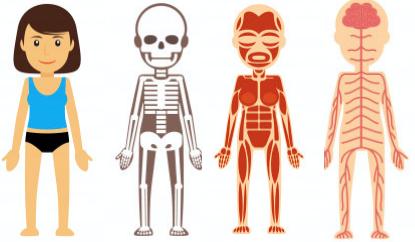
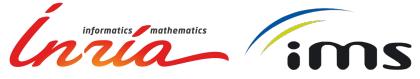


Mathématique pour la robotique

David Daney, Inria Auctus

david.daney@inria.fr

Équipe Auctus

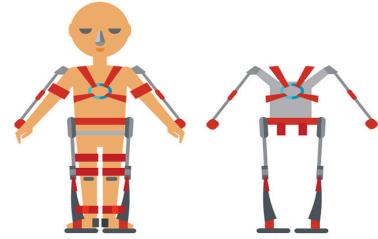


Modélisation et analyse du comportement humain

Axe A
physique/cognitique



activités, postures,
gestes, mouvements
humain



Axe B

Couplage
opérateur / cobot



optimisation et médiation des performances d'un couple opérateur/cobot



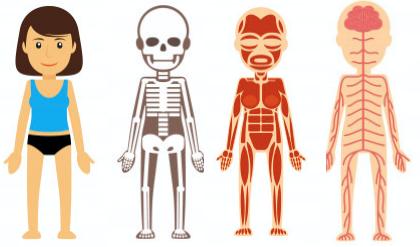
Conception des systèmes cobotiques

Axe C1 architecture



Axe C2 commande





Modélisation et analyse du comportement humain

Axe A

physique/cognitique



*activités, postures,
gestes, mouvements
humains*

Identifier des fragilités cognitives et physiques des opérateurs

- Proposer de nouveaux indices
ergonomique, bio-mécanique, cognitive, robotique
ex : fatigue, expertise, situations accidentogènes
- Modélisation et analyse du mouvement humain
- Découpage d'activités, mouvement, posture
- Reconnaissance d'activités, ...

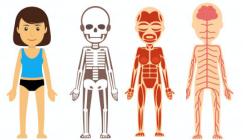
Méthodologies :

Expérimentation avec des humains

Analyse par intervalles, (incertitudes, variabilités)

Traitement du signal,

Apprentissage, machine-learning



Analyse et modélisation
du comportement
humain

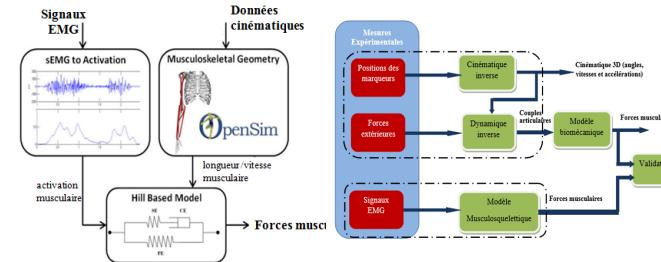
Contexte : limite des indices ergonomiques

Variabilité motrice : dispersion spatio-temporelle des mouvements articulaires, de la coordination et des activités musculaires entre les répétitions successives d'une même tâche.

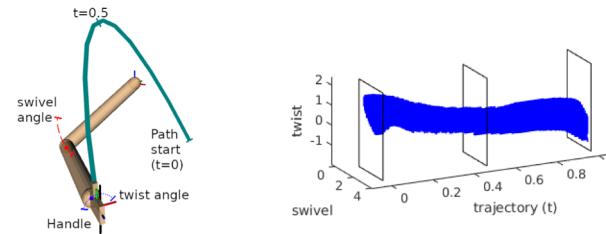
En liens avec la fatigue et l'expertise



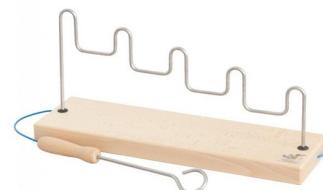
Fatigue via la variabilité musculaire

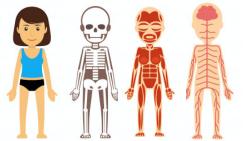


Modélisation ensembliste des
redondances humaines



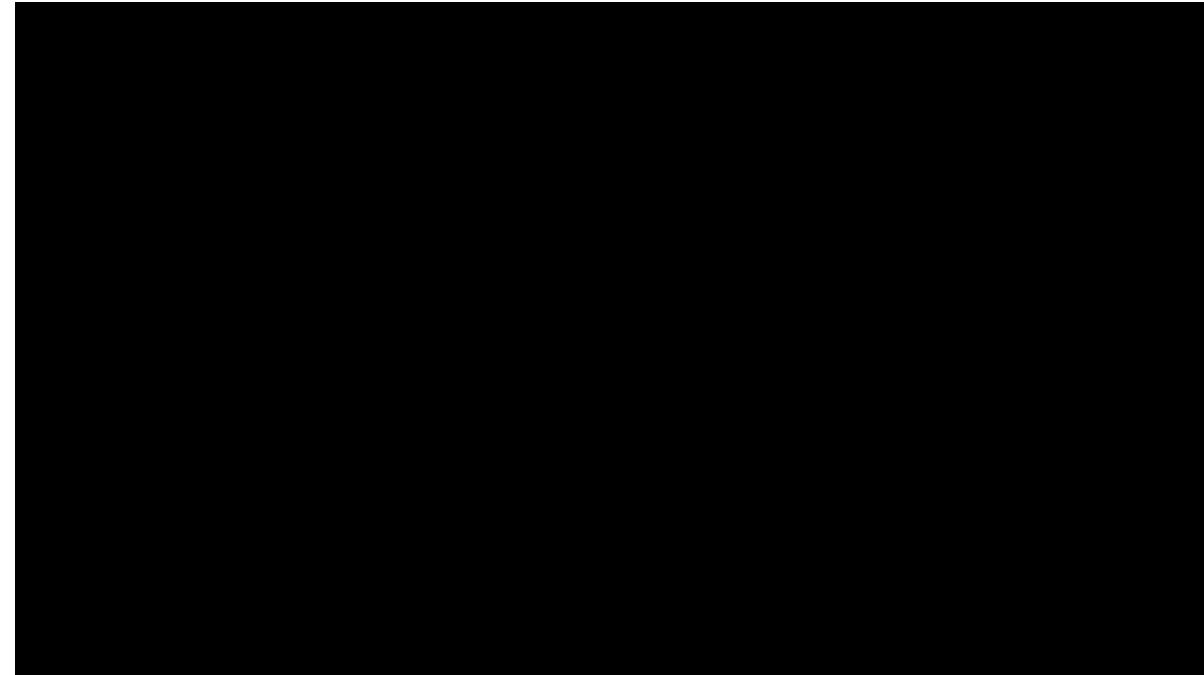
Mesure via une interaction
Humain/Robot



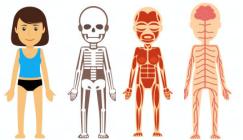


Analyse et modélisation
du comportement
humain

Évaluation ergonomique du mouvement humain



Modalités de représentation du mouvement permettant d'extraire des informations physiques et cognitives



Analyse et modélisation
du comportement
humain



Conception des
systèmes
cobotiques

Conception outils d'assistance métier du bâtiment



Étude de cas (plomberie, pose câbles)



Étude ergonomique



Test utilisation exosquelette



Pré-étude robotisation gazelle



Analyse et modélisation
du comportement
humain

Développement technologique

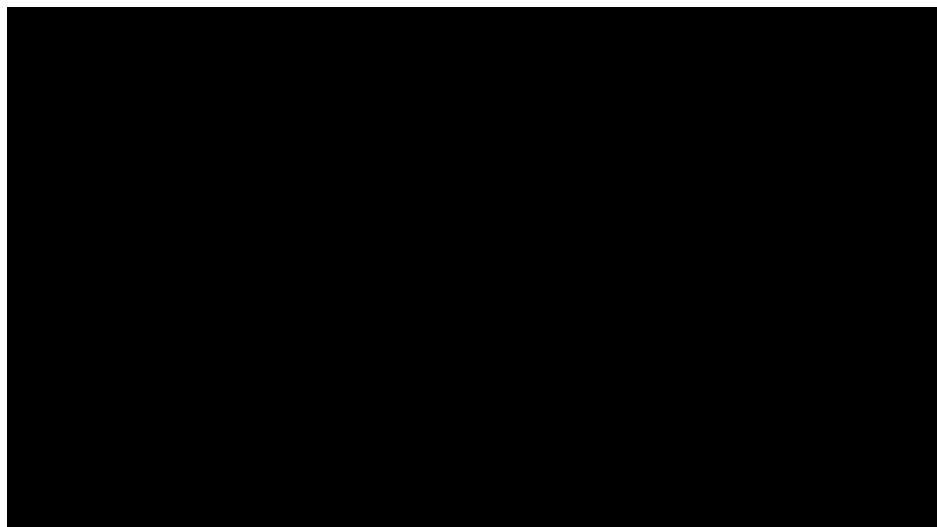
Postex, ADT, inria

*Comment mesurer le mouvement humain
in-situ ?*



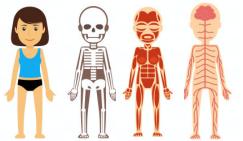
Interprétation le changement de résistance du tissu conducteur pendant son extension /
déformation causée par le mouvement humain.

Patent: number FR1860192 (Smart textile adapted for motion and/or deformation detection)



Creation start up : Touch sensity



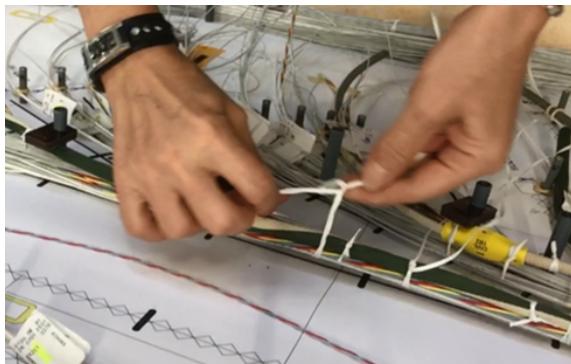


Analyse et modélisation
du comportement
humain



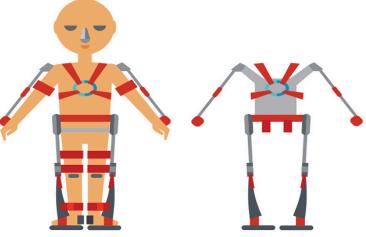
Analyse des humeurs et de la satisfaction

Conception d'une expérience avec un chatbot simplifié dans le but de tester les changements émotionnels et d'observer les changements de comportements



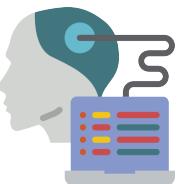
Amélioration du processus de frettage des harnais électriques

- Caractérisation du nœud actuellement réalisé
- Proposer des solution(s) pour le remplacer



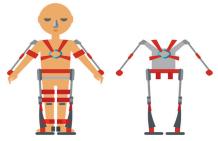
Axe B

Couplage opérateur / cobot



optimisation et médiation des performances d'un couple opérateur/cobot

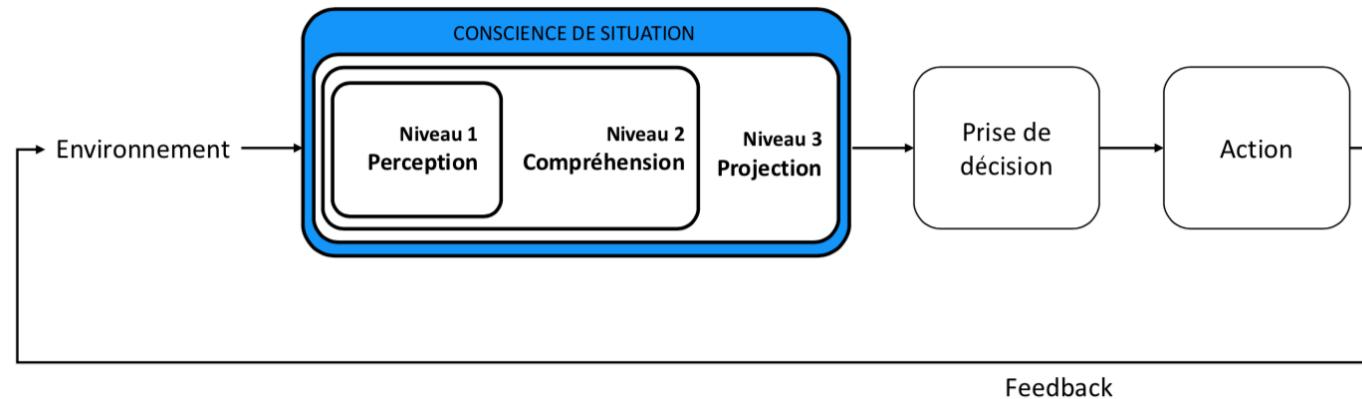
- Modéliser le couple
- Évaluer l'impact
 - du robot sur l'opérateur
Facteur humain : satisfaction ; nouvelle relation à la tâche, au métier ; Influence sur l'expertise et le confort de l'opérateur
 - du design sur la performance du couple
Intuitivité, adaptabilité, accessibilité, acceptabilité physique et/ou cognitive, sociale, facteurs de motivation
 - de l'utilisateur sur l'exploitation des cobots
Risques d'erreur humaine
- Synthétiser la médiation
 - préconisation, contraintes, critères pour le design
 - partage d'autorité (décision, choix d'un leader)
 - niveau d'autonomie du robot



Couplage opérateur / cobot

« *Situation Awareness is being aware of what is happening around you and understanding what information means to you now and in the future* »

- Mica R. Endsley, 1995b



8 Démons de Endsley

Attentional Tunneling – Focalisation sur une zone ou un problème en excluant ainsi les signaux d'autres zones ou problèmes.

Requisite Memory Trap – Sur-dépendance envers la mémoire à court terme de l'opérateur.

Workload, Anxiety, Fatigue, and Other Stressors – (WAFOS) Stress physiques et psychologiques qui peuvent impacter la performance.

Data Overload – Sur-dépendance en la capacité de l'opérateur à traiter une grande quantité ou fréquence de données

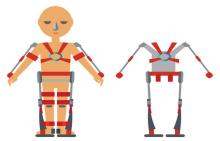
Misplaced Salience – Incorrecte priorisation ou représentation de l'importance de l'information par l'IHM.

Complexity Creep – complexité élevé du système rendant difficile la construction d'une représentation suffisante du fonctionnement de l'outil

Errant Mental Model – Utilisation d'un modèle mental incomplet ou mauvais résultant en une mauvaise interprétation de l'information

Out-of-the-Loop Syndrome – Automatisation d'un processus causant la perte de SA de l'opérateur et l'empêchant de savoir comment agir en cas de panne de ce système.

Adapter ces travaux application des démons de Endsley pour évaluer le travail collaboratif.



Couplage opérateur / cobot

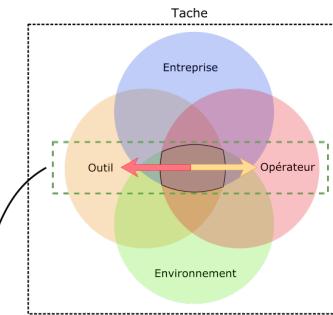
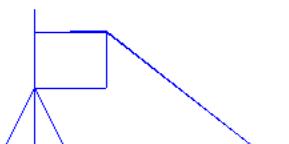
Élaboration d'une méthode d'analyse et d'évaluation d'un système complexe humain / outil, avec application au levé de tampon



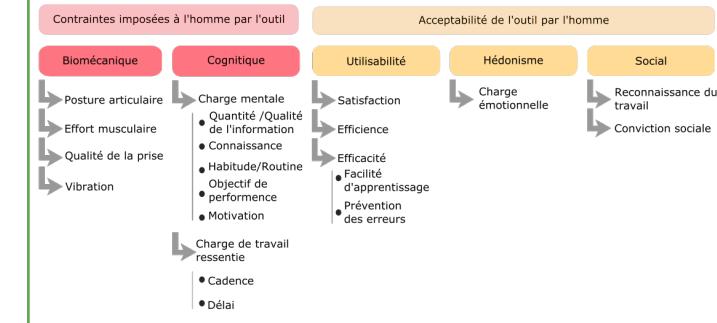
Conception des systèmes cobotiques



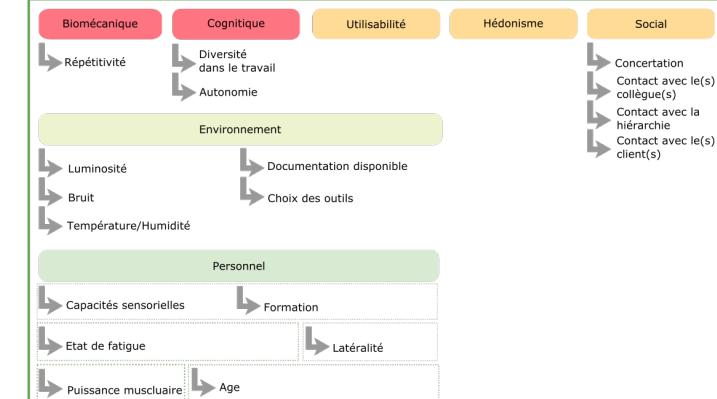
Étude de cas
Levé de tampon

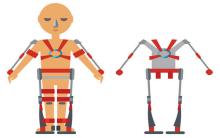


Evaluation du couple Outil/Opérateur



Facteurs d'influence





Couplage
opérateur / cobot

PlatefOrme de RoboTisation et d'Autonomisation GénériquE de bâtiS industriels

Portage



Conception des
systèmes
cobotiques

AKKA ez-wheel IIDRE ims
The Electric Wheel

Avec le soutien de :

DASSAULT BIBUS[®]
AVIATION SUPPORTING YOUR SUCCESS



Faciliter les opérations de « logistique interne » dans l'industrie et notamment le transport de bâtiS lourds et volumineux, dans le domaine de l'aéronautique.



Conception ergonomique et influence des facteurs humains

- Quel comportement (télé-opéré, semi-autonome, autonome...) ?
- Quelles interfaces ? Anticiper les possible accidents : démons d'Endsley



Conception des systèmes cobotiques

Axe C1 architecture



Axe C2 commande



Architecture et conception

- Quel architecture mécanique ?
- Quelle architecture d'interaction ?
- Quels sont les placements, des dimensions ?
 - Étudier les échanges d'information (nature et les flux)
 - Étudier les rôles et les types d'interactions (humain-robot-tâche)
 - Modéliser les espaces de travail (utilisation d'une approche ensembliste)

Commande : adaptation du comportement du robot

- en fonction de la nature de l'assistance du robot selon l'état de l'opérateur (ergo, cognitif, ...)
- en fonction des phases du processus de production, apprendre pour gérer les transitions



Conception des
systèmes cobotiques

Woobot



Analyse et modélisation du
comportement humain

Élaboration d'une commande collaborative pour améliorer la sécurité d'un opérateur lors de l'accomplissement d'une tâche artisanale tout en préservant son savoir faire.



Augmenter la sécurité, le confort.
Réduire la pénibilité
Préserver l'expertise de l'opérateur
Diminuer TMS
Simplifier des tâches complexes

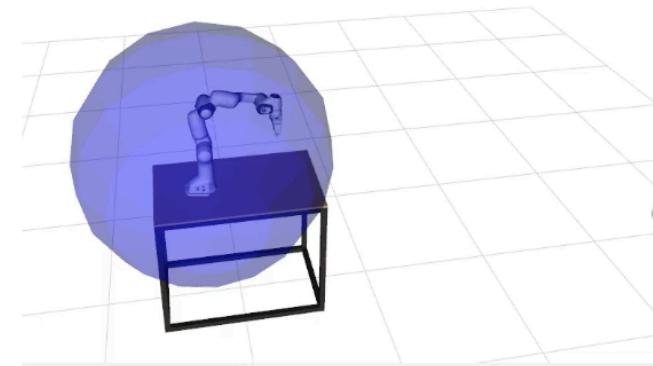




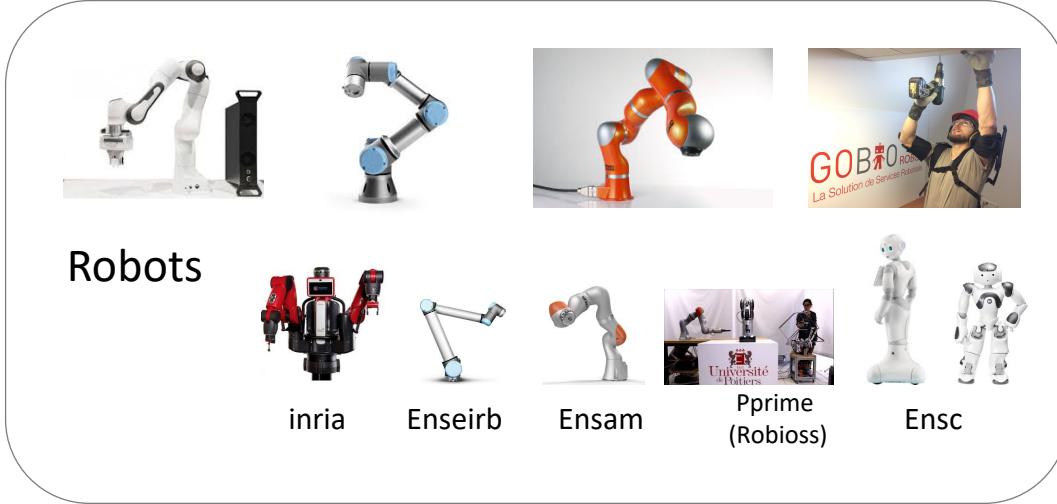
Conception des systèmes cobotiques

Modulation continue du comportement robotique par une analyse en ligne de l'espace de travail en Robotique collaborative

- Détection, estimation et prédition en-ligne de l'état de l'humain
- Calcul ensembliste en-ligne de l'espace de travail partageable de manière sûre ([ISO TS 15066](#))



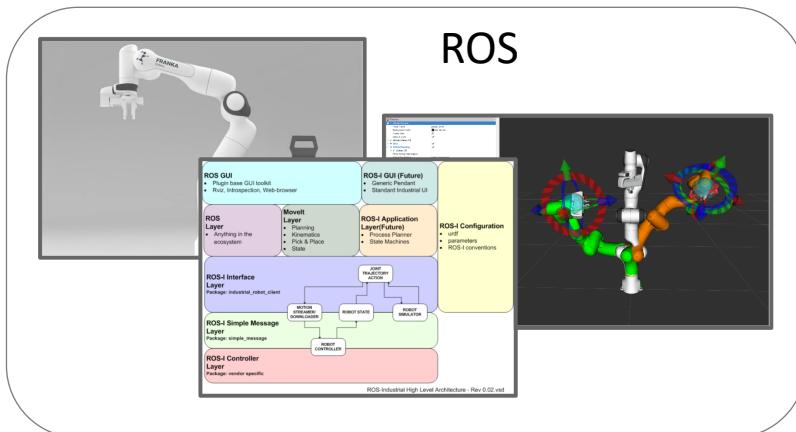
Plateformes robotiques



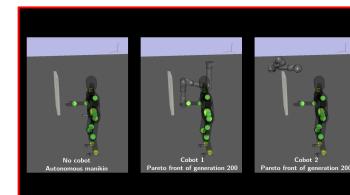
+ Capteurs (observation de l'humain)



ArCol



HuMoSoft



Todo : mannequin virtuel

Utilisation de Matlab, Scilab, Octave

<https://www.scilab.org/fr>

<https://www.gnu.org/software/octave/>

Notions utiles

- Modélisation de problèmes robotique
- Résolution des modèles *directe et inverse*
- Résolution de systèmes linéaires
- Inversion de matrice
- Singularité
- Résolution de systèmes non-linéaire (moindres carrés)
- + si le temps le permet ...

Vérification du planning : 20h

- 2.5h : 7 octobre 09:30-12:00
- 2.5h : 14 octobre 09:30-12:00
- 2.5h : 21 octobre 09:30-12:00
- **2.5h : 4 novembre 09:30-12:00**
- 2.5h : 18 novembre 09:30-12:00
- 2.5h : 25 novembre 09:30-12:00
- 2.5h : 02 décembre 09:30-12:00
- **2.5h : 09 décembre 09:30-12:00**

Questions ouvertes :

- Slides ou pas ?
- Restons ouvert
- Vos origines ?
- Intérêt du cours

Question ouverte : Pourquoi DS et pas un projet ?

