB$ en termes d'indices et de composantes:

Une matrice multipliée par une autre matrice est équivalent en termes de sommation de composante à l'opération :

$$A = BC \quad \Leftrightarrow \quad A_{ik} = \sum_{j} B_{ij}C_{jk} \tag{18.4}$$

18.1.5 L'opération matricielle A^T

Définition 18.4 Transposée d'une matrice:

Prendre la transposée d'une matrice est une opération qui consiste à inverser les indices i et j de la matrice, ce qu'on peut aussi voir comme une réflexion selon la diagonale de la matrice :

$$A^T_{ij} = A_{ji} \tag{18.5}$$

18.1.6 Vecteur-ligne multiplié par une matrice $\boldsymbol{y}^T = \boldsymbol{x}^T A$

Définition 18.5 Multiplication vecteur-ligne par matrice:

L'opération d'un vecteur-ligne multiplié par une matrice donne un nouveau vecteur-ligne, en termes d'indices, cela correspond à :

$$\boldsymbol{y}^T = \boldsymbol{x}^T A \quad \Leftrightarrow \quad y_j = \sum_{i=1}^{m} x_i A_{ij} \tag{18.6}$$

18.1.7 Forme quadratique $f = \boldsymbol{x}^T A \boldsymbol{x}$

Définition 18.6 Forme quadratique:

Une forme quadratique est une opération qui transforme un vecteur et une matrice carrée en un scalaire :

$$f = \boldsymbol{x}^T A \boldsymbol{x} = \sum_i \sum_j A_{ij} x_i x_j \tag{18.7}$$

18.1.8 Propriétés des opérations matricielles

Propriété 18.2: Loi commutative

L'addition de matrices est commutative, c'est-à-dire que l'ordre des termes n'a pas d'importance :

$$A + B = B + A \tag{18.8}$$

A l'opposé, la multiplication de deux matrices n'est généralement pas commutative :

$$\text{Attention!!} \qquad AB \neq BA \qquad \text{en général} \tag{18.9}$$

Propriété 18.3: Loi associative

L'addition et la multiplication de matrices sont associatives, on peut grouper les opérations dans l'ordre que l'on veut :

$$(A + B) + C = A + (B + C) \tag{18.10}$$
$$A(BC) = (AB)C \tag{18.11}$$

Propriété 18.4: Loi distributive

La multiplication de matrices est distributive sur l'addition (à gauche et à droite) :

$$C(A + B) = CA + CB \tag{18.12}$$
$$(A + B)C = AC + BC \tag{18.13}$$

De même pour la multiplication par un scalaire :

$$c(A + B) = cA + cB \tag{18.14}$$

Propriété 18.5: Lois pour la transposition et l'inversion

La transposée (et l'inverse) d'une multiplication de matrice affecte l'ordre des matrices :

$$(AB)^T = B^T A^T \tag{18.15}$$
$$(AB)^{-1} = B^{-1} A^{-1} \tag{18.16}$$

La transposée et l'inverse sont commutatif :

$$(A^{-1})^T = (A^T)^{-1} \tag{18.17}$$

Finalement, l'addition :

$$(A+B)^T = A^T + B^T \tag{18.18}$$

18.2 Classes de matrices particulières

Certaines classes de matrices possèdent des propriétés spécifiques.

Définition 18.7 Matrice Identité:

La matrice identité, notée I, est une matrice carrée avec des 1 sur la diagonale principale et des 0 partout ailleurs. Elle agit comme l'élément neutre de la multiplication matricielle :

$$I = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \quad \Longrightarrow \quad AI = IA = A \tag{18.19}$$

Définition 18.8 Matrice Diagonale:

Une matrice diagonale D est une matrice carrée dont tous les éléments hors de la diagonale principale sont nuls ($D_{ij} = 0$ pour $i \neq j$).

$$D = \text{diag}(d_1, d_2, \ldots, d_n) = \begin{bmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_n \end{bmatrix} \tag{18.20}$$

Ses propriétés incluent une inversion facile (voir section 18.4) ($D^{-1} = \text{diag}(1/d_1, \ldots, 1/d_n)$) et un calcul de puissance aisé (voir propriété 18.88) ($D^k = \text{diag}(d_1^k, \ldots, d_n^k)$).

Définition 18.9 Matrice Symétrique:

Une matrice A est dite symétrique si elle est égale à sa transposée :

$$A = A^T \quad \Longleftrightarrow \quad A_{ij} = A_{ji} \tag{18.21}$$

Les matrices symétriques ont la propriété remarquable d'avoir toutes leurs valeurs propres réelles (voir section 18.9) et des vecteurs propres orthogonaux.

Définition 18.10 Matrice Anti-symétrique (Skew-symmetric):

Une matrice A est dite anti-symétrique si sa transposée est égale à son opposé :

$$A^T = -A \quad \Longleftrightarrow \quad A_{ij} = -A_{ji} \tag{18.22}$$

Cela implique nécessairement que les éléments de la diagonale sont nuls. En robotique, ces matrices sont omniprésentes pour représenter le produit vectoriel (voir section 17.3.2) sous forme matricielle.

Définition 18.11 Matrice Orthogonale:

Une matrice carrée Q est orthogonale si ses colonnes (et lignes) forment une base orthonormée. Cela se traduit par :

$$Q^T Q = QQ^T = I \quad \Longrightarrow \quad Q^{-1} = Q^T \text{ (voir section 18.4)} \tag{18.23}$$

Les matrices de rotation sont des matrices orthogonales (avec déterminant $+1$, voir section 18.5.1). Elles préservent la norme des vecteurs ($\|Q\boldsymbol{x}\| = \|\boldsymbol{x}\|$).

Définition 18.12 Matrice Définie Positive:

Une matrice symétrique A est dite définie positive si pour tout vecteur non nul $\boldsymbol{x} \neq \mathbf{0}$, le scalaire suivant est strictement positif :

$$\boldsymbol{x}^T A \boldsymbol{x} > 0 \tag{18.24}$$

Si l'inégalité est large (≥ 0), elle est dite semi-définie positive.

18.3 Interprétations de l'opération $\boldsymbol{y} = A\boldsymbol{x}$

18.3.1 Système d'équations linéaires

L'opération matricielle $\boldsymbol{y} = A\boldsymbol{x}$ peut d'abord être vue comme un système de m équations linéaires.

Définition 18.13 Système d'équations:

De façon générale, l'opération matricielle $\boldsymbol{y} = A\boldsymbol{x}$ est un système d'équations linéaires. Une matrice d'un tel système d'équation a une largeur n, correspondant au nombre de variable dans les équations, et une hauteur m, correspondant au nombre d'équations.

$$\begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \tag{18.25}$$

$$m: \quad \text{nombre de ranges} \quad = \quad \text{nombre d'équations / sorties} \tag{18.26}$$

$$n: \quad \text{nombre de colonnes} \quad = \quad \text{nombre de variables / entrées} \tag{18.27}$$

Par exemple, deux fonctions linéaires, f_1 et f_2, de deux variables (entrées), x_1 et x_2, peuvent s'écrire :

$$y_1 = f_1(x_1, x_2) = a_{11}\, x_1 + a_{12}\, x_2 \tag{18.28}$$

$$y_2 = f_2(x_1, x_2) = a_{21}\, x_1 + a_{22}\, x_2 \tag{18.29}$$

ou y_1 et y_2 sont les variables résultats (sorties) des fonctions, et a_{ii} les paramètres internes des fonctions. Ces équations, avec la notation matricielle prennent la forme :

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \leftrightarrow \quad \boldsymbol{y} = A\boldsymbol{x} \tag{18.30}$$

ou $\boldsymbol{y}$ et $\boldsymbol{x}$ sont des vecteur-colonnes de dimension 2 et A est une matrice 2x2.

18.3.2 Combinaison de vecteur-colonnes

La multiplication d'une matrice par un vecteur-colonne aussi être vue comme une combinaison linéaire des vecteur-colonnes de la matrice.

Propriété 18.6: Combinaison linéaire de vecteur-colonnes

De façon générale, le vecteur-colonne $\boldsymbol{y}$ résultant d'une multiplication d'une matrice A avec un vecteur-colonne $\boldsymbol{x}$, peut-être exprimée comme une combinaison linéaire des colonnes de la matrice A :

$$\begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} \underbrace{\begin{bmatrix} a_{11} \\ \vdots \\ a_{1m} \end{bmatrix}}_{\boldsymbol{c}_1} & \dots & \underbrace{\begin{bmatrix} a_{i1} \\ \vdots \\ a_{im} \end{bmatrix}}_{\boldsymbol{c}_i} & \dots & \underbrace{\begin{bmatrix} a_{n1} \\ \vdots \\ a_{nm} \end{bmatrix}}_{\boldsymbol{c}_n} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \tag{18.31}$$

$$\boldsymbol{y} = \sum_{i=1}^{n} \boldsymbol{c}_i x_i \tag{18.32}$$

Exemple 18.1 Combinaison linéaire 2x2:

L'équation (18.30), correspondant a la multiplication d'un vecteur-colonne de dimension 2 par une matrice 2×2 peut être réécrite comme l'addition pondérée des deux colonnes de la matrice :

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad \leftrightarrow \quad \boldsymbol{y} = \underbrace{\begin{bmatrix} a_{11} \\ a_{21} \end{bmatrix}}_{\boldsymbol{c}_1} x_1 + \underbrace{\begin{bmatrix} a_{12} \\ a_{22} \end{bmatrix}}_{\boldsymbol{c}_2} x_2 \tag{18.33}$$

Cette interprétation est très importante pour comprendre toutes les propriétés importantes et utiles des matrices, qui seront détaillées dans les prochaines sections. Graphiquement, tel illustré à la figure 18.1, la multiplication d'une matrice peux être représenté par une addition pondérée (combinaison linéaire) des vecteurs correspondants aux colonnes de la matrice, dans l'espace sortie $\boldsymbol{y} \in \mathbb{R}^m$.

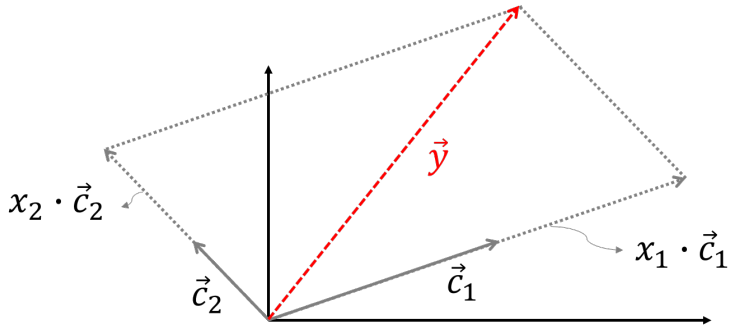


FIGURE 18.1 – La multiplication matricielle comme une combinaison lineaire de vecteurs

18.3.3 Produits scalaires avec les rangées de la matrice

Une autre interprétation de l'opération matricielle $\boldsymbol{y} = A\boldsymbol{x}$ est que chaque élément y_j du résultat correspond au produit scalaire du vecteur-rangé j avec le vecteur colonne $\boldsymbol{x}$.

Propriété 18.7: Produits scalaires avec les rangées

De façon générale, chaque élément du y_j vecteur-colonne $\boldsymbol{y}$ résultant d'une multiplication d'une matrice A avec un vecteur-colonne $\boldsymbol{x}$, peut-être exprimé comme le résultat du produit scalaire entre le vecteur-rangé $\boldsymbol{r}_j$ et le vecteur-colonne $\boldsymbol{x}$:

$$\begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} \underbrace{\begin{bmatrix} a_{11} & \dots & a_{1n} \end{bmatrix}}_{\boldsymbol{r}_1} \\ \vdots \\ \underbrace{\begin{bmatrix} a_{m1} & \dots & a_{mn} \end{bmatrix}}_{\boldsymbol{r}_m} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \tag{18.34}$$

$$y_j = \boldsymbol{r}_j \, \boldsymbol{x} = \sum_i a_{ij} x_i \tag{18.35}$$

Exemple 18.2 Produit scalaire 2x2:

L'équation (18.30) peut prendre la forme :

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \underbrace{\begin{bmatrix} a_{11} & a_{12} \end{bmatrix}}_{\boldsymbol{r}_1} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ \underbrace{\begin{bmatrix} a_{21} & a_{22} \end{bmatrix}}_{\boldsymbol{r}_2} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \vec{r}_1 \cdot \vec{x} \\ \vec{r}_2 \cdot \vec{x} \end{bmatrix} \tag{18.36}$$

18.4 L'opération matricielle A^{-1}

Définition 18.14 Inverse d'une matrice:

L'inverse d'une matrice carrée A, notée A^{-1}, est la matrice qui, lorsqu'elle est multipliée par A, donne la matrice identité I :

$$AA^{-1} = A^{-1}A = I \tag{18.37}$$

L'inverse n'existe que si le déterminant de la matrice est non-nul ($\det(A) \neq 0$). Une matrice qui possède un inverse est dite *inversible*, sinon elle est dite *singulière*.

L'opération d'inversion est cruciale pour résoudre des systèmes d'équations linéaires de la forme $A\boldsymbol{x} = \boldsymbol{y}$ lorsque A est carrée et inversible, car la solution est donnée par $\boldsymbol{x} = A^{-1}\boldsymbol{y}$.

18.4.1 Formule pour une matrice 2x2

Propriété 18.8: Inverse d'une matrice 2x2

Pour une matrice 2×2, l'inverse peut être calculé analytiquement avec la formule suivante :

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \underbrace{\frac{1}{ad - bc}}_{\text{Déterminant}} \underbrace{\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}}_{\text{Adjoint}} \tag{18.38}$$

18.4.2 Pseudo-Inverse

Définition 18.15 Pseudo-Inverse Gauche:

Lorsque la matrice A est rectangulaire avec $m \geq n$ (plus ou autant d'équations que d'inconnues) et de plein rang colonne, elle ne possède pas d'inverse classique (sauf si $m = n$), mais on peut définir un inverse à gauche qui donne la matrice identité lorsque multiplié à gauche :

$$A^{\#} = (A^T A)^{-1} A^T \quad \text{tel que} \quad A^{\#} A = I \tag{18.39}$$

Définition 18.16 Pseudo-Inverse Droite:

Lorsque la matrice A est rectangulaire avec $n \geq m$ (plus ou autant d'inconnues que d'équations) et de plein rang ligne, on peut définir un inverse à droite qui donne la matrice identité lorsque multiplié à droite :

$$A^{+} = A^T (A A^T)^{-1} \quad \text{tel que} \quad A A^{+} = I \tag{18.40}$$

Le concept de décomposition par valeurs singulières (SVD) permet de généraliser les deux concepts de pseudo-inverse, voir **??**.

18.5 Propriétés d'une matrice

18.5.1 Déterminant

Définition 18.17 Déterminant:

Le déterminant d'une matrice carrée A, noté $\det(A)$ ou $|A|$, est une valeur scalaire qui caractérise certaines propriétés de la matrice. Une matrice est inversible si et seulement si son déterminant est non nul :

$$\det(A) \neq 0 \iff A \text{ est inversible} \tag{18.41}$$

Géométriquement, la valeur absolue du déterminant correspond au facteur d'échelle du volume (ou de l'aire en 2D) lors de la transformation linéaire définie par la matrice.

Formules analytiques

Propriété 18.9: Déterminant 2x2

Pour une matrice 2×2 :

$$\det\left(\begin{bmatrix} a & b \\ c & d \end{bmatrix}\right) = ad - bc \tag{18.42}$$

Propriété 18.10: Déterminant 3x3

Pour une matrice 3×3 :

$$\det\left(\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}\right) = aei + bfg + cdh - ceg - bdi - afh \tag{18.43}$$

18.5.2 Trace

Définition 18.18 Trace:

La trace d'une matrice carrée A de dimension $n \times n$ est définie comme la somme des éléments de sa diagonale principale :

$$\text{tr}(A) = \sum_{i=1}^{n} a_{ii} = a_{11} + a_{22} + \cdots + a_{nn} \tag{18.44}$$

18.5.3 Indépendance linéaire et Base

Définition 18.19 Base:

Un ensemble $\{\boldsymbol{v}_1\ \boldsymbol{v}_2\ \ldots\ \boldsymbol{v}_n\}$ de vecteurs de dimensions m dans un espace vectoriel appartenant à $\mathbb{R}^m$, forme une base de cet espace si : **1)** les vecteurs $\boldsymbol{v}_1\ \boldsymbol{v}_2\ \ldots\ \boldsymbol{v}_n$ sont linéairement indépendants et **2)** tout vecteur $\boldsymbol{w}$ de cet espace peut être exprimé comme une combinaison linéaire de ces vecteurs :

$$\boldsymbol{w} = x_1\boldsymbol{v}_1 + x_2\boldsymbol{v}_2 + ... + x_n\boldsymbol{v}_n \tag{18.45}$$

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Exemple 18.3 Base pour un plan:

Par exemple, dans un espace tri-dimensionnel qui correspond à des coordonnées $\begin{bmatrix} x \\ y \\ z \end{bmatrix}$, les vecteur-colonnes $\left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \right\}$ forment une base pour le sous-espace qui correspond au plan $z = 0$.

Un vecteur est dit linéairement dépendant d'un autre vecteur (ou un ensemble de vecteurs), lorsque celui-ci peut être exprimé comme une combinaison pondéré des autres.

Définition 18.20 Indépendance linéaire:

Un ensemble de vecteurs $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n\}$ est **linéairement dépendant** s'il existe des coefficients non nuls x_i tels que la somme pondérée des vecteurs est nulle :

$$\mathbf{0} = x_1\boldsymbol{v}_1 + x_2\boldsymbol{v}_2 + ... + x_n\boldsymbol{v}_n \tag{18.46}$$

Inversement, un ensemble de vecteurs $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n\}$ est **linéairement indépendant** si :

$$\mathbf{0} = x_1\boldsymbol{v}_1 + x_2\boldsymbol{v}_2 + ... + x_n\boldsymbol{v}_n \quad \text{implique que :} \quad 0 = x_1 = x_2 = ... = x_n \tag{18.47}$$

18.5.4 Rang d'une matrice

Définition 18.21 Rang d'une matrice:

Le rang r d'une matrice correspond au nombre de lignes (ou de colonnes) linéairement indépendantes. Il est égal à la dimension de l'espace colonne et de l'espace rangée. Ce nombre correspond aussi au nombre de pivots obtenus lors d'une élimination de Gauss :

$$r = \text{rank}(A) = \dim(\text{col}(A)) = \dim(\text{row}(A)) = \#\,\text{pivots} \tag{18.48}$$

Propriété 18.11: Dimensions du rang

Le rang d'une matrice est toujours inférieur ou égale aux dimensions m (nombre de lignes) et n (nombre de colonnes) :

$$r \leq m \quad \text{et} \quad r \leq n \implies r \leq \min(m, n) \tag{18.49}$$

Propriété 18.12: Matrice de plein rang

Dans le cas d'une matrice carrée ($m = n$), la matrice est dite de *plein rang* si $r = m = n$. Cela équivaut à dire que son déterminant est non-nul ($\det(A) \neq 0$, voir section 18.5.1) et qu'elle est inversible.

18.6 Les quatre espaces fondamentaux

La figure 18.2 résume les quatre espaces fondamentaux d'une matrice, deux espaces caractérisent les entrées $\boldsymbol{x}$ et deux espaces caractérisent les sorties $\boldsymbol{y}$. La figure 18.3 illustre comment les dimensions de ces espaces sont reliés au rang et dimensions d'une matrice.

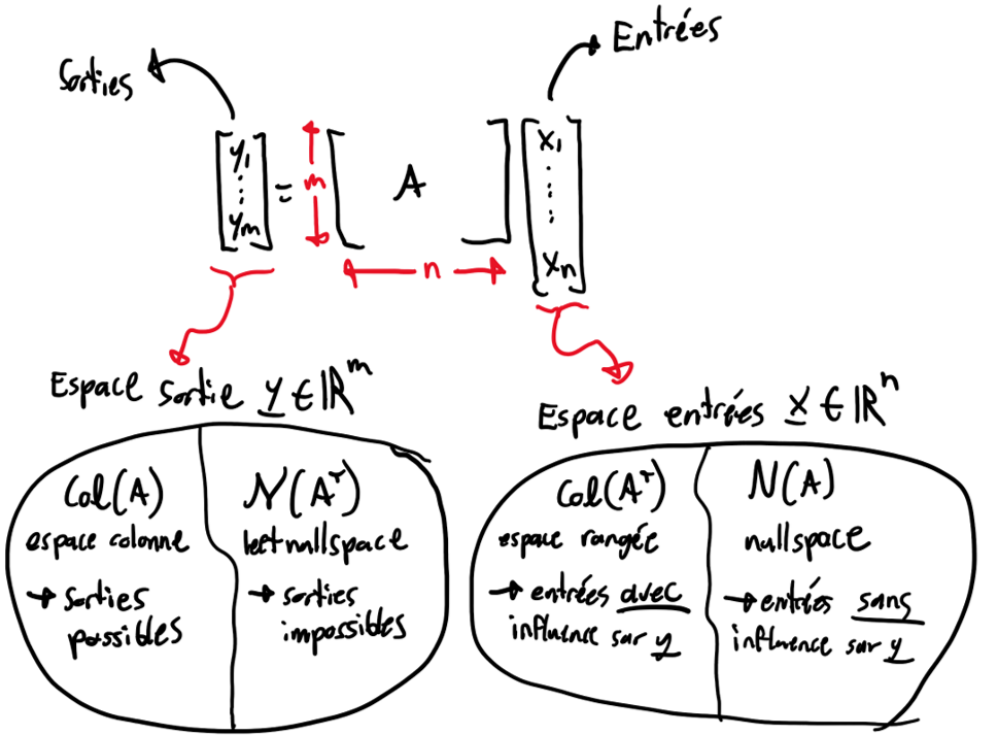


FIGURE 18.2 – Quatres espaces fondamentaux d'une matrice

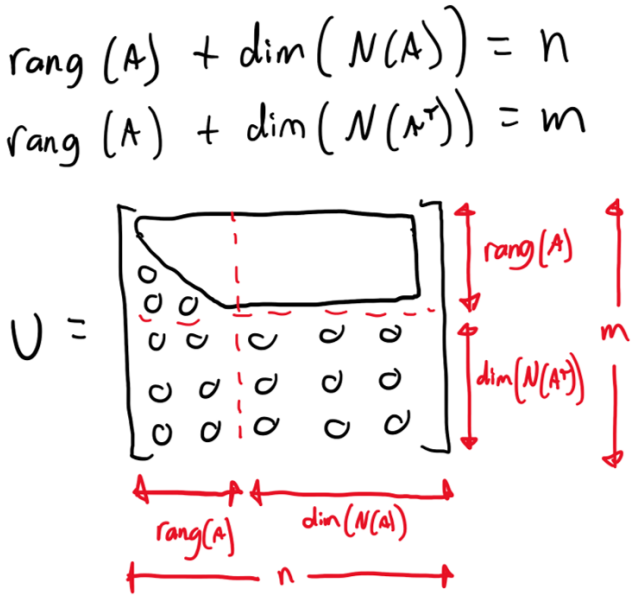


FIGURE 18.3 – Dimensions des quatre espaces fondamentaux d'une matrice

18.6.1 Espace colonne

L'espace colonne est un sous-espace de la sortie $\boldsymbol{y}$ pour l'opération $\boldsymbol{y} = A\boldsymbol{x}$. L'espace colonne d'une matrice A représente l'ensemble de toutes les valeurs possible de la sortie $\boldsymbol{y}$ qui résulte de l'opération $\boldsymbol{y} = A\boldsymbol{x}$. Avec l'interprétation présentée à la section 18.3.2, il est possible de voir que la sortie $\boldsymbol{y}$ a comme domaine toutes les combinaisons linéaires possibles des vecteur-colonnes de la matrice. Comme illustré à la Figure 18.4, pour une matrice qui aurait deux colonnes, si on interprète ces deux colonnes comme les composantes de vecteurs spatiaux, alors le domaine possible pour la sortie $\boldsymbol{y}$ va être le plan formé par ces deux vecteurs.

