



# Navigation de multirobot par exploitation du contact physique

Mini-séminaire 02/03/2020

Quentin Delamare  
AGPR ENS Rennes, IRISA/Inria Rennes  
Mécatronique | Rainbow group

# Contexte

**Robotique aérienne** : un domaine étudié depuis quelques temps

- Nombreuses méthodes développées :
  - Contrôle du mouvement en dépit du **sous - actionnement**
  - **Plannification** efficace sous contraintes
- Mais aussi plus récemment :
  - **Préemption** d'objets
  - Application d' **efforts**
  - **Atterrissages** sous contrainte



# Contexte

**Robotique aérienne** : un domaine étudié depuis quelques temps

- Nombreuses méthodes développées :
  - Contrôle du mouvement en dépit du **sous-actionnement**
  - **Plannification** efficace sous contraintes
- Mais aussi plus récemment :
  - **Préemption** d'objets
  - Application d' **efforts**
  - **Atterrissages** sous contrainte

**L'environnement est toujours un facteur limitant**

- Les mouvements robotiques sont restreints à cause des **collisions** potentielles



# Contexte

A l'inverse, la navigation pourrait être améliorée en **tirant partie** des contacts

- Exemples chez les animaux :
  - Singes
  - Insectes
  - Spéléologie
  - ...



- Il faut de la précision aux évènements de **transition** , malgré les fortes **incertitudes** sur la dynamique globale

# Concept de la locomotion aérienne

- La navigation de multirobot en environnement **encombré** est difficile
- A cause des **contacts**



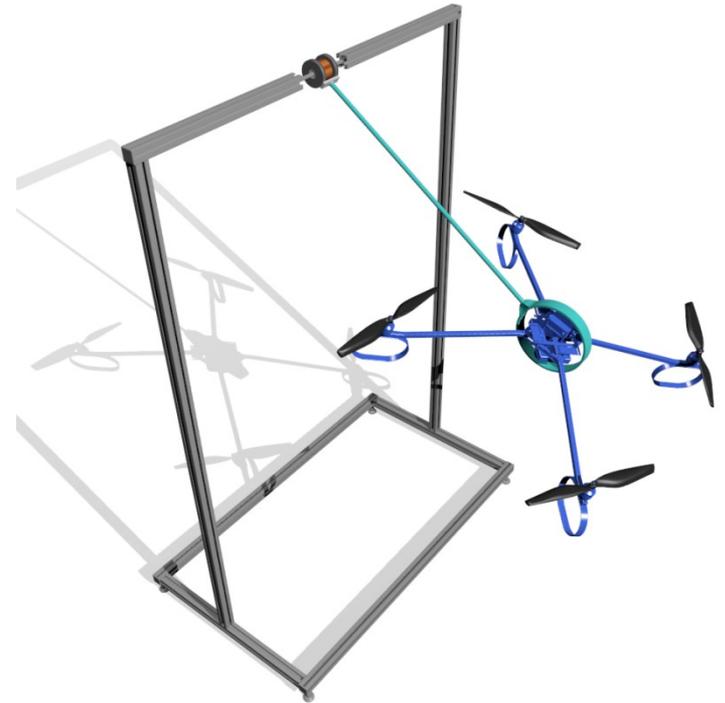
# Concept de la locomotion aérienne

- La navigation de multirobot en environnement **encombré** est difficile
- A cause des **contacts**
- Peut-on améliorer la précision aux passages critiques en prenant appui ?
- Et bénéficier des dynamiques particulières émergentes ?



# Un cas d'étude pour commencer : le « MonkeyRotor »

- Quadrotor + Bras
- 2 modalités de fonctionnement :
  - ✓ Attaché à une « branche »
  - ✓ En vol libre
- Locomotion avec contacts intermittents



# Propriétés

Système à 4 états et 3 actionnements :

$$\mathbf{q} = \begin{pmatrix} x \\ z \\ \theta_B \\ \theta_{1B} \end{pmatrix} \quad \mathbf{u} = \begin{pmatrix} f_r \\ f_l \\ u_r \end{pmatrix}$$

- **Dynamique attachée :**

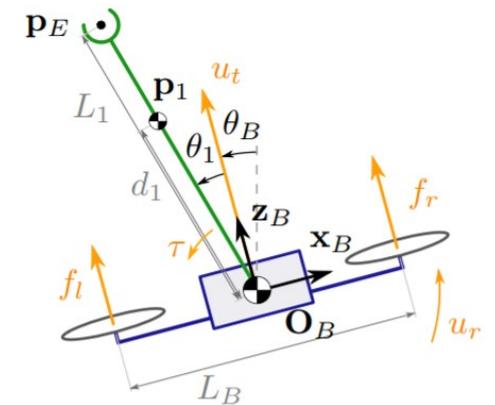
$x, z$  imposés par contrainte

$$\mathbf{y}_h = \begin{pmatrix} \theta_B \\ \theta_{1B} \end{pmatrix} \Rightarrow \text{Sur-actionnement}$$