Modélisation

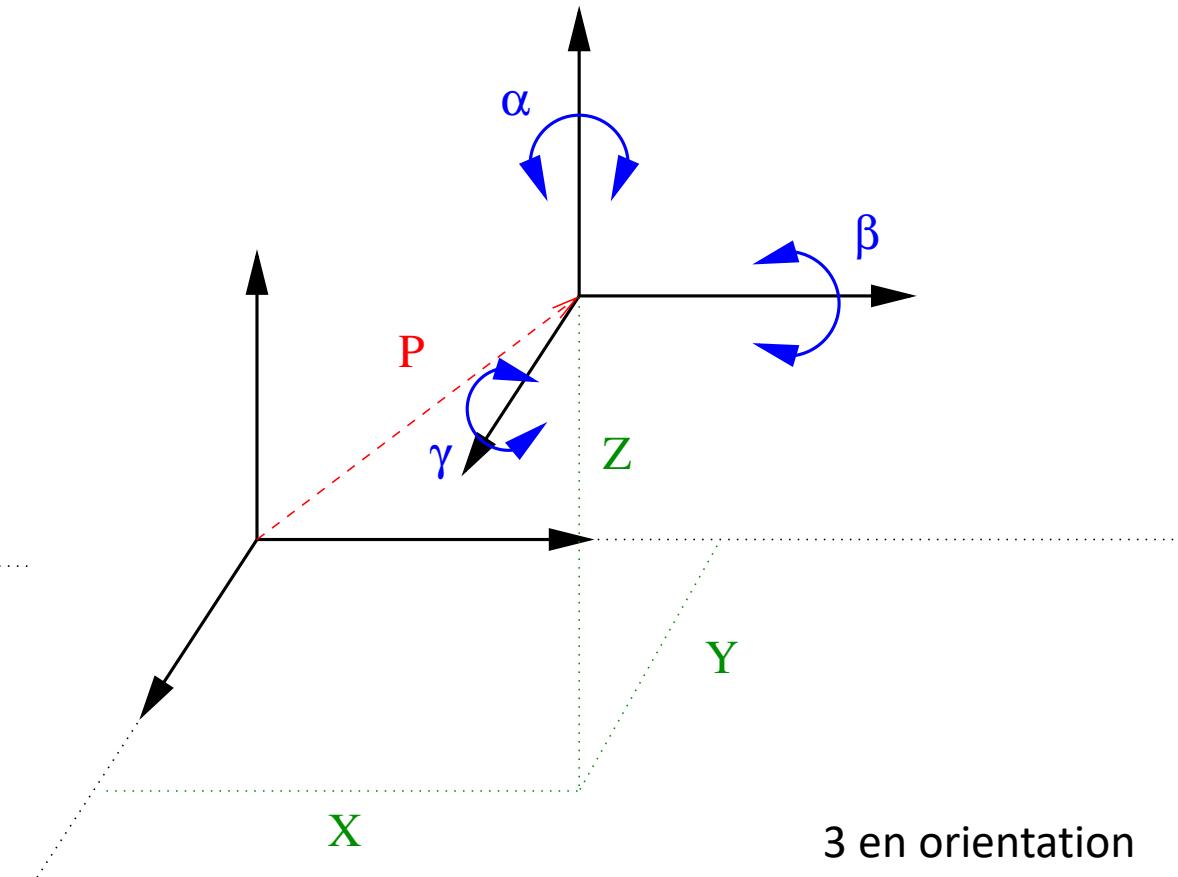
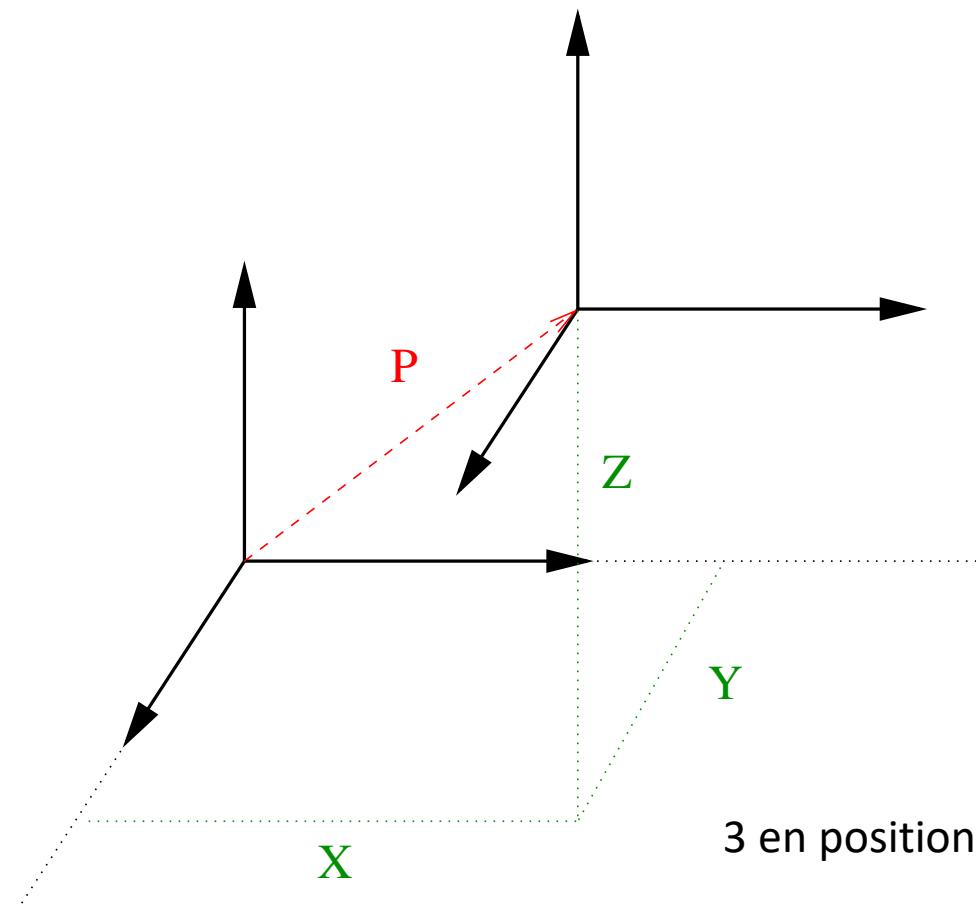
Combien de degrés de libertés a un solide dans l'espace ?

ou encore... Combien de paramètres indépendants (nombre minimal) sont-ils nécessaires pour définir la situation (positionnement) du solide dans l'espace (par rapport à un repère de référence) ?

Combien de degrés de libertés a un solide dans l'espace ?

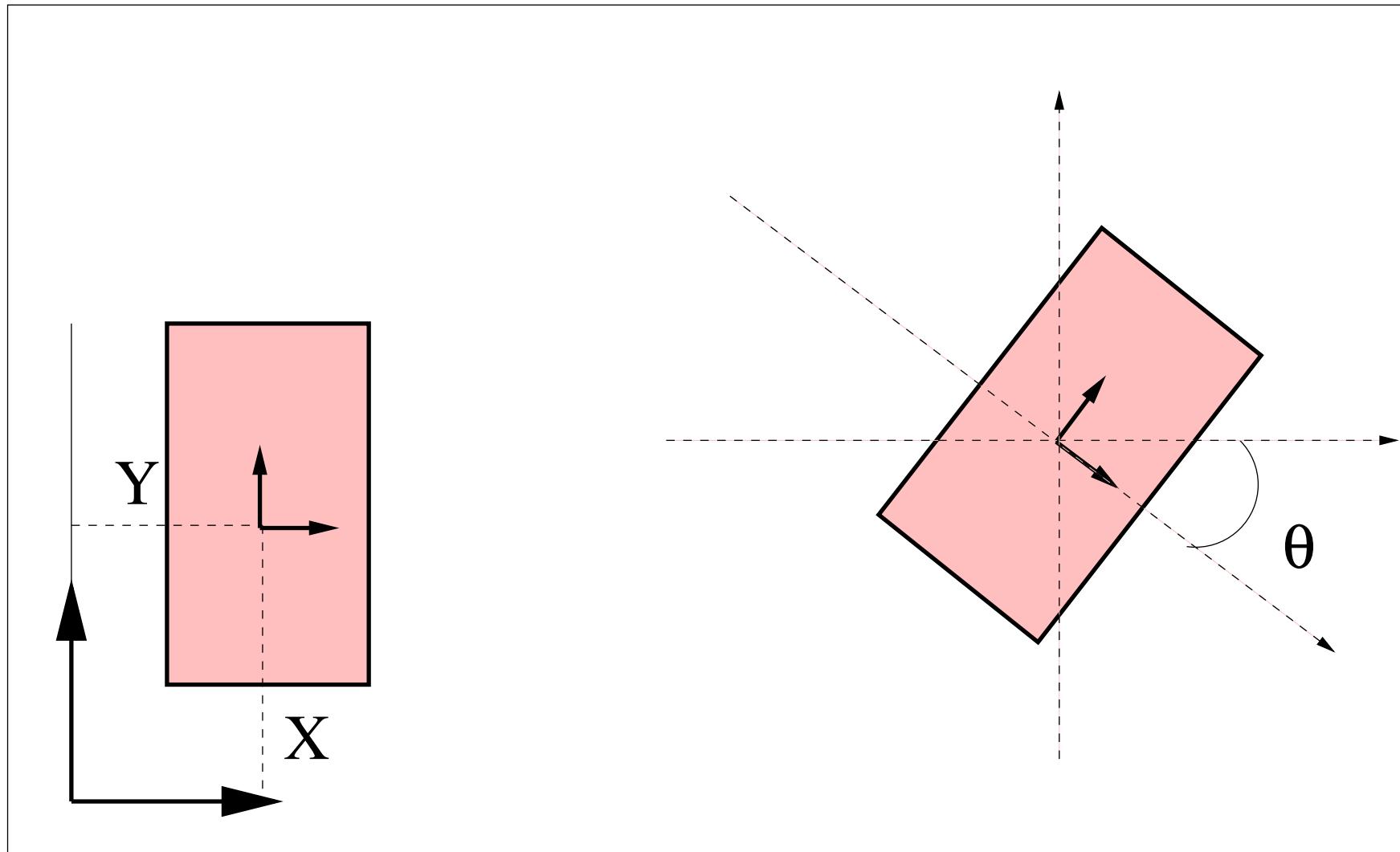
ou encore... Combien de paramètres indépendants (nombre minimal) sont-ils nécessaires pour définir la situation (positionnement) du solide dans l'espace (par rapport à un repère de référence) ?

6 degrés de liberté



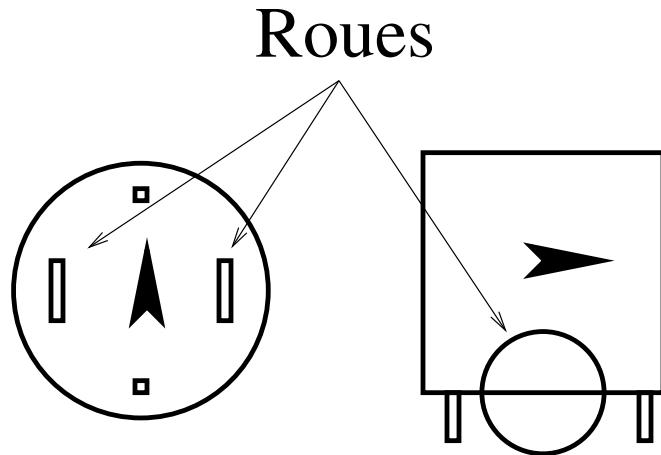
Quels sont les **degrés de liberté** de la "brosse à effacer" se déplaçant sur le tableau ?

Quels sont les **degrés de liberté** de la "brosse à effacer" se déplaçant sur le tableau ?



3 dll :
• 2 translations
• 1 orientation

Un exemple simple robot Toto



Vue de haut Vue de profile

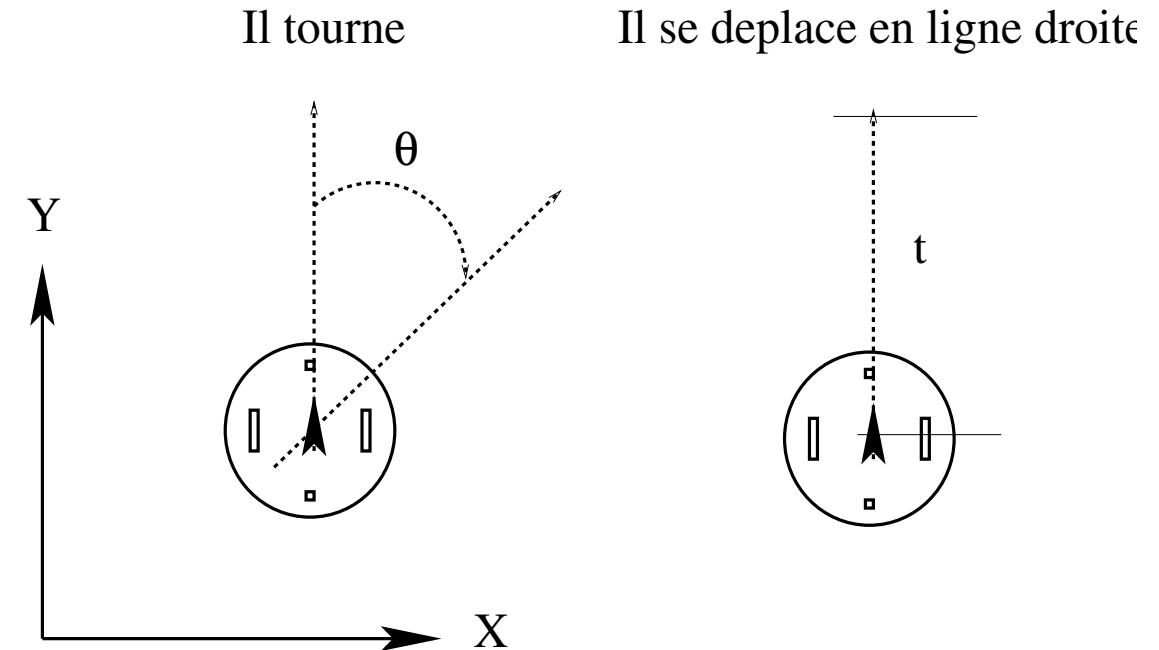
Questions :

Déplacements de Toto.

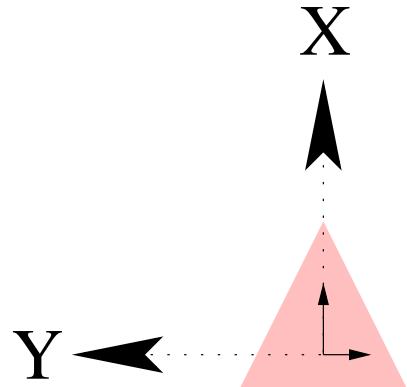
Dans le plan, quel sont les coordonnées d'un solide ? Quel sont les degrés de liberté du robot ?

Est-ce équivalent ?

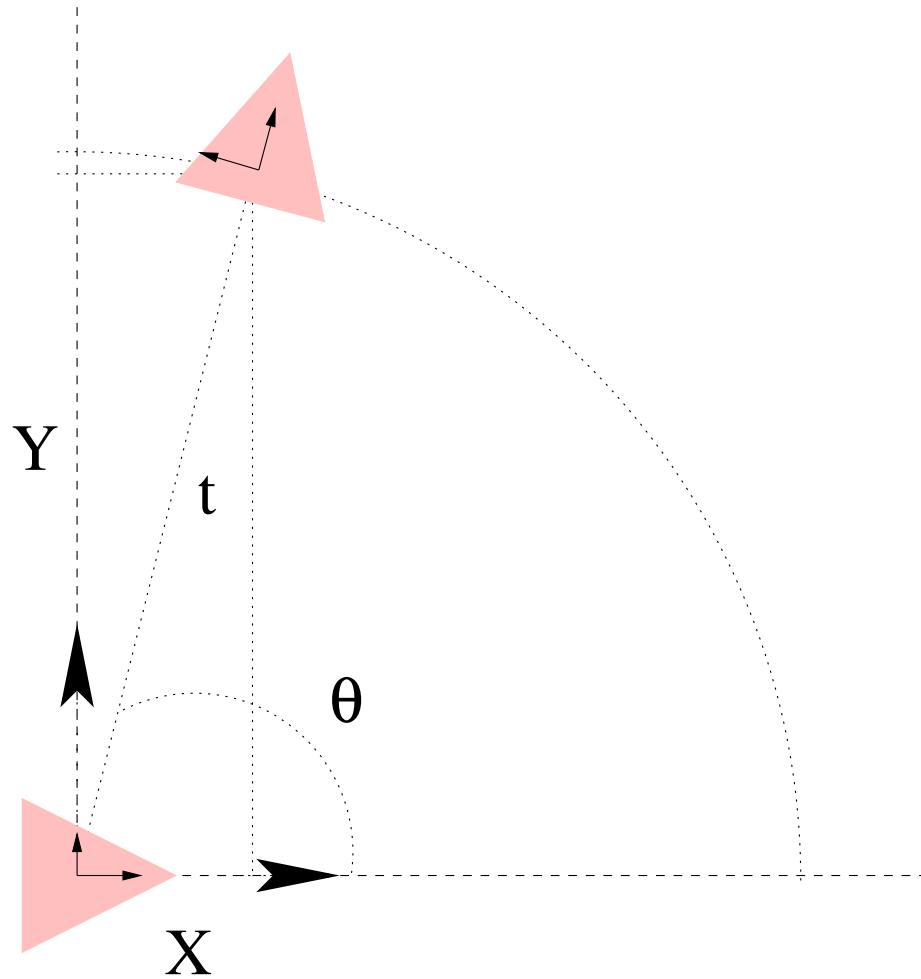
- Le robot avance de t puis tourne de θ .
- Le robot tourne de θ puis avance de t .
- Donner les coordonnées du robot.



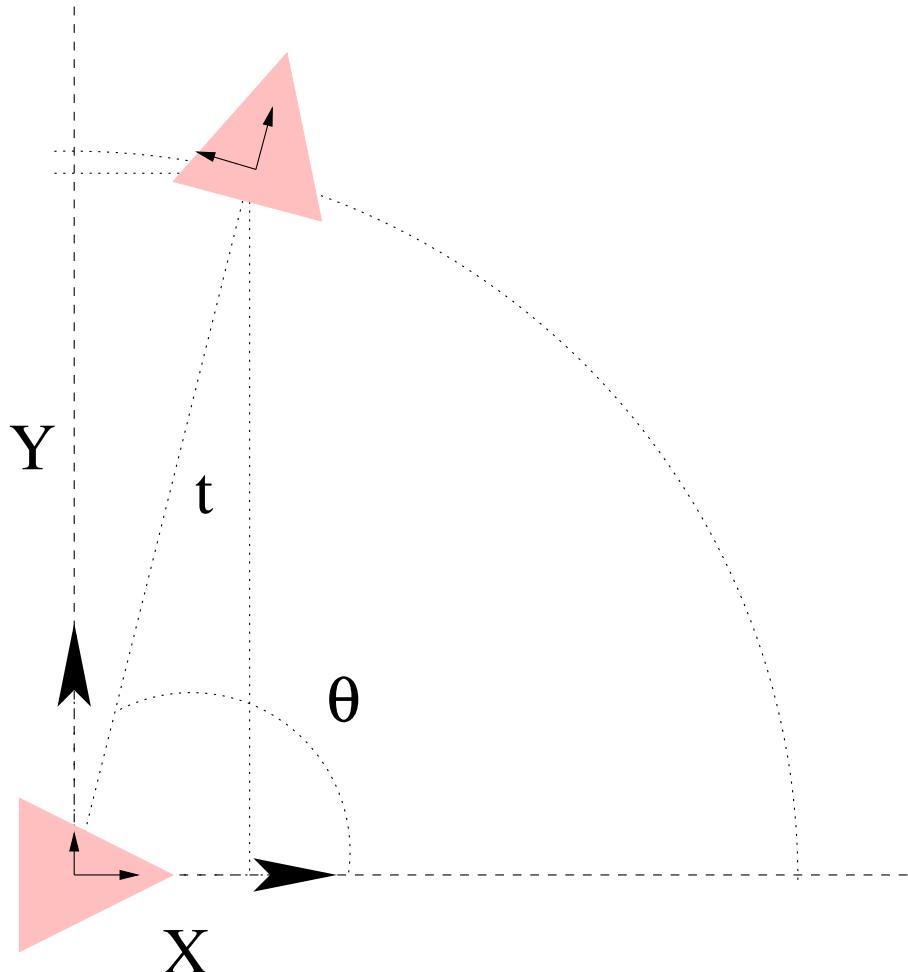
A partir d'une position initiale, le robot **tourne de θ puis avance de t puis tourne de α puis avance de d .**
Donner son positionnement.



A partir d'une position initiale, le robot tourne de θ puis avance de t . Donner sa nouvelle position



A partir d'une position initiale, le robot **tourne de θ puis avance de t puis tourne de α puis avance de d** .
Donner son positionnement.



ou bien

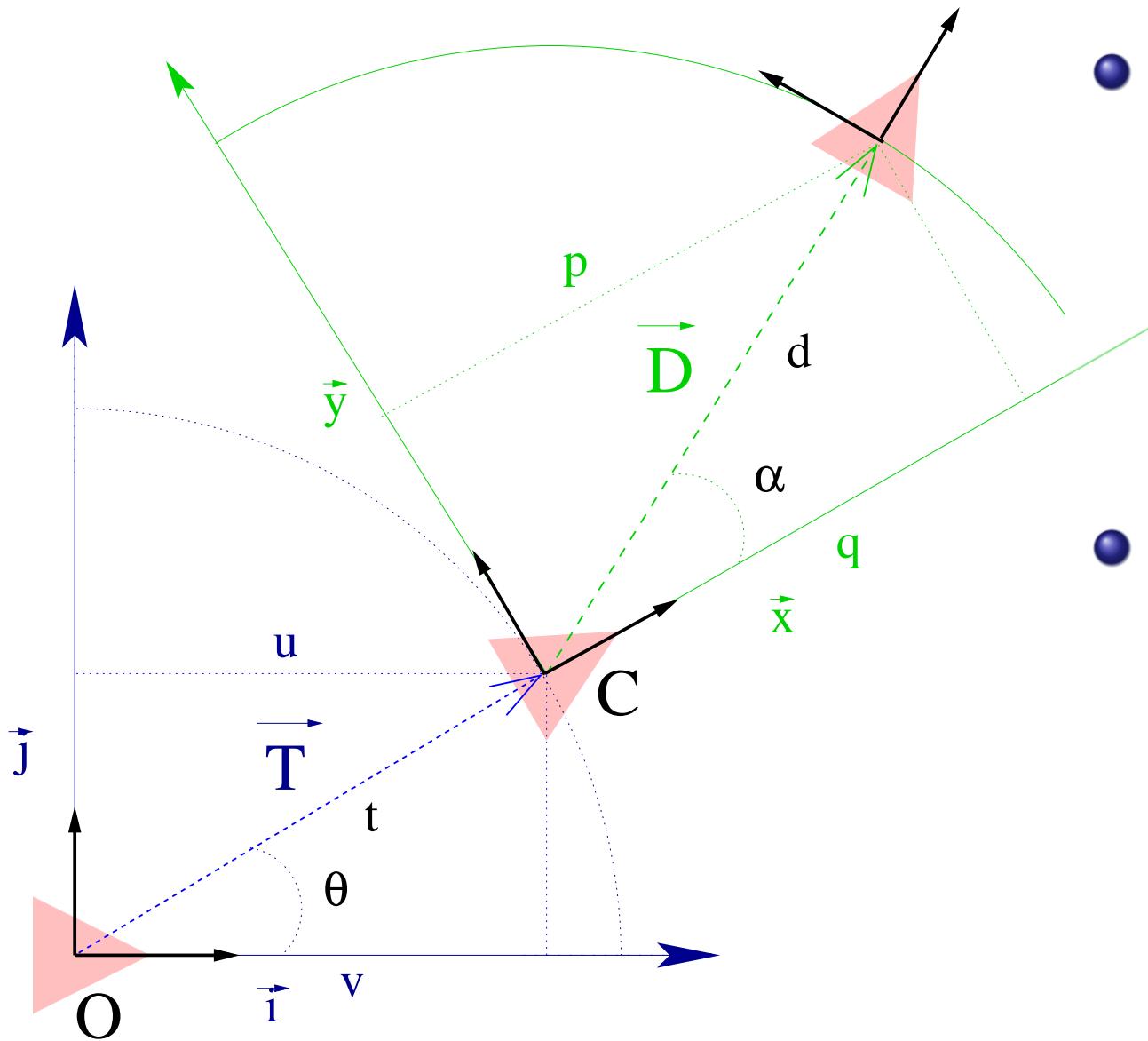
$$X = t \times \cos(\theta)$$

$$Y = t \times \sin(\theta)$$

$$\Theta = \theta$$

$$\mathcal{X} = \begin{pmatrix} t \cdot \cos(\theta) \\ t \cdot \sin(\theta) \\ \theta \end{pmatrix}$$

A partir d'une position initiale, le robot tourne de θ puis avance de t puis tourne de α puis avance de d .
Donner son positionnement.



- Le robot tourne de θ puis avance de t .

Dans le repère $\Omega_O = (O, \vec{i}, \vec{j})$

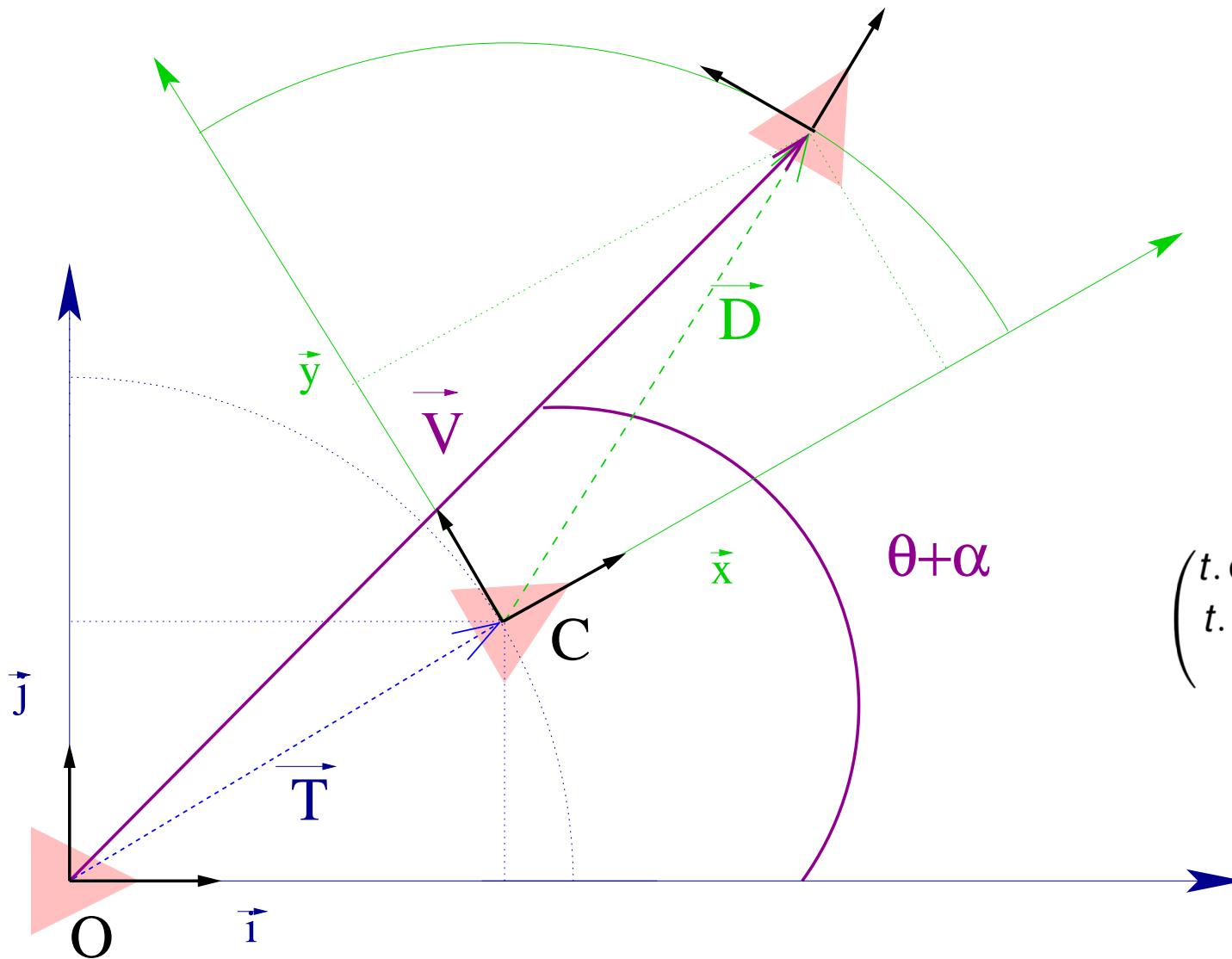
$$\vec{T}_{\Omega_O} = \begin{pmatrix} u \\ v \end{pmatrix} = t \cdot \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix} \quad (1)$$

- Puis, le robot tourne de α puis avance de d .

Dans le repère $\Omega_C = (C, \vec{x}, \vec{y})$

$$\vec{D}_{\Omega_C} = \begin{pmatrix} p \\ q \end{pmatrix} = d \cdot \begin{pmatrix} \cos \alpha \\ \sin \alpha \end{pmatrix} \quad (2)$$

Question : Déterminer la position du robot dans Ω_o .