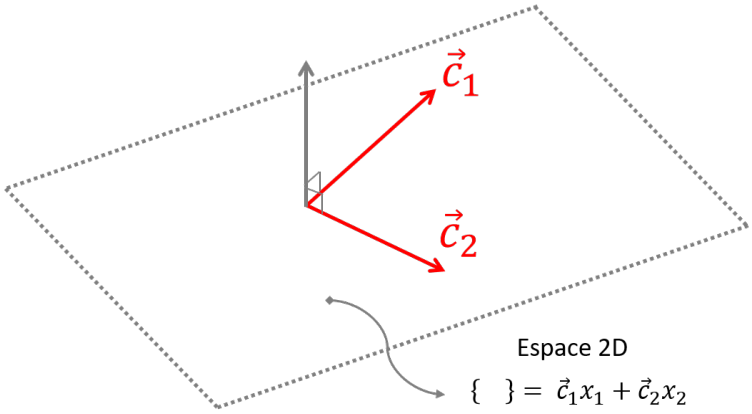


FIGURE 18.4 – Visualisation graphique d'un espace colonne de dimension 2

Définition 18.22 Espace Colonne:

L'espace colonne d'une matrice A sera noté $col(A)$ et est défini comme suit :

$$col(A) = \{A\boldsymbol{x} \mid \boldsymbol{x} \in \mathbb{R}^n\} \tag{18.50}$$

L'espace colonne est aussi appelé l'image de la matrice.

Notation mathématique pour les ensembles :

Les crochets {} représentent un ensemble, à l'intérieur on note la liste des éléments de l'ensemble ou bien la définition. La barre verticale | signifie *tel que* et le symbole ∈ signifie *appartient à*. L'équation (18.50) peut donc se lire *l'espace colonne de la matrice A est égale à l'ensemble des résultats possible de l'opération $A\boldsymbol{x}$ tel que $\boldsymbol{x}$ appartient à l'espace vectoriel de dimension n.*

Lorsque la matrice A est inversible, i.e. de rang plein, l'espace colonne correspond à $\mathbb{R}^m$, ou m est le nombre de variables de sortie. Par exemple, pour une matrice 3×3 avec une sortie $\boldsymbol{y}$ qui représente une position spatiale (x,y,z), si les colonnes ne sont pas co-planaire, alors la sortie $\boldsymbol{y}$ va être une valeur arbitraire en 3D. On va donc dire que la matrice est inversible, car ici il est toujours possible de trouver une combinaison des colonnes pour obtenir un point (x,y,z) arbitraire. Toutefois, si les vecteurs-colonnes de la matrice sont co-linéaire (donc ils forment un plan), alors l'espace colonne est restreint à ce plan. La sortie $\boldsymbol{y}$ peut seulement prendre des valeurs qui correspondent à ce plan. Dans cette situation la matrice est dite non-inversible.

18.6.2 Espace nul gauche

L'espace nul gauche correspond au domaine de la sortie $\boldsymbol{y}$ qui n'est pas dans l'espace colonne. Une condition qui revient à dire que l'espace nul gauche correspond à l'ensemble des vecteurs $\boldsymbol{y}$ qui sont perpendiculaires à toutes les colonnes de la matrice A.

Définition 18.23 Espace Nul Gauche:

L'espace nul gauche d'une matrice A, noté $N(A^T)$, correspond a l'ensemble de tous les vecteur-colonnes $\boldsymbol{y}$ pour lesquels la multiplication avec la matrice A^T donne un vecteur-colonne nulle $\mathbf{0}$:

$$N(A^T) = \left\{ \boldsymbol{y} \in \mathbb{R}^m \mid A^T \boldsymbol{y} = \mathbf{0} \right\} \tag{18.51}$$

Ce domaine correspond donc aux vecteurs $\boldsymbol{y}$ qui sont impossible à obtenir avec l'opération $\boldsymbol{y} = A\boldsymbol{x}$, peut importe le vecteur d'entrée $\boldsymbol{x}$.

L'espace des vecteur-colonnes sorties $\boldsymbol{y}$ est formé par r (rang) bases qui forment l'espace colonne et $m-r$ bases qui forment l'espace nul gauche.

Propriété 18.13: Dimensions de l'espace nul gauche

La dimension de l'espace nul gauche est liée au nombre de lignes m et au rang r de la matrice :

$$\dim(N(A^T)) = m - r \tag{18.52}$$

Ainsi, la somme des dimensions de l'espace colonne et de l'espace nul gauche est égale au nombre total de lignes :

$$m = \dim(col(A)) + \dim(N(A^T)) \tag{18.53}$$

Les vecteur-colonnes $\boldsymbol{y}$ appartenant à l'espace colonne ont une solution $\boldsymbol{x}$ possible, tandis que ceux appartenant à l'espace nul gauche n'ont aucune solution exacte $\boldsymbol{x}$ possible. Une matrice A va avoir un espace nul gauche seulement lorsqu'elle a une rang déficient ($r < m$). La présence d'un espace nul gauche indique un système sur-contraint, i.e. un surplus d'équations.

Exemple 18.4 Espace-nul gauche de la matrice jacobienne d'un robot manipulateur:

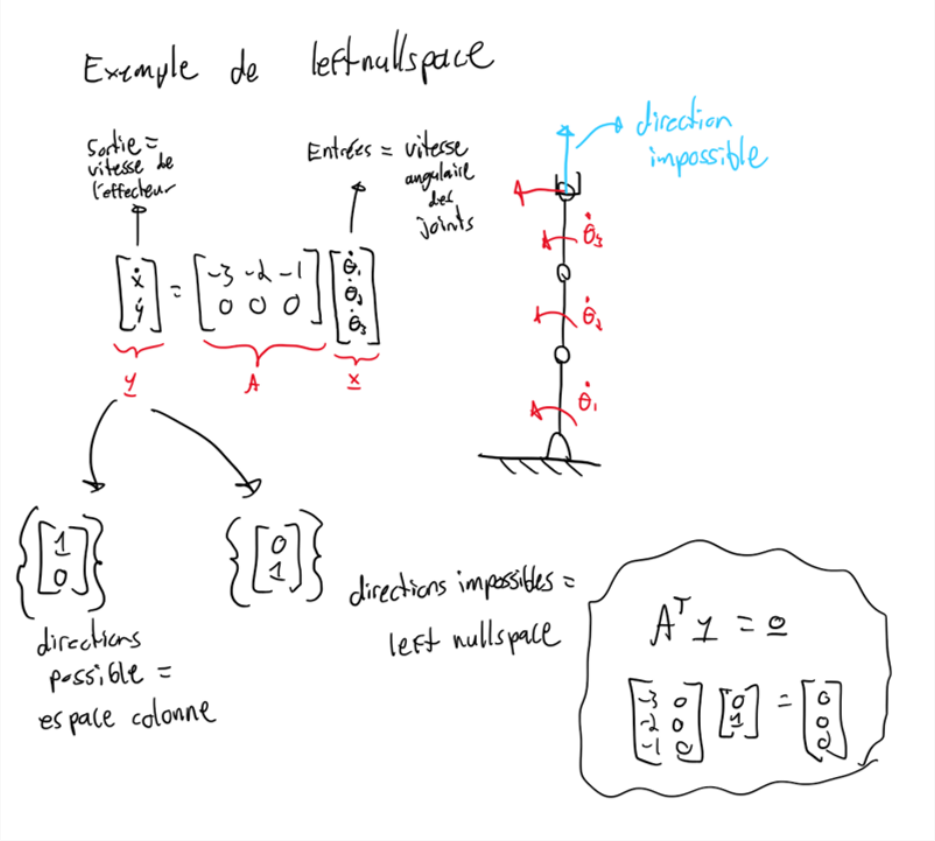


FIGURE 18.5 – Exemple de l'espace-nul gauche d'une matrice

18.6.3 Espace nul

Définition 18.24 Espace Nul:

L'espace nul d'une matrice A, noté $N(A)$ correspond a l'ensemble de tous les vecteur-colonnes $\boldsymbol{x}$ pour lesquels la multiplication avec la matrice A donne un vecteur-colonne nulle $\mathbf{0}$:

$$N(A) = \{\boldsymbol{x} \in \mathbb{R}^n \,|\, A\boldsymbol{x} = \mathbf{0}\} \tag{18.54}$$

L'espace nul est aussi appelé le noyau de la matrice.

Une matrice A va avoir un espace nul seulement lorsqu'elle a une rang déficient ($r < n$), donc des colonnes linéairement dépendantes. La présence d'un espace nul indique un système sous-contraint ou il y a un surplus de variables, donc plusieurs solutions possibles d'entrées $\boldsymbol{x}$ pour obtenir une même sortie $\boldsymbol{y}$.

Propriété 18.14: Dimensions de l'espace nul

La dimension de l'espace nul est liée au nombre de variables n et au rang r de la matrice :

$$\dim(N(A)) = n - r \tag{18.55}$$

Exemple 18.5 Espace-nul de la matrice jacobienne d'un robot manipulateur:

La figure 18.6 illustre la signification physique de l'espace nul d'une matrice qui relie la vitesse des joints d'un robot manipulateur à la vitesse de son effecteur.

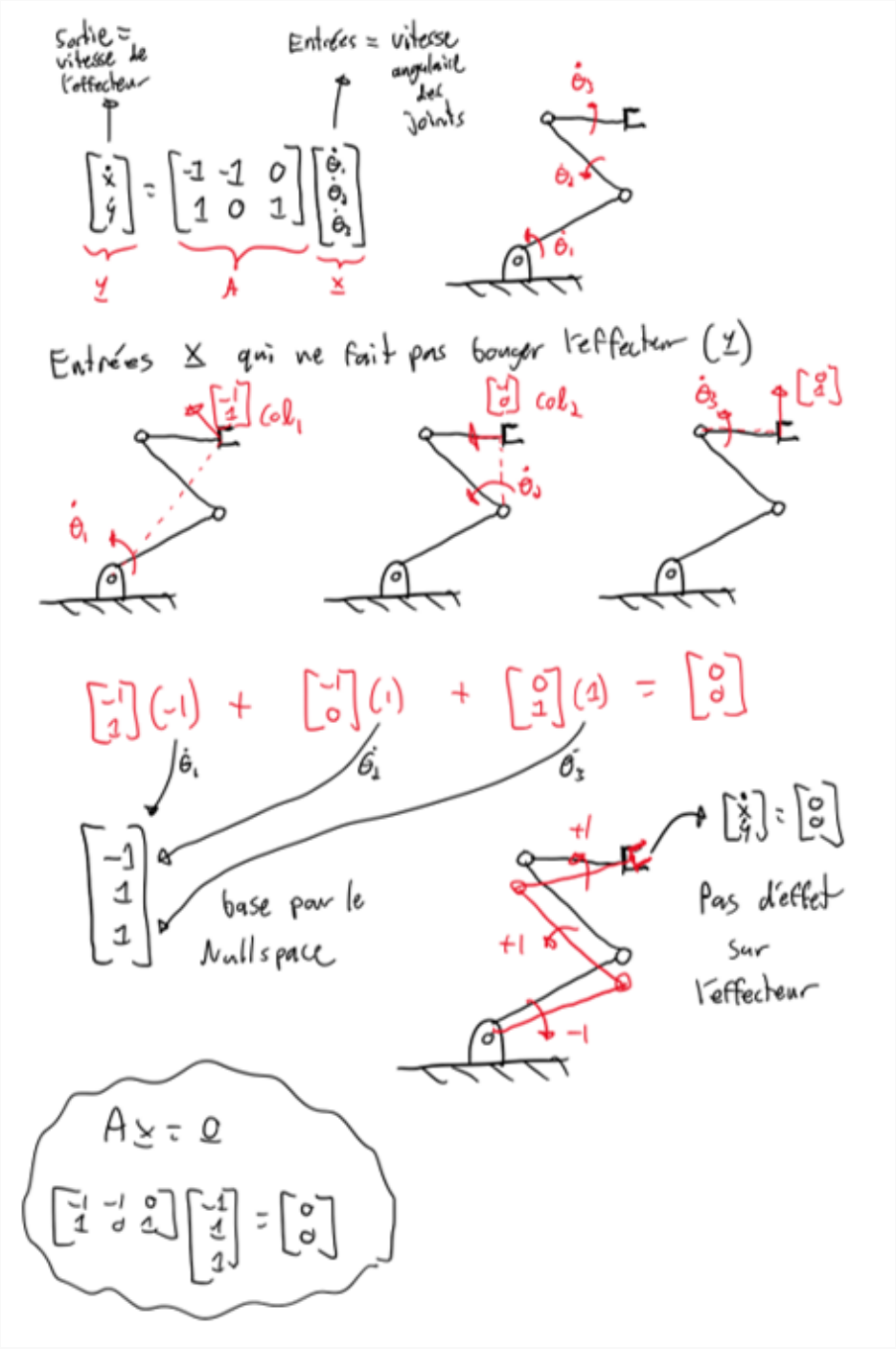


FIGURE 18.6 – Exemple de l'espace nul d'une matrice

18.6.4 Espace rangée

L'espace rangée correspond à toutes les combinaisons possibles des vecteurs-lignes de la matrice A. Les entrées $\boldsymbol{x}$ qui influencent la sortie sont celles qui appartiennent à l'espace rangée.

Définition 18.25 Espace Rangée:

L'espace rangé d'une matrice correspond à l'ensemble des vecteur-colonnes entrées $\boldsymbol{x}$ qui donnent un vecteur-colonne sortie $\boldsymbol{y}$ non-nul lorsque multiplié par la matrice A.L'espace rangée d'une matrice A, noté $row(A)$, est équivalent à l'espace colonne de la matrice transposée A^T :

$$row(A) = col(A^T) = \{A^T\boldsymbol{y} \mid \boldsymbol{y} \in \mathbb{R}^m\} \tag{18.56}$$

Lorsque la matrice A est inversible, i.e. de rang plein, l'espace rangé correspond à $\mathbb{R}^n$, ou n est le nombre de variables d'entrées. L'espace des vecteur-colonnes entrées $\boldsymbol{x}$ est formé par r (rang) bases qui forment l'espace rangée et $n - r$ bases qui forment l'espace nul. Les vecteur-colonnes appartenant à l'espace rangée influencent la sortie alors que les vecteur-colonnes appartenant à l'espace nul n'influent pas la sortie.

Propriété 18.15: Dimensions de l'espace rangée

La dimension de l'espace rangée est égale au rang r de la matrice, tout comme l'espace colonne :

$$\dim(row(A)) = r \tag{18.57}$$

Ainsi, la somme des dimensions de l'espace rangée et de l'espace nul est égale au nombre total de colonnes (variables) :

$$n = \dim(row(A)) + \dim(N(A)) \tag{18.58}$$

18.6.5 Quatre situations possibles pour un système d'équations

Un système d'équations peut être parfaitement contraint (une seule solution existe), sur-contraint (aucune solution exacte n'existe) ou bien sous-contraint (plusieurs solutions sont possible. Dans le cas d'une relation matricielle de type $A\boldsymbol{x} = \boldsymbol{y}$, qui représente un système d'équations linéaires, le rang de la matrice A permet de déterminer la situation, voir figure 18.7.

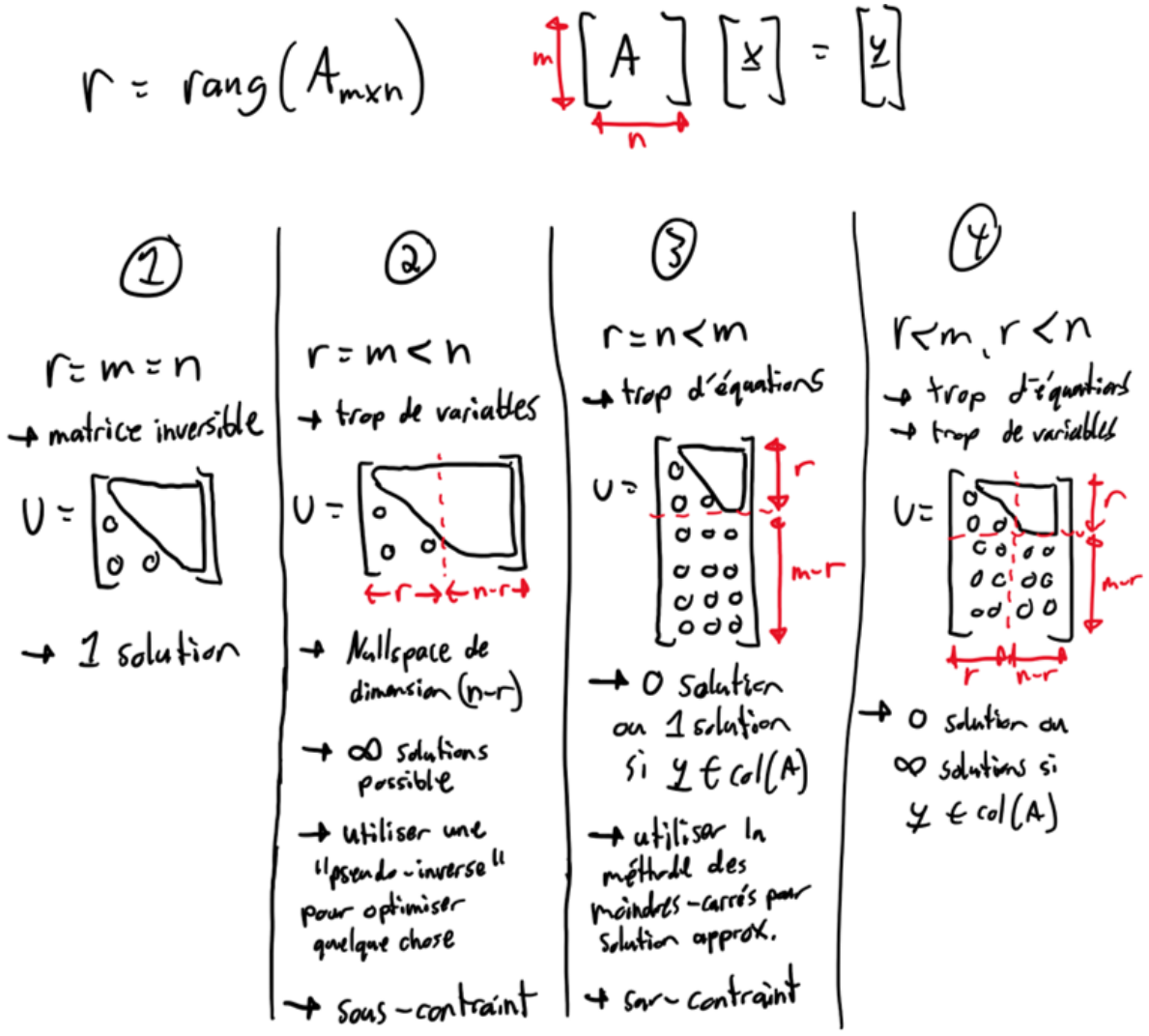


FIGURE 18.7 – Quatre situations possibles pour un système d'équations

Les grandes catégories de situations possibles illustrés à la figure 18.7 sont exemplifiés avec les exemples illustrés aux figures 18.8, 18.9, 18.10 et 18.11.

Le tableau 18.1 ci-dessous synthétise les caractéristiques de ces quatre situations, notamment les dimensions des espaces nuls qui en résultent. La dernière colonne suggère la méthode de résolution privilégiée pour chaque cas : l'inverse classique A^{-1} lorsque possible, ou les pseudo-inverses (notées A^{+}) qui généralisent le concept d'inversion pour trouver des solutions optimales (au sens des moindres carrés ou de la norme minimale) lorsque le système n'admet pas de solution unique exacte.

Situation	Dimensions	Noyau $N(A)$	Noyau Gauche $N(A^T)$	Solutions	Résolution typique
Parfaitement contraint	$r = m = n$	-	-	1 (Unique)	L'inverse matricielle A^{-1} permet de trouver l'unique solution exacte $\boldsymbol{x} = A^{-1}\boldsymbol{y}$.
Sous-contraint (Trop de variantes)	$r = m < n$	dim $n - r$	-	∞	La pseudo-inverse droite A^+ donne la solution unique qui minimise la norme $\|\boldsymbol{x}\|$.
Sur-contraint (Trop d'équations)	$r = n < m$	-	dim $m - r$	0 (approx.)	La méthode des moindres carrés (pseudo-inverse gauche A^+) minimise l'erreur d'approximation.
Singulier (Général)	$r < \min(m, n)$	dim $n - r$	dim $m - r$	0 ou ∞	La décomposition en valeurs singulières (SVD) permet de calculer une pseudo-inverse A^+.