Solution :

- la position du robot est égale à :

$$\vec{V} = \vec{T}_{\Omega_O} + \vec{D}_{\Omega_O} \quad (3)$$

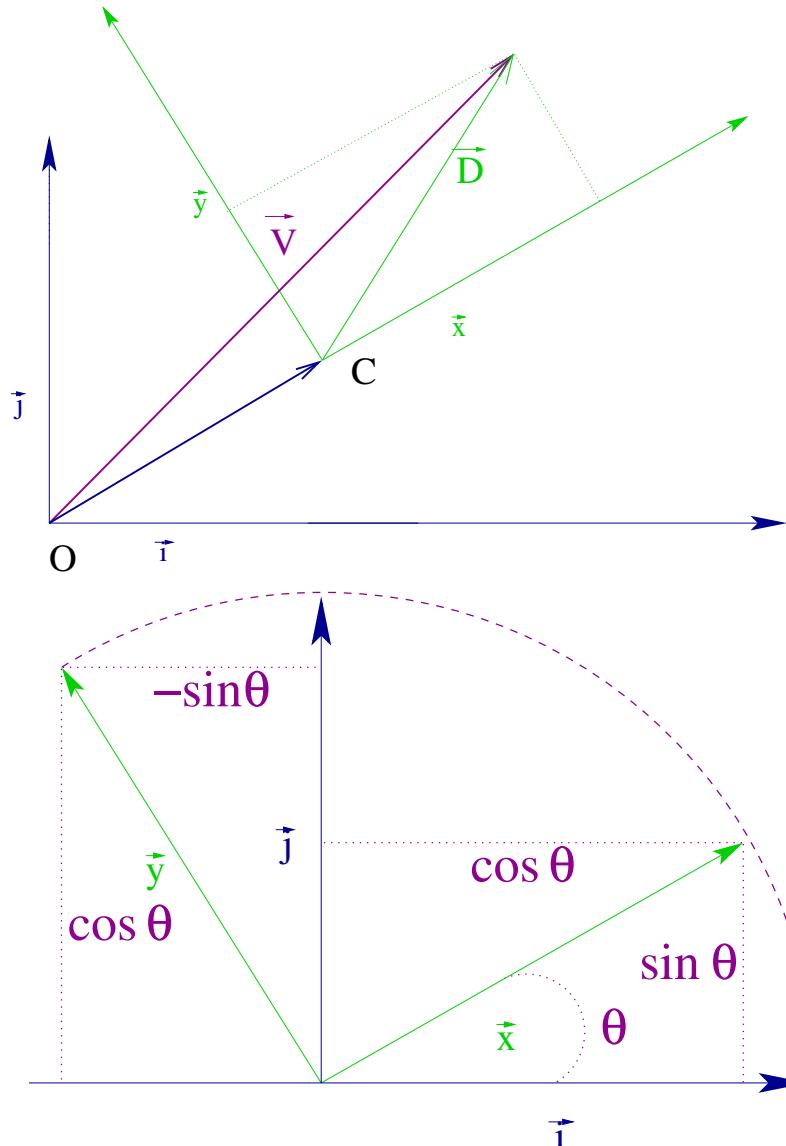
Sous-problème :

Déterminer \vec{D} dans Ω_O

- l'orientation de Toto est égale à $\theta + \alpha$.

$$\begin{pmatrix} t \cdot \cos \theta + p_{\Omega_O} \\ t \cdot \sin \theta + q_{\Omega_O} \\ \theta + \alpha \end{pmatrix} \quad (4)$$

Déterminer D dans Ω_O



Dans Ω_O le vecteur \vec{D} peut s'exprimer en fonction des axes de \vec{x} et \vec{y} .

$$\vec{D}_{\Omega_O} = p \cdot \vec{x}_{\Omega_O} + q \cdot \vec{y}_{\Omega_O} \quad (5)$$

Exprimons les axes \vec{x} et \vec{y} dans Ω_O

$$\vec{x}_{\Omega_O} = \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix} \quad \vec{y}_{\Omega_O} = \begin{pmatrix} -\sin \theta \\ \cos \theta \end{pmatrix} \quad (6)$$

Déterminer D dans Ω_0

eq. (6) \rightarrow eq. (5) \rightarrow eq. (3)

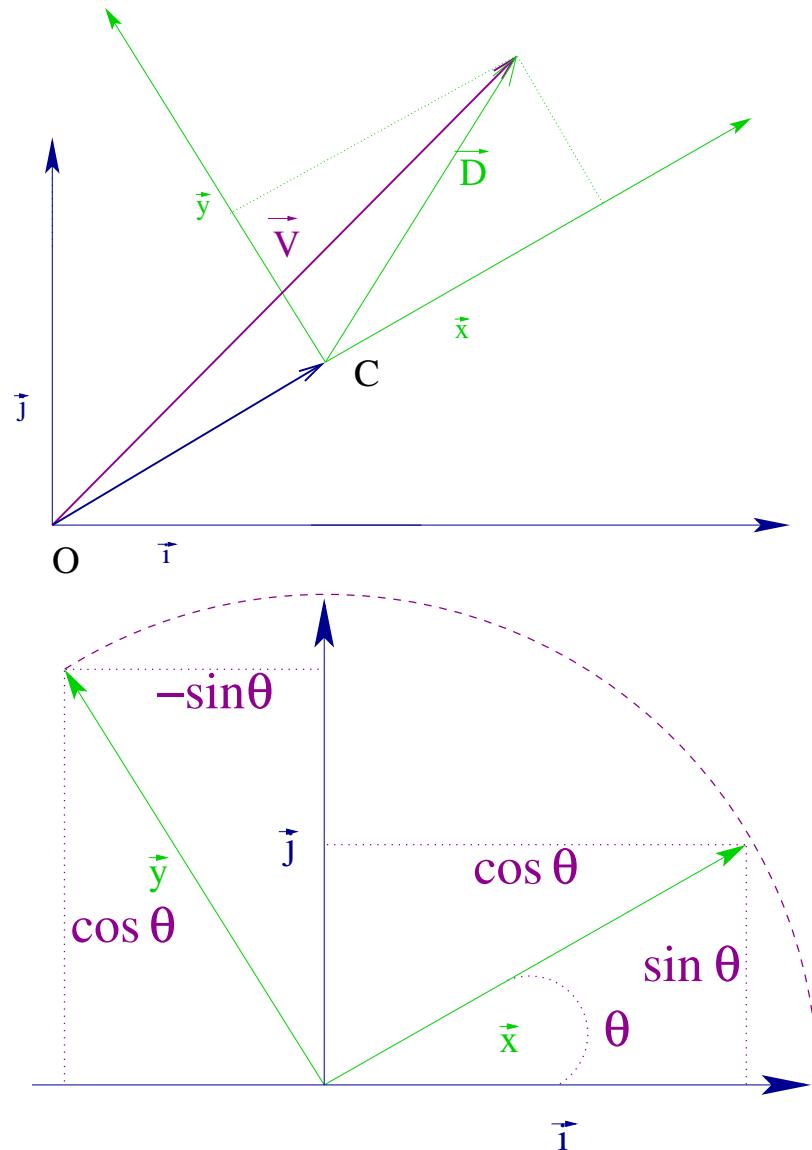
$$\vec{V} = \vec{T}_{\Omega_0} + \vec{D}_{\Omega_0} = t \cdot \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix} + p \cdot \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix} + q \cdot \begin{pmatrix} -\sin \theta \\ \cos \theta \end{pmatrix} \quad (7)$$

eq. (2) \rightarrow eq. (7)

$$\vec{V} = t \cdot \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix} + d \cdot \cos \alpha \cdot \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix} + d \cdot \sin \alpha \cdot \begin{pmatrix} -\sin \theta \\ \cos \theta \end{pmatrix} \quad (8)$$

$$= t \cdot \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix} + d \cdot \begin{pmatrix} \cos(\theta + \alpha) \\ \sin(\theta + \alpha) \end{pmatrix} \quad (9)$$

Déterminer D dans Ω_O



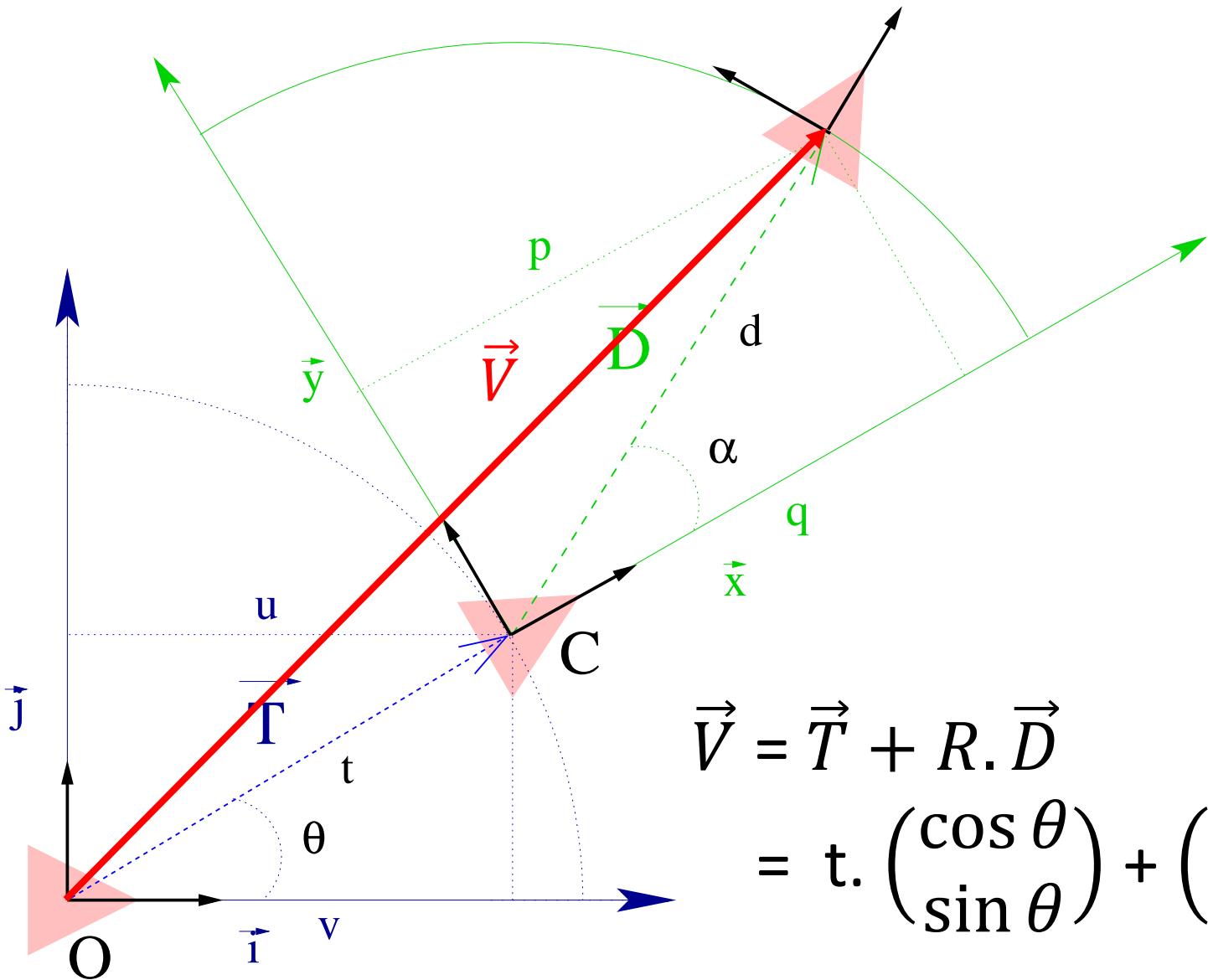
Dans Ω_O le vecteur \vec{D} peut s'exprimer en fonction des axes de \vec{x} et \vec{y} .

$$\vec{D}_{\Omega_O} = p \cdot \vec{x}_{\Omega_O} + q \cdot \vec{y}_{\Omega_O}$$

Exprimons les axes \vec{x} et \vec{y} dans Ω_O

$$\vec{x}_{\Omega_O} = \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix} \quad \vec{y}_{\Omega_O} = \begin{pmatrix} -\sin \theta \\ \cos \theta \end{pmatrix}$$

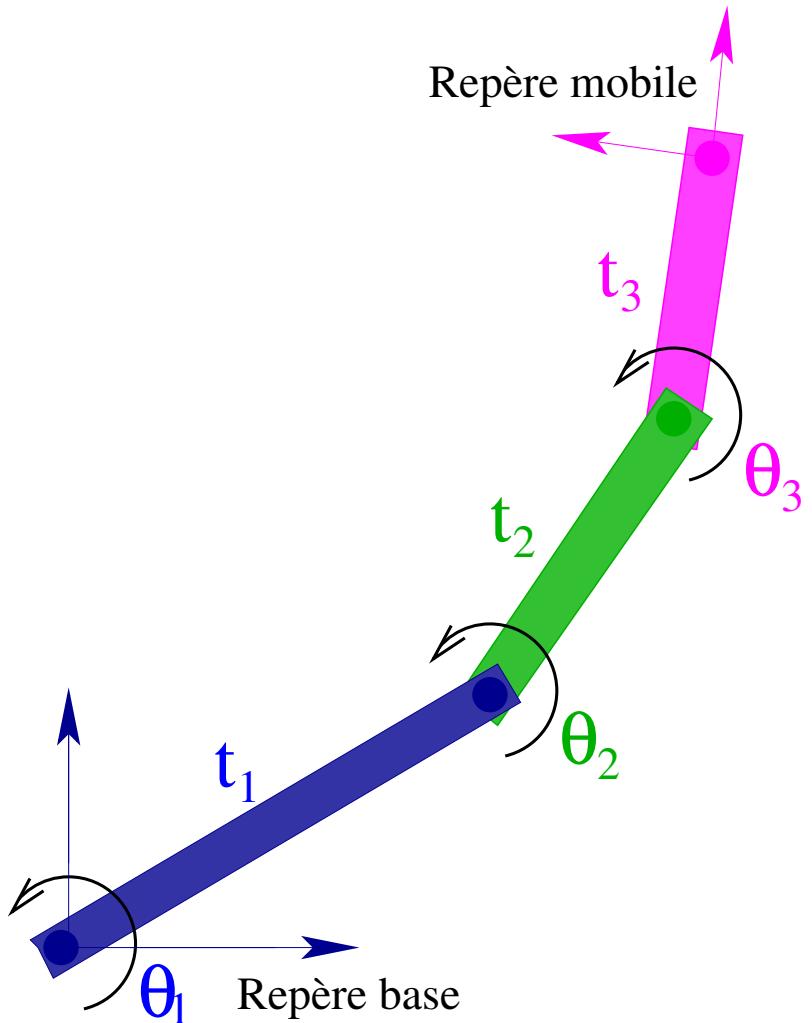
$$\begin{aligned}\vec{D}_{\Omega_O} &= p \cdot \vec{x}_{\Omega_O} + q \cdot \vec{y}_{\Omega_O} \\ &= p \cdot \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix} + q \cdot \begin{pmatrix} -\sin \theta \\ \cos \theta \end{pmatrix} \\ &= \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} p \\ q \end{pmatrix} \\ &= \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \vec{D}_{\Omega_C} \\ &= (\vec{x}_{\Omega_O} \quad \vec{y}_{\Omega_O}) \vec{D}_{\Omega_C} \\ &= R \cdot \vec{D}_{\Omega_C}\end{aligned}$$



$$\begin{aligned}
 \vec{V} &= \vec{T} + R \cdot \vec{D} \\
 &= t \cdot \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix} + \begin{pmatrix} \cos \theta \\ \sin \theta \end{pmatrix}
 \end{aligned}$$

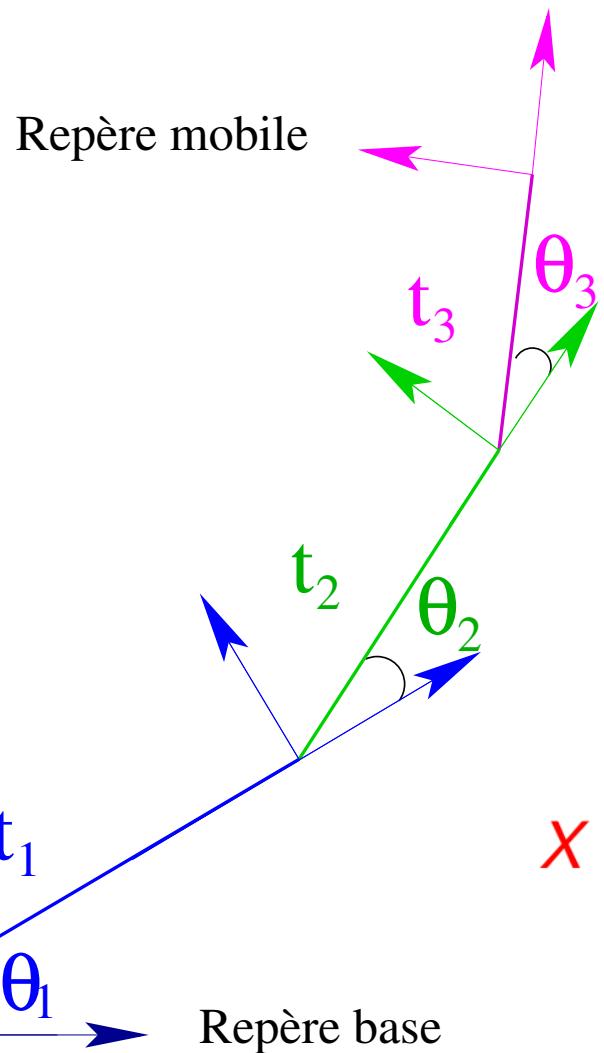
$$\begin{pmatrix} -\sin \theta \\ \cos \theta \end{pmatrix} \cdot d \cdot \begin{pmatrix} \cos \alpha \\ \sin \alpha \end{pmatrix}$$

Modèles Directe / Inverse



- **Dans quel espace ?** Quels sont les degrés de liberté de ce mécanisme plan 3R ?
- **Quel sont les variables ?** Entrées-sorties coordonnées articulaires, généralisé, paramètres qui définissent le système
- **Poser le problème ?**
Sortie = $F_{\text{directe}}(\text{entrée})$, Entrée = $F_{\text{directe}}(\text{sortie})$
- **Écrire les équations/constraints ?**
 - Associer à chacune des articulations un repère
 - Déterminer le positionnement (matrice R , vecteur P) de chaque repères par rapport au précédent.
 - Mettre ces changements de repères sous la forme de matrice homogène
 - Montrer comment calculer le MGD de ce mécanisme

Modèles Directe : sorties en fonction des entrées

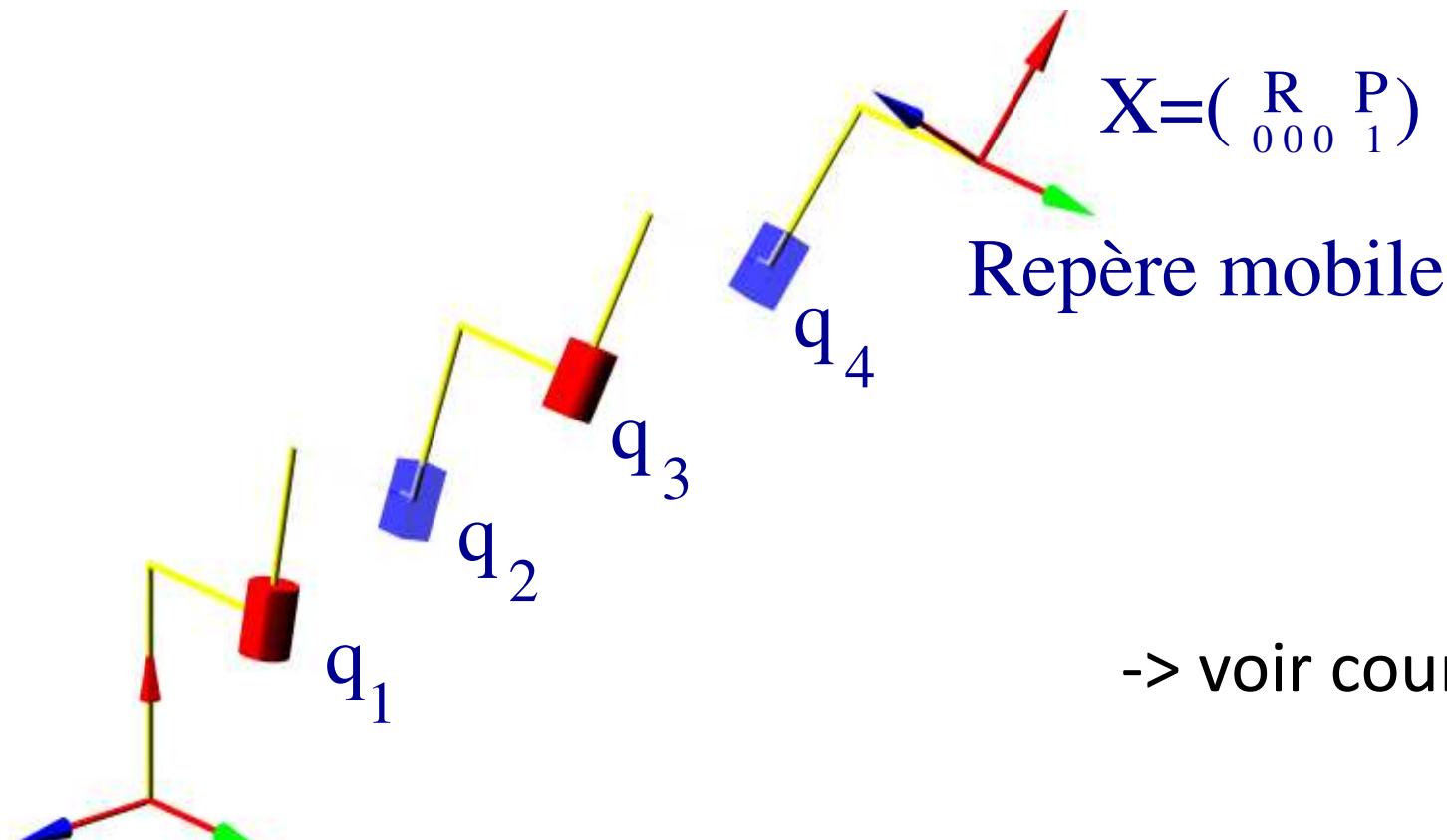


$$H_{0,3} = \begin{pmatrix} \cos \theta_1 & -\sin \theta_1 & t_1 \cdot \cos \theta_1 \\ \sin \theta_1 & \cos \theta_1 & t_1 \cdot \sin \theta_1 \\ 0 & 0 & 1 \end{pmatrix} \times \dots$$

$$= \begin{pmatrix} \cos \theta_2 & -\sin \theta_2 & t_2 \cdot \cos \theta_2 \\ \sin \theta_2 & \cos \theta_2 & t_2 \cdot \sin \theta_2 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos \theta_3 & -\sin \theta_3 & t_3 \cdot \cos \theta_3 \\ \sin \theta_3 & \cos \theta_3 & t_3 \cdot \sin \theta_3 \\ 0 & 0 & 1 \end{pmatrix}$$

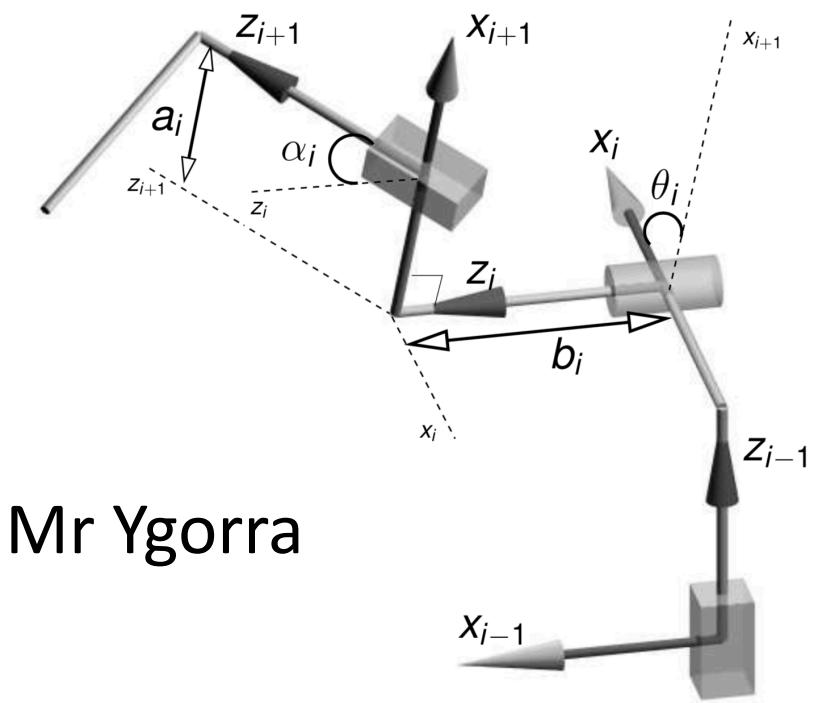
$$X = \begin{pmatrix} t_1 \cdot \cos \theta_1 + t_2 \cdot \cos(\theta_1 + \theta_2) + t_3 \cdot \cos(\theta_1 + \theta_2 + \theta_3) \\ t_1 \cdot \sin \theta_1 + t_2 \cdot \sin(\theta_1 + \theta_2) + t_3 \cdot \sin(\theta_1 + \theta_2 + \theta_3) \\ \theta_1 + \theta_2 + \theta_3 \end{pmatrix}$$

Modèle de robot série



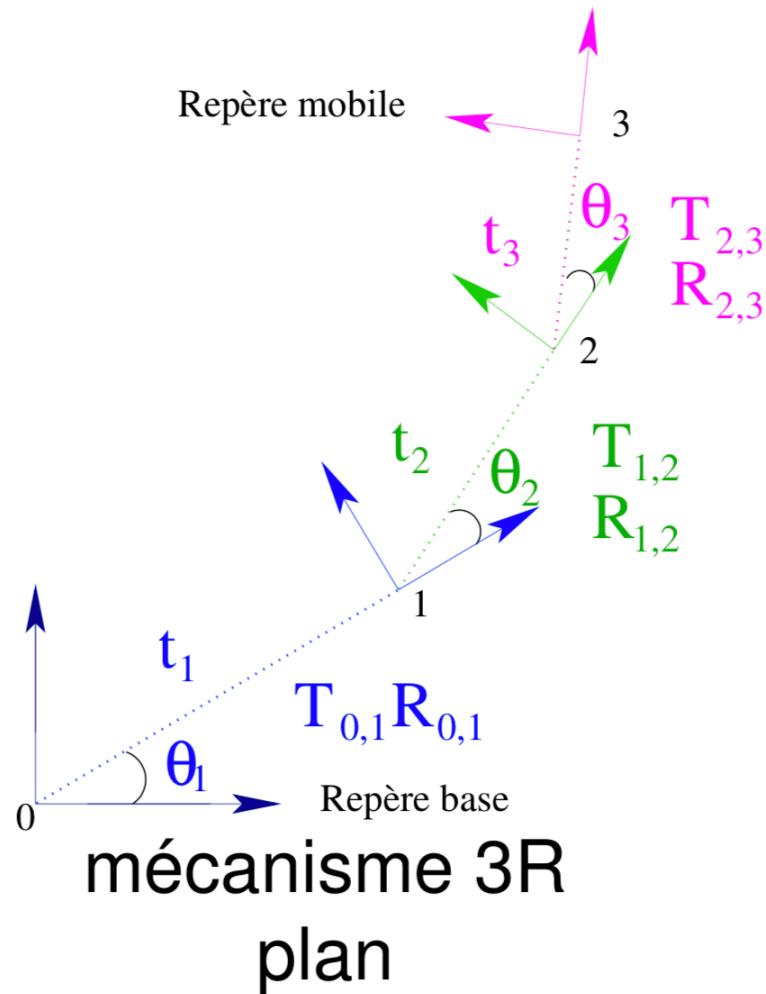
Repère mobile

-> voir cours Mr Ygorra



Modèles Inverse : entrées en fonction des sorties

Souvent -> résolution de système



$$X = \dots \\ \begin{pmatrix} t_1 \cdot \cos \theta_1 + t_2 \cdot \cos(\theta_1 + \theta_2) + t_3 \cdot \cos(\theta_1 + \theta_2 + \theta_3) \\ t_1 \cdot \sin \theta_1 + t_2 \cdot \sin(\theta_1 + \theta_2) + t_3 \cdot \sin(\theta_1 + \theta_2 + \theta_3) \\ \theta_1 + \theta_2 + \theta_3 \end{pmatrix}$$