TABLE 18.1 – Résumé des quatre situations possibles pour un système d'équations linéaires

Exemple 18.6 Système d'équation parfaitement contraint:

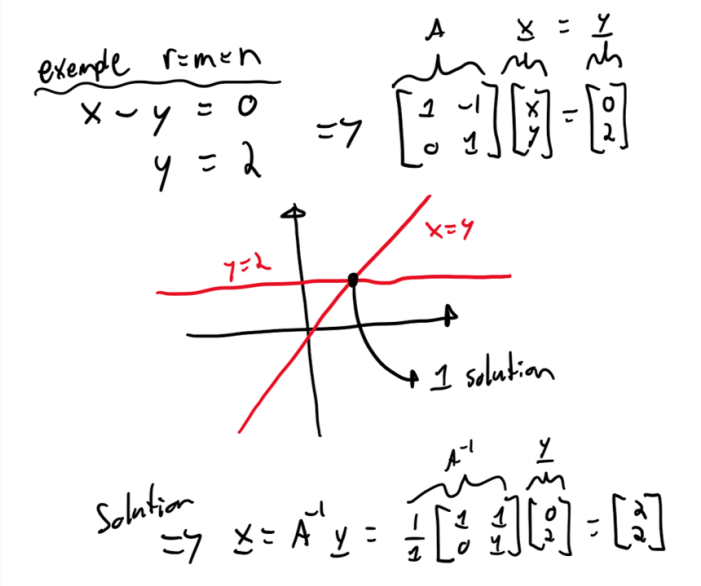


FIGURE 18.8 – Exemple d'un système d'équation parfaitement contraint

Exemple 18.7 Système d'équation sous-contraint:

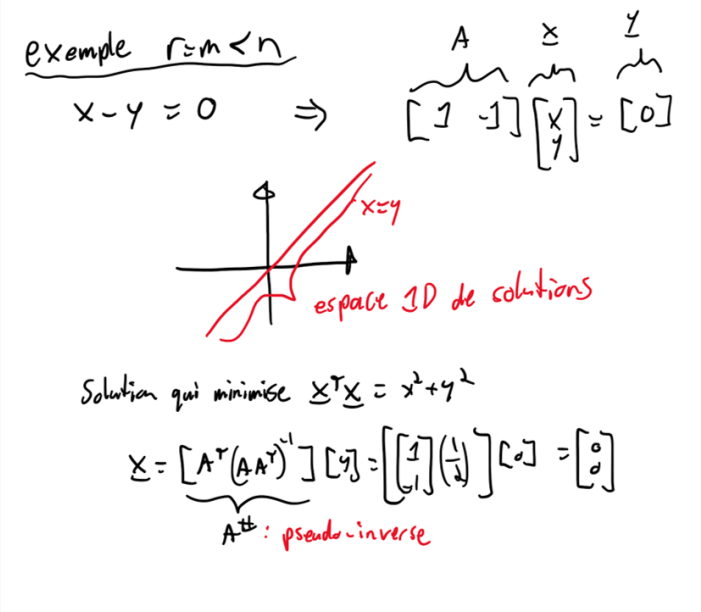


FIGURE 18.9 – Exemple d'un système d'équation sous-contraint

Exemple 18.8 Système d'équation sur-contraint:

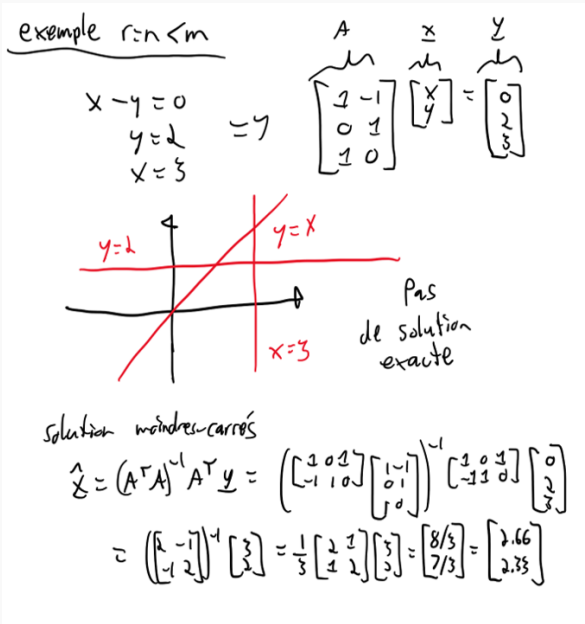


FIGURE 18.10 – Exemple d'un système d'équation sur-contraint

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Exemple 18.9 Système d'équation non-inversible:

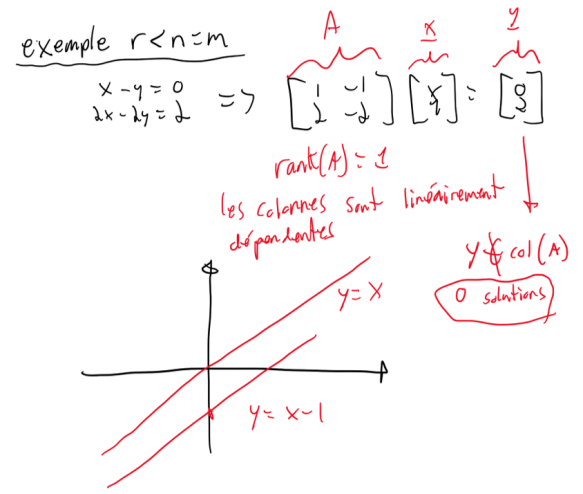


FIGURE 18.11 – Exemple d'un système d'équation non-inversible

18.7 Solution des moindres carrés pour un système sur-contraint

Lorsqu'un système d'équation linéaire $\boldsymbol{y} = A\boldsymbol{x}$ est sur-contraint, c'est-à-dire que le vecteur $\boldsymbol{y}$ n'est pas dans l'espace colonne de la matrice A, il n'y a pas de solution exacte possible. C'est définitivement le cas si il y a plus d'équations que d'inconnues ($m > n$), mais cela peut aussi être dû à des équations redondantes. Il est toutefois possible de calculer une solution approximée $\hat{\boldsymbol{x}}$ qui minimise la norme de l'erreur :

Théorème 18.26 Solution des moindres carrés:

La solution optimale, pour inverser un système d'équation linéaire sur-contraint, au sens de minimiser la norme euclidienne de l'erreur $\boldsymbol{e} = A\hat{\boldsymbol{x}} - \boldsymbol{y}$, est donnée par :

$$\hat{\boldsymbol{x}}^* = \underset{\hat{\boldsymbol{x}}}{\operatorname{argmin}} \|A\hat{\boldsymbol{x}} - \boldsymbol{y}\|^2 = (A^T A)^{-1} A^T \boldsymbol{y} \tag{18.59}$$

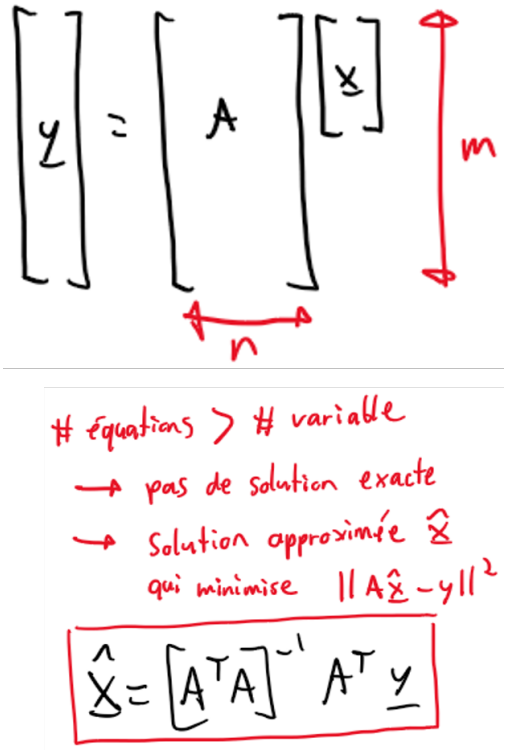


FIGURE 18.12 – Système sur-contraint : solution des moindres-carrés

La matrice $(A^T A)^{-1} A^T$ est aussi appelée inverse gauche de Moore–Penrose et notée :

$$A^+ = (A^T A)^{-1} A^T \quad \Rightarrow \quad A^+ A = I \tag{18.60}$$

Note

Il est à noter que la matrice A doit avoir des colonnes indépendantes pour que la matrice $A^T A$ soit inversible et donc pour que la méthode des moindres-carrés fonctionne.

18.7.1 Application à des problèmes d'identification de paramètres

La méthode des moindres-carrés est très utile pour identifier des paramètres inconnus dans une relation linéaire. Lorsqu'on a un modèle linéaire qui implique des signaux connus et des paramètres inconnus qui

peut être exprimé sous la forme :

$$\underline{\varphi}^T \underline{\theta} = b \quad \Rightarrow \quad \underbrace{\begin{bmatrix} \varphi_1 & \dots & \varphi_m \end{bmatrix}}_{m \text{ signaux connus}} \underbrace{\begin{bmatrix} \theta_1 \\ \vdots \\ \theta_m \end{bmatrix}}_{m \text{ paramètres inconnus}} = \underbrace{b}_{\text{signal connu}} \tag{18.61}$$

Il est possible de regrouper plusieurs mesures expérimentales dans un système matricielle :

$$A\underline{\theta} = \underline{b} \quad \Rightarrow \quad \underbrace{\begin{bmatrix} \underline{\varphi}^T(1) \\ \vdots \\ \underline{\varphi}^T(N) \end{bmatrix}}_{A \,:\, N \times m \text{ échantillons connus}} \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_m \end{bmatrix} = \underbrace{\begin{bmatrix} b(1) \\ \vdots \\ b(N) \end{bmatrix}}_{\underline{b} \,:\, N \text{ échantillons connus}} \tag{18.62}$$

Pour ensuite utiliser la méthode des moindres-carrés pour obtenir un estimé des paramètres inconnus qui minimise l'erreur avec les mesures expérimentales :

$$\hat{\boldsymbol{\theta}}^* = \underset{\hat{\boldsymbol{\theta}}}{\operatorname{argmin}} \left\| A\hat{\boldsymbol{\theta}} - \boldsymbol{b} \right\|^2 = (A^T A)^{-1} A^T \boldsymbol{b} \tag{18.63}$$

Exemple 18.10 Régression linéaire avec les moindres-carrés:

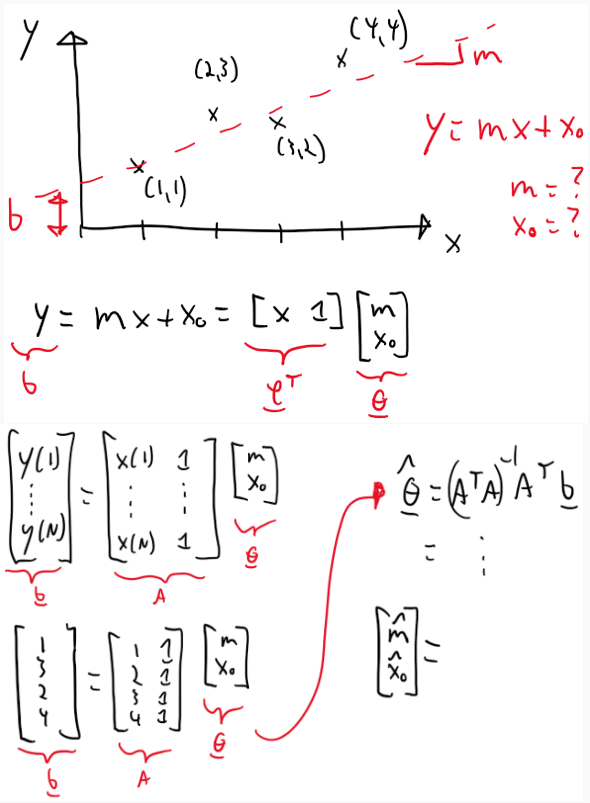


FIGURE 18.13 – Régression linéaire avec les moindres-carrés pour identifier les paramètres d'une ligne

Démonstration. La solution des moindres-carrés minimise une fonction coût qui est défini comme le carré de la norme du vecteur d'erreur, qui correspond à la somme des carrés des erreurs individuelles pour chaque

sortie du système :

$$J = \|\boldsymbol{e}\|^2 = \boldsymbol{e}^T\boldsymbol{e} = \sum e_i^2 \tag{18.64}$$

L'erreur est définie comme la différence entre les sorties $\boldsymbol{y}$ et le résultat du calcul $A\hat{\boldsymbol{x}}$:

$$\boldsymbol{e} = A\hat{\boldsymbol{x}} - \boldsymbol{y} \tag{18.65}$$

Il est ensuite possible de trouver le minimum de la fonction coût J en trouvant le point pour lequel toutes les dérivées par rapport à $\hat{\boldsymbol{x}}$ sont nulles :

$$\mathbf{0} = \frac{\partial J}{\partial \hat{\boldsymbol{x}}} \tag{18.66}$$

$$\mathbf{0} = \frac{\partial \boldsymbol{e}^T\boldsymbol{e}}{\partial \hat{\boldsymbol{x}}} \tag{18.67}$$

$$\mathbf{0} = 2\boldsymbol{e}^T\frac{\partial e}{\partial \hat{\boldsymbol{x}}} \tag{18.68}$$

$$\mathbf{0} = 2(A\hat{\boldsymbol{x}} - \boldsymbol{y})^T\frac{\partial(A\hat{\boldsymbol{x}} - \boldsymbol{y})}{\partial \hat{\boldsymbol{x}}} \tag{18.69}$$

$$\mathbf{0} = 2(A\hat{\boldsymbol{x}} - \boldsymbol{y})^T A \tag{18.70}$$

$$\mathbf{0} = \hat{\boldsymbol{x}}^T A^T A - \boldsymbol{y}^T A \tag{18.71}$$

$$A^T\boldsymbol{y} = A^T A\hat{\boldsymbol{x}} \tag{18.72}$$

$$(A^T A)^{-1}A^T\boldsymbol{y} = \hat{\boldsymbol{x}} \tag{18.73}$$

□

18.8 Solution de norme minimale pour un système sous-contraint

Lorsqu'un système d'équation linéaire $\boldsymbol{y} = A\boldsymbol{x}$ est sous-contraint, c'est-à-dire que plusieurs vecteurs $\boldsymbol{x}$ sont des solutions, il est possible d'utiliser une matrice pseudo-inverse pour obtenir une solution qui optimise une fonction coût quadratique :

Théorème 18.27 Solution de norme minimale (Pseudo-Inverse Droite):

La solution optimale $\boldsymbol{x}^*$ qui minimise la norme pondérée (matrice de poids Q) ou la norme euclidienne ($Q = I$) tout en respectant la contrainte $\boldsymbol{y} = A\boldsymbol{x}$ est donnée par :

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmin}} \left(\boldsymbol{x}^T Q \boldsymbol{x}\right) \quad \text{sujet à} \quad \boldsymbol{y} = A\boldsymbol{x} \tag{18.74}$$

$$\boldsymbol{x}^* = Q^{-1}A^T(AQ^{-1}A^T)^{-1}\,\boldsymbol{y} \tag{18.75}$$

Dans le cas simplifié où $Q = I$, la solution minimise la norme $\|\boldsymbol{x}\|$:

$$\boldsymbol{x}^* = A^T(AA^T)^{-1}\,\boldsymbol{y} \tag{18.76}$$

La matrice $A^{\#} = A^T(AA^T)^{-1}$ est appelée inverse droit de Moore–Penrose.

Note

Il est à noter que la matrice A doit avoir des rangées indépendantes (plein rang ligne) pour que la matrice AA^T soit inversible et donc pour pouvoir utiliser la pseudo-inverse droite.

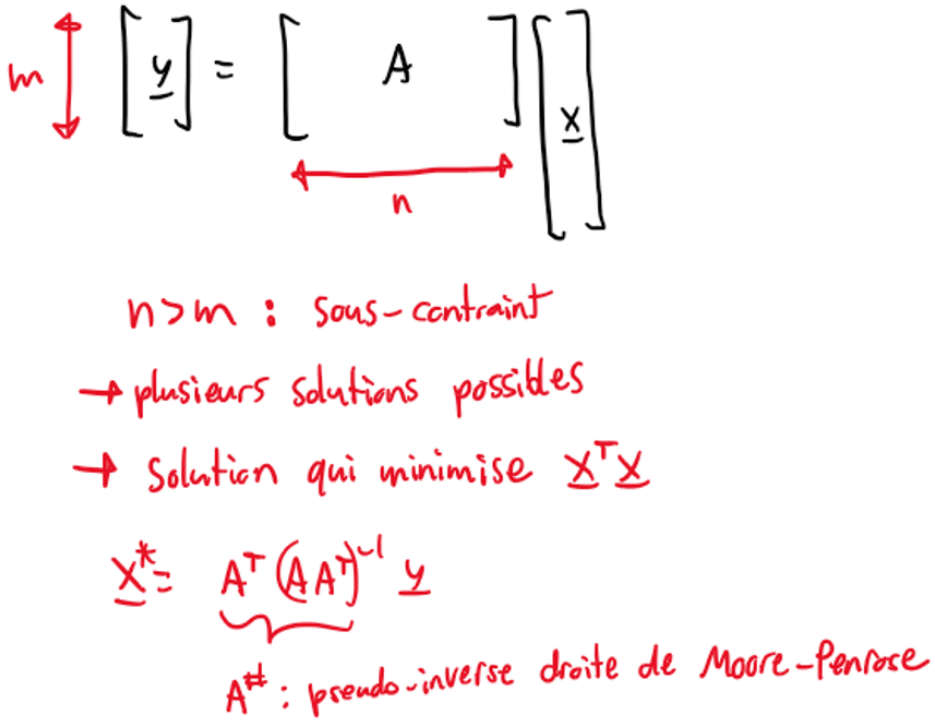


FIGURE 18.14 – Système sous-contraint - solution avec matrice pseudo-inverse

Exemple 18.11 Utilisation d'une matrice pseudo-inverse:

Pour cet exemple, un robot doit produire une force de 14 N sur un mur. Pour y parvenir le courant électrique dans les trois moteurs doit être ajusté. La relation entre la force et le courant dans les moteurs est linéaire et illustrée ci-dessous. On cherche ici une solution qui minimise les pertes joules ri^2 dans les moteurs. Comme il y a plusieurs solutions possibles pour atteindre le niveau de force désiré, et que la fonction coût à minimiser est quadratique, une matrice pseudo-inverse droite peut être utilisée pour calculer la solution optimale directement, voir Figure 18.15.

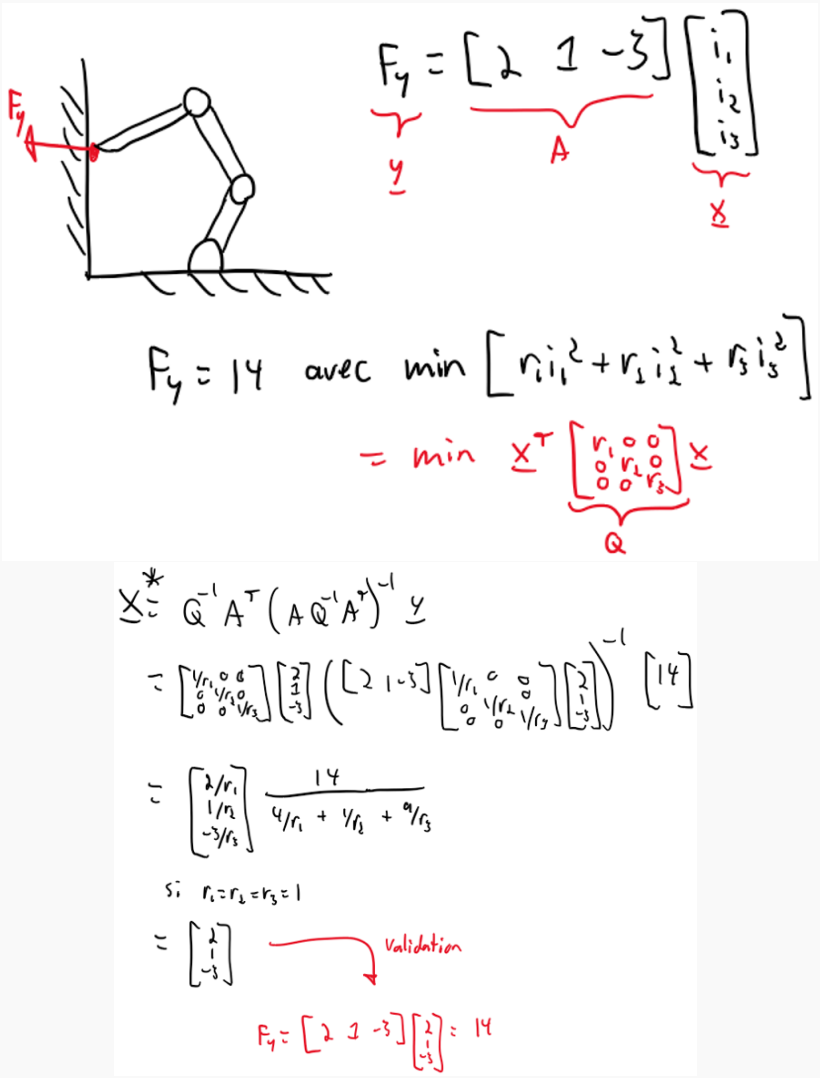


FIGURE 18.15 – Exemple d'utilisation d'une matrice pseudo-inverse

18.9 Vecteurs et valeurs propres

Pour les matrices carrées ($m = n$), une autre caractéristique utile est l'ensemble de ses vecteurs et valeurs propres.

Définition 18.28 Vecteurs et valeurs propres:

Un vecteur $\boldsymbol{v}$ est un vecteur propre d'une matrice A si le résultat $\boldsymbol{y} = A\boldsymbol{v}$ est parallèle à l'entrée $\boldsymbol{v}$, autrement dit s'il existe un scalaire λ (appelé valeur propre) tel que :

$$A\boldsymbol{v} = \lambda\boldsymbol{v} \tag{18.77}$$

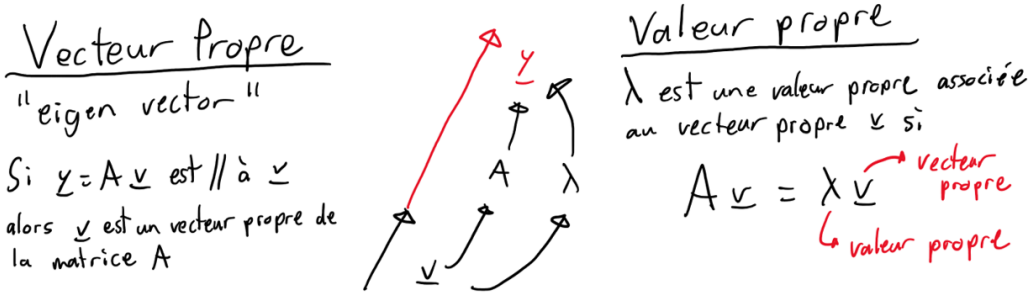


FIGURE 18.16 – Vecteurs et valeurs propres

Propriété 18.16: Nombre de solutions

Une matrice $n \times n$ possède exactement n paires de vecteurs et valeurs propres (qui peuvent être répétés et aussi des nombres complexes).

Propriété 18.17: Matrice singulière

Si une ou plusieurs valeurs propres de la matrice est égale à zéro, alors la matrice est singulière (non-inversible). Les vecteurs propres associés forment une base de l'espace nul de la matrice.

Propriété 18.18: Trace et Déterminant

La trace d'un matrice est égale à la somme des valeurs propres et le déterminant d'une matrice est égale à la multiplication de toutes les valeurs propres :

$$trace(A) = \sum_{1}^{n} \lambda_i \tag{18.78}$$

$$det(A) = \prod_{1}^{n} \lambda_i \tag{18.79}$$

18.9.1 Calcul des valeurs et vecteurs propres

Pour déterminer les valeurs et vecteurs propres d'une matrice, le problème est posé ainsi :

$$A\boldsymbol{x} = \lambda\boldsymbol{x} \tag{18.80}$$

$$(A - \lambda I)\boldsymbol{x} = \boldsymbol{0} \tag{18.81}$$

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

La matrice $(A - \lambda I)$ doit être singulière pour qu'il existe des solutions non-nulles à cette équation. Il est donc possible de trouver les valeurs propres λ_i en solutionnant l'équation suivante :

$$det(A - \lambda I) = 0 \tag{18.82}$$

Pour chaque solution, i.e pour chaque valeur propre λ_i, il est possible de déterminer le vecteur propre associé $\boldsymbol{v}_i$ en déterminant l'espace nul de $(A - \lambda I)$:

$$\boldsymbol{v}_i \in nul(A - \lambda I) \tag{18.83}$$

Il est à noté que pour chaque valeur propre, le vecteur propre n'est pas unique, seule la direction l'est. Si $\boldsymbol{v}_i$ est un vecteur propre alors tout multiple de lui même est aussi un vecteur propre. Donc autrement dit pour chaque valeur propre λ_i, on cherche en fait une base pour l'espace 1D associé aux vecteurs propres de cette valeur.