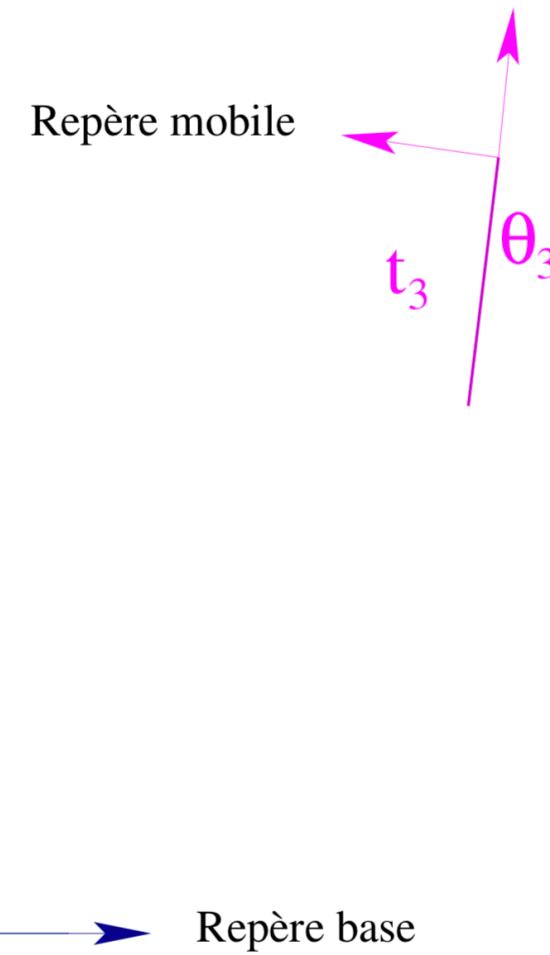
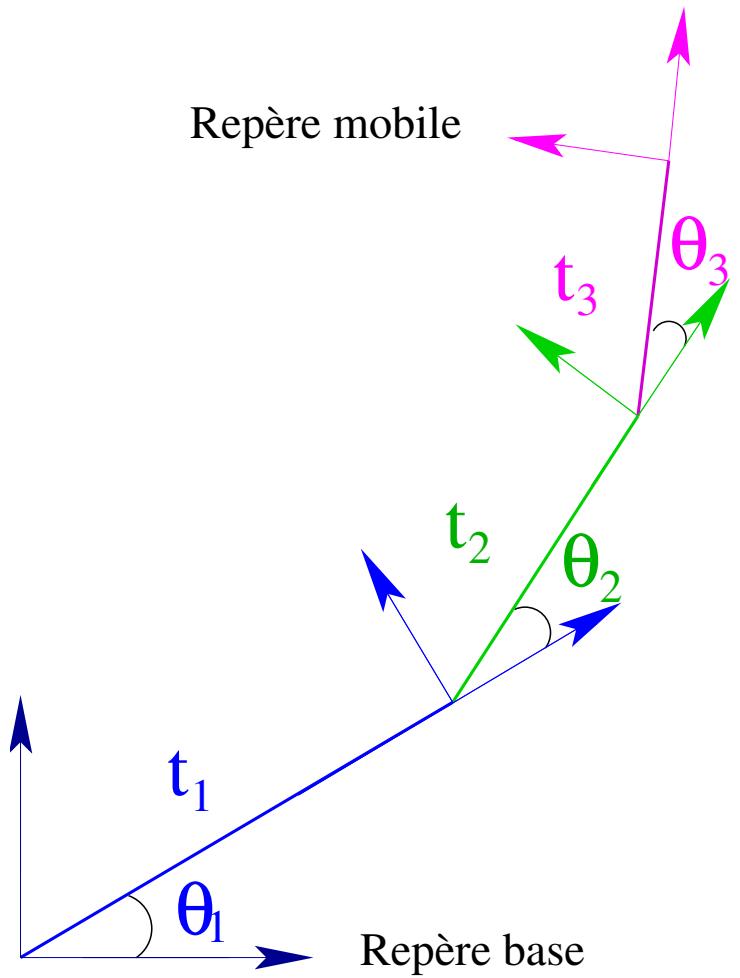
Calculer le MGI, c'est déterminer:

$$[\theta_1, \theta_2, \theta_3] = \mathcal{F}_{\text{MGI}}(X_1, X_2, X_3, \zeta)$$

avec $\zeta = [t_1, t_2, t_3]$

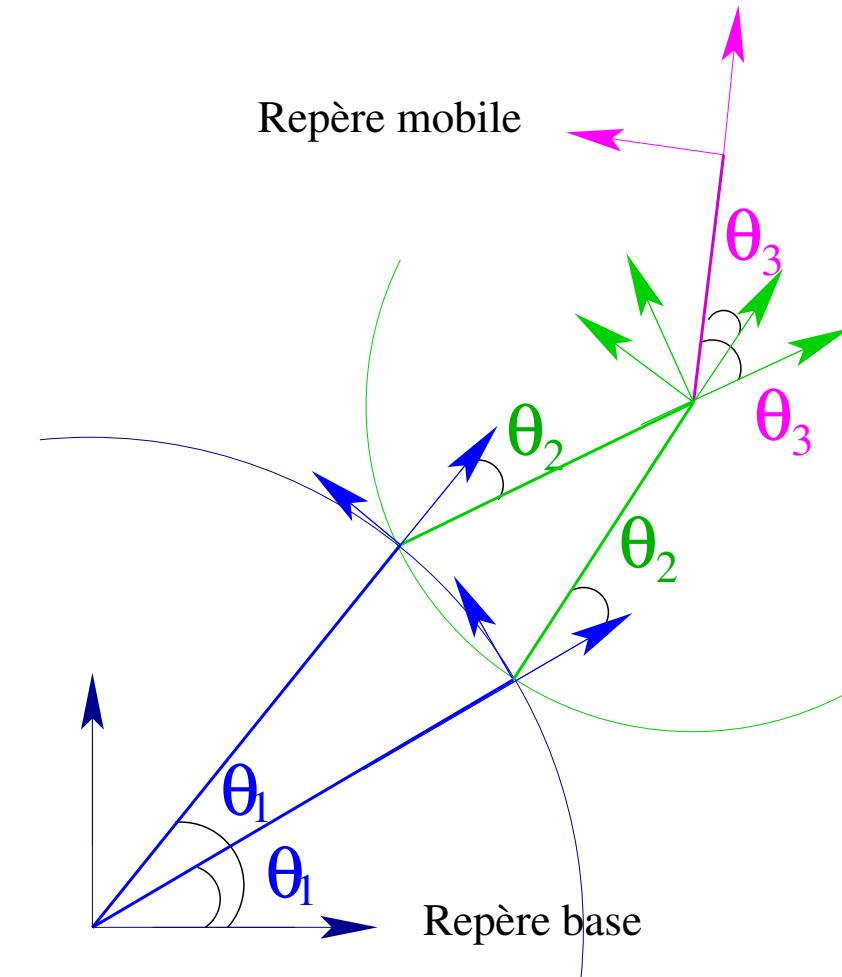
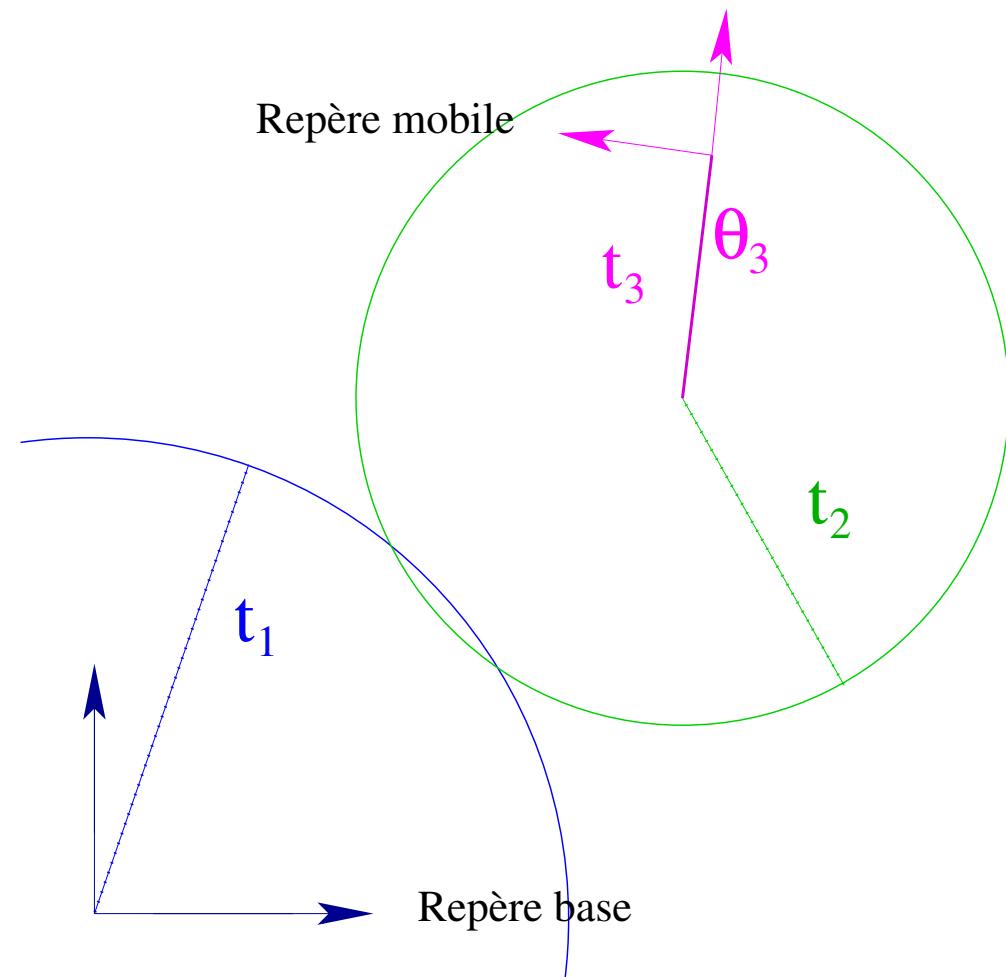
Modèles Inverse : entrées en fonction des sorties

Souvent -> résolution de système



Modèles Inverse : entrées en fonction des sorties

Souvent -> résolution de système



Modèles Inverse : entrées en fonction des sorties

résolution Algébrique 1

$$t_1 \cdot \cos \theta_1 + t_2 \cdot \cos (\theta_1 + \theta_2) + t_3 \cdot \cos (\theta_1 + \theta_2 + \theta_3) - X_1 = 0$$

$$t_1 \cdot \sin \theta_1 + t_2 \cdot \sin (\theta_1 + \theta_2) + t_3 \cdot \sin (\theta_1 + \theta_2 + \theta_3) - X_2 = 0$$

$$\theta_1 + \theta_2 + \theta_3 = X_3$$

$$t_1 \cdot \cos \theta_1 + t_2 \cdot \cos (\theta_1 + \theta_2) + t_3 \cdot \cos X_3 - X_1 = 0$$

$$t_1 \cdot \sin \theta_1 + t_2 \cdot \sin (\theta_1 + \theta_2) + t_3 \cdot \sin X_3 - X_2 = 0$$

$$t_1 \cdot \cos \theta_1 + t_2 \cdot \cos (\theta_1 + \theta_2) = u_1 \tag{18}$$

$$t_1 \cdot \sin \theta_1 + t_2 \cdot \sin (\theta_1 + \theta_2) = u_2$$

On sait que

$$\cos^2 (\theta_1 + \theta_2) + \sin^2 (\theta_1 + \theta_2) = 1 \tag{19}$$

Modèles Inverse : entrées en fonction des sorties

résolution Algébrique 2

En reportant, les équations 18 dans l'équation 19.

$$(u_1 - t_1 \cdot \cos \theta_1)^2 + (u_2 - t_1 \cdot \sin \theta_1)^2 = t_2^2$$

Nous obtenons

$$u_1 \cdot \cos \theta_1 + u_2 \cdot \sin \theta_1 = \frac{t_1^2 - t_2^2 + u_1^2 + u_2^2}{2 \cdot t_1}$$

sachant que pour l'équation $X \cdot \sin \alpha + Y \cdot \cos \alpha = Z$:

$$\cos \alpha = \frac{YZ - \epsilon X \sqrt{X^2 + Y^2 - Z^2}}{X^2 + Y^2}$$

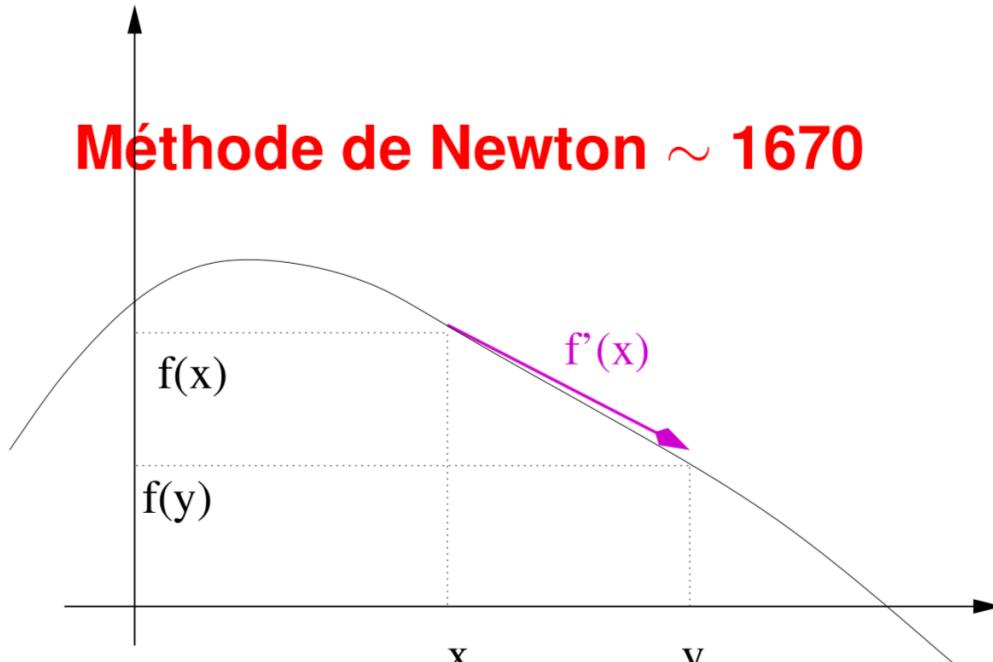
$$\sin \alpha = \frac{XZ + \epsilon Y \sqrt{X^2 + Y^2 - Z^2}}{X^2 + Y^2}$$

avec $\epsilon = + / - 1$.

On en déduit donc θ_1 puis $\theta_1 + \theta_2 \rightarrow \theta_2$ (en utilisant eq. (18)), puis enfin θ_3 .

Modèles Inverse : entrées en fonction des sorties

Résolution numérique



$$\lim_{h \rightarrow 0} \frac{f(x) - f(x+h)}{h} = f'(x)$$

Le schéma de Newton est donc :

Nous cherchons à déterminer x tel que $f(x) = 0$, Nous connaissons une approximation de x noté x_0 .

Nous avons

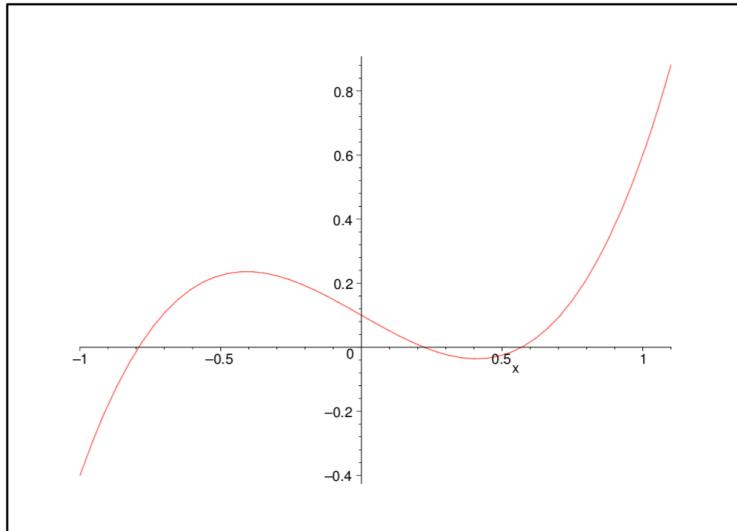
$f(x_0) - f(x) = f'(x_0).(x_0 - x)$ avec
 $f(x) = 0$ nous obtenons :

$$x = x_0 - \frac{f(x_0)}{f'(x_0)}$$

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

Modèles Inverse : entrées en fonction des sorties

Newton



$$x^3 - 0.5 \times x + 0.1 = 0$$

- $f(x) = x^3 - 0.5 \times x + 0.1$

- $f'(x) = 3x^2 - 0.5$

- $x_{k+1} = x_k - \frac{x^3 - 0.5 \times x + 0.1}{3x^2 - 0.5}$

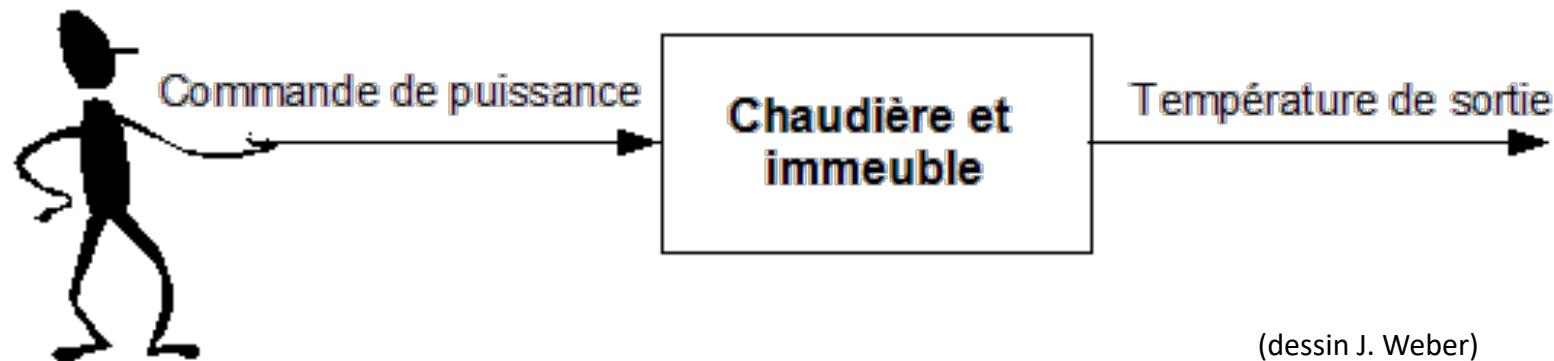
x_0	0	1	-0.5	-0.4
x_1	0.2	0.76	-1.4	11.4
x_2	0.2211	0.6310	-1.0387	7.6095
x_3	0.2218	0.5796	-0.8555	5.0871
x_4		0.5699	-0.7975	3.4121
x_5		0.5696	-0.7915	2.3048
x_6			-0.7914	1.5799
x_7				1.1143
x_8				0.8270
x_9				0.6645
x_{10}				0.5903
x_{11}				0.5710
x_{12}				0.5696

Modèle Inverse : entrées en fonction des sorties

Souvent -> résolution de système

- A quoi ça sert en robotique ?

Exemple



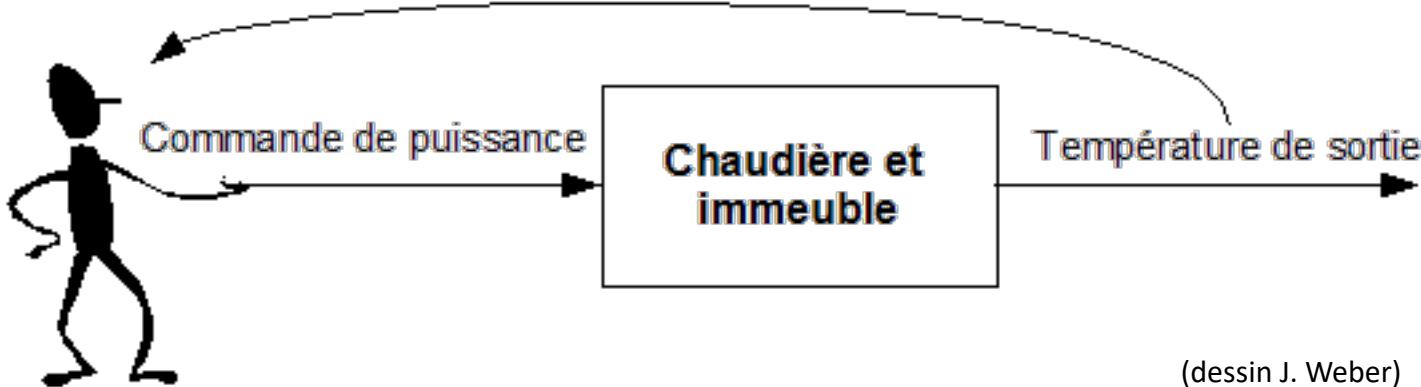
(dessin J. Weber)

Modèle Inverse : entrées en fonction des sorties

Souvent -> résolution de système

- A quoi ça sert en robotique ?

Exemple : nécessité d'une boucle de retro-action

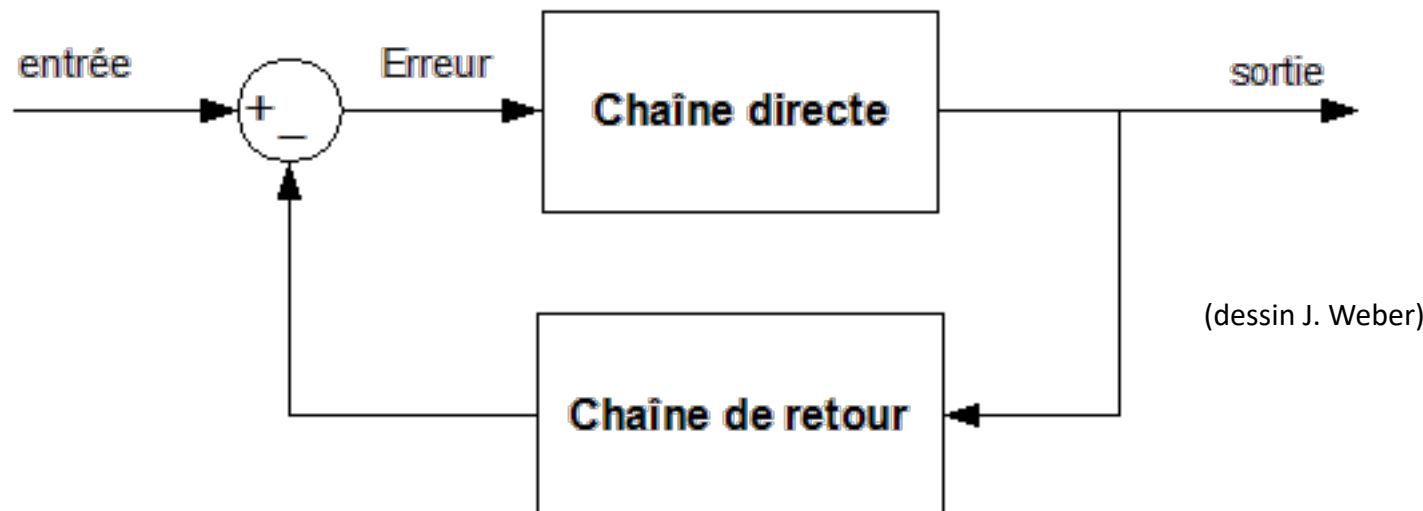


Modèle Inverse : entrées en fonction des sorties

Souvent -> résolution de système

- A quoi ça sert en robotique ?

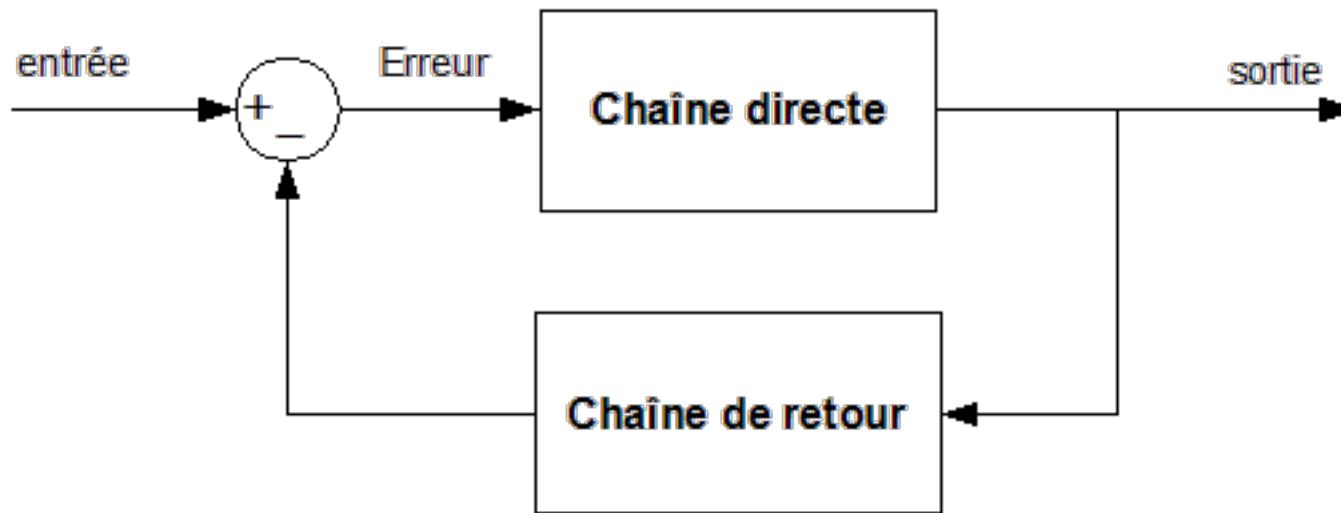
Exemple : nécessité d'une boucle de retro-action



Modèle Inverse : entrées en fonction des sorties

Souvent -> résolution de système

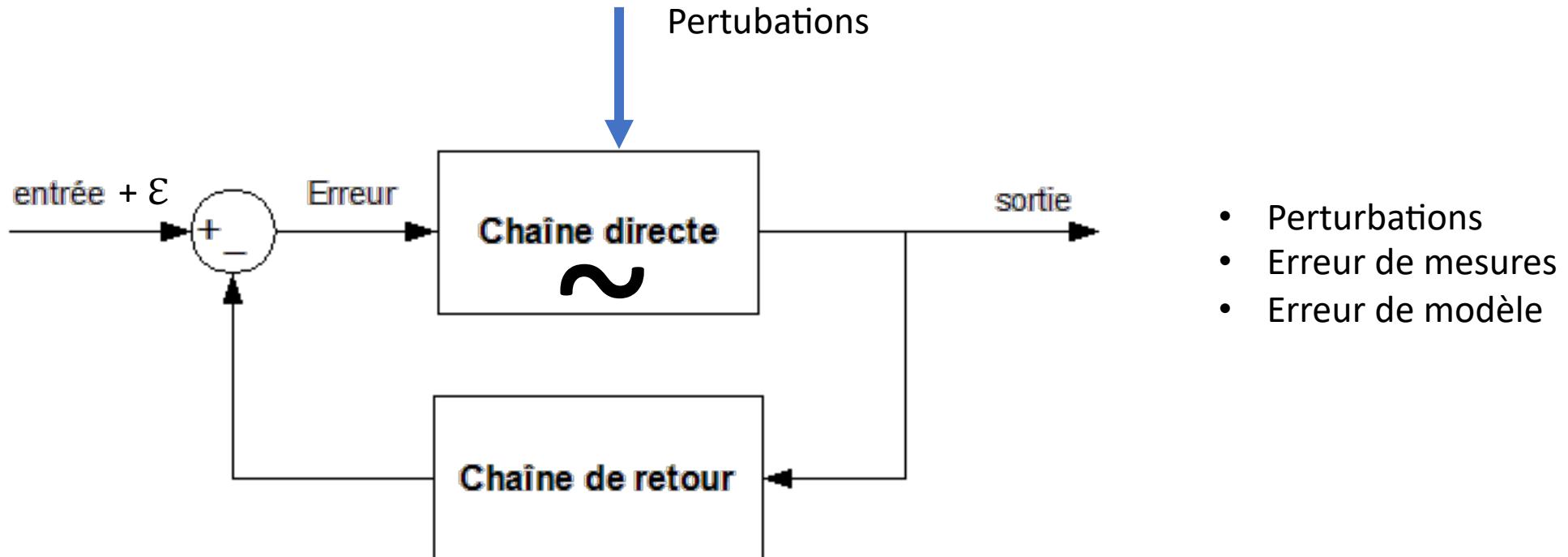
- Pourquoi une boucle de retro-action ?