Exemple 18.12 Exemple:

La figure 18.17 illustre un exemple de vecteurs et valeurs propres dans un contexte de transformations géométriques.

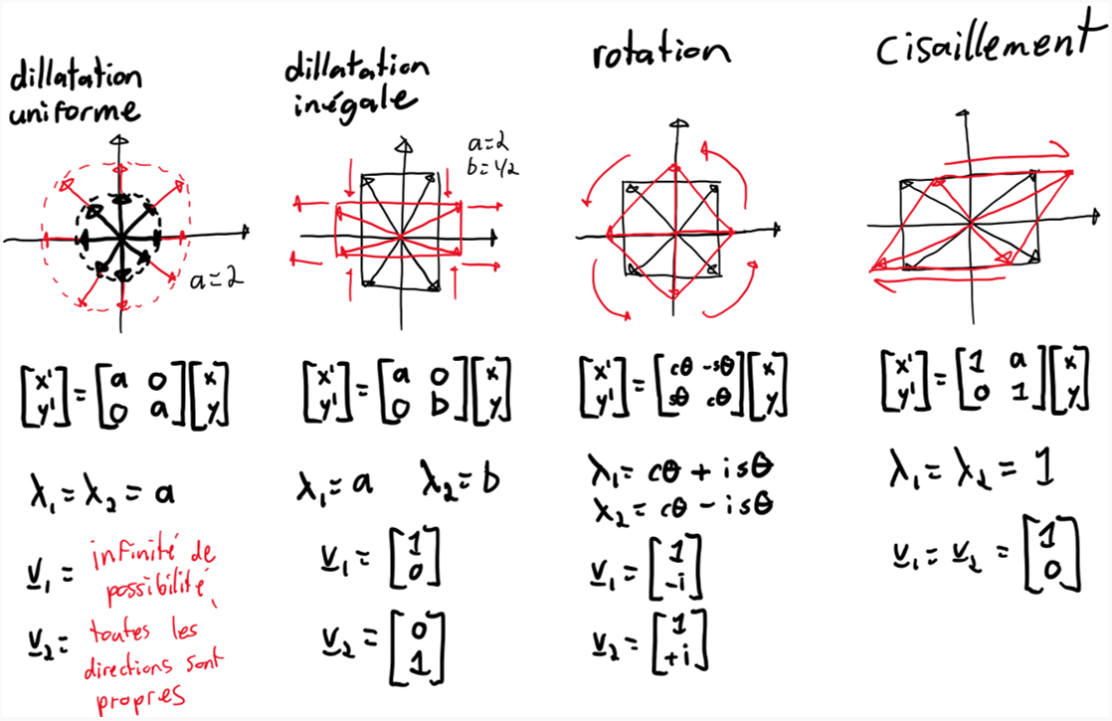


FIGURE 18.17 – Exemple de vecteurs et valeurs propres dans un contexte de transformations géométriques

18.9.2 Diagonalisation

La diagonalisation d'une matrice peut être interprété comme un changement de base, vers des coordonnées dites propres, pour lesquelles l'effet de la multiplication de cette matrice est indépendant pour chaque axe.

Définition 18.29 Forme Diagonale:

Si une matrice a n vecteurs propres indépendants, alors la matrice peut être diagonalisée, c'est à dire décomposée sous la forme :

$$A = V\Lambda V^{-1} \tag{18.84}$$

où la matrice V regroupe tous les vecteurs propres de A, et la matrice diagonale Λ toutes les valeurs propres de A :

$$V = \begin{bmatrix} \begin{bmatrix} \\ \underline{v}_1 \\ \\ \end{bmatrix} \dots \begin{bmatrix} \\ \underline{v}_n \\ \\ \end{bmatrix} \end{bmatrix} \qquad \Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \lambda_2 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \lambda_n \end{bmatrix} \tag{18.85}$$

Lorsqu'une matrice a plusieurs vecteur propres associés à la même valeur propre alors la matrice n'est pas diagonalisable. Il est toutefois possible de faire appel à une méthode alternative appelée la réduction de Jordan.

Démonstration. Par définition, si les vecteurs propres $\boldsymbol{v}_i$ sont multipliés leur matrice A, la multiplication est équivalente à les multiplier par les valeurs propres associées λ_i :

$$AV = \begin{bmatrix} A \begin{bmatrix} \\ \underline{v}_1 \\ \\ \end{bmatrix} \dots A \begin{bmatrix} \\ \underline{v}_n \\ \\ \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \lambda_1 \begin{bmatrix} \\ \underline{v}_1 \\ \\ \end{bmatrix} \dots \lambda_n \begin{bmatrix} \\ \underline{v}_n \\ \\ \end{bmatrix} \end{bmatrix} = V\Lambda \tag{18.86}$$

Ensuite, si la matrice V est inversible (ce qui explique la condition d'indépendance des vecteurs propres pour que la matrice soit diagonalisable), il suffit de multiplier l'équation (18.84) par V^{-1} par la droite pour obtenir l'équation (18.84). □

Exemple 18.13 Exemple de diagonalisation pour une dillatation en 2D:

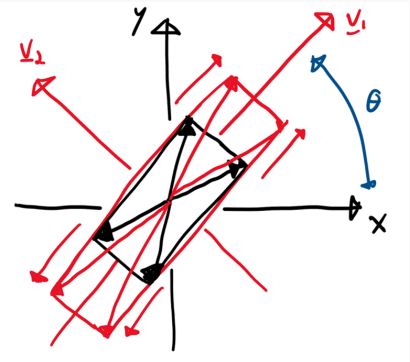


FIGURE 18.18 – Exemple de diagonalisation pour une transformation géométrique

$$A = \begin{bmatrix} c\theta^2+1 & c\theta s\theta \\ c\theta s\theta & s\theta^2+1 \end{bmatrix} = \underbrace{\begin{bmatrix} c\theta & -s\theta \\ s\theta & c\theta \end{bmatrix}}_{V} \underbrace{\begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}}_{\Lambda} \underbrace{\begin{bmatrix} c\theta & s\theta \\ -s\theta & c\theta \end{bmatrix}}_{V^{-1}} \tag{18.87}$$

18.9.3 Puissance d'une matrice

Propriété 18.19: Puissance d'une matrice

La puissance p d'une matrice est définie comme la multiplication répétée de p copies de cette matrice. Si la matrice est diagonalisable, la puissance de la matrice A est égale à

$$A^p = V\Lambda^p V^{-1} \tag{18.88}$$

avec :

$$\Lambda^p = \begin{bmatrix} \lambda_1^p t & 0 & 0 & 0 \\ 0 & \lambda_2^p & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \lambda_n^p \end{bmatrix} \tag{18.89}$$

Cette propriété découle de la diagonalisation :

$$A^p = A\,A\,\ldots\,A = V\Lambda V^{-1}\,V\Lambda V^{-1}\,\ldots\,V\Lambda V^{-1} = V\Lambda\Lambda\ldots\Lambda V^{-1} = V\Lambda^p V^{-1} \tag{18.90}$$

Exemple 18.14 Exemple de calcul de l'évolution d'un système discret:

La suite de Fibonacci est définie par une séquence de chiffre pour lequel le prochain est la somme des deux précédents :

$$0\ 1\ 1\ 2\ 3\ 5\ 8\ 13\ 21\ 34\ 55 \tag{18.91}$$

Le chiffre suivant est calculé en fonction des deux précédents :

$$f_{k+1} = f_k + f_{k-1} \tag{18.92}$$

il faut donc un vecteur d'état $\boldsymbol{x}$ de deux variables (f_k et f_{k-1}) pour prédire le prochain chiffre. L'évolution du vecteur d'état peut être décrite par l'équation matricielle suivante :

$$\underbrace{\begin{bmatrix} f_{k+1} \\ f_k \end{bmatrix}}_{\boldsymbol{x}_{k+1}} = \underbrace{\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} f_k \\ f_{k-1} \end{bmatrix}}_{\boldsymbol{x}_k} \tag{18.93}$$

Il est possible de calculer la valeur future d'une évolution discrète comme celle-ci sans calculer toute la séquence grâce la propriété (18.88) :

$$\boldsymbol{x}_{k+1} = A\boldsymbol{x}_k \tag{18.94}$$
$$\boldsymbol{x}_{k+2} = A\boldsymbol{x}_{k+1} = AA\boldsymbol{x}_k \tag{18.95}$$
$$\boldsymbol{x}_{k+3} = A\boldsymbol{x}_{k+2} = AAA\boldsymbol{x}_k \tag{18.96}$$
$$\boldsymbol{x}_{k+p} = A^p\boldsymbol{x}_k = V\Lambda^p V^{-1}\boldsymbol{x}_k \tag{18.97}$$

Pour la suite de Fibonacci, la matrice A qui décrit l'évolution peut être diagonalisée. Premièrement on détermine les valeurs propres :

$$0 = det(A - \lambda I) = det\left(\begin{bmatrix} 1-\lambda & 1 \\ 1 & -\lambda \end{bmatrix}\right) = \lambda^2 - \lambda - 1 \tag{18.98}$$

Il y a donc deux solutions :

$$\lambda_1 = \frac{1+\sqrt{5}}{2} = 1.61803399 \qquad \lambda_2 = \frac{1-\sqrt{5}}{2} = -0.61803399 \tag{18.99}$$

ou la première valeur propre est égale au très fameux nombre d'or. Il est possible de déterminer les vecteurs propres associé en substituant dans l'équation $A\boldsymbol{v} = \lambda\boldsymbol{v}$, on trouve alors :

$$\boldsymbol{v}_1 = \begin{bmatrix} \lambda_1 \\ 1 \end{bmatrix} \qquad \boldsymbol{v}_2 = \begin{bmatrix} \lambda_2 \\ 1 \end{bmatrix} \tag{18.100}$$

Il est donc maintenant possible de calculer n'importe quelle position future dans la séquence directement grâce à l'équation (18.97). Si on cherche la position $p = 1E9$ à partir du début de la suite de Fibonacci :

$$\boldsymbol{x}_p = V\Lambda^p V^{-1}\boldsymbol{x}_0 \tag{18.101}$$

$$\begin{bmatrix} f_p \\ f_{p-1} \end{bmatrix} = \begin{bmatrix} \lambda_1 & \lambda_2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1^p & 0 \\ 0 & \lambda_2^p \end{bmatrix} \begin{bmatrix} \lambda_1 & \lambda_2 \\ 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{18.102}$$

lorsque p est très grand il est possible de simplifier le calcul en négligeant la contribution $\lambda_2^p \approx 0$ puisque lorsque que la valeur absolue de la base est inférieur à l'unité, la puissance tend vers zéro lorsque p tend vers l'infini. Il est donc possible de simplifier :

$$\begin{bmatrix} f_p \\ f_{p-1} \end{bmatrix} = \begin{bmatrix} \lambda_1 & \lambda_2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1^p & 0 \\ 0 & 0 \end{bmatrix} \frac{1}{\lambda_1 - \lambda_2} \begin{bmatrix} 1 & -\lambda_2 \\ -1 & \lambda_1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{18.103}$$

$$\begin{bmatrix} f_p \\ f_{p-1} \end{bmatrix} = \begin{bmatrix} \lambda_1^{p+1} & 0 \\ \lambda_1^p & 0 \end{bmatrix} \frac{1}{\lambda_1 - \lambda_2} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \tag{18.104}$$

$$\begin{bmatrix} f_p \\ f_{p-1} \end{bmatrix} = \frac{1}{\lambda_1 - \lambda_2} \begin{bmatrix} \lambda_1^{p+1} \\ \lambda_1^p \end{bmatrix} \tag{18.105}$$

Il est donc possible de calculer directement la valeur dans la suite pour une positon p lorsque p est très grand :

$$f_p = \frac{\lambda_1^{p+1}}{\lambda_1 - \lambda_2} = \frac{1.618^{(p+1)}}{\sqrt{5}} \tag{18.106}$$

18.9.4 Exponentielle de matrice

De manière similaire à la puissance d'une matrice pour les systèmes discrets, l'exponentielle de matrice permet de prédire l'évolution des systèmes continus.

Définition 18.30 Exponentielle de matrice:

L'exponentielle d'une matrice carrée A, notée e^A ou $\exp(A)$, est définie par la série de Taylor :

$$e^A = \sum_{k=0}^{\infty} \frac{1}{k!} A^k = I + A + \frac{1}{2!}A^2 + \frac{1}{3!}A^3 + \dots \tag{18.107}$$

Cette série converge toujours pour toute matrice carrée.

Le calcul de cette série infinie est généralement impraticable directement. Cependant, si la matrice est diagonalisable, une propriété très utile permet le calcul exact.

Propriété 18.20: Calcul par diagonalisation

Si une matrice A est diagonalisable, telle que $A = V\Lambda V^{-1}$, alors l'exponentielle de la matrice est donnée par :

$$e^{A} = Ve^{\Lambda}V^{-1} \tag{18.108}$$

où l'exponentielle de la matrice diagonale se calcule élément par élément :

$$e^{\Lambda} = \begin{bmatrix} e^{\lambda_1} & 0 & 0 & 0 \\ 0 & e^{\lambda_2} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & e^{\lambda_n} \end{bmatrix} \tag{18.109}$$

18.10 Décomposition en valeurs singulières (SVD)

La décomposition en valeurs singulières (SVD) est une méthode de factorisation matricielle qui généralise la diagonalisation à toutes les matrices, même rectangulaires ($m \times n$).

Définition 18.31 SVD:

Toute matrice $A \in \mathbb{R}^{m \times n}$ peut être décomposée sous la forme :

$$A = U\Sigma V^T \tag{18.110}$$

où :

- U est une matrice orthogonale $m \times m$ ($U^T U = I$) dont les colonnes sont les vecteurs propres de AA^T, i.e. les directions principales de l'espace de sortie de la matrice.
- V est une matrice orthogonale $n \times n$ ($V^T V = I$) dont les colonnes sont les vecteurs propres de $A^T A$, i.e. les directions principales de l'espace d'entrée de la matrice.
- Σ est une matrice diagonale $m \times n$ contenant les **valeurs singulières** σ_i sur sa diagonale, en ordre décroissant : $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$. Ces valeurs sont les racines carrées des valeurs propres non-nulles de $A^T A$.

On peut interpréter l'opération de multiplication d'un vecteur $\boldsymbol{x}$ par la matrice A comme trois opérations, une rotation U^T, un étirement selon les axes Σ, et une autre rotation V.

18.10.1 Calcul de la pseudo-inverse avec SVD

La SVD est l'outil de choix pour calculer la pseudo-inverse de Moore-Penrose A^+ de manière robuste, surtout lorsque la matrice est singulière ou mal conditionnée.

Propriété 18.21: Pseudo-inverse via SVD

Si la décomposition SVD de A est $A = U\Sigma V^T$, alors sa pseudo-inverse A^+ est donnée par :

$$A^+ = V\Sigma^+ U^T \tag{18.111}$$

où Σ^+ est obtenue en transposant Σ et en inversant chaque valeur singulière non-nulle ($\frac{1}{\sigma_i}$).

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix} \quad \Rightarrow \quad \Sigma^+ = \begin{bmatrix} 1/\sigma_1 & 0 & \dots & 0 \\ 0 & 1/\sigma_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}^T \tag{18.112}$$

Pour la stabilité numérique, les valeurs singulières nulles (inférieures à un seuil ϵ) sont mises à zéro.

Définition du problème d'optimisation

L'utilisation de la pseudo-inverse via SVD permet de résoudre un problème d'optimisation bien précis qui couvre tous les cas (sur-contraint, sous-contraint et singulier).

Théorème 18.32 Solution de norme minimale aux moindres carrés:

La solution $\boldsymbol{x}^* = A^+ \boldsymbol{y}$ calculée via la SVD résout le problème d'optimisation suivant en deux étapes :

$$\boldsymbol{x}^* = \underset{\boldsymbol{x} \in \mathcal{S}}{\operatorname{argmin}} \|\boldsymbol{x}\| \quad \text{où} \quad \mathcal{S} = \underset{\boldsymbol{z}}{\operatorname{argmin}} \|A\boldsymbol{z} - \boldsymbol{y}\| \tag{18.113}$$

1. **Minimiser l'erreur** : Trouver l'ensemble $\mathcal{S}$ des vecteurs $\boldsymbol{x}$ qui minimisent la norme de l'erreur $\|A\boldsymbol{x} - \boldsymbol{y}\|$ (Moindres carrés).

2. **Minimiser la norme de la solution** : Parmi tous ces vecteurs minimisant l'erreur, choisir celui qui a la plus petite norme $\|\boldsymbol{x}\|$ (Norme minimale).

18.10.2 Équivalence entre formules de pseudo-inverse

Les formules classiques sont des cas particuliers de la SVD lorsque la matrice est de rang plein :
— **Inverse à gauche** ($m > n$, rang plein colonne) : $A^+ = (A^TA)^{-1}A^T$.
— **Inverse à droite** ($n > m$, rang plein ligne) : $A^+ = A^T(AA^T)^{-1}$.

Cas du plein rang colonne ($m \geq n$)

Sous l'hypothèse de rang n, on peut écrire l'identité $\Sigma^+ = (\Sigma^T\Sigma)^{-1}\Sigma^T$.

$$\begin{aligned} A^+ &= V\left[(\Sigma^T\Sigma)^{-1}\Sigma^T\right]U^T \\ &= V(\Sigma^T\Sigma)^{-1}(V^TV)\Sigma^TU^T \quad \text{(on insère } V^TV = I\text{)} \\ &= (V(\Sigma^T\Sigma)V^T)^{-1}(V\Sigma^TU^T) \\ &= (A^TA)^{-1}A^T \end{aligned}$$

Cas du plein rang ligne ($n \geq m$)

Sous l'hypothèse de rang m, on utilise $\Sigma^+ = \Sigma^T(\Sigma\Sigma^T)^{-1}$.

$$\begin{aligned} A^+ &= V\left[\Sigma^T(\Sigma\Sigma^T)^{-1}\right]U^T \\ &= V\Sigma^T(U^TU)(\Sigma\Sigma^T)^{-1}U^T \quad \text{(on insère } U^TU = I\text{)} \\ &= (V\Sigma^TU^T)(U(\Sigma\Sigma^T)U^T)^{-1} \\ &= A^T(AA^T)^{-1} \end{aligned}$$

La définition via SVD est donc la plus universelle et générique.

18.11 Matrices de Projection

Une matrice de projection P projette un vecteur sur un sous-espace vectoriel. Si la projection est orthogonale, la matrice est aussi symétrique ($P^T = P$).

Définition 18.33 Projection sur l'espace sortie (Espace Colonne):

La projection orthogonale d'un vecteur $\boldsymbol{y} \in \mathbb{R}^m$ sur l'espace colonne de A (notée P_{col}) donne le vecteur $\boldsymbol{p}$ le plus proche de $\boldsymbol{y}$ dans $col(A)$. La matrice de projection est donnée par :

$$P_{col} = AA^+ \tag{18.114}$$

où A^+ est la pseudo-inverse de A. Si A est de plein rang colonne, alors $P_{col} = A(A^TA)^{-1}A^T$. Cette projection est étroitement liée à la solution des moindres carrés, car $\hat{\boldsymbol{y}} = P_{col}\boldsymbol{y}$.

Définition 18.34 Projection sur l'espace entrée (Espace Rangée):

La projection orthogonale d'un vecteur $\boldsymbol{x} \in \mathbb{R}^n$ sur l'espace rangée de A (notée P_{row}) donne le vecteur $\boldsymbol{p}$ le plus proche de $\boldsymbol{x}$ dans $row(A)$. La matrice de projection est donnée par :

$$P_{row} = A^+A \tag{18.115}$$

Si A est de plein rang ligne, alors $P_{row} = A^T(AA^T)^{-1}A$.

Définition 18.35 Projection sur l'espace nul gauche:

La projection sur l'espace nul gauche (orthogonal à l'espace colonne) est donnée par le complément de la projection sur l'espace colonne :

$$P_{lnull} = I - P_{col} = I - AA^+ \tag{18.116}$$

Définition 18.36 Projection sur l'espace nul:

La projection sur l'espace nul (orthogonal à l'espace rangée) est donnée par le complément de la projection sur l'espace rangée :

$$P_{null} = I - P_{row} = I - A^+A \tag{18.117}$$

Les matrices de projection permettent de décomposer tout vecteur en deux composantes orthogonales. Pour le vecteur de sortie $\boldsymbol{y} \in \mathbb{R}^m$:

$$\boldsymbol{y} = \underbrace{P_{col}\boldsymbol{y}}_{\in col(A)} + \underbrace{P_{lnull}\boldsymbol{y}}_{\in N(A^T)} \tag{18.118}$$

Le système $A\boldsymbol{x} = \boldsymbol{y}$ n'admet une solution que si $\boldsymbol{y}$ réside entièrement dans l'espace colonne, c'est-à-dire si $P_{col}\boldsymbol{y} = \boldsymbol{y}$ (ou de manière équivalente $P_{lnull}\boldsymbol{y} = \boldsymbol{0}$). Si ce n'est pas le cas, $P_{col}\boldsymbol{y}$ représente la meilleure approximation possible de $\boldsymbol{y}$ qui peut être atteinte par $A\boldsymbol{x}$, ce qui correspond à la solution des **moindres carrés** (voir section 18.7).

Pour le vecteur d'entrée $\boldsymbol{x} \in \mathbb{R}^n$:

$$\boldsymbol{x} = \underbrace{P_{row}\boldsymbol{x}}_{\in row(A)} + \underbrace{P_{null}\boldsymbol{x}}_{\in N(A)} \tag{18.119}$$

Seule la partie $P_{row}\boldsymbol{x}$ "travaille" pour produire la sortie, car la partie $P_{null}\boldsymbol{x}$ est annulée par la multiplication avec A. Ainsi, parmi toutes les solutions possibles d'un système sous-contraint, celle donnée par $\boldsymbol{x}^* = P_{row}\boldsymbol{x}$ est celle qui a la plus petite norme, car elle ne contient aucune composante "inutile" de l'espace nul. C'est la solution de **norme minimale** (voir section 18.8).

Chapitre 19

Trigonométrie

19.0.1 Fonctions trigonométriques

À venir !