Modèle Inverse : entrées en fonction des sorties

Souvent -> résolution de système

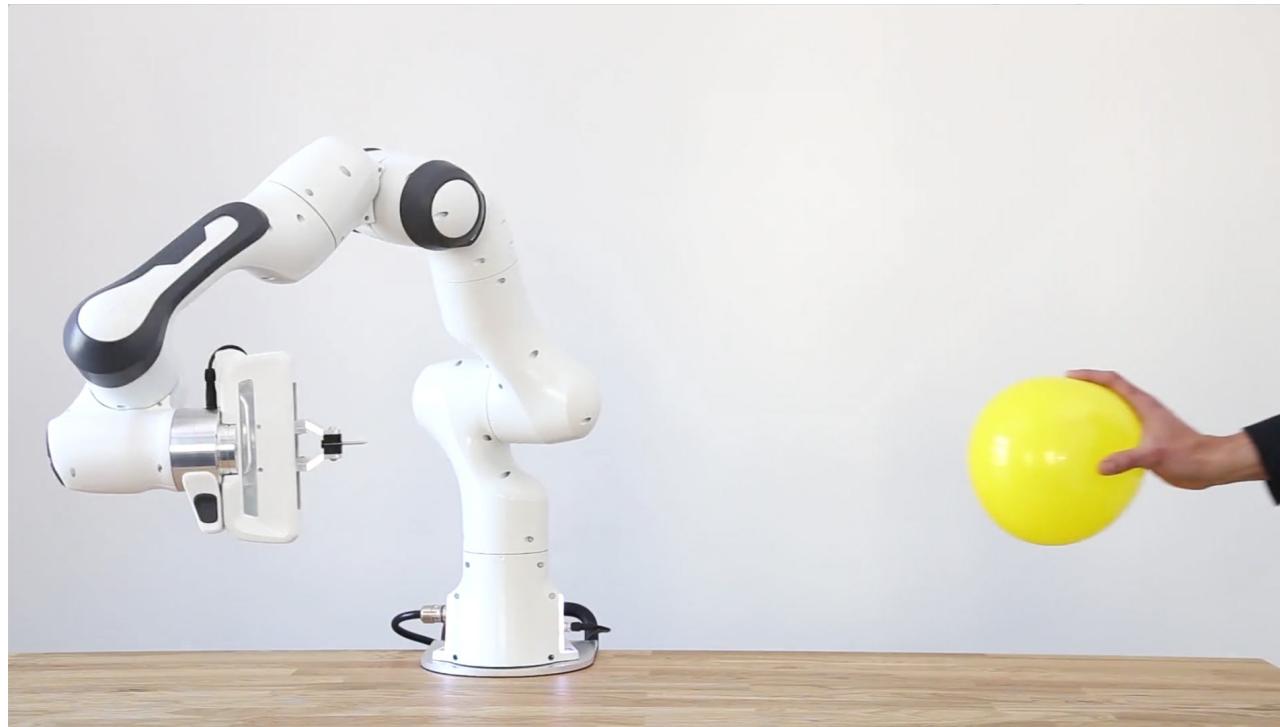
- Pourquoi une boucle de retro-action ?



Modèle Inverse : entrées en fonction des sorties

Souvent -> résolution de système

Réfléchir sur la commande d'un robot

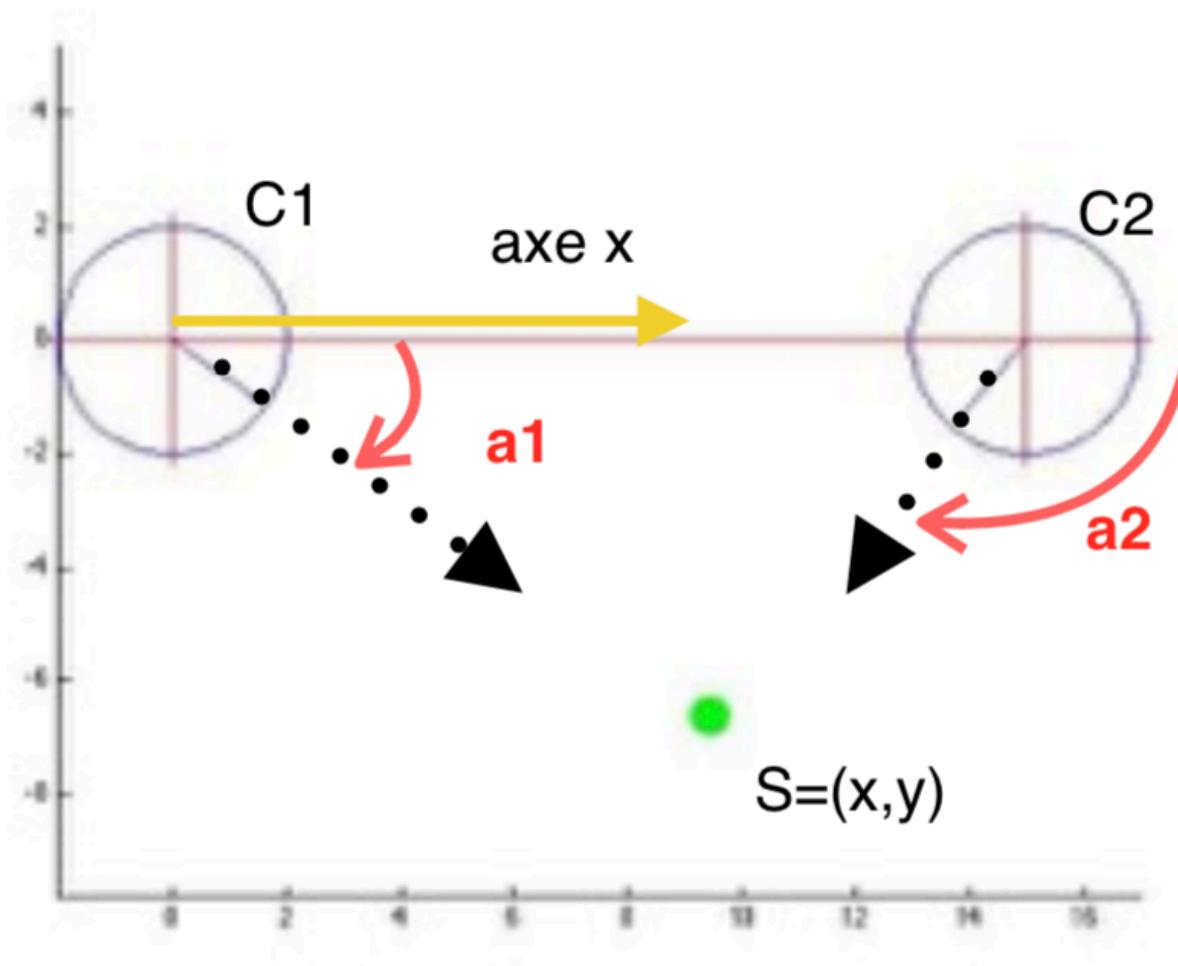


Modèle Inverse : entrées en fonction des sorties

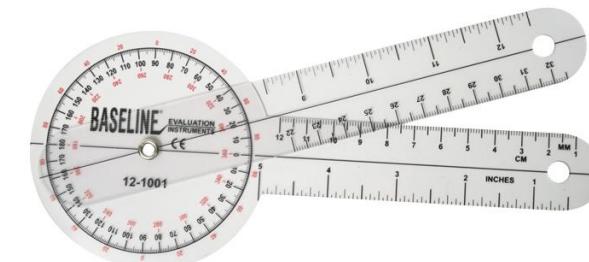
Souvent -> résolution de système

- Quel est la forme, la complexité du système étudié ?
- Quel est la caractérisation de la solution ?

Un exemple simple



C1, C2 position des goniomètres



a1, a2 mesures angularaire

S position du robot

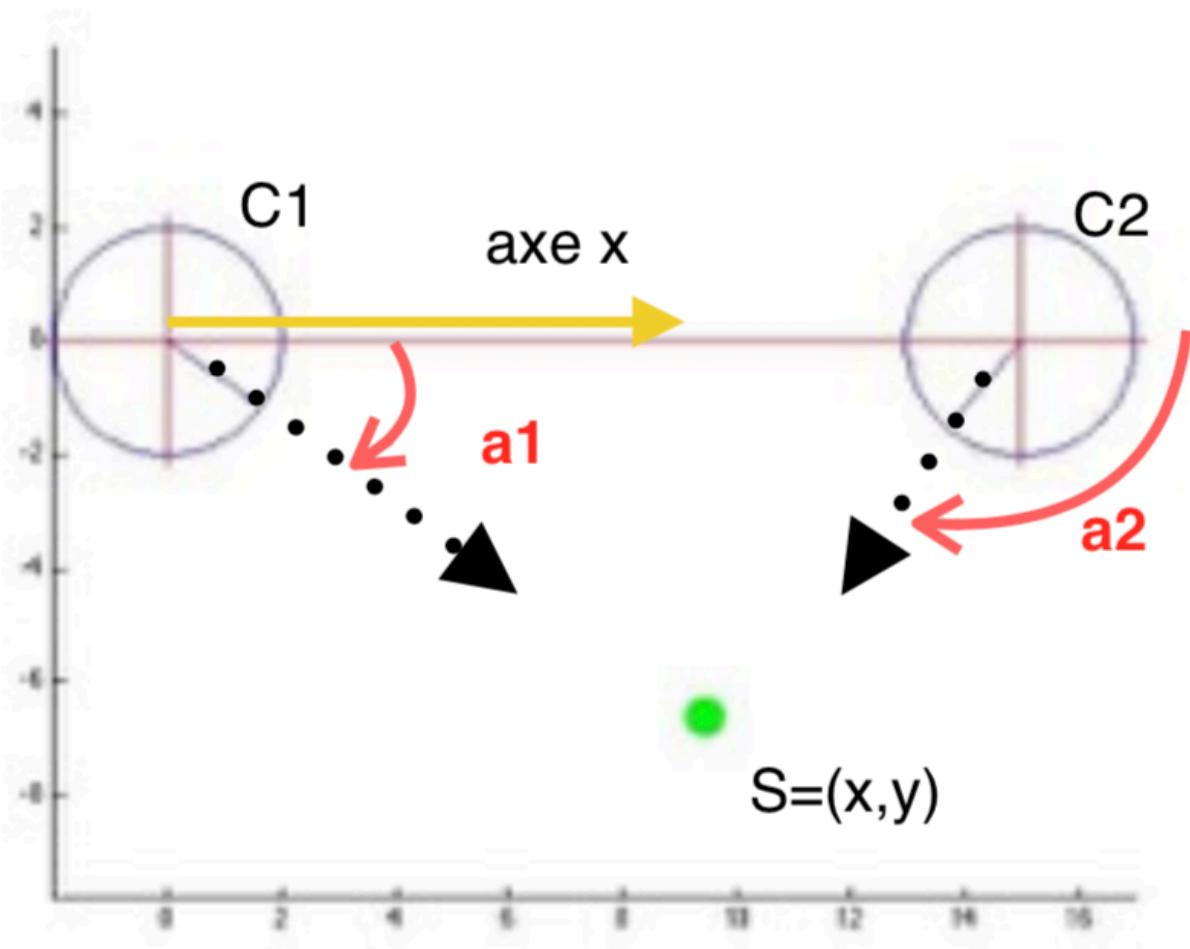
Mettre en équations:
Quels sont les paramètres du problème ?
Donner des coordonnées à C1 et C2
Relier x,y avec C1 et C2 ...

Modèle Inverse : entrées en fonction des sorties

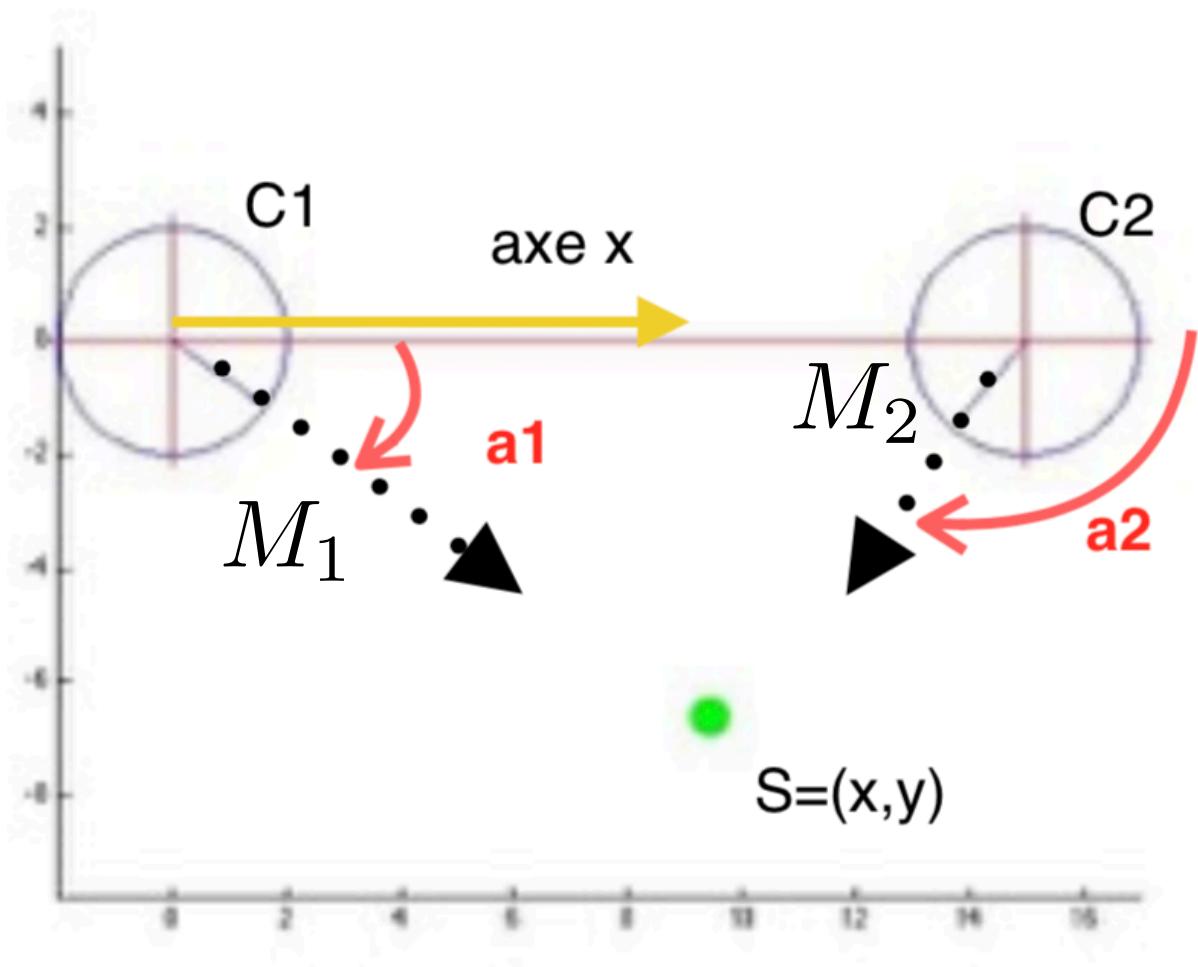
Souvent -> résolution de système

- Quel est la forme, la complexité du système étudié ?
-> Intersection de deux droite : système linéaire
- Quel est la caractérisation de la solution ?
 - -> Résolution d'un système bien contraint
(autant d'équation que d'inconnue)

Les équations



Les équations



$$M_1 = \begin{pmatrix} u_1 \\ v_1 \end{pmatrix} = \begin{pmatrix} \cos a_1 \\ \sin a_1 \end{pmatrix}$$

$$M_2 = \begin{pmatrix} u_2 \\ v_2 \end{pmatrix} = \begin{pmatrix} \cos a_2 \\ \sin a_2 \end{pmatrix}$$

$$\begin{pmatrix} x - c_{1x} \\ y - c_{1y} \end{pmatrix} \cdot \begin{pmatrix} \cos\left(\frac{\pi}{2}\right) & -\sin\left(\frac{\pi}{2}\right) \\ \sin\left(\frac{\pi}{2}\right) & \cos\left(\frac{\pi}{2}\right) \end{pmatrix} \cdot \begin{pmatrix} u_1 \\ v_1 \end{pmatrix} = 0$$

$$\begin{pmatrix} x - c_{2x} \\ y - c_{2y} \end{pmatrix} \cdot \begin{pmatrix} \cos\left(\frac{\pi}{2}\right) & -\sin\left(\frac{\pi}{2}\right) \\ \sin\left(\frac{\pi}{2}\right) & \cos\left(\frac{\pi}{2}\right) \end{pmatrix} \cdot \begin{pmatrix} u_2 \\ v_2 \end{pmatrix} = 0$$

Les équations

$$\begin{pmatrix} x - c_2x \\ y - c_2y \end{pmatrix} \cdot \begin{pmatrix} \cos\left(\frac{\pi}{2}\right) & -\sin\left(\frac{\pi}{2}\right) \\ \sin\left(\frac{\pi}{2}\right) & \cos\left(\frac{\pi}{2}\right) \end{pmatrix} \cdot \begin{pmatrix} u_2 \\ v_2 \end{pmatrix} = 0$$

$$\begin{pmatrix} x - c_2x \\ y - c_2y \end{pmatrix} \cdot \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} u_2 \\ v_2 \end{pmatrix} = 0$$

$$\begin{pmatrix} x - c_2x \\ y - c_2y \end{pmatrix} \cdot \begin{pmatrix} -v_2 \\ u_2 \end{pmatrix} = 0$$

$$-(x - c_2x).v_2 + (y - c_2y).u_2 = 0$$

$$(x - c_2x).v_2 - (y - c_2y).u_2 = 0$$

$$v_2.x - u_2.y + (u_2.c_2y - v_2.c_2x) = 0$$

$$v_2.x - u_2.y = v_2.c_2x - u_2.c_2y$$

Les équations

$$v_1 \cdot x - u_1 \cdot y = v_1 \cdot c_{1x} - u_1 \cdot c_{1y}$$

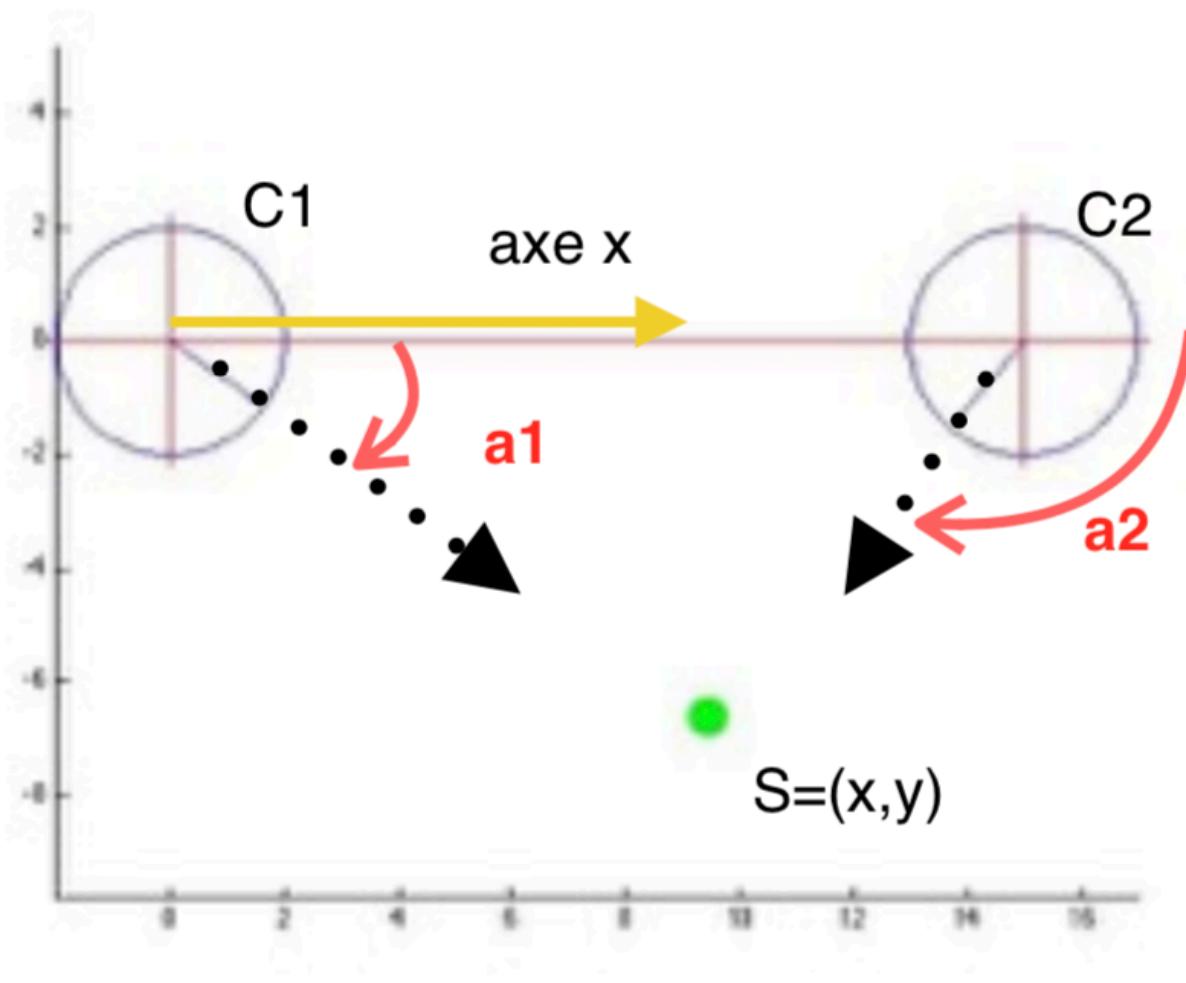
$$v_2 \cdot x - u_2 \cdot y = v_2 \cdot c_{2x} - u_2 \cdot c_{2y}$$

$$\sin a_1 \cdot x - \cos a_1 \cdot y = \sin a_1 \cdot c_{1x} - \cos a_1 \cdot c_{1y}$$

$$\sin a_2 \cdot x - \cos a_2 \cdot y = \sin a_2 \cdot c_{2x} - \cos a_2 \cdot c_{2y}$$

$$\begin{pmatrix} \sin(a_1) & -\cos(a_1) \\ \sin(a_2) & -\cos(a_2) \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \sin(a_1) \cdot c_{1x} - \cos(a_1) \cdot c_{1y} \\ \sin(a_2) \cdot c_{2x} - \cos(a_2) \cdot c_{2y} \end{pmatrix}$$

Les équations



Déterminer la position du robot
=> Résoudre le système

$$A \cdot S = B$$

avec

$$S = \begin{pmatrix} x \\ y \end{pmatrix}$$

$$A = \begin{pmatrix} \sin(a_1) & -\cos(a_1) \\ \sin(a_2) & -\cos(a_2) \end{pmatrix}$$

$$B = \begin{pmatrix} \sin(a_1).c_{1x} - \cos(a_1).c_{1y} \\ \sin(a_2).c_{2x} - \cos(a_2).c_{2y} \end{pmatrix}$$

TP dessine la scène

Dessine un cercle

```
function dessin_robot( a,c )  
[m,n]=size(c);  
  
function circle(x,y,r)  
    ang=0:0.01:2*pi;  
    xp=r*cos(ang);  
    yp=r*sin(ang);  
    plot(x+xp,y+yp);  
end  
  
for i=1:m  
    circle(c(i,1),c(i,2),2);  
    plot(c(i,1), c(i,2), '+r', 'MarkerSize',100);  
    R=[cos(a(i)) -sin(a(i)) ; sin(a(i)) cos(a(i))] ;  
    v=R*[30. 0]';  
    quiver(c(i,1),c(i,2),v(1),v(2),'LineWidth',1,'Color',[0 0 1]);  
end  
end
```

```

% efface tous
clear all;
clf;

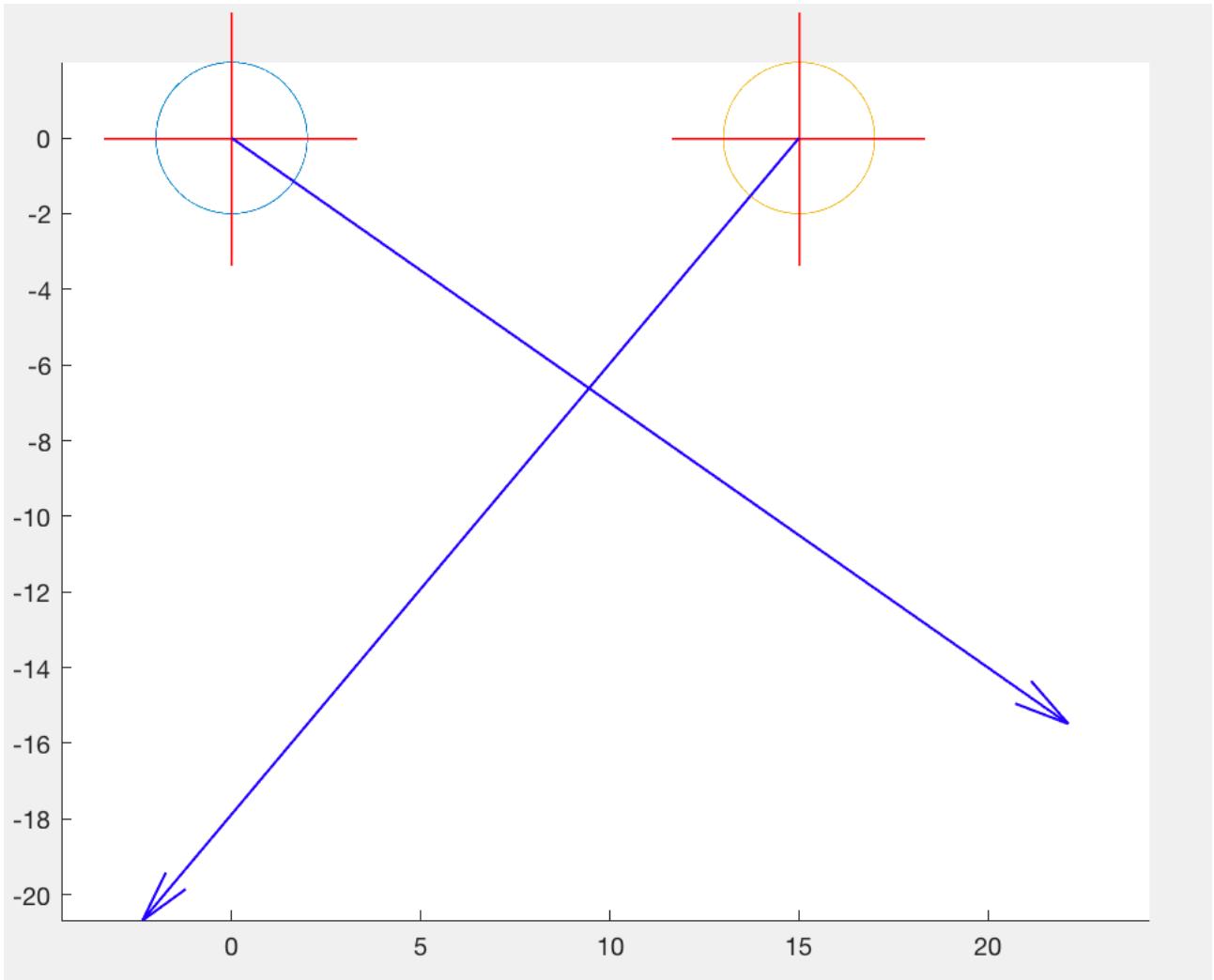
% gestion des axes des graphiques
hold on
axis equal;

% Coordonnées des balises
c1=[0 0];
c2=[15 0];

% Angles des deux capteurs
a1=-35*pi/180;
a2=-130*pi/180;

% dessine le robot
dessin_robot([a1 a2],[c1 ; c2])