19.0.2 Identités trigonométriques

Somme au carré :

$$\sin^2(\theta) + \cos^2(\theta) = 1 \tag{19.1}$$

Inversion du signe de l'angle :

$$\sin(-\theta) = -\sin(\theta) \tag{19.2}$$
$$\cos(-\theta) = \cos(\theta) \tag{19.3}$$

Déphasage de 90^o :

$$\sin\left(\frac{\pi}{2} - \theta\right) = \cos(\theta) \tag{19.4}$$
$$\cos\left(\frac{\pi}{2} - \theta\right) = \sin(\theta) \tag{19.5}$$

Déphasage de 180^o :

$$\sin(\pi - \theta) = \sin(\theta) \tag{19.6}$$
$$\cos(\pi - \theta) = -\cos(\theta) \tag{19.7}$$

Somme de deux angles :

$$\sin(\theta_1 + \theta_2) = \sin(\theta_1)\cos(\theta_2) + \cos(\theta_1)\sin(\theta_2) \tag{19.8}$$
$$\sin(\theta_1 - \theta_2) = \sin(\theta_1)\cos(\theta_2) - \cos(\theta_1)\sin(\theta_2) \tag{19.9}$$
$$\cos(\theta_1 + \theta_2) = \cos(\theta_1)\cos(\theta_2) - \sin(\theta_1)\sin(\theta_2) \tag{19.10}$$
$$\cos(\theta_1 - \theta_2) = \cos(\theta_1)\cos(\theta_2) + \sin(\theta_1)\sin(\theta_2) \tag{19.11}$$

Dérivés :

$$\frac{d}{d\theta}\sin(\theta) = \cos(\theta) \tag{19.12}$$
$$\frac{d}{d\theta}\cos(\theta) = -\sin(\theta) \tag{19.13}$$

Chapitre 20

Analyse de la stabilité

20.1 Introduction

Lorsqu'on dispose des équations du mouvement, il est toujours possible de simuler le futur à partir de conditions initiales spécifiques $\mathbf{x}(t=0)$. Cependant, cette simulation ne décrit qu'une trajectoire unique. Rien ne garantit qu'un comportement très différent n'émergera pas si l'on modifie légèrement ces conditions initiales. L'objectif de l'analyse de stabilité d'un système dynamique est de **tirer des conclusions génériques sur l'évolution future du système pour un ensemble de conditions initiales**, sans avoir à calculer chaque trajectoire. On s'intéresse typiquement au comportement asymptotique, c'est-à-dire l'état du système lorsque $t \to \infty$, souvent on cherche à savoir s'il va converger sur une position cible :

$$\lim_{t\to\infty} \mathbf{x}(t) = \mathbf{x}_d \,? \tag{20.1}$$

Capsule vidéo
Introduction à l'analyse de stabilité pour les systèmes non-linéaires
`https://youtu.be/q0Oqa5J3zEk`

20.1.1 Cadre de référence

Dans cette section, le niveau d'abstraction le plus générique d'équations dynamiques est utilisé, avec deux variantes : les systèmes dynamiques dit autonome ou non-autonomes.

Définition 20.1 Système Autonome:

Un système dynamique est dit **autonome** si sa dynamique ne dépend pas explicitement du temps t, mais uniquement de l'état actuel $\mathbf{x}$.

$$\dot{\mathbf{x}} = f(\mathbf{x}) \tag{20.2}$$

Définition 20.2 Système Non-Autonome:

Un système dynamique est dit **non-autonome** si sa dynamique dépend explicitement de la variable temporelle t.

$$\dot{\mathbf{x}} = f(\mathbf{x}, t) \tag{20.3}$$

C'est équations peuvent représenter tout genre de systèmes dynamique, en boucle ouverte ou fermée, tant qu'on est en présence d'équations en temps continu. Typiquement, un système en boucle fermée, avec une loi de commande π pour une cible fixe, va donner un système autonome :

$$\dot{\mathbf{x}} = f_{BO}(\mathbf{x}, \mathbf{u}) \quad \text{avec} \quad \mathbf{u} = \pi(\mathbf{x}) \quad \Rightarrow \quad \dot{\mathbf{x}} = f_{BO}(\mathbf{x}, \pi(\mathbf{x})) = f_{BF}(\mathbf{x}) \tag{20.4}$$

Alors qu'un système en boucle fermée, avec une loi de commande π ciblant une trajectoire temporelle $\mathbf{x}_d(t)$, va donner un système non-autonome :

$$\dot{\mathbf{x}} = f_{BO}(\mathbf{x}, \mathbf{u}) \quad \text{avec} \quad \mathbf{u} = \pi(\mathbf{x}, \mathbf{x}_d(t)) \quad \Rightarrow \quad \dot{\mathbf{x}} = f_{BO}(\mathbf{x}, \pi(\mathbf{x}, \mathbf{x}_d(t))) = f_{BF}(\mathbf{x}, t) \tag{20.5}$$

Généralité de l'analyse à l'origine En pratique, on va chercher à analyser si un système dynamique va se stabiliser à une configuration arbitraire désirée $\mathbf{x}_d$. Dans cette section, les techniques sont présentés pour analyser la stabilité de l'origine du système $\mathbf{x} = \mathbf{0}$. Cette focalisation sur l'origine n'est pas une restriction, mais une simplification notationnelle. Il est toujours possible de ramener l'étude d'un point d'équilibre quelconque à l'étude de l'origine via un changement de variable.

Théorème 20.3 Translation du Point d'Équilibre:

L'analyse de stabilité d'un système autonome $\dot{\mathbf{x}} = f(\mathbf{x})$ pour un point arbitraire $\mathbf{x}_d$ est équivalente à l'analyse de stabilité de l'origine d'un système autonome $\dot{\mathbf{x}}_e = f_e(\mathbf{x}_e)$ défini avec les transformation suivantes :

$$\mathbf{x}_e = \mathbf{x}_d - \mathbf{x} \tag{20.6}$$
$$\dot{\mathbf{x}}_e = \dot{\mathbf{x}}_d - \dot{\mathbf{x}} = -f(\mathbf{x}) \tag{20.7}$$

On peut donc toujours définir l'état du système comme l'erreur par rapport à la cible désirée, et ainsi tout les analyses peuvent viser l'origine sans perte de généralité.

20.2 Définitions de la stabilité

Comme établi à la section précédente, nous considérons ici la stabilité de l'origine $\mathbf{x} = \mathbf{0}$. L'analyse de stabilité vise à caractériser le comportement des trajectoires du système lorsqu'elles sont perturbées par rapport à cet équilibre. Trois définitions fondamentales sont typiquement utilisés :

Définition 20.4 Stabilité (au sens de Lyapunov):

L'équilibre $\mathbf{x} = \mathbf{0}$ est dit **stable** si, pour tout rayon $\epsilon > 0$, il existe un rayon $r > 0$ tel que :

$$\|\mathbf{x}(t_0)\| < r \implies \|\mathbf{x}(t)\| < \epsilon, \quad \forall t \geq t_0 \tag{20.8}$$

Interprétation : Si le système démarre proche de l'équilibre (dans la boule r), il ne s'éloignera jamais au-delà d'une certaine limite supérieure (la boule ϵ).

Définition 20.5 Stabilité Asymptotique:

L'équilibre est dit **asymptotiquement stable** s'il est stable (au sens de Lyapunov) et qu'il existe un rayon $r > 0$ tel que :

$$\|\mathbf{x}(t_0)\| < r \implies \lim_{t \to \infty} \mathbf{x}(t) = \mathbf{0} \tag{20.9}$$

Interprétation : Le système ne fait pas que rester borné, il finit par retourner exactement à l'équilibre. C'est la condition minimale requise pour la plupart des systèmes de commande en robotique, i.e. l'erreur tend vers zéro.

Définition 20.6 Stabilité Exponentielle:

L'équilibre est dit **exponentiellement stable** s'il existe des constantes positives $\alpha > 0$ et $\lambda > 0$ et qu'il existe un rayon $r > 0$ tel que : :

$$\|\mathbf{x}(t_0)\| < r \implies \|\mathbf{x}(t)\| \leq \alpha \|\mathbf{x}(t_0)\| e^{-\lambda(t-t_0)} \tag{20.10}$$

Interprétation : La convergence vers l'équilibre est rapide et bornée par une enveloppe exponentielle décroissante.

20.2.1 Stabilité Locale vs Globale

Dans les définitions ci-dessus, si conditions sont valides pour des états initiaux situés à l'intérieur d'un certain rayon r, on parle alors de **stabilité locale**.

Définition 20.7 Stabilité Globale:

Une propriété de stabilité est dite **globale** si elle est valide pour n'importe quelle condition initiale, aussi grande soit-elle. Autrement dit, si le rayon d'attraction est infini :

$$r \to \infty \tag{20.11}$$

20.3 Méthode Directe de Lyapunov

L'approche privilégiée pour analyser la stabilité des systèmes non-linéaires est appelée la *Méthode Directe de Lyapunov*. Le principe est de proposer une fonction scalaire $V(\mathbf{x})$, appelée *fonction candidate de Lyapunov*, qui représente une sorte d'énergie généralisée du système, et d'observer son évolution temporelle. L'idée de base est de vérifier que 1) l'origine du système est un minimum d'énergie, et 2) que l'énergie est toujours décroissante. Ces deux conditions permettre de conclure que le système va nécessairement converger vers l'origine. Quelques définitions sont nécessaires pour formaliser ces concepts :

Définition 20.8 Voisinage:

Un ensemble $\Omega \subset \mathbb{R}^n$ qui contient l'origine $\mathbf{x} = \mathbf{0}$ en son intérieur, est appelé un **voisinage de l'origine**.

$$\mathbf{0} \in \Omega \subseteq \mathbb{R}^n \tag{20.12}$$

C'est donc une région autour de l'origine dans l'espace d'état. Ce sous-ensemble est utile pour définir des propriétés locales.

Définition 20.9 Fonction Définie Positive:

Une fonction $V(\mathbf{x})$ est dite **définie positive** dans Ω si :

1. $V(\mathbf{0}) = 0$
2. $V(\mathbf{x}) > 0$ pour tout $\mathbf{x} \in \Omega$ tel que $\mathbf{x} \neq \mathbf{0}$.

*Si $V(\mathbf{x}) \geq 0$, la fonction est dite **semi-définie positive**.*

Définition 20.10 Fonction Définie Négative:

Une fonction $V(\mathbf{x})$ est dite **définie négative** dans Ω si :

1. $V(\mathbf{0}) = 0$
2. $V(\mathbf{x}) < 0$ pour tout $\mathbf{x} \in \Omega$ tel que $\mathbf{x} \neq \mathbf{0}$.

*Si $V(\mathbf{x}) \leq 0$ la fonction est dite **semi-définie négative**.*

Définition 20.11 Fonction Radialement Non-Bornée:

Une fonction $V(\mathbf{x})$ est dite **radialement non-bornée** si elle tend vers l'infini dans toutes les directions :

$$V(\mathbf{x}) \to \infty \quad \text{quand} \quad \|\mathbf{x}\| \to \infty \tag{20.13}$$

Cette propriété est nécessaire pour garantir la stabilité **globale**.

20.3.1 Systèmes Autonomes

Pour les systèmes autonomes $\dot{\mathbf{x}} = f(\mathbf{x})$, l'analyse repose sur le signe de la dérivée de V par rapport au temps :

$$\dot{V}(\mathbf{x}) = \frac{\partial V}{\partial \mathbf{x}} f(\mathbf{x}) \tag{20.14}$$

Théorème 20.12 Stabilité de Lyapunov (Locale):

Soit Ω un voisinage de l'origine. S'il existe une fonction scalaire $V(\mathbf{x})$ continûment différentiable telle que, pour tout $\mathbf{x} \in \Omega$:

1. $V(\mathbf{x})$ est **définie positive** ($V(\mathbf{0}) = 0$ et $V(\mathbf{x}) > 0$ ailleurs).
2. $\dot{V}(\mathbf{x})$ est **semi-définie négative** ($\dot{V}(\mathbf{x}) \leq 0$).

Alors l'origine est **stable au sens de Lyapunov**.

Théorème 20.13 Stabilité Asymptotique Locale:

Soit Ω un voisinage de l'origine. S'il existe une fonction scalaire $V(\mathbf{x})$ continûment différentiable telle que, pour tout $\mathbf{x} \in \Omega$:

1. $V(\mathbf{x})$ est **définie positive** ($V(\mathbf{0}) = 0$ et $V(\mathbf{x}) > 0$ ailleurs).
2. $\dot{V}(\mathbf{x})$ est **définie négative** ($\dot{V}(\mathbf{x}) < 0$).

Alors l'origine est **asymptotiquement stable**.

Théorème 20.14 Stabilité Asymptotique Globale:

S'il existe une fonction scalaire $V(\mathbf{x})$ continûment différentiable telle que, pour tout $\mathbf{x} \in \mathbb{R}^n$:

1. $V(\mathbf{x})$ est **définie positive** ($V(\mathbf{0}) = 0$ et $V(\mathbf{x}) > 0$ ailleurs).
2. $\dot{V}(\mathbf{x})$ est **définie négative** ($\dot{V}(\mathbf{x}) < 0$).
3. $V(\mathbf{x})$ est radialement non-bornée ($V(\mathbf{x}) \to \infty \quad \text{quand} \quad \|\mathbf{x}\| \to \infty$)

Alors l'origine est **globalement asymptotiquement stable**.

Le problème de la dérivée semi-définie Avec les systèmes mécaniques, la dérivée de l'énergie totale est souvent liée à la puissance dissipée par les frottements, et cette expression est souvent seulement semi-définie négative car elle s'annule dès que la vitesse est nulle, même si la position n'est pas à l'équilibre. Le théorème de Lyapunov simple ne permet pas de conclure à la convergence asymptotique (seulement à la stabilité). On peut toutefois utilise le principe d'invariance pour prouver une convergence asymptotique.

Définition 20.15 Ensemble Invariant:

Un ensemble M est dit invariant si toute trajectoire démarrant dans M reste dans M pour tout $t \geq 0$.

Théorème 20.16 Principe d'Invariance de LaSalle:

Pour un système autonome avec une fonction Lyapunov $V(\mathbf{x})$ définie positive et $\dot{V}(\mathbf{x}) \leq 0$. Si on défini un ensemble de point E où la dérivée de l'énergie s'annule :

$$E = \{\mathbf{x} \mid \dot{V}(\mathbf{x}) = 0\} \tag{20.15}$$

Alors, $\mathbf{x}$ converge vers M, le plus grand **ensemble invariant** contenu dans E.

Application : Si $\dot{V} = 0$ implique que l'accélération est non-nulle (le système ne peut pas rester là, donc ce n'est pas un ensemble invariant) sauf lorsque $\mathbf{x} = 0$, alors le plus grand ensemble invariant contenu dans E est simplement $\mathbf{x} = 0$. Dans ce cas on retrouve la même conclusion de stabilité asymptotique que l'on aurait obtenue avec la dérivée de l'énergie définie négative.

20.3.2 Systèmes Non-Autonomes

Pour les systèmes dépendant du temps $\dot{\mathbf{x}} = f(\mathbf{x}, t)$ on peut utiliser le Lemme de Barbalat pour analyser la convergence.

Théorème 20.17 Lemme de Barbalat:

Soit $f : [0, \infty) \to \mathbb{R}$ une fonction différentiable. Si les deux conditions suivantes sont respectées :

1. $f(t)$ possède une limite finie :
$$\lim_{t\to\infty} f(t) = L < \infty \tag{20.16}$$
2. $\dot{f}(t)$ est uniformément continue (condition souvent vérifiée en montrant que $\ddot{f}$ est bornée).

Alors, la dérivée converge vers zéro :
$$\lim_{t\to\infty} \dot{f}(t) = 0 \tag{20.17}$$

Ce résultat mathématique est utilisé pour formuler un théorème fondamental pour l'analyse des systèmes variables dans le temps, souvent appelé "Lyapunov-like Lemma" :

Théorème 20.18 Lemme de type Lyapunov:

Pour un système non-autonome, si une fonction scalaire $V(\mathbf{x}, t)$ satisfait les trois conditions suivantes :

1. $V(\mathbf{x}, t)$ est bornée inférieurement (souvent par 0) :
$$V(\mathbf{x}, t) \geq V_{min} \tag{20.18}$$
2. $\dot{V}(\mathbf{x}, t)$ est semi-définie négative :
$$\dot{V}(\mathbf{x}, t) \leq 0 \tag{20.19}$$
3. $\dot{V}(\mathbf{x}, t)$ est uniformément continue (souvent vérifié si $\ddot{V}$ est bornée).

Alors, la dérivée de la fonction de Lyapunov converge vers zéro :
$$\lim_{t\to\infty} \dot{V}(\mathbf{x}, t) = 0 \tag{20.20}$$

20.4 Passivité

Intuitivement, un système est **passif** s'il ne génère pas d'énergie interne. Il ne peut que stocker l'énergie qu'on lui fournit ou la dissiper. Cette propriété est utile car la mise en connexion (de systèmes passifs) conserve cette propriété. On peut donc utiliser ce concept pour conclure sur la stabilité de plusieurs systèmes interconnecté sans connaître le détail de certains sous-système sauf qu'ils ont la propriété d'être passif.

20.4.1 Définitions Formelles

Considérons un système dynamique avec un état $\mathbf{x}$, une entrée $\mathbf{u}$ et une sortie $\mathbf{y}$.

Définition 20.19 Système Passif:

Un système est dit **passif** s'il existe une fonction scalaire continue $S(\mathbf{x}) \geq 0$, appelée **fonction de stockage**, telle que $\forall t \geq t_0$:
Forme intégrale :

$$\underbrace{S(\mathbf{x}(t)) - S(\mathbf{x}(t_0))}_{\text{Énergie stockée}} \leq \underbrace{\int_{t_0}^{t} \mathbf{y}(\tau)^T \mathbf{u}(\tau) d\tau}_{\text{Énergie fournie}} \tag{20.21}$$

Forme différentielle :

$$\dot{S}(\mathbf{x}) \leq \mathbf{y}^T \mathbf{u} \tag{20.22}$$

Interprétation : Le taux de variation de l'énergie interne est inférieur à la puissance fournie. La différence est l'énergie dissipée.

Il existe des variantes plus strictes de la passivité, utiles pour prouver la convergence asymptotique.

Définition 20.20 Variantes de la Passivité:

Soit $S(\mathbf{x})$ une fonction de stockage.

— **Strictement Passif :** S'il existe une fonction définie positive $\psi(\mathbf{x})$ (dissipation interne) telle que :

$$\dot{S}(\mathbf{x}) \leq \mathbf{y}^T \mathbf{u} - \psi(\mathbf{x}) \tag{20.23}$$

— **Strictement Passif en Sortie :** S'il existe $\varepsilon > 0$ tel que :

$$\dot{S}(\mathbf{x}) \leq \mathbf{y}^T \mathbf{u} - \varepsilon \|\mathbf{y}\|^2 \tag{20.24}$$

— **Sans Perte :** Si l'égalité stricte est respectée :

$$\dot{S}(\mathbf{x}) = \mathbf{y}^T \mathbf{u} \tag{20.25}$$

Exemple : Un système purement inertiel (masse pure) ou purement conservatif (ressort).

20.4.2 Lien avec Lyapunov

Il existe un lien direct entre passivité et stabilité au sens de Lyapunov. La fonction de stockage $S(\mathbf{x})$, qui représente l'énergie interne, peut être vue comme une fonction candidate de Lyapunov. Si un système est passif, alors en l'absence d'apport d'énergie extérieur (entrée $\mathbf{u} = \mathbf{0}$), l'énergie stockée ne peut pas augmenter :

$$\mathbf{u} = \mathbf{0} \implies \dot{S}(\mathbf{x}) \leq 0 \tag{20.26}$$

Ainsi, **tout système passif est stable** (au sens de Lyapunov) si sa fonction de stockage est définie positive.

20.4.3 Passivité des systèmes linéaires (LTI)

Pour les systèmes linéaires, la passivité peut être vérifiée directement dans le domaine fréquentiel, sans chercher explicitement la fonction $S(\mathbf{x})$.

Théorème 20.21 Passivité d'un système LTI:

Pour un système LTI stable avec une matrice de fonction de transferts $H(s)$, le système est passif si et seulement si :

$$H(j\omega) + H(j\omega)^H \geq 0 \quad \forall \omega \in \mathbb{R}$$

où $H(j\omega)^H$ est la transposée hermitienne de $H(j\omega)$.

Théorème 20.22 Passivité d'une fonction de transfert:

Pour un système linéaire stable de fonction de transfert $H(s)$. Le système est passif si et seulement si :

$$\text{Re}\{H(j\omega)\} \geq 0, \quad \forall \omega \in \mathbb{R} \tag{20.27}$$

Cela signifie que que la phase de $H(j\omega)$ reste bornée à $\pm$ 90 degrées :

$$|\arg(H(j\omega))| \leq \frac{\pi}{2} \tag{20.28}$$

Chapitre 21

Optimisation

21.1 Formulation du problème

Tout problème d'optimisation peut être ramené à la forme canonique suivante :

Définition 21.1 Formulation d'un problème d'optimisation:

Soit $\boldsymbol{x} \in \mathbb{R}^n$ le vecteur des variables de décision. On cherche :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ \text{sujet à} \quad & \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0} \\ & \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0} \end{aligned}$$

Où $f(\boldsymbol{x})$ est la fonction objectif, $\boldsymbol{g}(\boldsymbol{x})$ sont les contraintes d'inégalité et $\boldsymbol{h}(\boldsymbol{x})$ les contraintes d'égalité.

Pour bien définir un problème, il est essentiel d'identifier ses cinq composantes :

- **Variables de décision ($\boldsymbol{x}$) :** Les paramètres que l'algorithme a la liberté de modifier pour trouver la meilleure solution (ex : les gains d'un PID, les points de passage d'une trajectoire).
- **Espace de recherche ($\boldsymbol{x} \in \mathcal{X}$) :** Le domaine mathématique dans lequel évolue $\boldsymbol{x}$.
- **Fonction objectif (f) :** Le critère de performance. Elle transforme un vecteur de décision en un scalaire unique permettant de comparer deux solutions.
- **Contraintes d'inégalité ($\boldsymbol{g}$) :** Elles définissent les limites pour les variables décisions (ex : ne pas dépasser le couple maximal du moteur, éviter un obstacle).
- **Contraintes d'égalité ($\boldsymbol{h}$) :** Elles imposent des relations strictes (ex : le robot doit terminer exactement sur la cible, respecter les lois de la physique $f = ma$).

21.1.1 Coût minimum vs récompense maximum

Bien que la forme canonique soit présentée comme une **minimisation**, il est simple de transformer un problème de **maximisation** sous cette forme canonique de minimisation :

Théorème 21.2 Max vs min:

La solution $\boldsymbol{x}^*$ qui maximise une fonction de récompense $V(\boldsymbol{x})$ est la même solution qui minimise son opposé :

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}} V(\boldsymbol{x}) = \arg\min_{\boldsymbol{x}} -V(\boldsymbol{x})$$

21.1.2 Typologie des problèmes et situations possibles

La difficulté de résolution d'un problème d'optimisation dépend de plusieurs facteurs critiques :

- **Connaissance de f (Analytique vs Échantillonnage) :** Dans certains cas, on possède l'équation exacte de f. Dans d'autres, f est une « boîte noire » dont on ne peut qu'échantillonner le résultat par des simulations ou des tests physiques coûteux.
- **Disponibilité des dérivées :** Les algorithmes les plus rapides utilisent le gradient (∇f) et la Hessienne ($\nabla^2 f$). Si ces dérivées sont inconnues ou impossibles à calculer, on doit se rabattre sur des méthodes « sans gradient », souvent beaucoup plus lentes.
- **Dimensionnalité ($\boldsymbol{x} \in \mathbb{R}^n$) :** Un problème à basse dimension (ex. optimiser les gains d'une loi de commande : $n < 10$) est plus facile à résoudre qu'un problème à haute dimension (ex. entraîner les poids d'un réseau de neurones : $n > 1000000$)
- **Convexité :** C'est la distinction la plus importante. Si le problème est **convexe**, on a la garantie de trouver le minimum global rapidement, si le problème est **non-convexe**, on risque de rester bloqué dans un minimum local.
- **Propriétés des fonctions :**
  - **Programmation Linéaire :** f, g, h sont des droites. Très simple à résoudre.
  - **Programmation Quadratique :** f est une parabole, g, h sont linéaires. Souvent on approxime le problème pour retrouver cette forme et avoir des garanties de trouver des solutions rapidement. C'est le standard pour la commande prédictive (MPC).
  - **Programmation Non-linéaire :** Le cas général en robotique (à cause de la trigonométrie et de la dynamique), nécessitant des méthodes itératives complexes.

Lorsque que la nature des variables décision est discrète, on appelle le problème la **programmation d'entiers**. La nature combinatoire complexifie beaucoup la résolution de ce type de problème. Cette annexe n'abordera pas cette catégorie d'optimisation, on va ce concentrer sur des variables décisions de nature continue.

Finalement, la **programmation dynamique**, est une autre classe de problème d'optimisation de nature légèrement différente : on cherche à optimiser/choisir une fonction plutôt que des variables décisions. Ce chapitre de traite pas cette classe d'optimisation. On vous réfère à un recueil de notes dédié à ce sujet : `https://www.alexandregirard.com/teaching/dp/PDF/DP_Notes.pdf`

21.2 Définitions importantes

Avant de classifier les problèmes d'optimisation, nous devons définir des propriétés de l'espace dans lequel nous cherchons nos solutions, ainsi que des propriétés des fonctions f,g et h.

21.2.1 Solutions

On va souvent parlé de minimum local vs global, ce qui est défini par :

Définition 21.3 Minimum local vs global:

- **Minimum local :** $\boldsymbol{x}^*$ est le meilleur point dans son voisinage immédiat.
- **Minimum global :** $\boldsymbol{x}^*$ est le meilleur point sur tout l'ensemble possible.

21.2.2 Propriété de l'espace des variables décisions

L'existence d'une solution, pour un problème d'optimisation, va dépendre d'une propriété pour l'espace des variable de décision appelée la compacité, i.e. est-ce que l'ensemble des x qui respectent les contraintes est compacte.

Définition 21.4 Ensembles ouverts, fermés et bornés:

Soit un ensemble $S \subset \mathbb{R}^n$:

- **Ouvert :** S ne contient pas sa frontière (ex : une contrainte de type $x < 5$).
- **Fermé :** S contient sa frontière (ex : une contrainte de type $x \leq 5$).
- **Borné :** S peut être contenu dans une sphère de rayon fini (il ne s'étend pas à l'infini).

Définition 21.5 Ensemble compact:

Un ensemble est dit **compact** s'il est à la fois **fermé** et **borné**.

Exemple 21.1 Intervalle compact vs non-compact:

Sur la droite réelle $\mathbb{R}$:
- $[0, 10]$ est un ensemble **compact** (fermé et borné).
- $[0, \infty[$ n'est pas compact (non borné).
- $]0, 10[$ n'est pas compact (ouvert).

Ensuite, les garanties que la solution trouvée est globale ou locale va dépendre d'un propriété pour l'espace des variable de décision appelée la convexité.

Définition 21.6 Ensemble convexe dans $\mathbb{R}^n$:

Un ensemble $C \subseteq \mathbb{R}^n$ est dit **convexe** si, pour tout couple de points $(\boldsymbol{x}, \boldsymbol{y}) \in C$, le segment les reliant est entièrement contenu dans C :

$$\forall \alpha \in [0, 1], \quad \alpha \boldsymbol{x} + (1 - \alpha)\boldsymbol{y} \in C$$

Exemple 21.2 Géométrie de la convexité:

- **Convexe :** Un cercle plein, un carré, un hexagone.
- **Non-convexe :** Une forme en "L", un anneau (le centre est vide), un croissant de lune.

21.3 Classes de fonctions

La nature de la fonction de coût $f(\boldsymbol{x})$ et des contraintes détermine la difficulté du problème. La propriété principale est la convexité d'une fonction.

Définition 21.7 Fonction convexe sur $\mathbb{R}^n$:

Soit $f : C \to \mathbb{R}$ une fonction définie sur un ensemble convexe $C \subseteq \mathbb{R}^n$. La fonction f est **convexe** si :

$$\forall (\boldsymbol{x}, \boldsymbol{y}) \in C, \forall \alpha \in [0, 1] : f(\alpha \boldsymbol{x} + (1 - \alpha)\boldsymbol{y}) \leq \alpha f(\boldsymbol{x}) + (1 - \alpha) f(\boldsymbol{y})$$

Autrement dit, la ligne sécante entre deux points sur la courbe de la fonction est toujours supérieure à la fonction.

Exemple 21.3 Convexité vs Non-convexité:

- **Convexe :** $f(x) = x^2$, $f(x) = e^x$, $f(x) = |x|$.
- **Non-convexe :** $f(x) = \sin(x)$, $f(x) = x^3$.

Définition 21.8 Fonctions linéaires et affines:

Soit une fonction $f : \mathbb{R}^n \to \mathbb{R}$:
- **Linéaire :** $f(\boldsymbol{x}) = \boldsymbol{c}^T \boldsymbol{x}$. Elle satisfait les propriétés d'additivité et d'homogénéité ($f(\alpha \boldsymbol{x} + \beta \boldsymbol{y}) = \alpha f(\boldsymbol{x}) + \beta f(\boldsymbol{y})$).
- **Affine :** $f(\boldsymbol{x}) = \boldsymbol{c}^T \boldsymbol{x} + d$. C'est une fonction linéaire décalée par une constante d. Elle ne passe pas par l'origine si $d \neq 0$.

Définition 21.9 Fonction quadratique:

Une fonction $f : \mathbb{R}^n \to \mathbb{R}$ est dite **quadratique** si elle peut s'écrire sous la forme :

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} + r$$

où $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ est la matrice Hessienne (généralement supposée symétrique), $\boldsymbol{c} \in \mathbb{R}^n$ est le vecteur gradient linéaire et $r \in \mathbb{R}$ est une constante.

Définition 21.10 Fonctions lisses (C^k):

Une fonction est dite de classe C^k si ses dérivées jusqu'à l'ordre k existent et sont continues.
- C^0 : Fonction continue.
- C^1 : Dérivable, avec un gradient $\nabla f(\boldsymbol{x})$ continu.
- C^2 : Deux fois dérivable, avec une Hessienne $\nabla^2 f(\boldsymbol{x})$ continue.

Exemple 21.4 Continuité d'une trajectoire:

n robotique, si on on s'intéresse à une fonction qui donne la position en fonction du temps :
- C^0 : Fonction continue (pas de téléportation du robot).
- C^1 : Vitesse continue (pas de saut de vitesse, donc accélération finie).
- C^2 : Accélération continue (pas de saut d'accélération, donc jerk fini).

21.4 Optimalité

21.4.1 Locale

Définition 21.11 Le Lagrangien:

Le Lagrangien $\mathcal{L}$ associé au problème 21.1 est défini par :

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = f(\boldsymbol{x}) + \boldsymbol{\lambda}^T \boldsymbol{g}(\boldsymbol{x}) + \boldsymbol{\nu}^T \boldsymbol{h}(\boldsymbol{x})$$

où $\boldsymbol{\lambda} \geq \mathbf{0}$ et $\boldsymbol{\nu}$ sont les vecteurs de multiplicateurs.

Théorème 21.12 Conditions de Karush-Kuhn-Tucker (KKT):

Pour qu'un point $\boldsymbol{x}^*$ soit un minimum local, il doit exister $\boldsymbol{\lambda}^*$ et $\boldsymbol{\nu}^*$ tels que :

1. $\nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\nu}^*) = \mathbf{0}$ (Stationnarité)
2. $\boldsymbol{h}(\boldsymbol{x}^*) = \mathbf{0}$ et $\boldsymbol{g}(\boldsymbol{x}^*) \leq \mathbf{0}$ (Primalité)
3. $(\boldsymbol{\lambda}^*)^T g(\boldsymbol{x}^*) = \mathbf{0}$ (Complémentarité)

21.4.2 Globale

Théorème 21.13 Propriété fondamentale de la convexité:

Si la fonction de coût $f(\boldsymbol{x})$ est **convexe** et que l'ensemble des contraintes est **convexe**, alors tout **minimum local est un minimum global**.

21.4.3 Existence

Les conditions suivantes doivent être respectées pour avoir la garantie mathématique qu'une solution optimale existe. Concrètement, un algorithme numérique pourrait chercher indéfiniment sans jamais converger si l'une de ces conditions fait défaut.

Théorème 21.14 Théorème de Weierstrass:

Soit $f : S \to \mathbb{R}$ une fonction définie sur un ensemble $S \subseteq \mathbb{R}^n$. Si les deux conditions suivantes sont satisfaites :

1. f est une fonction **continue** sur S,
2. S est un ensemble **compact** (à la fois fermé et borné),

alors la fonction f atteint son minimum global et son maximum global sur S. Autrement dit, il existe au moins un point $\boldsymbol{x}^* \in S$ tel que $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$ pour tout $\boldsymbol{x} \in S$.

21.5 Classes de problèmes

La complexité de résolution, et donc le type d'algorithme à utiliser, dépend de la forme des fonctions f, $\boldsymbol{g}$ et $\boldsymbol{h}$. Voici les trois classes principales de problèmes qui dictent le choix de la méthode numérique.

21.5.1 Programmation Linéaire (LP)

La programmation linéaire est la classe la plus simple et la plus robuste. Elle est caractérisée par une linéarité totale des relations.

Définition 21.15 Programmation Linéaire (LP):

Un problème est dit **LP** si la fonction objectif et toutes les contraintes sont des fonctions affines :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \boldsymbol{c}^T\boldsymbol{x} \\ \text{sujet à} \quad & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{x} \geq \boldsymbol{0} \end{aligned}$$

Un problème LP est toujours convexe. La solution optimale se trouve nécessairement sur l'un des sommets (coins) du polyèdre formé par les contraintes.

21.5.2 Programmation Quadratique (QP)

C'est la classe la plus utilisée en robotique et en commande optimale.

Définition 21.16 Programmation Quadratique (QP):

Un problème est dit **QP** si la fonction objectif est quadratique et les contraintes sont affines :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \frac{1}{2}\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{c}^T\boldsymbol{x} \\ \text{sujet à} \quad & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \end{aligned}$$

Avec la matrice $\boldsymbol{Q}$ semi-définie positive ($\boldsymbol{Q} \succeq 0$) pour un problème convexe.

— **Propriété :** La solution est alors globale, unique et peut être trouvée très rapidement (souvent en quelques millisecondes).

21.5.3 Programmation Non-Linéaire (NLP)

C'est la classe la plus générale. Elle regroupe tous les problèmes où les relations physiques ne peuvent pas être simplifiées en formes linéaires ou quadratiques.

Définition 21.17 Programmation Non-Linéaire (NLP):

Un problème est dit **NLP** si au moins une des fonctions ($f, \boldsymbol{g}$ ou $\boldsymbol{h}$) est non-linéaire (présence de fonctions trigonométriques, de produits de variables, etc.) :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ \text{sujet à} \quad & \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0} \\ & \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0} \end{aligned}$$

21.6 Solutions analytiques

Cette section présente les situations privilégiées où il existe une solution explicite à un problème d'optimisation, évitant ainsi le recours à des algorithmes itératifs.

21.6.1 Multiplicateurs de Lagrange

Capsule vidéo
Multiplicateurs de Lagrange
https://youtu.be/W7kzp927xXc

Définition 21.18 Problème : Optimisation sous contraintes d'égalité:

On cherche à minimiser une fonction sous un ensemble de contraintes strictes :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & f(\boldsymbol{x}) \\ \text{sujet à} \quad & \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0} \end{aligned}$$

Théorème 21.19 Solution : Le Lagrangien:

On définit le Lagrangien $\mathcal{L}$ comme :

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\nu}) = f(\boldsymbol{x}) + \boldsymbol{\nu}^T \boldsymbol{h}(\boldsymbol{x})$$

Les points candidats à l'optimum satisfont le système d'équations :

$$\nabla_{\boldsymbol{x}} \mathcal{L} = \boldsymbol{0} \quad \text{et} \quad \nabla_{\boldsymbol{\nu}} \mathcal{L} = \boldsymbol{0}$$

21.6.2 Objectif quadratique avec contraintes d'égalité

Définition 21.20 Problème : QP avec égalités:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \frac{1}{2} \boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x} + \boldsymbol{c}^T \boldsymbol{x} \\ \text{sujet à} \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

Théorème 21.21 Solution analytique directe:

La solution optimale au problème QP avec égalité est :

$$\boldsymbol{x}^* = -\boldsymbol{Q}^{-1}\boldsymbol{c} + \boldsymbol{Q}^{-1}\boldsymbol{A}^T \left(\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T \right)^{-1} \left(\boldsymbol{b} + \boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{c} \right)$$

21.6.3 Moindres carrés

Définition 21.22 Problème : Moindres carrés:

On cherche à minimiser l'erreur quadratique d'un système surdéterminé :

$$\min_{\boldsymbol{x}} \quad f(\boldsymbol{x}) = \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2$$

Théorème 21.23 Solution des moindres carrés:

La solution unique (si $\boldsymbol{A}$ est de plein rang) est :

$$\boldsymbol{x}^* = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{b}$$

21.6.4 Moindres carrés pondérés

Définition 21.24 Problème : Moindres carrés pondérés:

On accorde une importance différente à chaque mesure via une matrice de poids $\boldsymbol{W} \succ 0$:

$$\min_{\boldsymbol{x}} \quad f(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})^T\boldsymbol{W}(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})$$

Théorème 21.25 Solution analytique:

La solution intègre la matrice de pondération :

$$\boldsymbol{x}^* = (\boldsymbol{A}^T\boldsymbol{W}\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{W}\boldsymbol{b}$$

21.6.5 Moindres carrés régularisés

Définition 21.26 Problème : Régularisation:

On cherche un compromis entre la précision et la norme de la solution :

$$\min_{\boldsymbol{x}} \quad f(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|^2 + \frac{\lambda}{2}\|\boldsymbol{x}\|^2$$

Théorème 21.27 Solution : Inversion amortie:

$$\boldsymbol{x}^* = (\boldsymbol{A}^T\boldsymbol{A} + \lambda\boldsymbol{I})^{-1}\boldsymbol{A}^T\boldsymbol{b}$$

21.6.6 Solution de norme minimale (Redondance)

Définition 21.28 Problème : Norme Minimale:

Pour un système ayant plus de degrés de liberté que de contraintes, on cherche la solution avec la plus petite norme :

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \frac{1}{2}\|\boldsymbol{x}\|^2 \\ \text{sujet à} \quad & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \end{aligned}$$

Théorème 21.29 Solution : Pseudo-inverse droite:

Si $\boldsymbol{A}$ est de plein rang ligne, la solution est :

$$\boldsymbol{x}^* = \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T)^{-1}\boldsymbol{b}$$

21.7 Méthodes de résolution numériques

Lorsque le problème ne possède pas de solution analytique, on utilise des algorithmes itératifs. Le principe général est de générer une suite de points $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha\Delta\boldsymbol{x}_k$ qui converge vers l'optimum, où $\Delta\boldsymbol{x}_k$ est la direction de recherche et α est le pas de calcul.

21.7.1 Optimisation sans contraintes

La méthode de base est de suivre le gradient de la fonction :

Définition 21.30 Descente de gradient (Ordre 1):

La direction de recherche est l'opposé du gradient local, soit la direction de la plus forte pente descendante :

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \alpha\nabla f(\boldsymbol{x}_k)$$

C'est très robuste et simple à implémenter, mais peut être très lent car il est dure d'avoir un paramètre α qui convient à toutes les situations. Une alternative est de calculer le pas en fonction de la dérivée seconde :

Définition 21.31 Méthode de Newton (Ordre 2):

Cette méthode utilise la courbure locale (Hessienne) pour corriger la direction de descente et viser directement le minimum d'une approximation quadratique :

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \alpha\left[\nabla^2 f(\boldsymbol{x}_k)\right]^{-1}\nabla f(\boldsymbol{x}_k)$$

Cette méthode a un convergence rapide (quadratique) près de l'optimum, mais nécessite le calcul et l'inversion de la Hessienne à chaque itération.

Définition 21.32 Méthodes Quasi-Newton (BFGS):

Ces méthodes imitent le comportement de Newton sans calculer explicitement la Hessienne. On construit une approximation itérative $\boldsymbol{B}_k \approx \nabla^2 f$ à partir des variations successives du gradient lors des itérations précédentes :

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \alpha B_k^{-1}\nabla f(\boldsymbol{x}_k) \tag{21.1}$$

$$\text{avec} \quad B_k \approx \nabla^2 f \tag{21.2}$$

C'est le standard quand le calcul analytique de la Hessienne est trop complexe ou coûteux.

21.7.2 Optimisation avec contraintes

Pour traiter les contraintes $\boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}$ et $\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}$, on utilise des stratégies de transformation ou de sous-problèmes :

Définition 21.33 Méthodes de Points Intérieurs (Barrière):

On remplace les contraintes d'inégalité par des fonctions barrières (souvent logarithmiques) intégrées directement au coût :

$$\min_{\boldsymbol{x}} \quad f(\boldsymbol{x}) - \mu\sum_i \ln(-g_i(\boldsymbol{x}))$$

L'algorithme reste à l'intérieur de la zone admissible. Très efficace pour les problèmes de grande taille (ex : solveur **IPOPT**).

Définition 21.34 Programmation Quadratique Successive (SQP):

À chaque itération, l'algorithme résout un sous-problème de type **QP** (Quadratic Programming) en linéarisant les contraintes et en approximant le coût par une forme quadratique locale.

Méthode de référence pour l'optimisation de trajectoires complexes en robotique nécessitant une grande précision sur les contraintes (ex : solveur **SNOPT**).

21.7.3 Méthodes de recherche globale

Lorsque le problème est fortement non-convexe, les méthodes de descente (ordre 1 ou 2) convergent vers le minimum local le plus proche du point initial. Les méthodes globales visent à explorer l'ensemble de l'espace pour trouver le véritable optimum global. Toutefois, contrairement aux méthodes locales qui convergent en quelques itérations, les méthodes globales nécessitent souvent des milliers, voire des millions d'évaluations de la fonction objectif. Elle sont donc significativement plus lente que les méthodes basées sur la descente d'un gradient.

Définition 21.35 Méthodes Départs-Multiples:

C'est la stratégie la plus simple : on lance plusieurs optimisations locales (ex : Newton ou Gradient) à partir de points initiaux $\boldsymbol{x}_0$ distribués aléatoirement dans l'espace de recherche. On conserve ensuite le meilleur résultat obtenu. **Usage :** Très facile à paralléliser sur des processeurs multi-cœurs.

Définition 21.36 Algorithmes Évolutionnaires :

Ces méthodes s'inspirent de la sélection naturelle. Elles gèrent une population de points et utilisent des mécanismes de mutation et de sélection pour déplacer cette population vers les zones de bas coût. **Usage :** Très efficace pour l'optimisation "boîte noire" (*black-box*) où le gradient n'est pas disponible ou trop bruité.

Définition 21.37 Recuit Simulé :

Inspirée de la métallurgie, cette méthode accepte parfois des mouvements "vers le haut" (augmentant le coût) pour s'extraire d'un minimum local. Cette probabilité diminue avec le temps (la "température"). **Usage :** Utile pour les problèmes d'optimisation combinatoire ou de grande dimension avec de nombreux creux locaux.

Quatrième partie

Exercices

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Annexe A

Cinématique

A.1 Intro cinématique

1. **Addition de vecteurs** Les vecteurs $\vec{u}$, $\vec{v}$ et $\vec{w}$ ont les composantes suivantes dans la base b :

$$\boldsymbol{u}^b = \begin{bmatrix} 2 \\ -1 \end{bmatrix} \quad \boldsymbol{v}^b = \begin{bmatrix} 5 \\ 2 \end{bmatrix} \quad \boldsymbol{w}^b = \begin{bmatrix} 0 \\ 3 \end{bmatrix} \tag{A.1}$$

   Effectuez le calcul $2\vec{u} + \vec{v} - \vec{w}$ graphiquement, par algèbre de vecteurs et avec les équations matricielles.
2. **Angle entre deux vecteurs à l'aide du produit scalaire** Calculez l'angle entre les deux vecteurs avec les composantes suivantes dans une base commune :

$$\boldsymbol{a} = \begin{bmatrix} \sqrt{3} \\ 1 \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \tag{A.2}$$

3. **Mitiguy, exercice 4.9 page 195** (création d'une matrice de rotation pour changement de bases)
4. **Mitiguy, exercice 4.13 page 198** (création d'une matrice de rotation pour changement de bases)

A.2 Traduire des énoncés en équations vectorielles

Convertissez les énoncés suivant en équations vectorielles avec des scalaires qui représentent des distances et des vecteur unitaires qui représentent des directions.

A) À partir d'un point A, on avance de 100m vers le nord, 15m vers l'ouest et 50m vers le sud-est, pour arriver sur le point B.

B) À partir d'un point A, on avance de 50m vers le sud, dévie de 30 degrés vers la gauche (bâbord) et avance de 100m, pour arriver sur le point B.

C) À partir d'un point A, on avance de 200m vers le l'est, dévie de 60 degrés vers la droite (tribord) et avance de 50 m. Ensuite, on s'oriente à l'opposé de la direction du point A et on double la distance par rapport au point A pour arriver sur le point B.

A.3 Substitution de vecteurs unitaires

Convertissez les équations vectorielles obtenues à l'exercice A.2 pour seulement contenir les deux vecteurs unitaires suivants :

$$\hat{n} : Nord \qquad \hat{o} : Ouest \tag{A.3}$$

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

A.4 Obtenir les composantes avec un produit scalaire

Le vecteur $\vec{v}$ est définie dans une base vectorielle orthogonale a :

$$\vec{v} = v_1\hat{a}_1 + v_2\hat{a}_2 + v_3\hat{a}_3 \tag{A.4}$$

démontrez que

$$v_1 = \vec{v} \cdot \hat{a}_1 \tag{A.5}$$
$$v_2 = \vec{v} \cdot \hat{a}_2 \tag{A.6}$$
$$v_3 = \vec{v} \cdot \hat{a}_3 \tag{A.7}$$

en se basant sur les définitions.