```



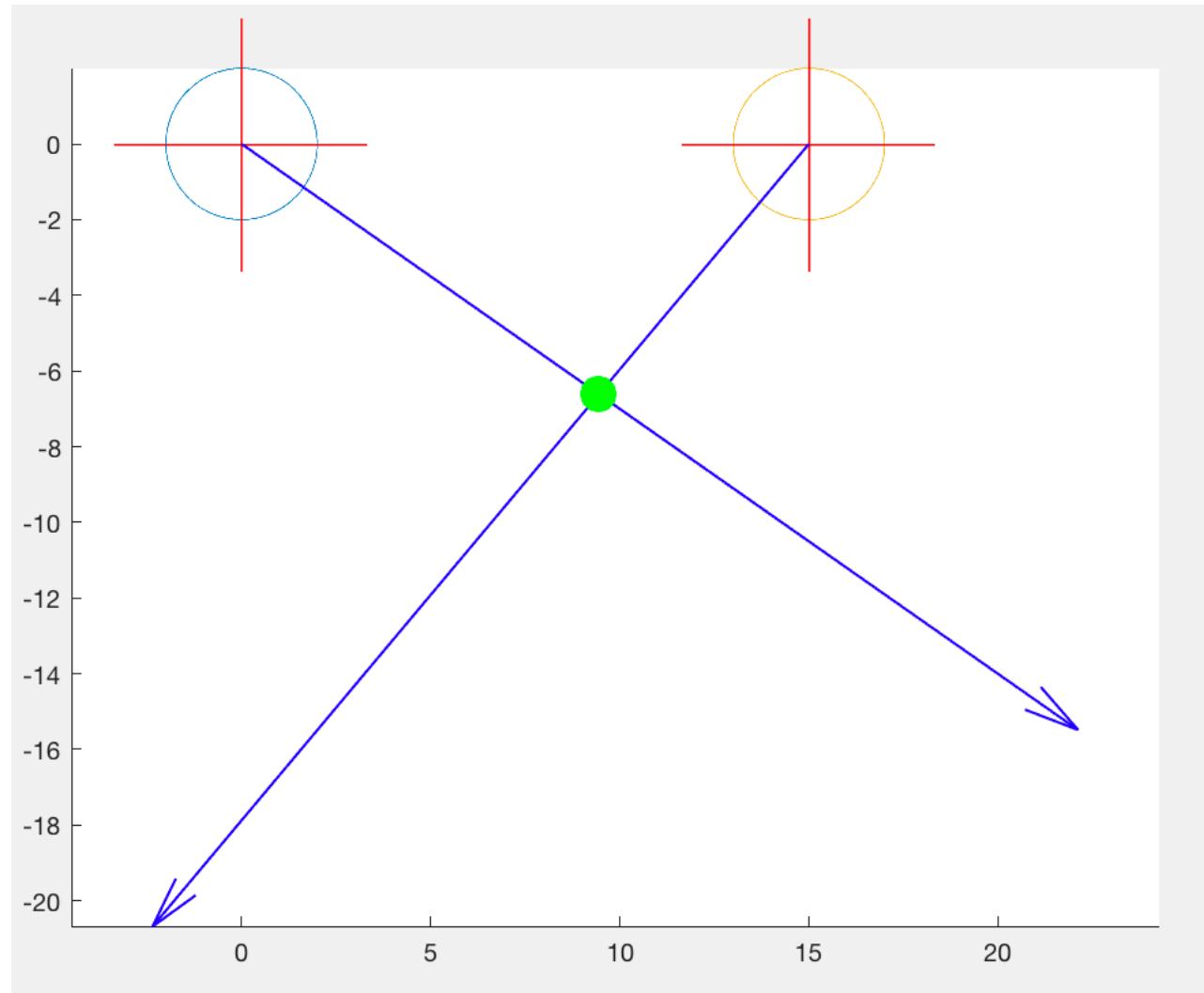
```
% equation droite passant par un point (cx,cy)
% de vecteur directeur (u,v) est
% v.x-u.y-(v.cx - u.cy) = 0
```

```
A=[sin(a1) -cos(a1) ;
 sin(a2) -cos(a2)]
```

```
b=[sin(a1)*c1(1)-cos(a1)*c1(2) ;
 sin(a2)*c2(1)-cos(a2)*c2(2) ]
```

```
s=A\b
```

```
plot(s(1), s(2), '.g', 'MarkerSize',50);
```



En faire une fonction ...

```
function [s,A,b] = sol_pos(a,c)
```

```
K=prod(size(a));
```

```
A=[];
```

```
b=[];
```

```
for i=1:K
```

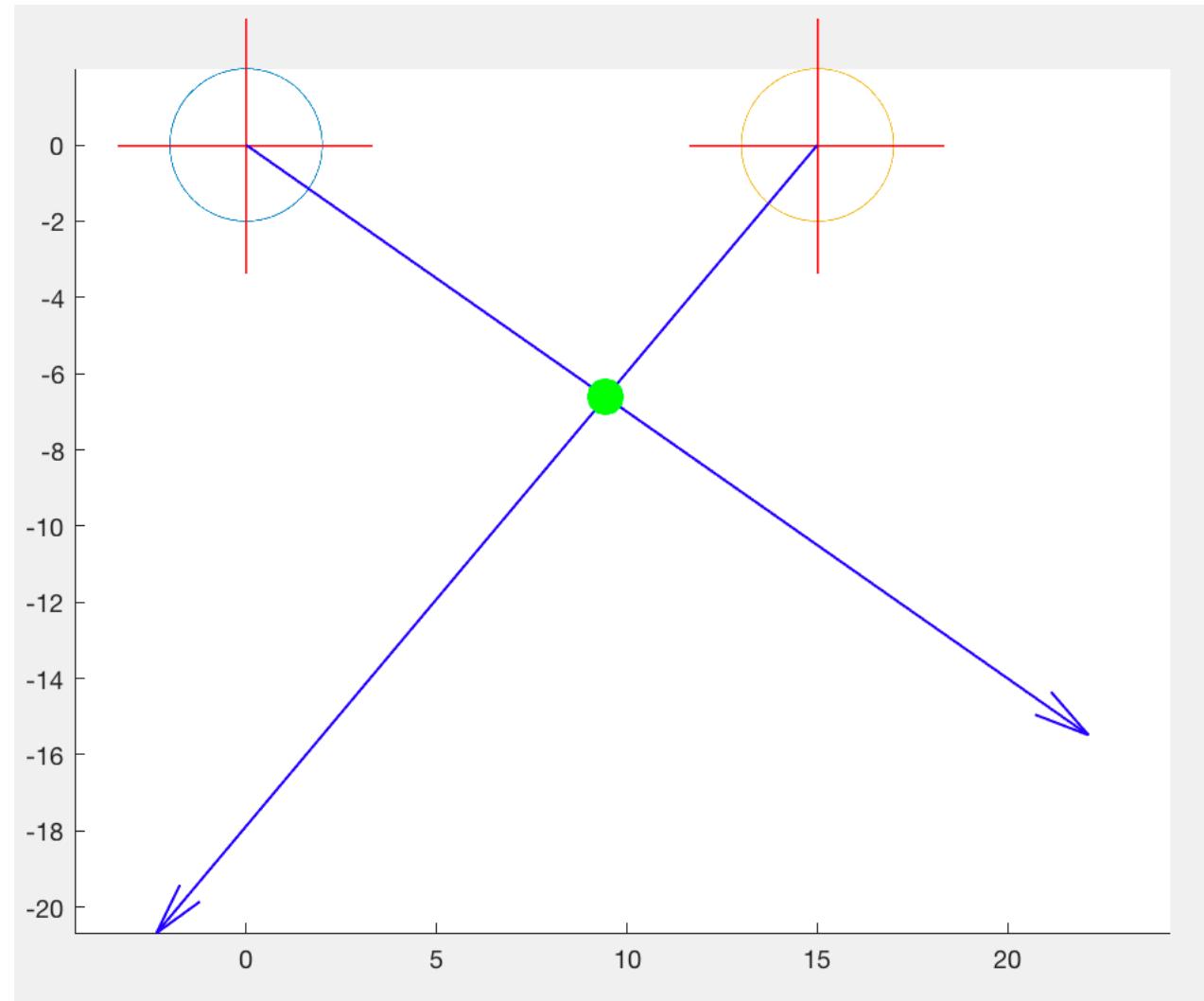
```
    A=[A ; sin(a(i)) -cos(a(i))];
```

```
    b=[b; sin(a(i))*c(i,1)-cos(a(i))*c(i,2)];
```

```
end
```

```
s=A\b;
```

```
end
```



Solve $A.x = b$

$$x = A^{-1}.b$$

Pas la bonne solution ...

$$[Q, R] = qr(A)$$

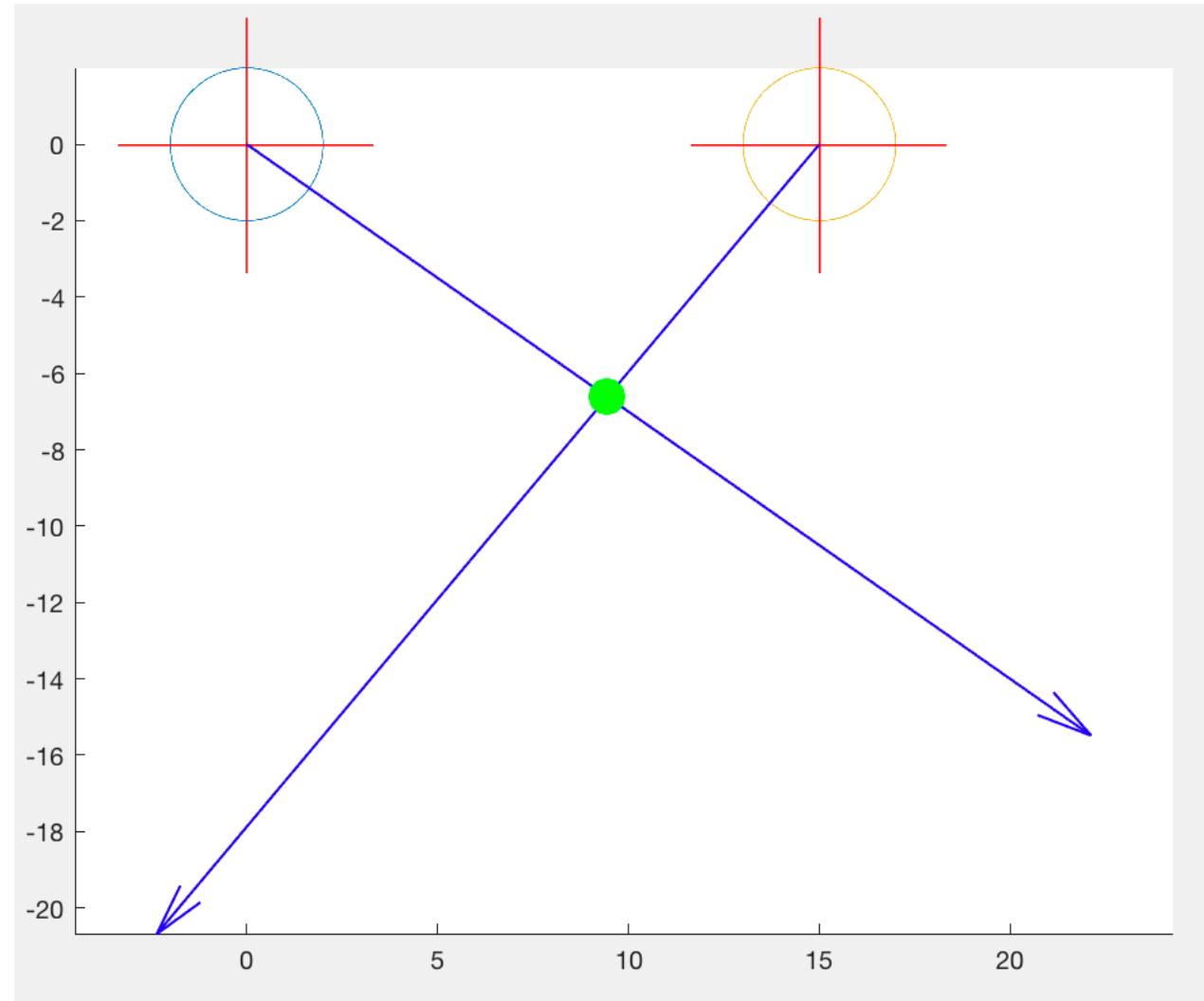
$$Q.R = A$$

$$Q.\underbrace{R.x}_y = b$$

Determine : $y = Q^T.b$

Solve : $x = R^{-1}.y$

avec R triangulaire



Solve $A.x = b$

$$x = A^{-1}.b$$

Pas la bonne solution ...

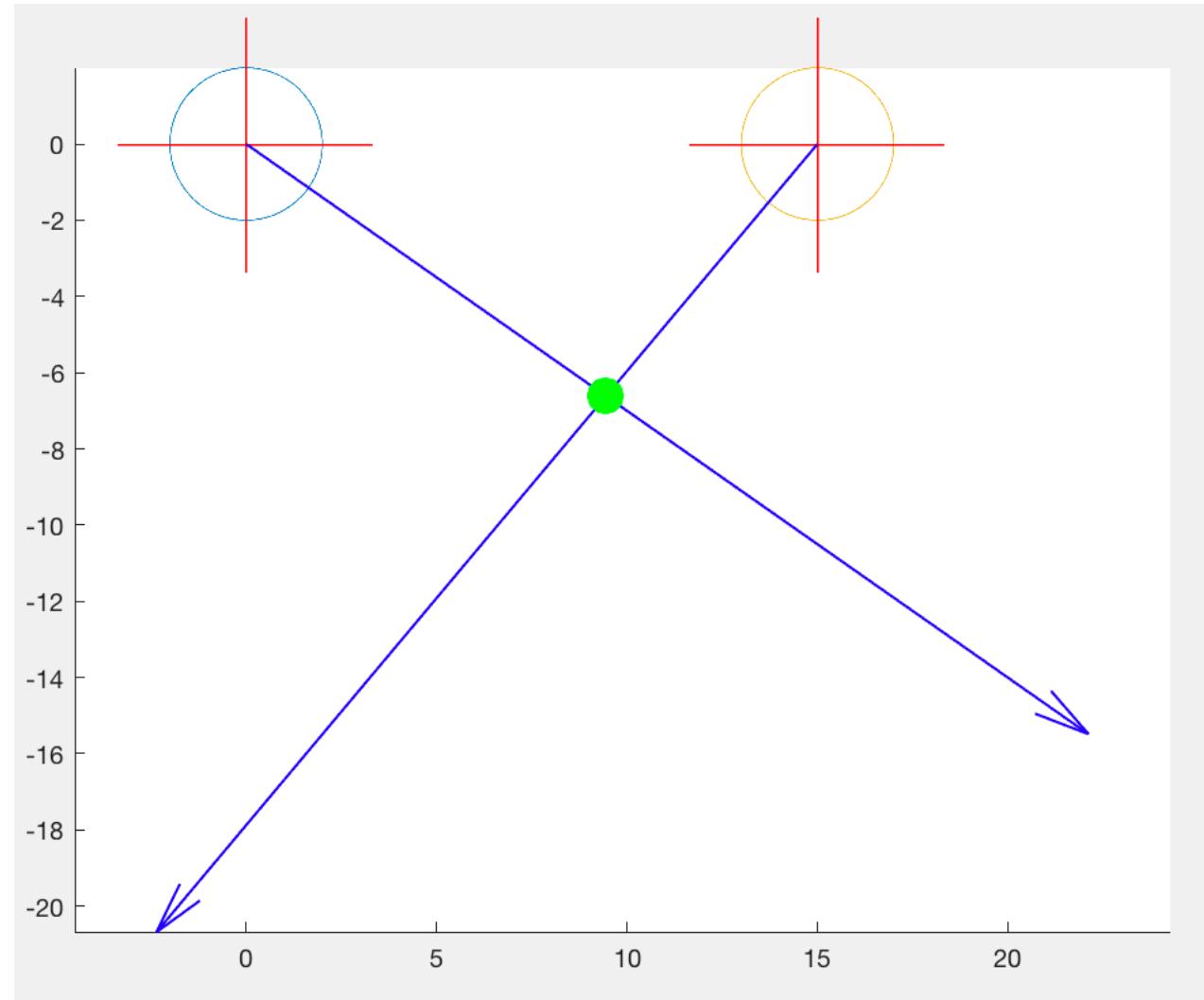
$$[L, U] = lu(A)$$

$$L.U = A$$

$$L.\underbrace{U.x}_y = b$$

Solve : $L.y = b$ descente

Solve : $U.x = y$ remontée



$$[U, S, V] = svd(A)$$

$$U.S.V^T = A$$

$$U^T.U = I, \quad V^T.V = I$$

$$n = rank(A)$$

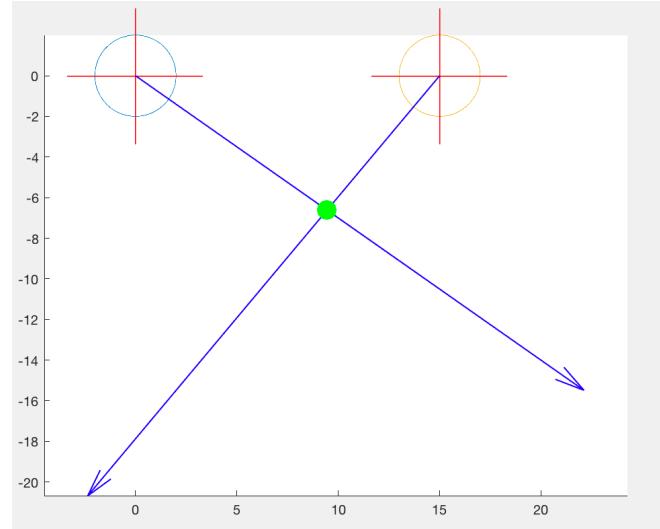
$$S = \begin{pmatrix} \sigma_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_n \end{pmatrix} \quad S^{-1} = \begin{pmatrix} \frac{1}{\sigma_1} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \frac{1}{\sigma_n} \end{pmatrix}$$

$$U.S.V^T.x = b$$

$$S.(V^T.x) = (U^T.b)$$

$$x = V.S^{-1}.U^T.b$$

Solve
 $A.x = b$



L'inverse

$$\begin{aligned} A^{-1} &= V \left[S_n^{-1} \right] U^T \\ &= \sum_{i=1}^m \sigma^{-1} v_i {u_i}^T \end{aligned}$$

Autre type de résolution

```
S1 = inv(A)*b
```

```
%% decompostion Q R %%%%%%%%
```

```
[Q,R]=qr(A)
```

```
%% Tester
```

```
Q*R - A
```

```
Q'
```

```
Q'*Q
```

```
%% Q*R*X=b
```

```
%% soit Y= R*X
```

```
Y = Q'*b
```

```
%% solve R*X=Y %%%%%%%%
```

```
t=Y(2)/R(2,2);
```

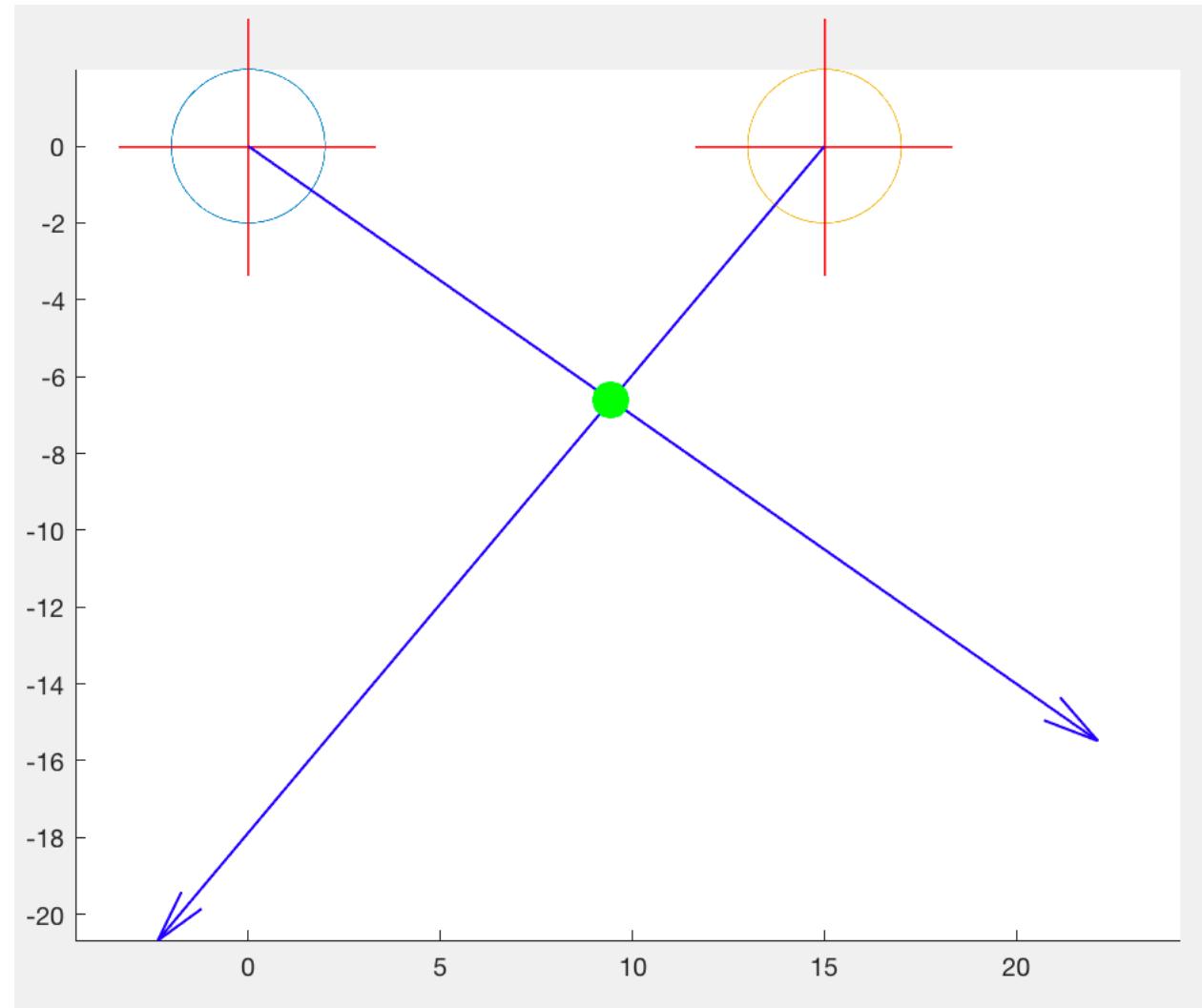
```
s2=[(Y(1)-R(1,2)*t)/R(1,1) t]';
```

```
%%%%%%%%
```

```
%% Si nous posons Y = U*X
```

```
%% Solve L.Y = b -> Descente
```

```
%% Solve U*X = Y -> Remonté
```



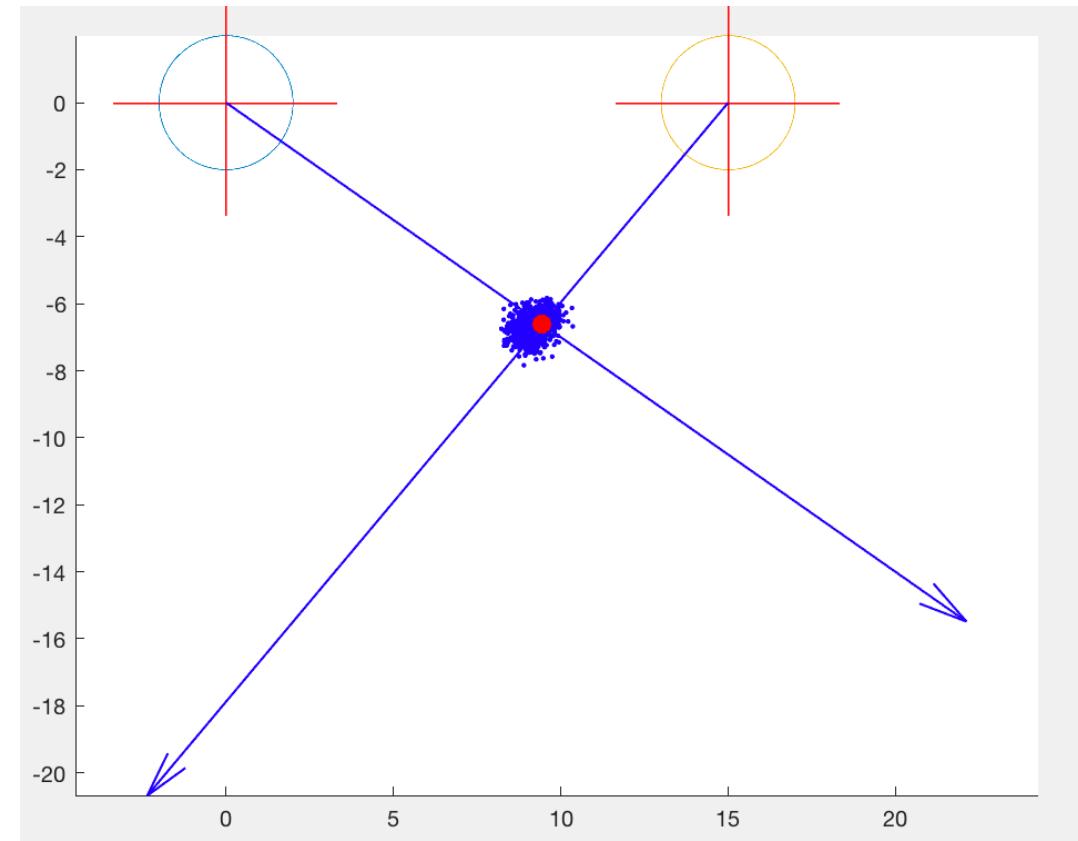
Ajouter une erreur aux mesures

```
function as = adderr( a,err )
[m,n]=size(a);
as=a+(err*(2*randn([m,n])-1))*pi/180;
end
```

Le script

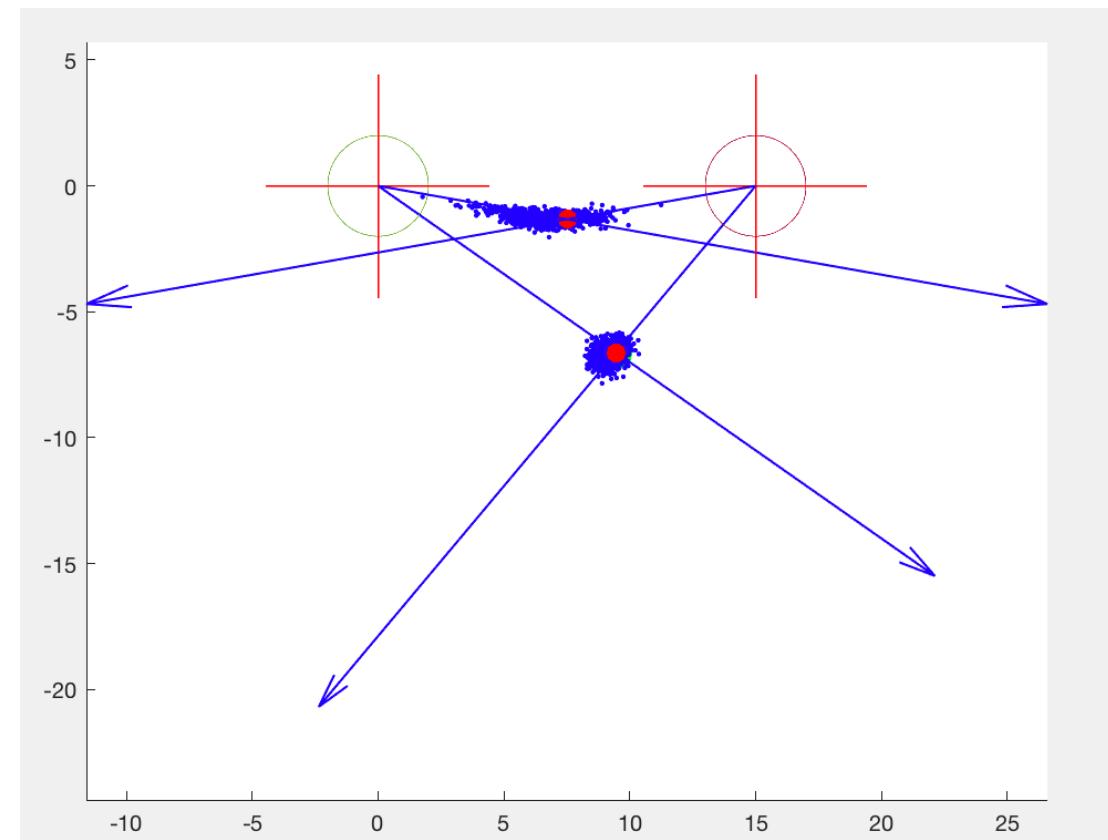
```
serr = sol_pos([adderr(a1,3) adderr(a2,3)],[c1;c2])
Serr=[]
for i=1:1000
    Serr=[Serr ; sol_pos([adderr(a1,1) adderr(a2,1)],[c1;c2])'];
end

plot(Serr(:,1), Serr(:,2), '.b');
sexact = sol_pos([a1 a2],[c1;c2])
plot(sexact(1), sexact(2), '.r', 'MarkerSize',30);
```



Ajouter une erreur aux mesures pour une autre position du robot

```
a1=-10*pi/180;  
a2=-170*pi/180;  
  
serr = sol_pos([adderr(a1,3) adderr(a2,3)],[c1;c2])  
Serr=[]  
for i=1:1000  
    Serr=[Serr ; sol_pos([adderr(a1,1) adderr(a2,1)],[c1;c2])'];  
end  
  
plot(Serr(:,1), Serr(:,2), '.b');  
sexact = sol_pos([a1 a2],[c1;c2])  
plot(sexact(1), sexact(2), '.r', 'MarkerSize',30);  
  
dessin_robot([a1 a2],[c1 ; c2])
```



Estimation des erreurs en utilisant la SVD

```
a1=-15*pi/180;a2=-170*pi/180;  
c1=[0 0];c2=[15 0];
```

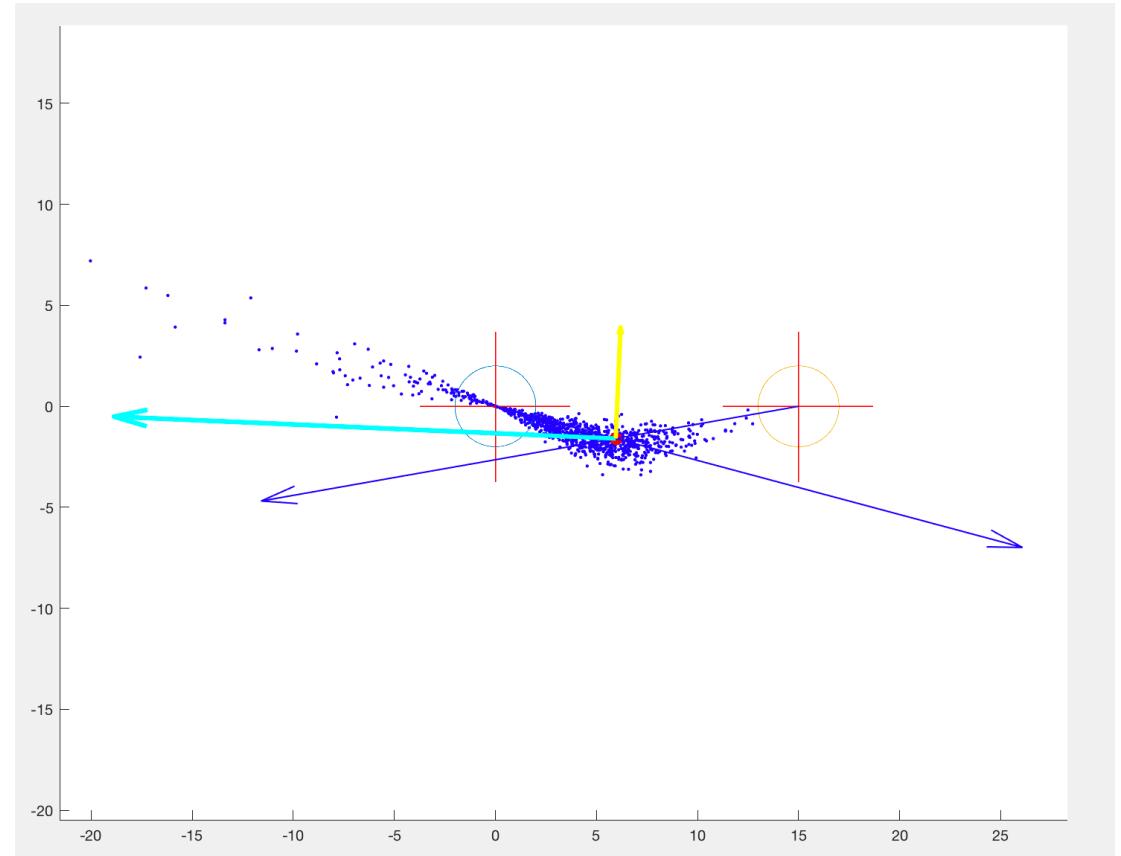
```
Serr=[]  
for i=1:1000  
    Serr=[Serr ; sol_pos([adderr(a1,3) adderr(a2,3)],[c1;c2])'];  
end
```

```
plot(Serr(:,1), Serr(:,2), '.b');  
[sexact,A,b] = sol_pos([a1 a2],[c1;c2])  
plot(sexact(1), sexact(2), '.r', 'MarkerSize',30);
```

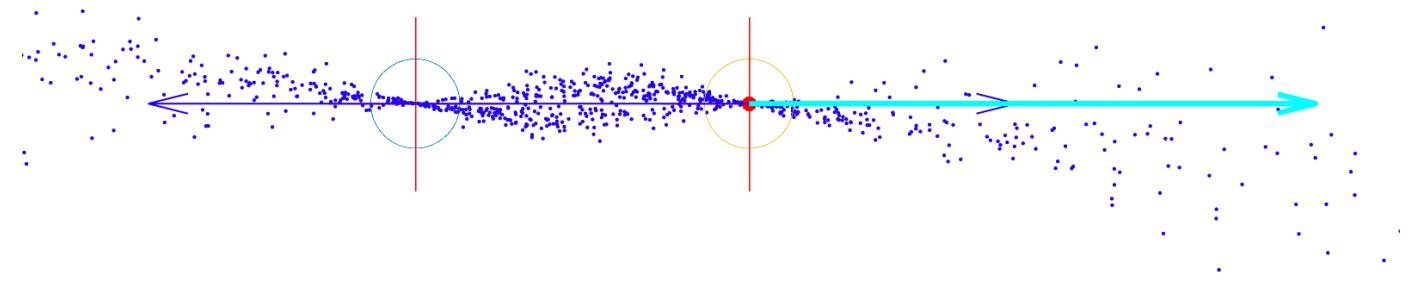
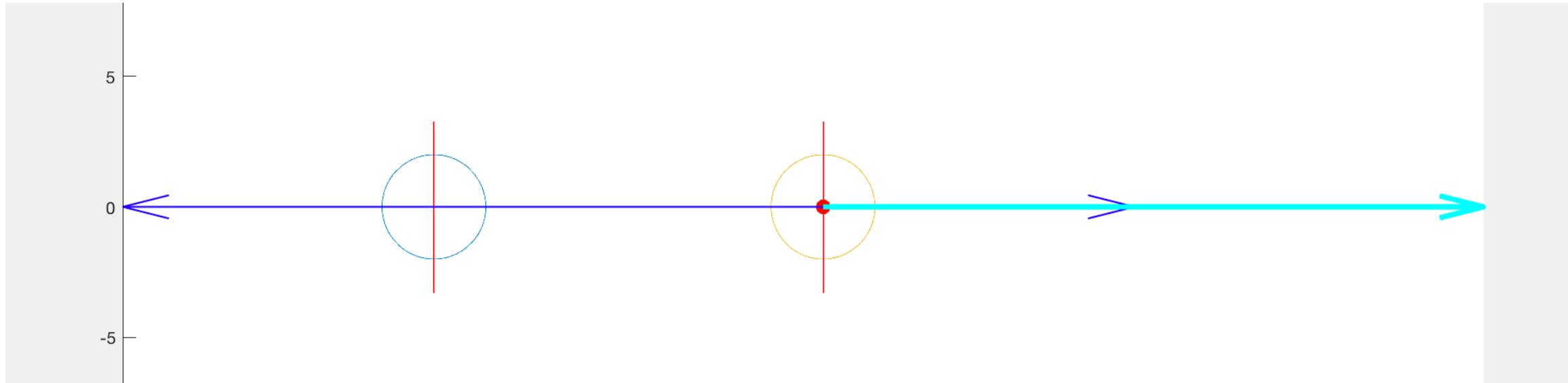
```
dessin_robot([a1 a2],[c1 ; c2])
```

```
[U,S,V]=svd(A)
```

```
quiver(sexact(1),sexact(2),20*S(1,1)*V(1,2),20*S(1,1)*V(2,2),'LineWidth',3,'Color',[0 1 1]);  
quiver(sexact(1),sexact(2),20*S(2,2)*V(1,1),20*S(2,2)*V(2,1),'LineWidth',3,'Color',[1 1 0]);
```



Que ce passe-t-il si le robot passe entre les deux capteurs ?



Résolution de systèmes linéaires sous-constraints

$$[A]_{n \times m} \cdot \begin{bmatrix} x \\ \vdots \\ x \end{bmatrix}_{m \times 1} = [b]_{n \times 1}$$

avec $n < m$

$$[U]_{n \times n} [S \quad 0]_{n \times m} \begin{bmatrix} V^T \\ \vdots \\ V^T \end{bmatrix}_{m \times m} \cdot \begin{bmatrix} x \\ \vdots \\ x \end{bmatrix}_{m \times 1} = [b]_{n \times 1}$$

Résolution de systèmes linéaire sous-contraints

Exemple : $x_1 + x_2 = 4$

$$\begin{bmatrix} 1 & 1 \end{bmatrix} x = 4$$
$$\begin{bmatrix} 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} x = 4$$

Remarquons que :

$$x = \begin{bmatrix} 2 \\ 2 \end{bmatrix} + \alpha \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

Résolution de systèmes linéaire sous-constraints

Une infinité de solution -> comment en caractériser une ?

$$[A]_{n \times m} \cdot \begin{bmatrix} x \end{bmatrix}_{m \times 1} = [b]_{n \times 1}$$

Exemple : parmi tous les solutions possibles sélectionner celle tel que :

$$\min \|x\|^2 \text{ avec } A.x - b = 0$$

Résolution de systèmes linéaire sous-constraints

$$\arg \min_x H(x)$$

avec

$$H(x) = x^T x + \lambda^T (Ax - b)$$

Résolution de systèmes linéaire sous-constraints

La valeur du critère est minimale pour :

$$\frac{\partial H}{\partial x} = 0$$

$$2x^T + \lambda^T A = 0$$

$$x = \frac{1}{2}A^T \lambda$$

Comment choisir λ ?

$$Ax = b$$

$$A\left(\frac{1}{2}A^T \lambda\right) = b$$

$$\lambda = 2(AA^T)^{-1}b$$

Donc

$$x = A^T (AA^T)^{-1}b$$

La pseudo-inverse (droite)

$$A_R^+ = A^T (AA^T)^{-1}$$

Remarquons : $AA_R^+ = AA^T (AA^T)^{-1} = I$

MAIS (génétiquement) : $A_R^+ A \neq I$

$A_R^+ A = I$ ssi A carré et inversible

La pseudo-inverse (droite)

$$\begin{aligned} A_R^+ &= A^T (A A^T)^{-1} \\ &= \left(V \begin{bmatrix} S_n \\ 0 \end{bmatrix} U^T \right) (U S_n^{-2} U^T) \\ &= V \begin{bmatrix} S_n \\ 0 \end{bmatrix} S_n^{-2} U^T \\ &= V \begin{bmatrix} S_n^{-1} \\ 0 \end{bmatrix} U^T \\ &= \sum_{i=1}^N \sigma_i^{-1} v_i {u_i}^T \end{aligned}$$

Solution d'un système linéaire sous-constraint

$$\begin{aligned}x &= A_R^+ b + \sum_{i=n+1}^m \alpha_i v_i, \text{ for any } \alpha_i \\&= A_R^+ b + [I - A_R^+ A] \alpha\end{aligned}$$

Remarquons :

$$Ax = b$$

$$A(x + z) = b$$

avec $Az = 0$ (z choisi dans le noyau de la matrice A)

Solution d'un système linéaire sous-constraint

$$\begin{aligned}x &= A_R^+ b + \sum_{i=n+1}^m \alpha_i v_i, \text{ for any } \alpha_i \\&= A_R^+ b + [I - A_R^+ A] \alpha\end{aligned}$$

Exemple :

$$U \begin{bmatrix} \sigma_1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v_1^T \\ v_n^T \end{bmatrix} z = 0$$
$$U \cdot \sigma_1 \cdot v_1^T \cdot z = 0$$

si $z = v_n$ alors

$$U \cdot \sigma_1 \cdot v_1^T \cdot v_n = 0$$

Solution d'un système linéaire sous-constraint

$$x = A_R^+ b + \sum_{i=n+1}^m \alpha_i v_i , \text{ for any } \alpha_i$$
$$= A_R^+ b + [I - A_R^+ A] \alpha$$

Tester ...

```
%% Noyau de la matrix  
ka1=null(A)  
%% ou  
[U,S,V]=svd(A);  
ka2=V(:,2)
```

```
%%  
%% propriété du noyau  
A*ka1  
A*ka2
```

Projection dans le noyau

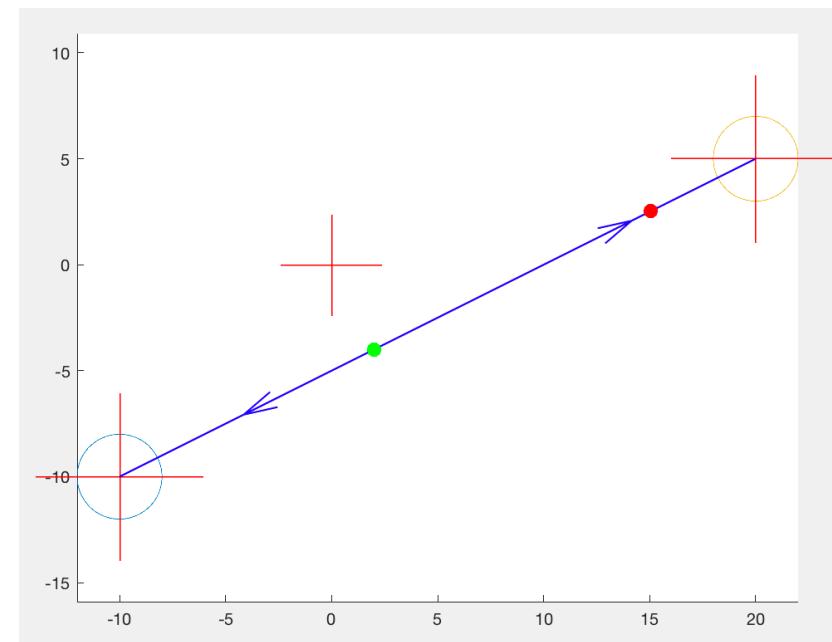
```
%% Projection sur le noyau  
(eye(2)-mypinv(A,1e-5)*A)*[rand(1) rand(1)]'
```

Système linéaire sur-constraint

Simulation de données

```
function [a] = pos_to_ang(p, c )  
  
[m,n]=size(c);  
a=[];  
for i=1:m  
    v=c(i,:)-p';  
    a=[a atan2(v(2),v(1))+pi];  
end  
end
```

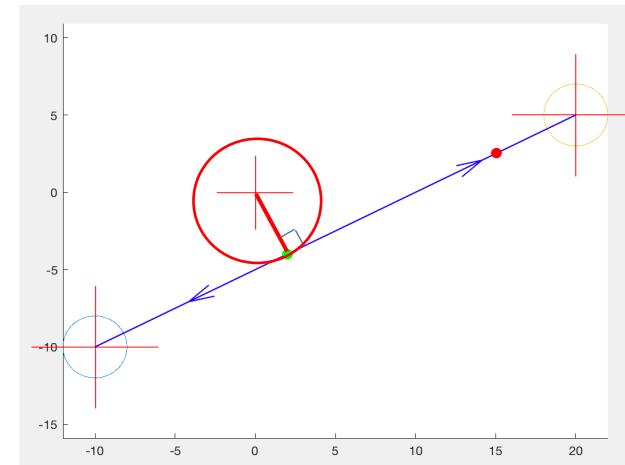
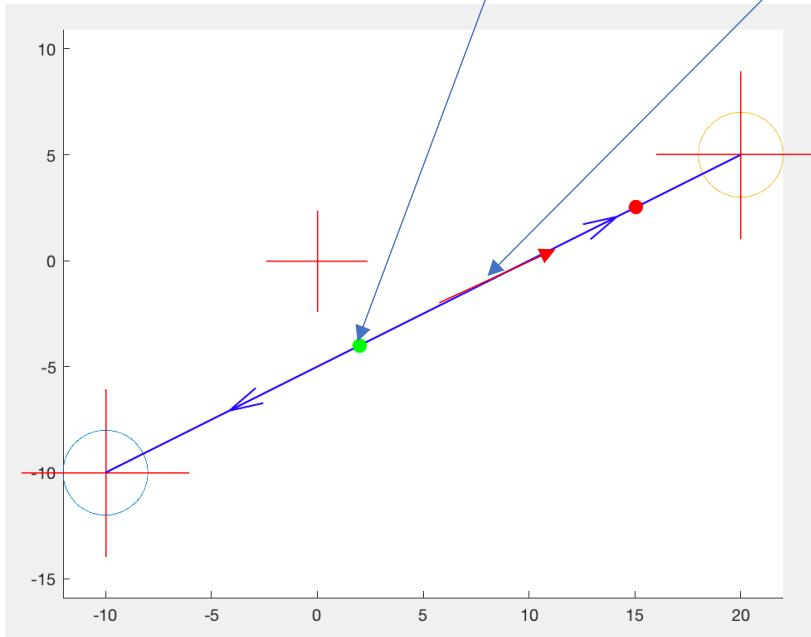
```
% Coordonnées des balises  
c=[-10 -10 ;  
    20 5];  
% simulation des mesures  
p=(c(1,:)+c(2,:))/2  
a = pos_to_ang(p, c )  
  
% dessine le robot  
dessin_robot(a,c)  
plot(0, 0, '+r', 'MarkerSize',60);  
  
[s1,A,b] = sol_pos(a,c)  
plot(s1(1), s1(2), 'r', 'MarkerSize',30);  
  
s2=pinv(A)*b  
plot(s2(1), s2(2), '.g', 'MarkerSize',30);
```



Système linéaire sur-constraint

$$\arg \min_x H(x) \quad \text{avec} \quad H(x) = x^T x + \lambda^T (Ax - b)$$

$$x = A_R^+ b + [I - A_R^+ A] \alpha$$

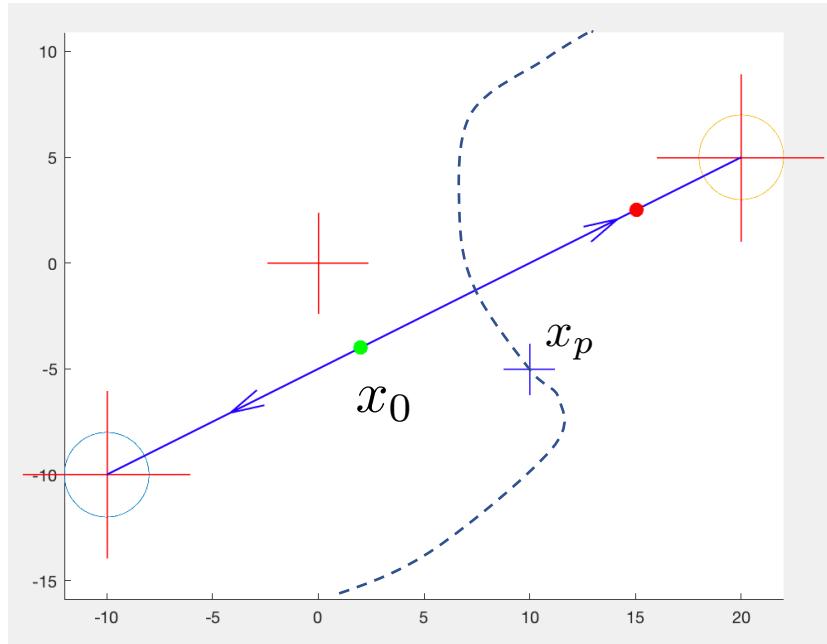


Adapter vos solutions au problème

Exemple trajectoire robot

Comment passer la singularité ?

$$x = A_R^+ b + [I - A_R^+ A] \alpha$$



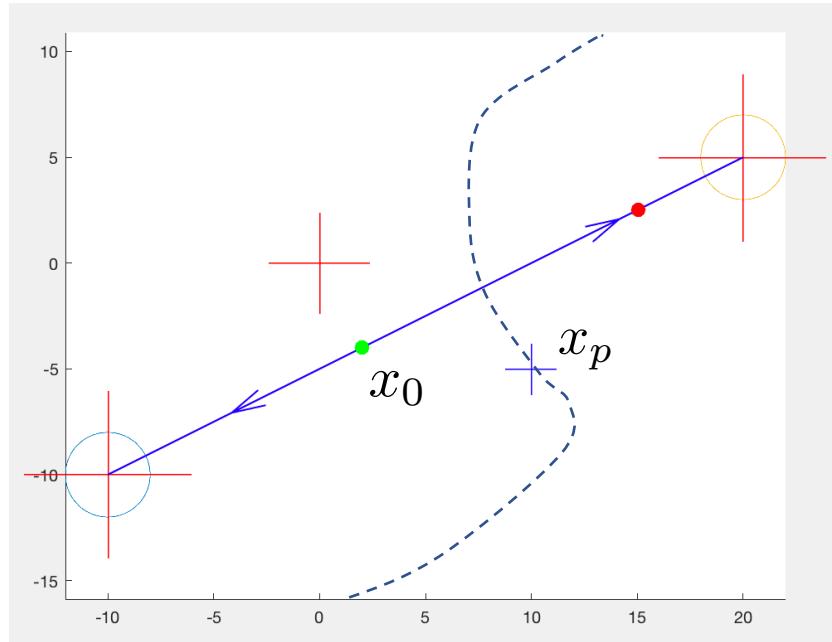
- 1) Poser le problème
- 2) Caractériser une solution
- 3) Résoudre le problème

Déterminer une solution particulière dans le noyau de la matrice A tel que ce point est proche du point précédent de la trajectoire (x_p)

Adapter vos solutions au problème

Exemple trajectoire robot

Comment passer la singularité ?



$$x = \underbrace{A_R^+ b}_{x_0} + \underbrace{[I - A_R^+ A] \alpha}_{k \cdot \omega}$$

$$k_s = \arg \min_k \|x_0 + k \cdot \omega - x_p\|^2$$

Posons : $g = x_0 - x_p$

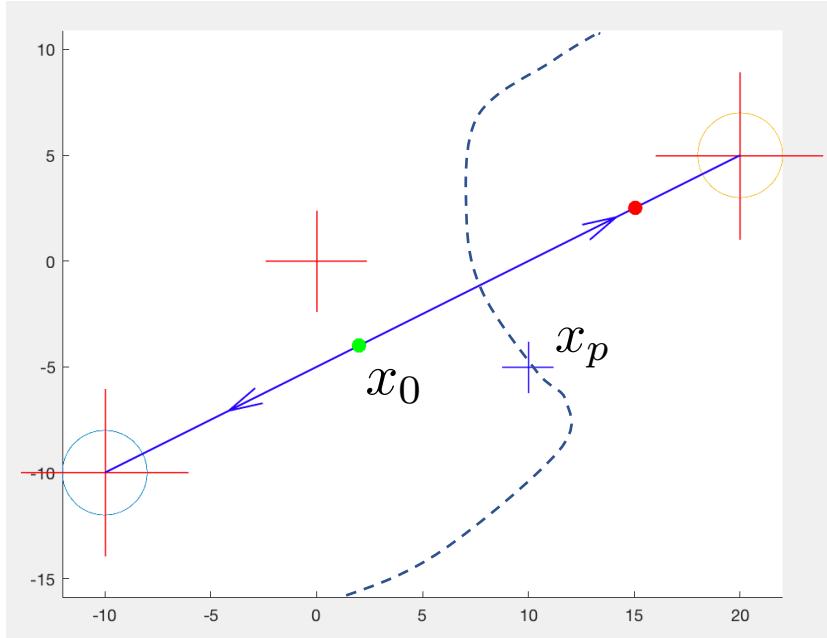
$$\mathcal{C} = \|g + k \cdot \omega\|^2 = (g + k \cdot \omega)^T \cdot (g + k \cdot \omega)$$

Adapter vos solutions au problème

Exemple trajectoire robot

Comment passer la singularité ?

$$x = \underbrace{A_R^+ b}_{x_0} + \underbrace{[I - A_R^+ A]\alpha}_{k.\omega}$$



\mathcal{C} est minimal, si $\frac{\partial \mathcal{C}}{\partial k} = 0$

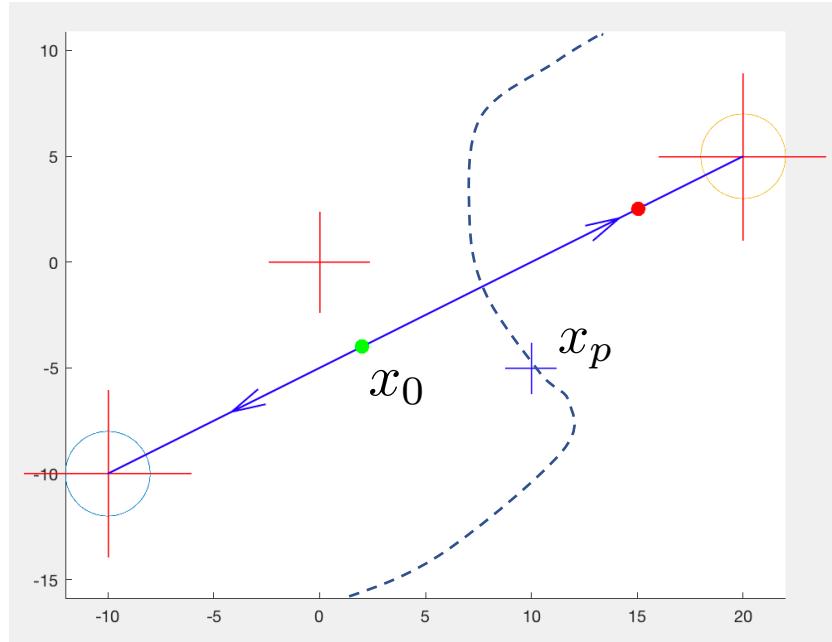
$$\begin{aligned}\mathcal{C} &= (g + k.w)^T (g + k.w) \\ &= g^T g + 2k.g^T w + k^2.w^T w\end{aligned}$$

Adapter vos solutions au problème

Exemple trajectoire robot

Comment passer la singularité ?