A.5 Matrice de rotation calculée avec toutes les définitions

Effectuer le calcul de la matrice R_3 discutée à la section 3.5.3 :

A) À partir de la définition basée sur les produit scalaires :

$$R_b^a = \begin{bmatrix} \hat{a}_1 \cdot \hat{b}_1 & \hat{a}_1 \cdot \hat{b}_2 & \hat{a}_1 \cdot \hat{b}_3 \\ \hat{a}_2 \cdot \hat{b}_1 & \hat{a}_2 \cdot \hat{b}_2 & \hat{a}_2 \cdot \hat{b}_3 \\ \hat{a}_3 \cdot \hat{b}_1 & \hat{a}_3 \cdot \hat{b}_2 & \hat{a}_3 \cdot \hat{b}_3 \end{bmatrix} \tag{A.8}$$

B) À partir de la définition basée sur l'angle entre les vecteurs unitaires :

$$R_b^a = \begin{bmatrix} \cos\angle(\hat{a}_1, \hat{b}_1) & \cos\angle(\hat{a}_1, \hat{b}_2) & \cos\angle(\hat{a}_1, \hat{b}_3) \\ \cos\angle(\hat{a}_2, \hat{b}_1) & \cos\angle(\hat{a}_2, \hat{b}_2) & \cos\angle(\hat{a}_2, \hat{b}_3) \\ \cos\angle(\hat{a}_3, \hat{b}_1) & \cos\angle(\hat{a}_3, \hat{b}_2) & \cos\angle(\hat{a}_3, \hat{b}_3) \end{bmatrix} \tag{A.9}$$

C) À partir de la définition basée sur les composants des vecteurs unitaires $\hat{b}_i$ dans la base a :

$$R_b^a = \begin{bmatrix} \boldsymbol{b}_1^a & \boldsymbol{b}_2^a & \boldsymbol{b}_3^a \end{bmatrix} \tag{A.10}$$

D) À partir de la définition basée sur les composants des vecteurs unitaires $\hat{a}_i$ dans la base b :

$$R_b^a = \begin{bmatrix} (\boldsymbol{a}_1^b)^T \\ (\boldsymbol{a}_2^b)^T \\ (\boldsymbol{a}_3^b)^T \end{bmatrix} \tag{A.11}$$

E) À partir de la définition basée sur les angles de Euler :

$$R_b^a = \begin{bmatrix} +c\phi\, c\psi - s\phi\, c\theta\, c\psi & -c\phi\, s\psi - s\phi\, c\theta c\psi & +s\phi\, s\theta \\ +s\phi\, c\psi + c\phi\, c\theta\, c\psi & -s\phi\, s\psi + c\phi\, c\theta c\psi & -c\phi\, s\theta \\ +s\theta\, s\psi & +s\theta\, c\psi & +c\theta \end{bmatrix} \tag{A.12}$$

A.6 Axe de rotation à partir d'une matrice de rotation

La figure A.1 illustre deux base vectorielle avec une orientation relative décrite par la matrice ${}^aR^b$.

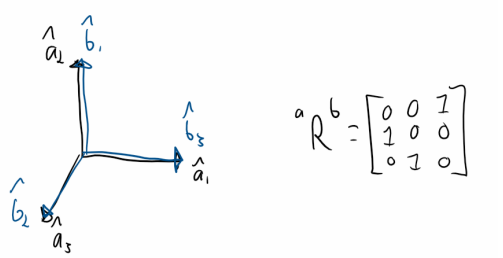


FIGURE A.1 – Calcul d'un axe de rotation à partir d'une matrice

A) Vérifiez que la matrice de rotation illustrée ci-dessus a une valeur propre réel $\lambda = 1$.

B) Calculez le vecteur propre associé.

C) Vérifiez graphiquement que le vecteur propre calculé correspond à l'axe de rotation.

A.7 Fonction pour un changement de repère

Un avion détecte un ovni et sa position dans son repère $\{V_o, \hat{v}_1, \hat{v}_2, \hat{v}_2\}$, il doit transmettre les coordonnées de cet ovni par rapport au repère $\{B_o, \hat{b}_1, \hat{b}_2, \hat{b}_2\}$ de la base, voir figure A.2. Écrivez une fonction, dans votre langage de programmation préféré, pour effectuer le calcul :

$$\boldsymbol{r}^b_{Ovni/B_o} = f\left(\boldsymbol{r}^v_{Ovni/V_o}, \theta, h, d\right) \tag{A.13}$$

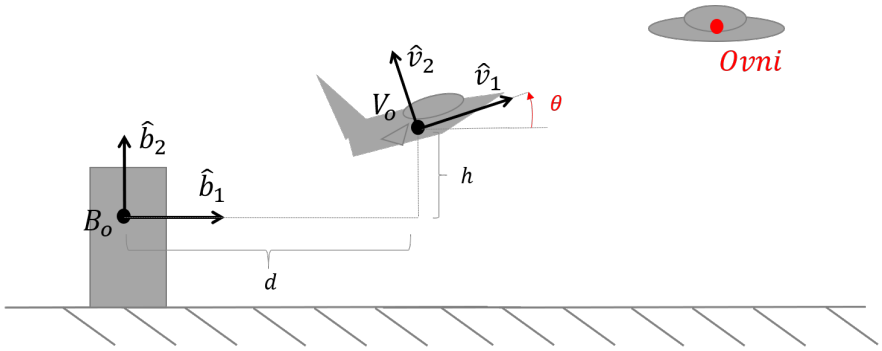


FIGURE A.2 – Calcul d'un changement de repère

A.8 Calcul de la cinématique d'un robot à deux joints

La figure A.3 illustre un robot planaire à deux joints.

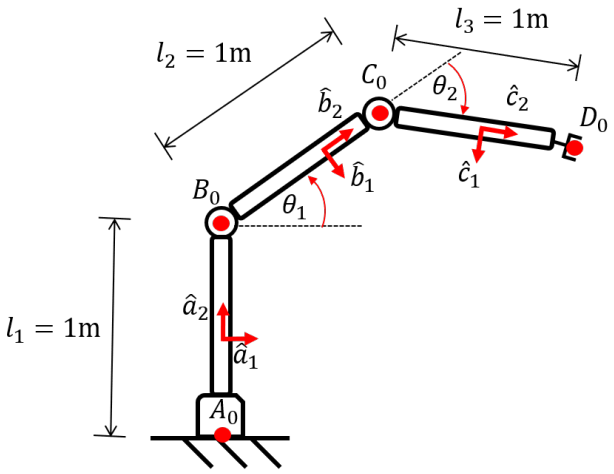


FIGURE A.3 – Calcul de la cinématique d'un robot à deux joints

A) Tracez le robot (ligne reliant les points A_o, B_o, C_o et D_o) lorsque la configuration des joints est (en Radians) :

$$\theta_1 = 0.3 \tag{A.14}$$
$$\theta_2 = -1.6 \tag{A.15}$$

B) Tracez la trajectoire de l'effecteur (point D_o) lorsque les joints bougent selon les fonctions temporelles (en Radians) :

$$\theta_1(t) = 0.5\sin(t) + 0.3 \tag{A.16}$$
$$\theta_2(t) = 0.3\sin(2t) - 1.6 \tag{A.17}$$

A.9 Calcul de la position de l'effecteur d'un robot

La figure A.4 illustre un robot avec quatre joints rotatifs.

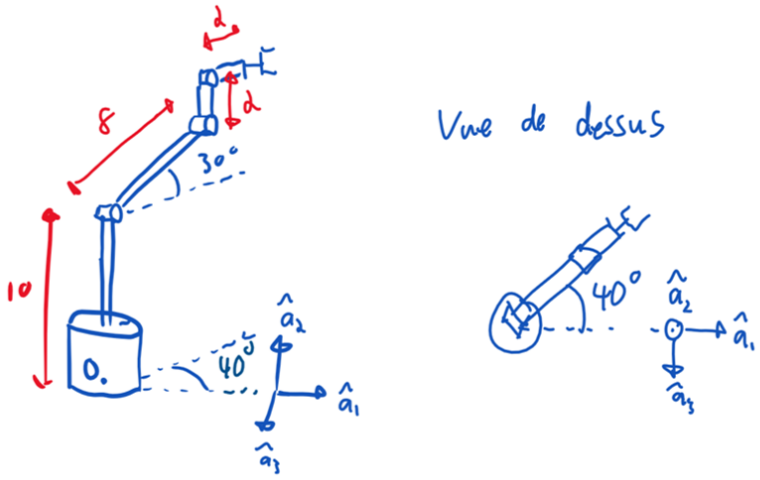


FIGURE A.4 – Calcul de la cinématique d'un robot en 3D

A) Identifier des points et bases vectorielles intermédiaires à utiliser.

B) Trouver le vecteur position de l'effecteur par rapport à l'origine O, exprimé dans la base vectorielle a.

A.10 Calcul de la cinématique d'une grue

La figure A.5 illustre une grue.

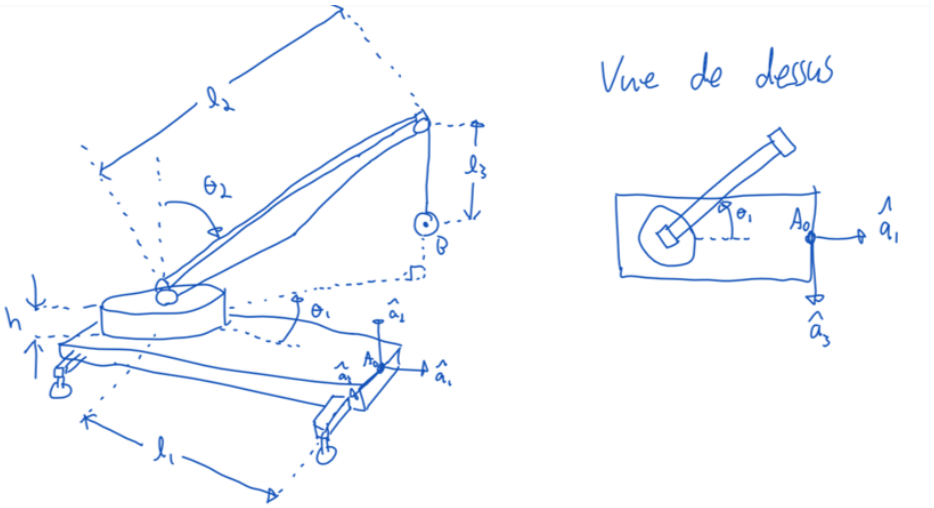


FIGURE A.5 – Calcul de la cinématique d'un robot en 3D

A) Identifier des points et bases vectorielles intermédiaires à utiliser.

B) Trouver le vecteur position du point B par rapport à l'origine A_o, exprimé dans la base vectorielle a.

A.11 Calcul de la cinématique d'une jambe

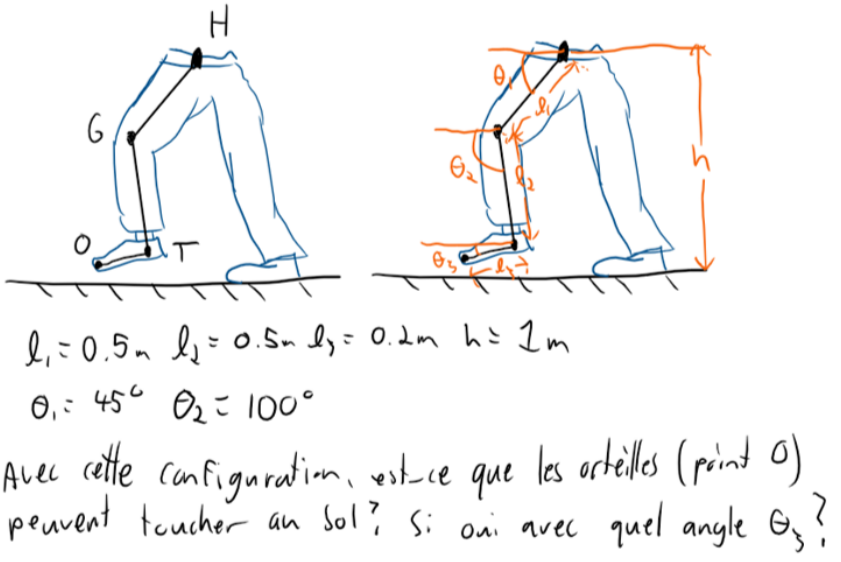


FIGURE A.6 – Calcul de la cinématique d'une jambe

A.11.1 Calcul de la cinématique d'un robot à trois joints

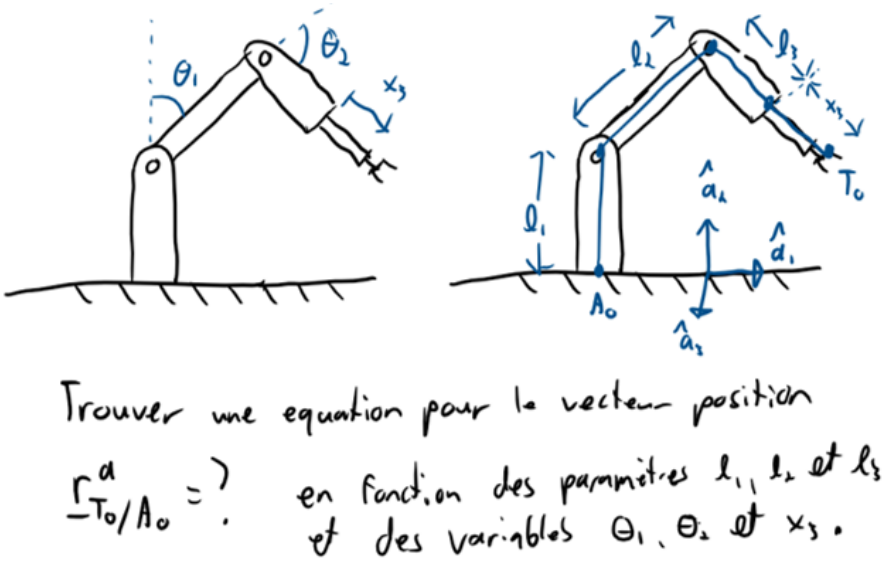


FIGURE A.7 – Calcul de la cinématique d'un robot à trois joints

Paramètres DH

a) Identifier les paramètres DH pour le lève charge illustré à la Figure A.8 et sa matrice de transformation reliant le repère B au repère C ci-dessous.

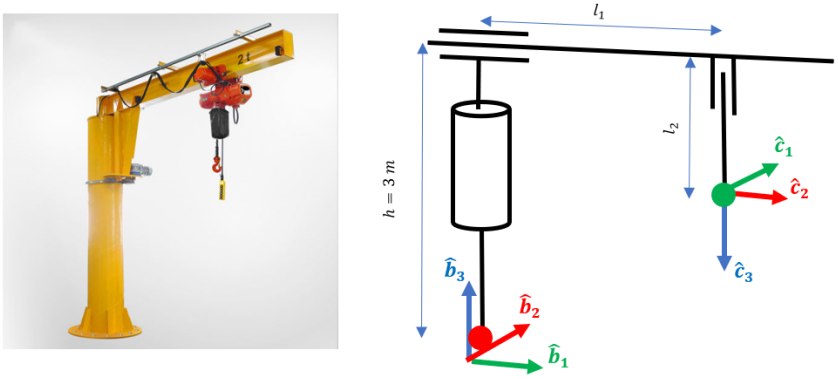


FIGURE A.8 – Lève-charge

b) Identifier les paramètres DH et la matrice associée reliant le repère H au repère P d'une patte du MiniDog V2 de James Bruton, voir Figure A.9.

c) Identifier les paramètres DH du robot Fanuc décrit par le CAD suivant : `https://cad.onshape.com/documents/12ec30dccc54eeca2294d694/w/61a90dbb68f97f2771010491/e/fe7737920bee5cf7c4202b51`)

d) Déterminer s'il est possible d'utiliser les paramètres DH pour le robot dércit par le CAD au lien suivant : `https://cad.onshape.com/documents/7dd54d71984b662e6cbc588e/w/2d33a12386161e8798cac451/e/5212be203761fdc` Déterminer une méthode possible pour utiliser les praramètres DH avec ce type de robot (à haut niveau sans aller dans le détail).

Plus d'information sur l'OpenDog, dont le CAD, se retrouve ici : `https://cad.onshape.com/documents/12ec30dccc54eeca2294d694/w/61a90dbb68f97f2771010491/e/fe7737920bee5cf7c4202b51`

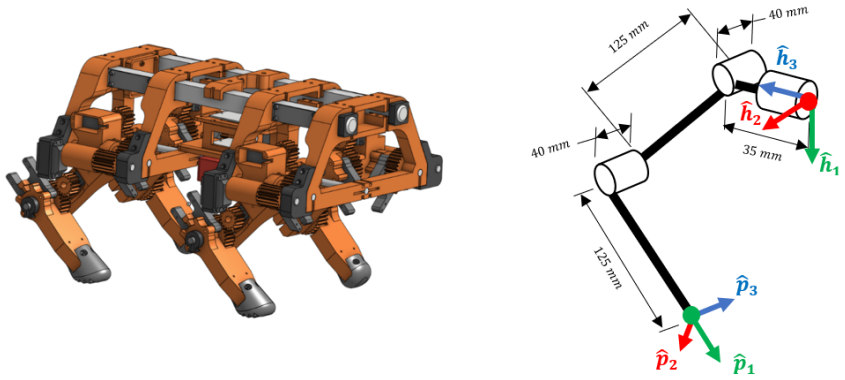


FIGURE A.9 – MiniDog V2 de James Bruton

Déterminer les paramètres DH pour le robot Fanuc à la Figure A.10. Calculer la matrice de transformation de l'effecteur en utilisant les longueurs indiqué par la Figure A.11 et le tableau A.1.

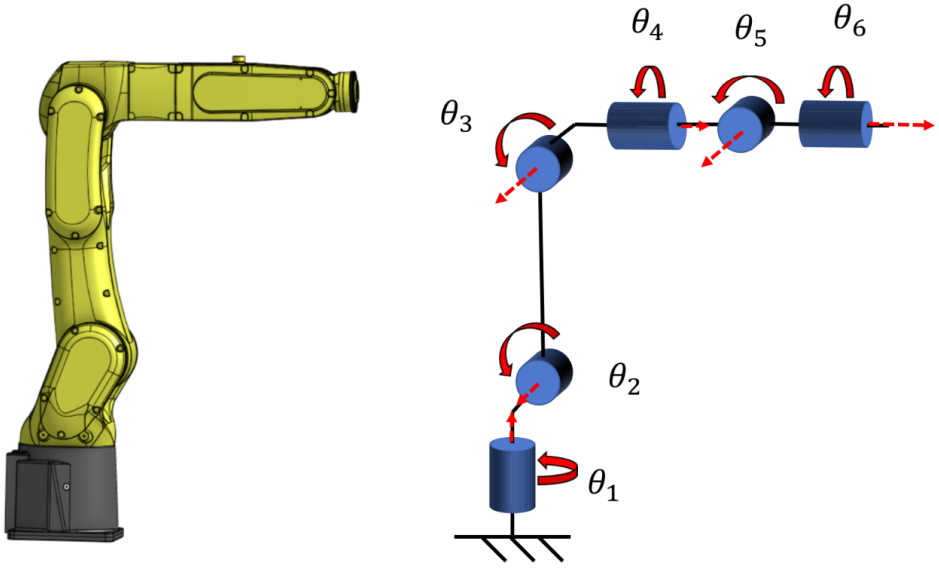


FIGURE A.10 – Robot Fanuc 6 DDLs et ses axes de rotations associés

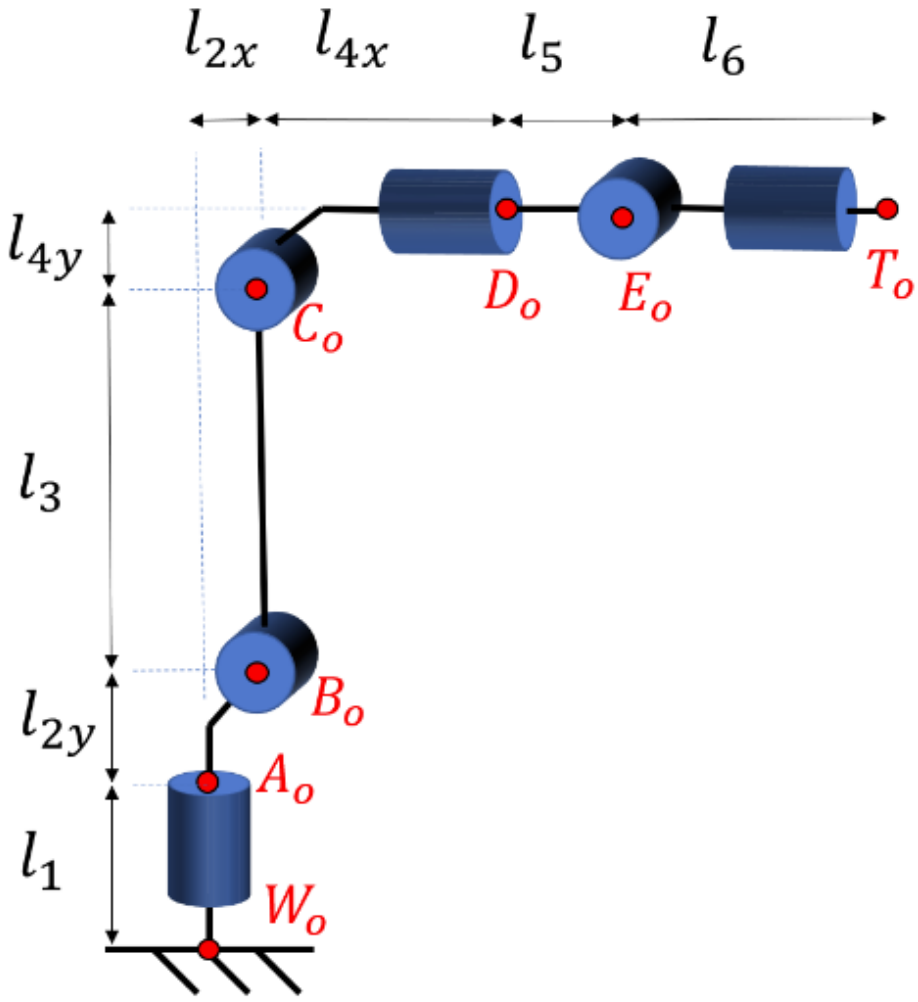


FIGURE A.11 – Dimensions du Robot Fanuc 6 DDLs

Longueur	Valeur
l_1	0.15 m
l_{2x}	0.05 m
l_{2y}	0.1 m
l_3	0.5 m
l_{4x}	0.1 m
l_{4y}	0.02 m
l_5	0.3 m
l_6	0.02 m

TABLE A.1 – Dimensions du Robot Fanuc 6 DDLs

Annexe B

Exercices d'algèbre linéaire

B.1 Moindres carrés pour estimer un ratio de transmission

La figure B.1 illustre un boiter de transmission et cinq paires de mesures angulaires à l'entrée et à la sortie.

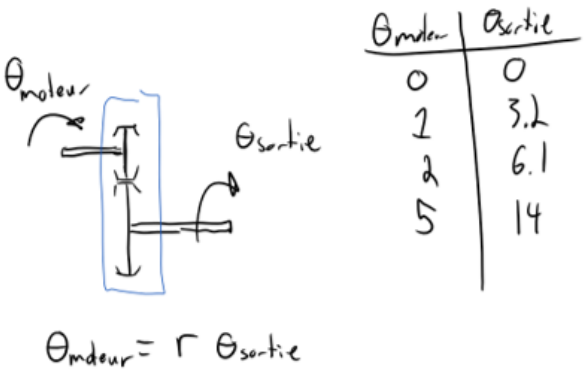


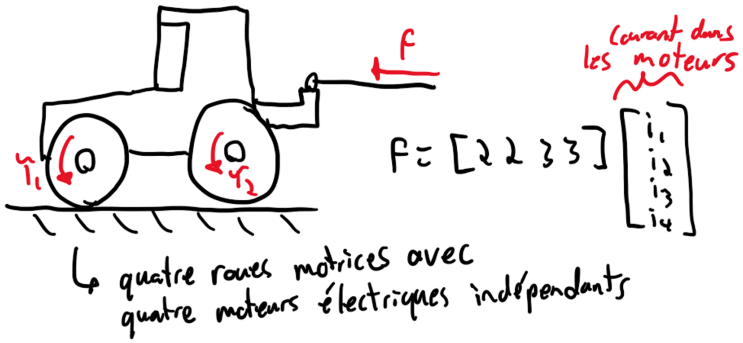
FIGURE B.1 – Moindres carrés pour estimer un ratio de transmission

A) Construisez le un système d'équation linéaire qui relie ces mesures expérimentales.