\mathcal{C} est minimal, si $\frac{\partial \mathcal{C}}{\partial k} = 0$



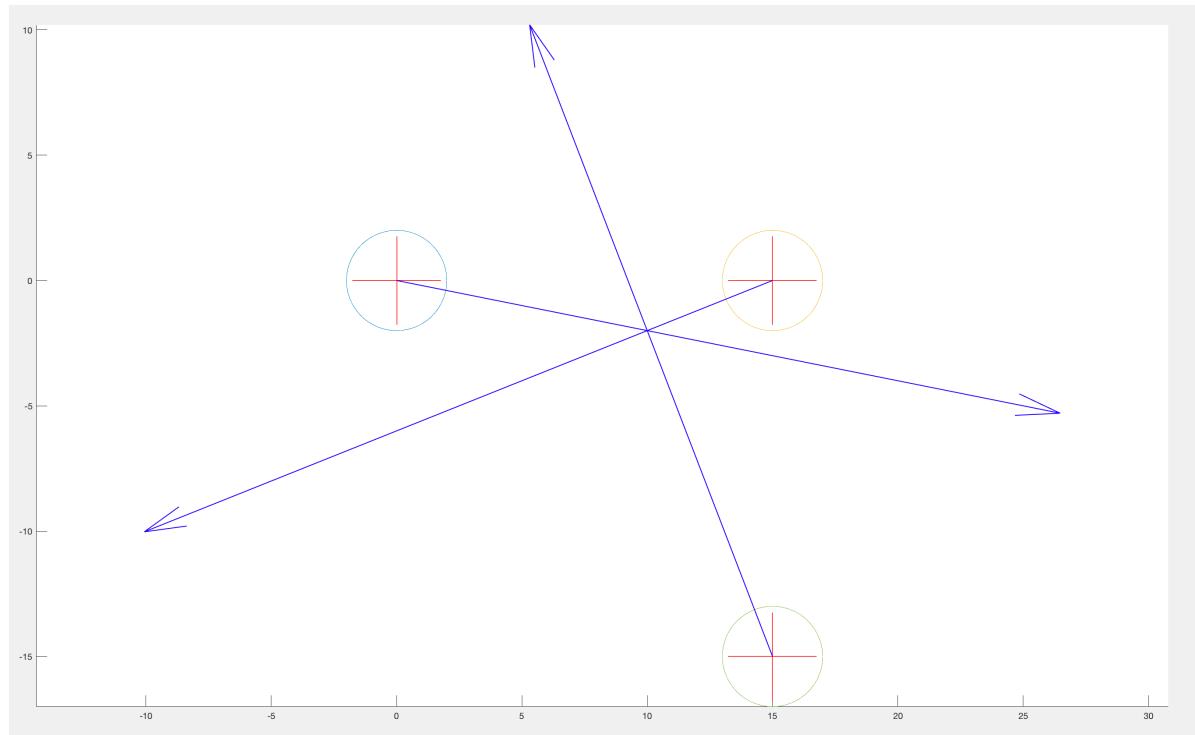
$$\frac{\partial [g^T g + 2kg^T \omega + k^2 \omega^T \omega]}{\partial k} = 0$$
$$2g^T \omega + 2k\omega^T \omega = 0$$

$$k = \frac{-g^T \omega}{\omega^T \omega}$$

Deux problématiques:

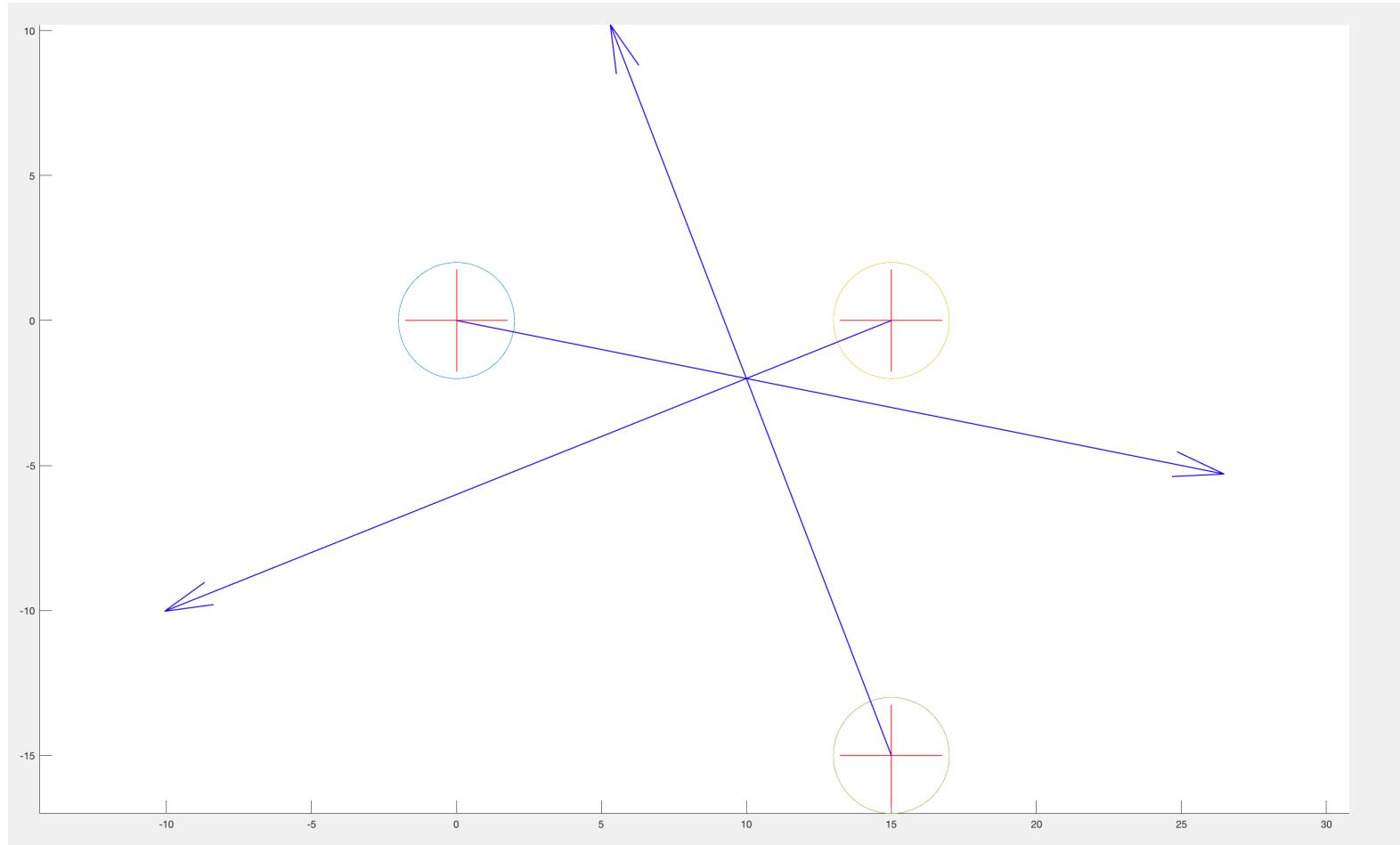
- Comment gérer les incertitudes ?
- Comment gérer les singularités ?

-> Ajouter des capteurs



Les systèmes sur-constraints

```
clear all; clf; hold on; axis equal;  
% Coordonnées des balises  
c=[0 0; 15 0 ; 15 -15 ];  
% simulation des mesures  
p=[10 -2]  
a = pos_to_ang(p, c )  
% dessine le robot  
dessin_robot(a,c)
```



... Dans la vraie vie



Système sur-contraint

... Dans la vraie vie

Système sur-constraint

```
% dans la vraie vie
```

```
clear all; clf; hold on; axis equal;
```

```
% Coordonnées des balises
```

```
c=[0 0; 15 0 ; 15 -15];
```

```
% simulation des mesures
```

```
p=[10 -2]
```

```
a = pos_to_ang(p, c)
```

```
plot(p(1), p(2), '.r', 'MarkerSize', 30);
```

```
plot(p(1)+1, p(2)+1, '.b', 'MarkerSize', 30);
```

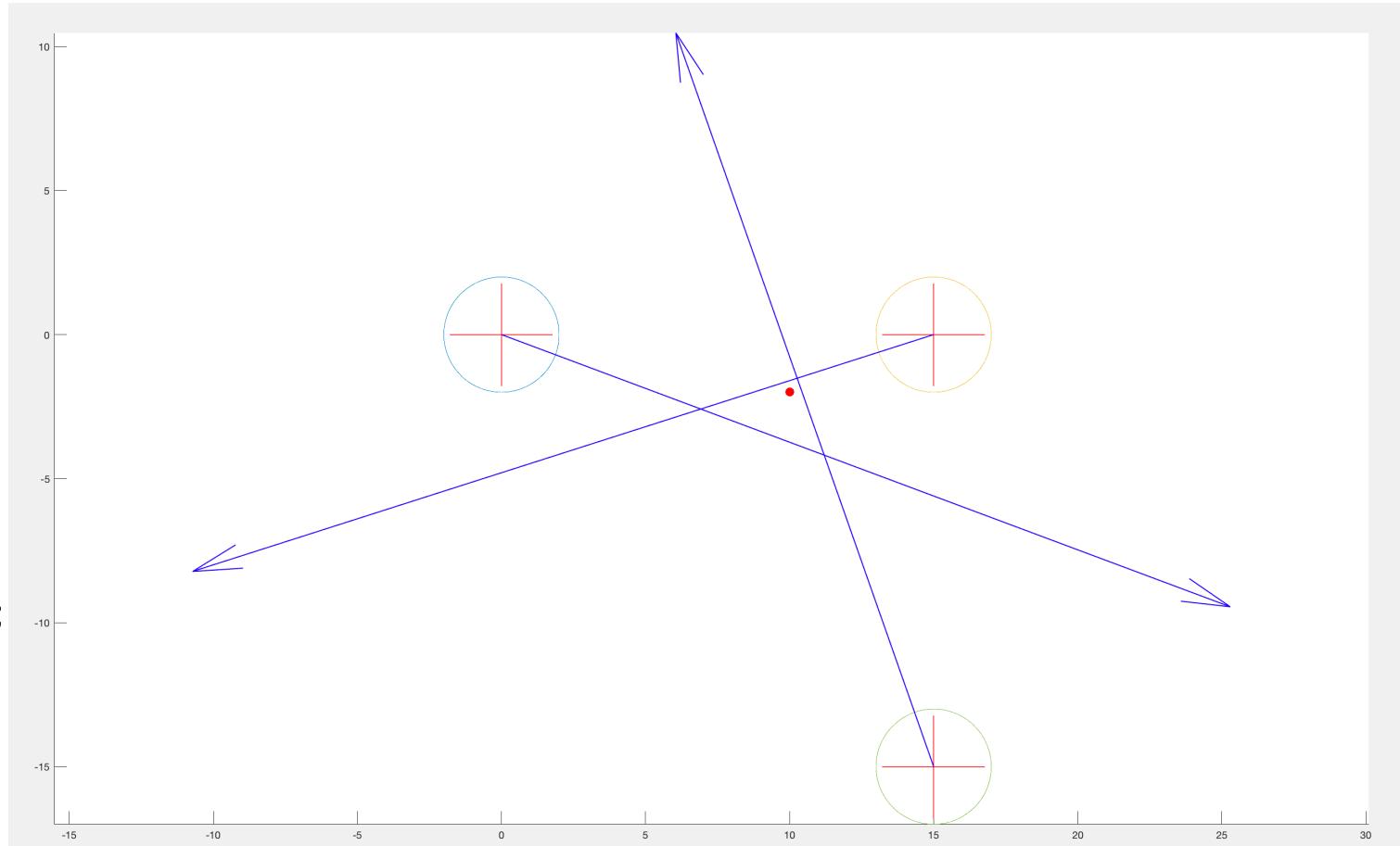
```
plot(p(1)+1, p(2)-1, '.y', 'MarkerSize', 30);
```

```
plot(p(1)-1, p(2)+1, '.m', 'MarkerSize', 30);
```

```
plot(p(1)-1, p(2)-1, '.c', 'MarkerSize', 30);
```

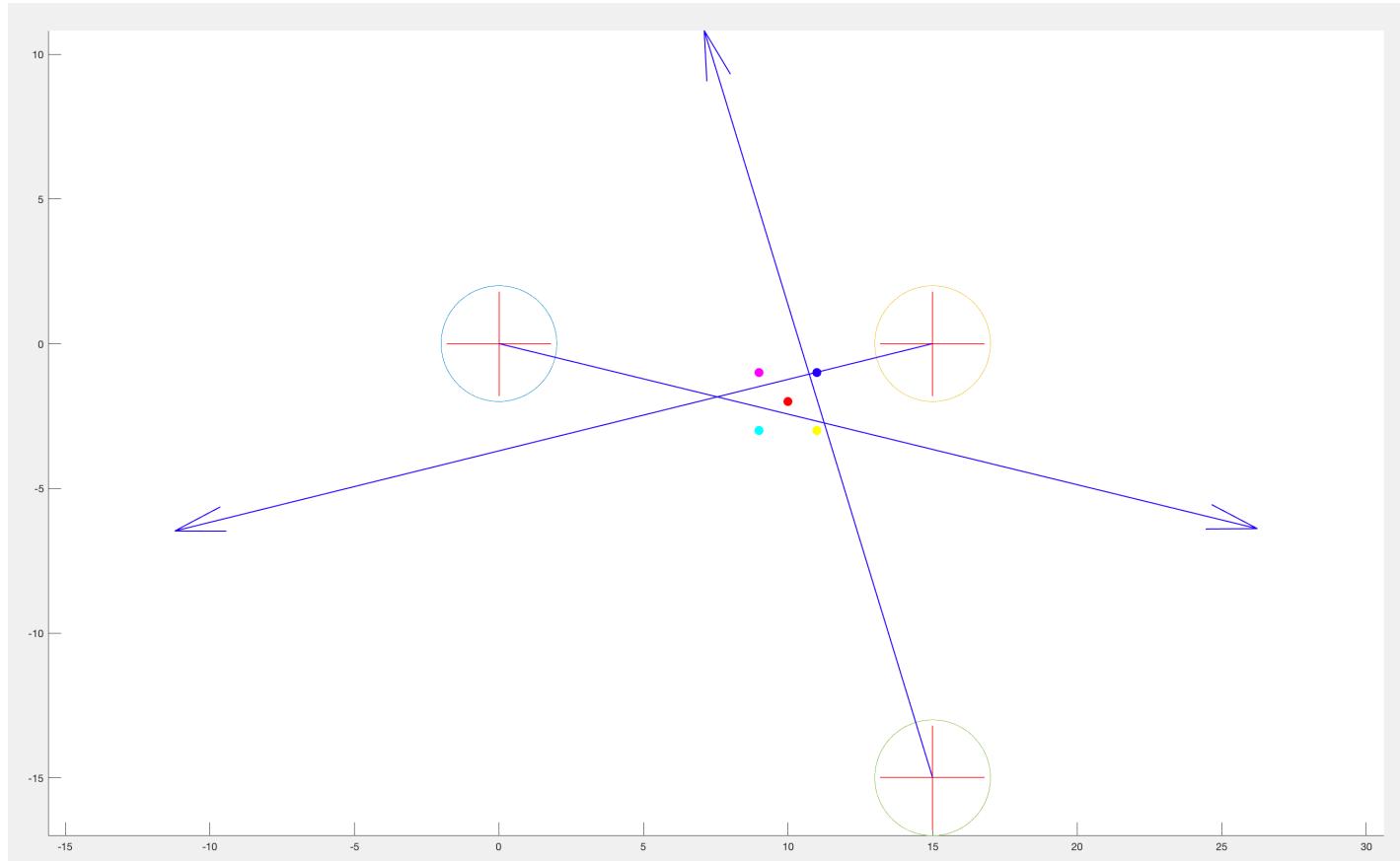
```
% dessine le robot
```

```
dessin_robot(adderr(a, 3), c)
```



... Déterminer une caractérisation d'une solution

Système sur-contraint



Évaluer les équations pour
différents points

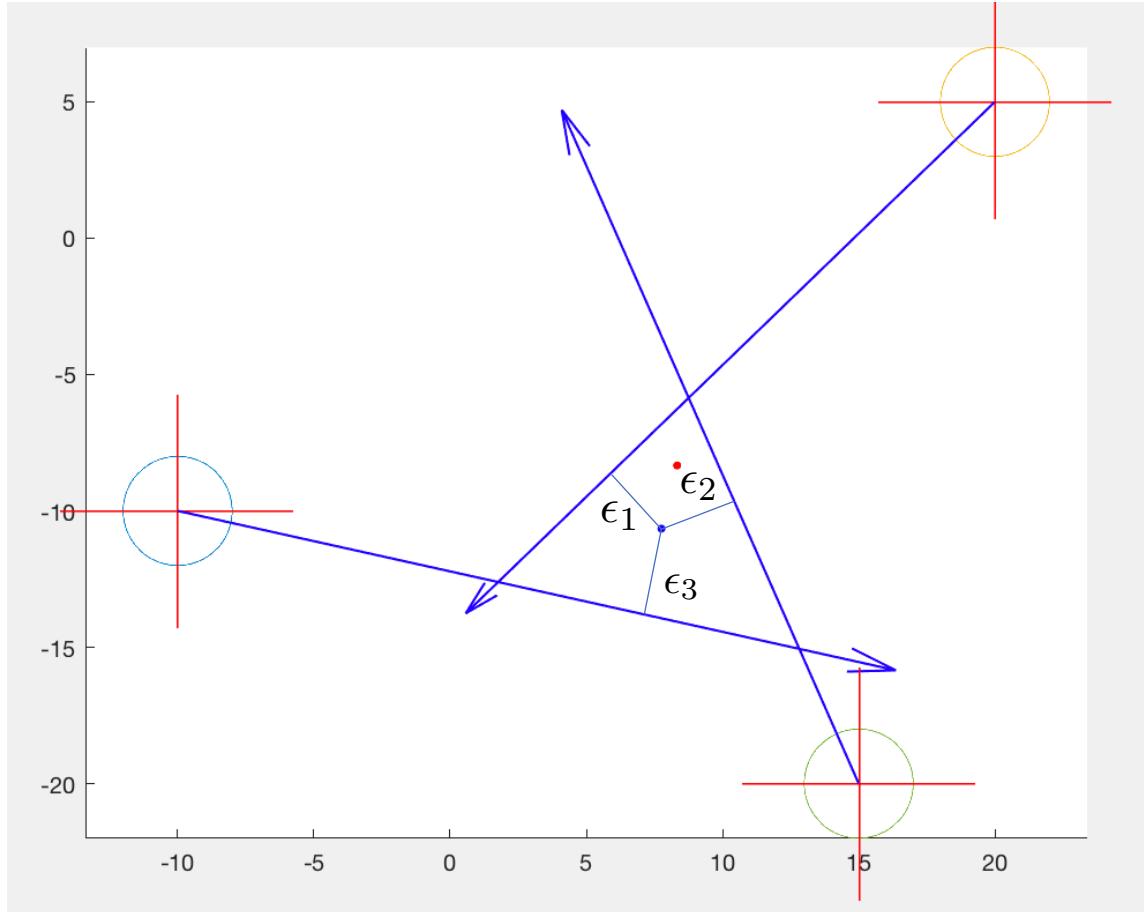
$$A.S_1 - b$$

$$A.S_2 - b$$

$$A.S_3 - b$$

...

Solution aux moindres carrés



$$\begin{aligned}\epsilon_1 &= A_{(1,:)}.x - b_1 \\ \epsilon_2 &= A_{(2,:)}.x - b_2 \\ \epsilon_3 &= A_{(3,:)}.x - b_3\end{aligned}$$

Solution tel que :

$$\min \sum_{i=1}^3 \epsilon_i$$

Solution aux moindres carrés

$$\begin{aligned}\mathcal{C}_o &= \|Ax - b\|^2 = (Ax - b)^T(Ax - b) \\ &= x^T A^T Ax - 2b^T Ax + b^T b\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{C}_o}{\partial x} &= 2x^T A^T A - 2b^T A = 0 \\ A^T A x - A^T b &= 0\end{aligned}$$

$$x = (A^T A)^{-1} A^T b$$

Prise en compte des incertitudes

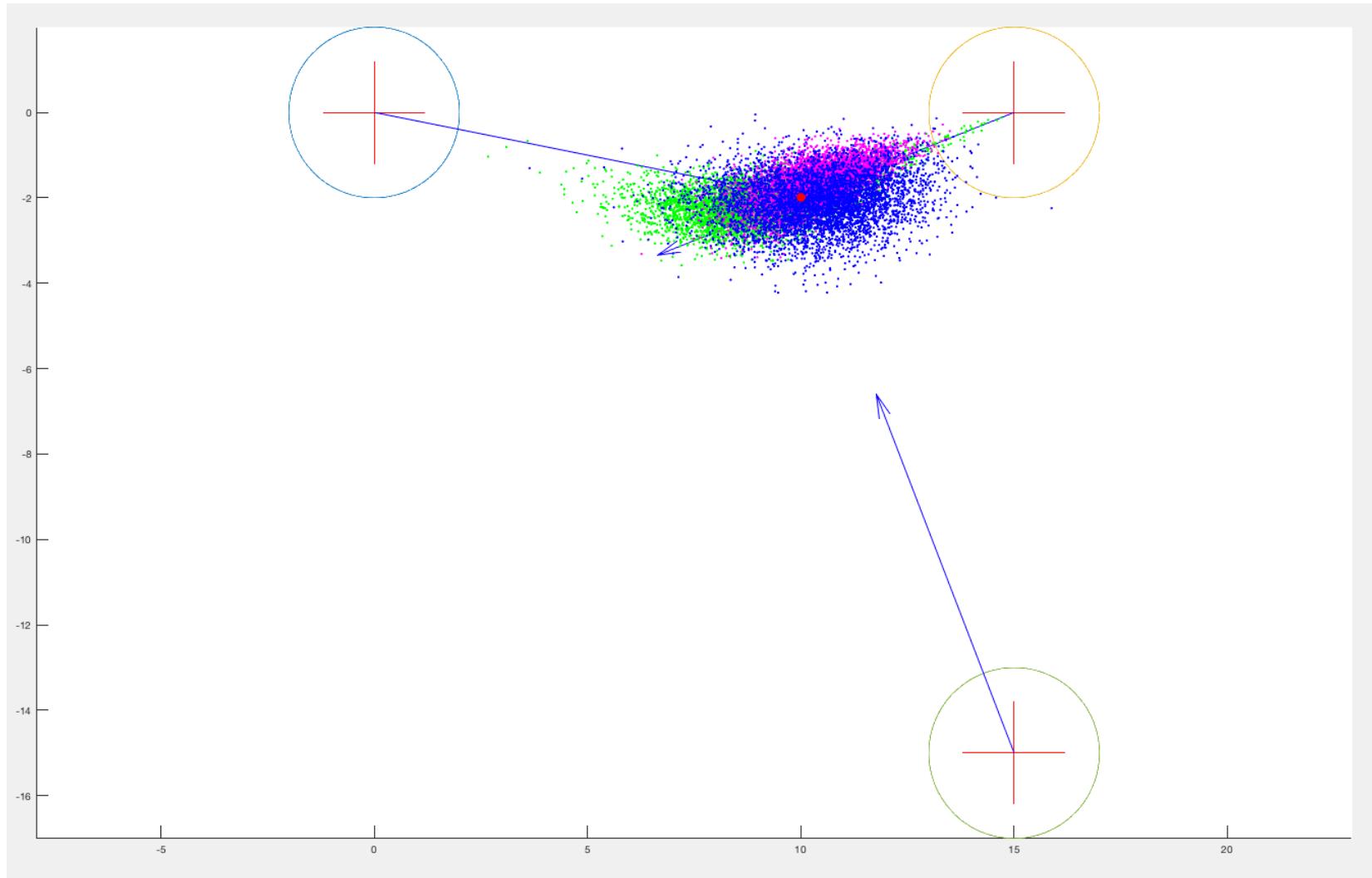
Système sur-constraint

```
% efface tous  
clear all; clf; hold on; axis equal;  
% Coordonnées des balises  
c=[0 0; 15 0 ; 15 -15 ];  
% simulation des mesures  
p=[10 -2]  
a = pos_to_ang(p, c )  
% dessine le robot  
dessin_robot(a,c)  
plot(p(1), p(2), '.r', 'MarkerSize', 30);
```

```
% Ajout d'erreur, prise en compte des balises 1 et 2  
Serr=[]  
for i=1:5000  
    Serr=[Serr ; sol_pos(adderr(a(1:2),2),c(1:2,:))'];  
end  
plot(Serr(:,1), Serr(:,2), '.g');  
% Ajout d'erreur, prise en compte des balises 2 et 3  
Serr=[]  
for i=1:5000  
    Serr=[Serr ; sol_pos(adderr(a(2:3),2),c(2:3,:))'];  
end  
plot(Serr(:,1), Serr(:,2), '.m');  
% Ajout d'erreur, prise en compte des balises 1, 2 et 3  
Serr=[]  
for i=1:5000  
    Serr=[Serr ; sol_pos(adderr(a,3),c)'];  
end  
plot(Serr(:,1), Serr(:,2), '.b');  
% dessine la position du robot  
plot(p(1), p(2), '.r', 'MarkerSize', 30);
```

Prise en compte des incertitudes

Système sur-constraint



Prise en compte des incertitudes

Système sur-constraint

Exemple

$$A = \begin{pmatrix} 2 & 0 \\ -1 & 1 \\ 0 & 2 \end{pmatrix} B = \begin{pmatrix} 1 \\ 0 \\ -1 \end{pmatrix}$$