B) Écrivez l'équation matricielle à résoudre pour estimer le ratio de transmission.

C) Effectuer le calcul numérique pour estimer le ratio de transmission.

B.2 Solution optimale pour un tracteur avec 4 moteurs indépendants



Déterminer la solution $F = 10$ qui minimise les pertes joules.

B.3 Identification des paramètres d'un circuit électrique

5) Estimation des paramètres d'un système électrique :



t	V	I	$\frac{dI}{dt}$
0	V_0	I_0	dI_0
1	V_1	I_1	dI_1
2	V_2	I_2	dI_2
3	V_3	I_3	dI_3

mesures expérimentale

$$V(t) = L\frac{dI(t)}{dt} + RI(t)$$

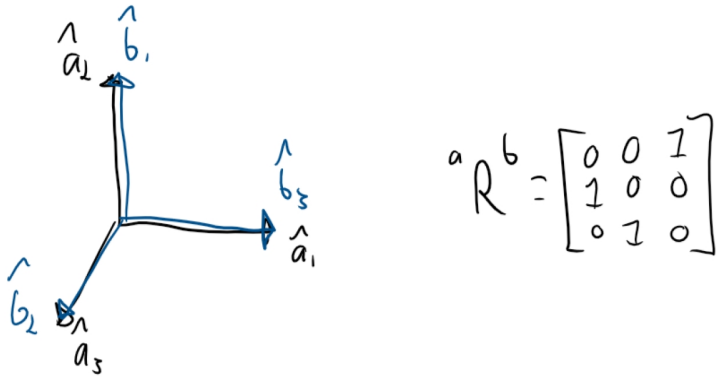
I) Construire une équation linéaire de la forme $[\quad][L\ R] = [\quad]$

II) Construire un système d'équation avec les données de la forme :

$$[\quad][L\ R] = [\quad]$$

III) Écrire l'équation matricielle à résoudre pour estimer L et R

B.4 Vecteur propre d'une matrice de rotation



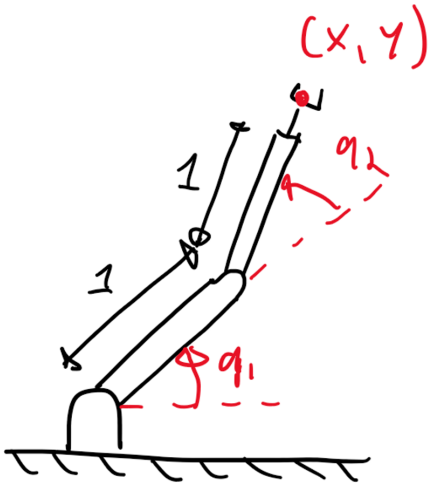
1. Vérifiez que la matrice de rotation illustré ci-dessus a une valeur propre réel $\lambda = 1$
2. Calculez le vecteur propre associé.
3. Vérifiez graphiquement que le vecteur propre calculé correspond à l'axe de rotation.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

Annexe C

Exercices sur la cinématique différentielle

C.1 Cinématique différentielle inverse d'un bras à 2 DDL



$$\underbrace{\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix}}_{\dot{r}} = \underbrace{\begin{bmatrix}-(s_1+s_{12}) & -s_{12}\\ (c_1+c_{12}) & c_{12}\end{bmatrix}}_{J(q)} \underbrace{\begin{bmatrix}\dot{q}_1\\ \dot{q}_2\end{bmatrix}}_{\dot{q}}$$

Trouver une solution $\dot{q}$ pour obtenir $\dot{x}=1$ et $\dot{y}=1$ lorsque

a) $q_1 = 0°$ et $q_2 = 90°$

b) $q_1 = 45°$ et $q_2 = 45°$

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard

C.2 Cinématique différentielle d'un robot à 3 DDL

Le robot illustré à la Figure C.1 est une configuration assez courante pour les 3 premiers joints d'un système robotique.

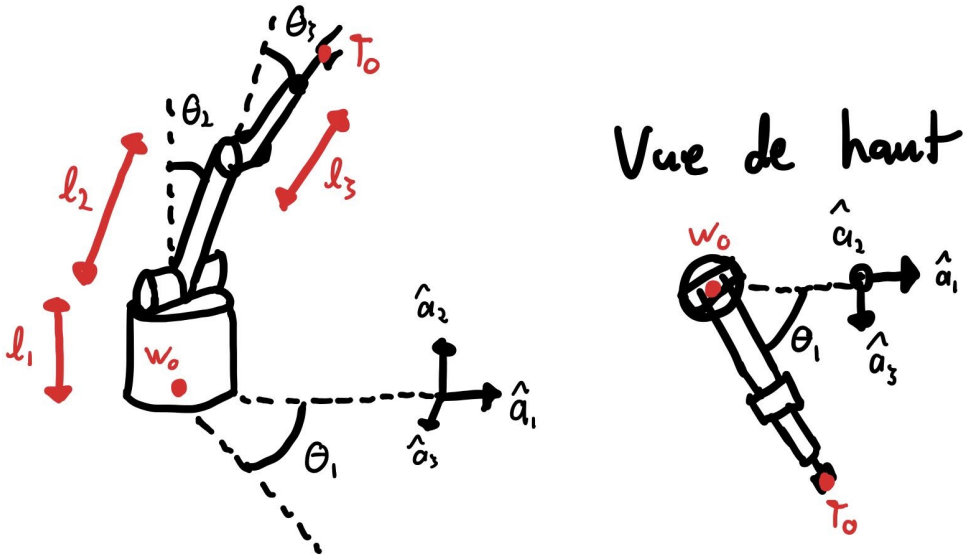


FIGURE C.1 – Robot à 3 joint rotatifs

a) Calculez la matrice Jacobienne J qui caractérise la relation de cinématique différentielle :

$$\dot{\boldsymbol{r}} = J(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{C.1}$$

Rappel :

$$J(\boldsymbol{q}) = \frac{\partial f(\boldsymbol{q})}{\partial \boldsymbol{q}} = \begin{bmatrix} \frac{\partial f_1}{\partial q_1} & \frac{\partial f_1}{\partial q_2} & \frac{\partial f_1}{\partial q_3} \\ \frac{\partial f_2}{\partial q_1} & \frac{\partial f_2}{\partial q_2} & \frac{\partial f_2}{\partial q_3} \\ \frac{\partial f_3}{\partial q_1} & \frac{\partial f_3}{\partial q_2} & \frac{\partial f_3}{\partial q_3} \end{bmatrix} \tag{C.2}$$

où f est est la fonction de cinématique directe.

b) déterminez les configurations pour lesquels la matrice J est singulière. Pour chacune de ces (familles de) configurations singulières : **1)** faites un dessin du robot dans la configuration singulière, **2)** déterminez et illustrez les trois vecteurs vitesses à l'effecteur que chacun des trois joints peut générer dans cette configuration, et **3)** déterminez et illustrez avec des vecteurs la ou les directions pour laquelle il est impossible de bouger l'outil du robot momentanément.

C.3 Matrice jacobienne

L'objectif pour les exercices suivants est de développer les équations de loi de commande pour le robot illustré à la Figure C.2 dans divers contextes.

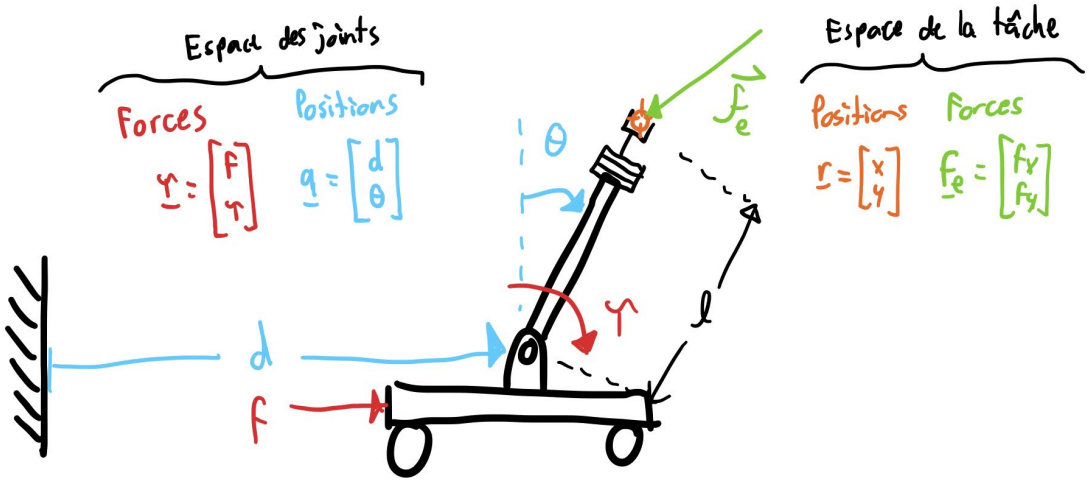


FIGURE C.2 – Robot à deux DDL

C.3.1 Trajectoire

Dans un premier temps, déterminez la relation mathématique qui permet de calculer les accélérations nécessaires aux joints, pour obtenir une accélération désirée à l'effecteur.

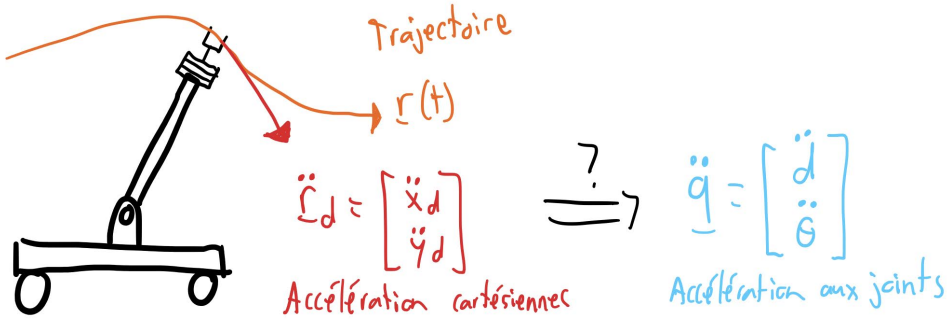


FIGURE C.3 – Robot à deux DDL

C.3.2 Contrôleur en force

Dans un deuxième temps, déterminez la loi de commande qui calculerait les efforts aux joints nécessaires pour produire une force de contacte désirée à l'effecteur.

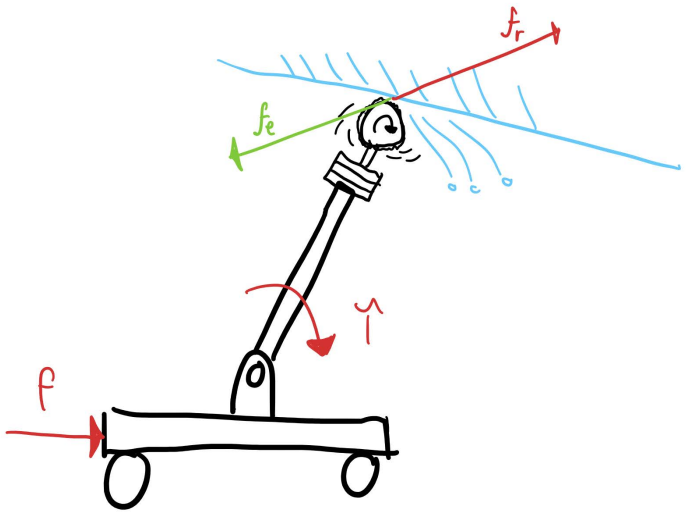


FIGURE C.4 – Robot à deux DDL

C.3.3 Contrôleur en impédance

Ensuite, déterminer une loi de commande en impédance qui ferait que le robot se comporte comme une rigidité cartésienne à l'effecteur.

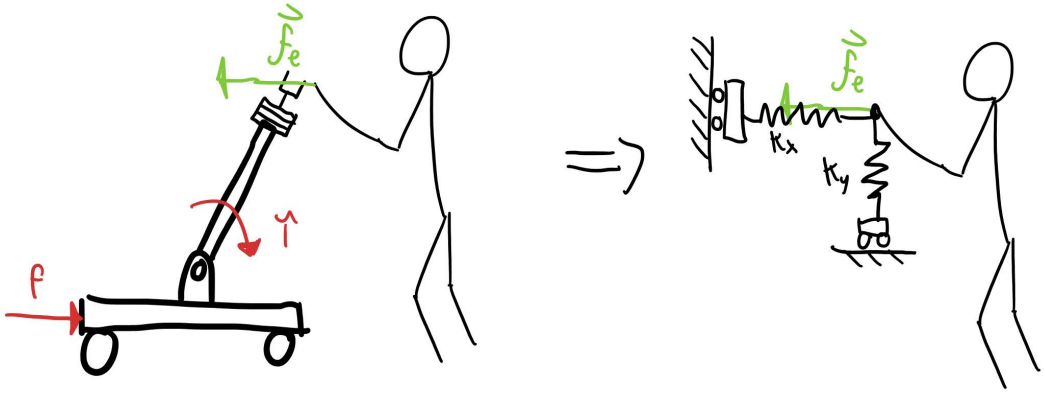


FIGURE C.5 – Robot à deux DDL

C.3.4 Contrôleur en admittance

Ensuite, déterminer une loi de commande en admittance qui ferait que le robot se déplace avec une vitesse à l'effecteur proportionnelle à une force appliquée à l'effecteur mesurée par une cellule de charge.

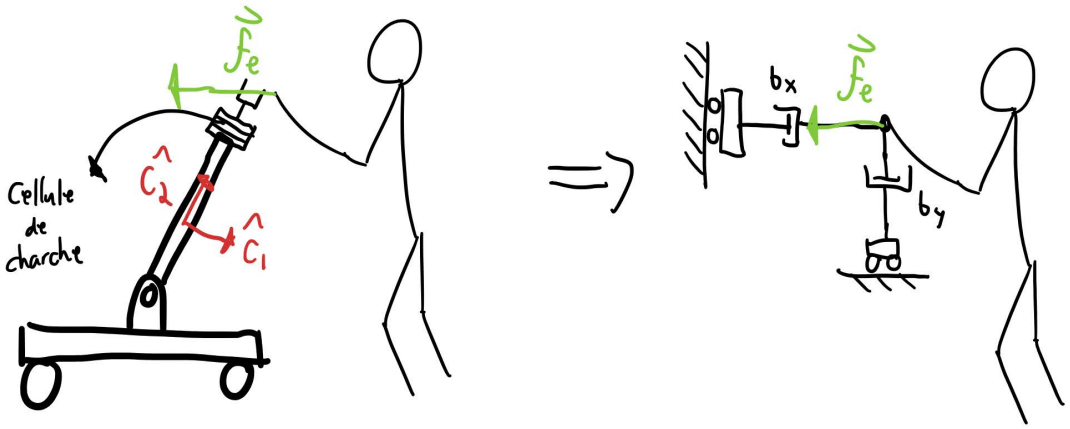


FIGURE C.6 – Robot à deux DDL

C.4 Rigidité d'un robot manipulateur

Pour le robot illustré à la Figure **??**, avec comme paramètres $l_1 = l_2 = 3$m, basé sur l'information que les servo-moteurs produisent des rigidités angulaires de $k_1 = 100\,000 \frac{Nm}{rad}$ et $k_2 = 200\,000 \frac{Nm}{rad}$ (comme illustré à la Figure C.7) :

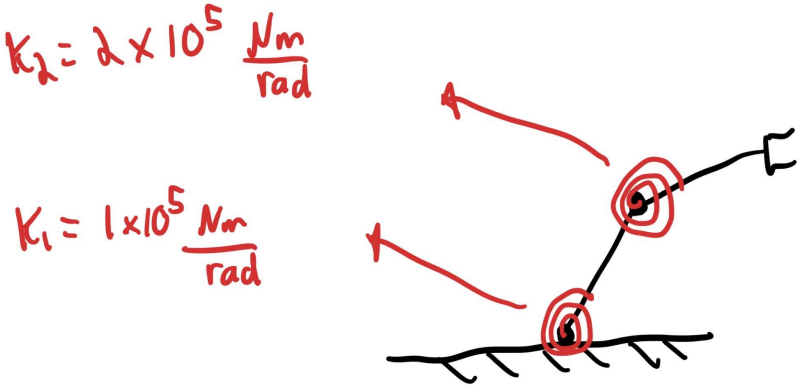


FIGURE C.7 – Robot avec des rigidités angulaires dans l'espace des joints

a) Déterminez la matrice de rigitité cartésienne à l'effecteur lorsque $\theta_1 = 30$ deg et $\theta_2 = 60$ deg.
b) Déterminez les directions de compliance maximum et minimum lorsque $\theta_1 = 30$ et $\theta_2 = 60$ deg.

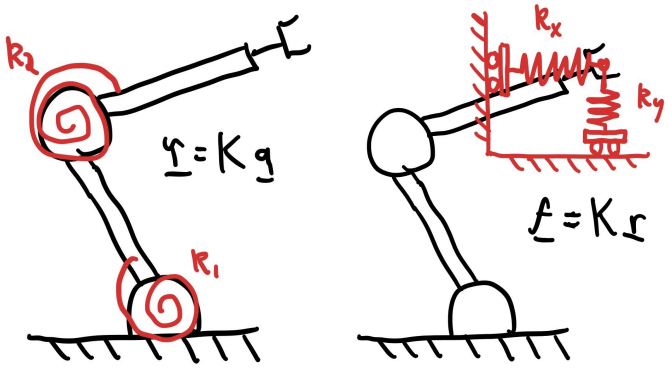
C.5 Commande en impédance d'un robot manipulateur

COLAB

Une page colab est disponible pour faire cet exercise au lien suivant : `https://colab.research.google.com/drive/1EM3hNEwz2aiBqx9GDHoM7QGuPF-Afv1Q?usp=sharing`

a) Comparer le comportement du robot à 2 DLL lorsque commandé avec un contrôleur en impédance dans le domaine des joints vs. lorsque commandé par un contrôleur en impédance dans le domaine de l'effecteur. Changez les paramètres des lois de commande et observez les changements de comportement.

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard



(a) Dans l'espace des joints (b) Dans l'espace de la tâche

FIGURE C.8 – Commande de l'impédance d'un robot

b) Vérifier la convergence des contrôleurs sur la position désirée. Est-ce qu'une approche est préférable à une autre ? Quels vous semble les avantages et inconvénients de ces variantes de la commande en impédance ?

c) Basé sur ces exemples, développez et simulez une version de ces contrôleurs qui incluent une compensation du vecteur de gravité. Vérifiez dans les simulations si la performance est améliorée avec la compensation de gravité.

C.6 Génération de trajectoires

COLAB

Vous pouvez utiliser le fichier amorce suivant sur colab `https://colab.research.google.com/drive/1M3O_nD8iLbRSvZzyUpBnOMo8KUclrG26?usp=sharing` pour faire cet exercice.

Pour le robot illustré à la Figure **??** avec comme paramètres $l_1 = 4$ et $l_2 = 3$, déterminez le profil temporelle de la position angulaire, la vitesse angulaire et l'accélération angulaire des joints, pour suivre la trajectoire illustrée à la Figure **??** mais cet fois avec des contraintes temporelles suivantes :

1. Le robot débute à l'arrêt et doit terminer avec une vitesse nulle.
2. Le robot doit complètement arrêter momentanément aux points (3,0) et (0,3).
3. Le robot doit avoir un profil de vitesse trapézoïdal à l'effecteur, avec une vitesse linéaire maximum de l'effecteur de 1 m/s.
4. Le robot doit avoir une accélération linéaire maximum à l'effecteur de $1m/s^2$.
5. Le robot doit effectuer le trajet le plus rapidement possible avec ses contraintes.

C.7 Espace nul d'une matrice et robots redondants

COLAB

Utiliser le fichier amorce suivant sur colab `https://colab.research.google.com/drive/16ACenFOLOHVNeReqJTbkATAB3281iVbp?usp=sharing` pour faire cet exercice.

a) Comparer le comportement du robot redondant à 5 DDL avec les différents types de lois de commande. Vous pouvez changer les paramètres et observer le changement de comportement.

b) Basé sur ces exemples, développez et simulez un contrôleur qui utilise l'espace nul de la matrice Jacobienne pour amener la troisième articulation à une position (0.5,0.5), pendant que l'effecteur va à une

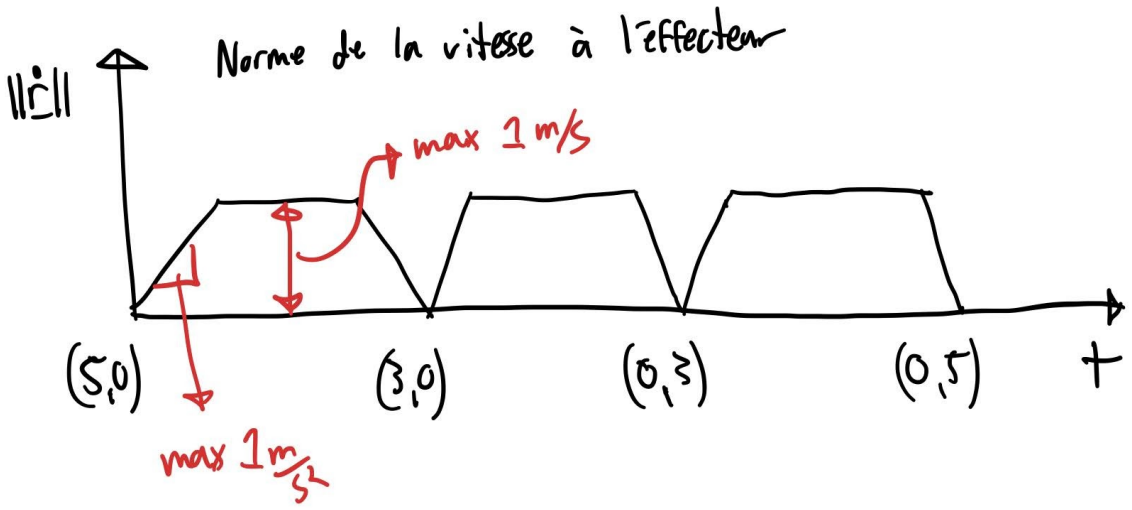


FIGURE C.9 – Profil de vitesse de l'effecteur souhaité

position (1,1) comme illustré à la Figure suivante (une configuration parmi plusieurs options qui rencontrent ces spécifications).

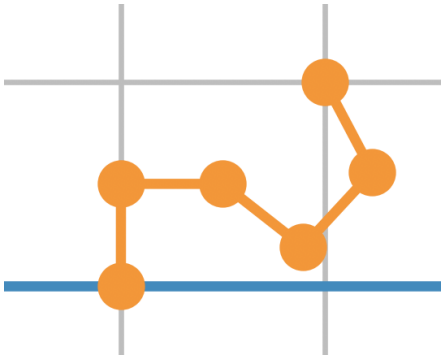


FIGURE C.10 – Configuration désirée du robot manipulateur (une solution parmi plusieurs options)

Notes de cours génie robotique - Copyright ©2026 Alexandre Girard