- **Dynamique libre :**

l'attitude  $\theta_B$  est sujette à une dynamique interne

$$\mathbf{y}_f = \begin{pmatrix} x \\ z \\ \theta_{1B} \end{pmatrix} \Rightarrow \text{Sous-actionnement}$$



# MonkeyRotor : génération de trajectoires

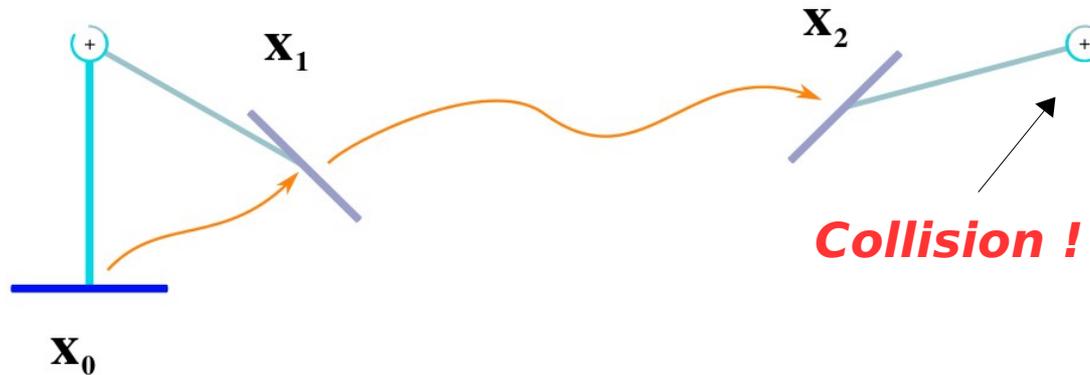
Le système commence par se déplacer en restant **attaché**



Dans cette phase il n'y a que **2 degré de liberté** : les 2 angles

# MonkeyRotor : génération de trajectoires

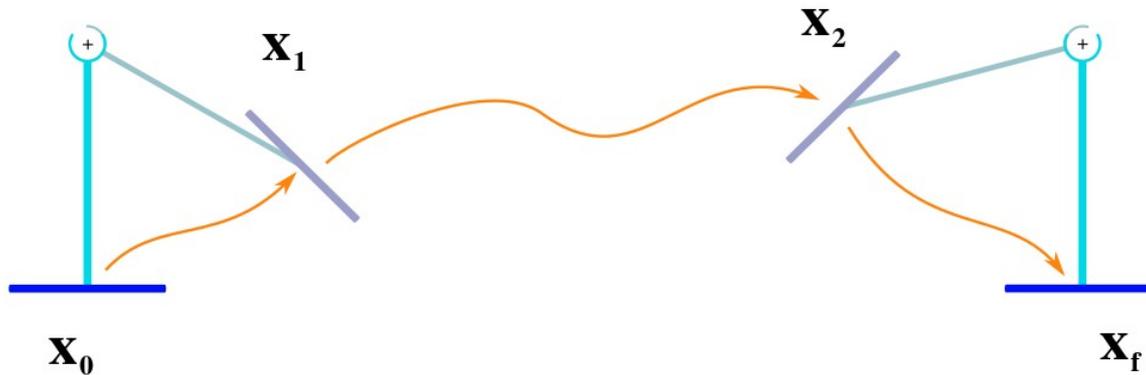
Ensuite débute une phase de **vol libre**



Une **collision inélastique** a lieu si la vitesse de l'effecteur est  $>0$

# MonkeyRotor : génération de trajectoires

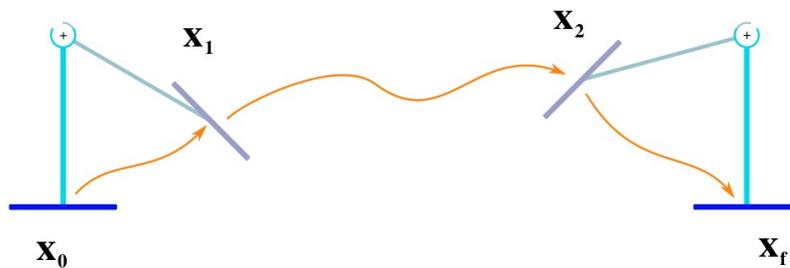
...pour finalement atteindre la **cible** et s'arrêter.



La poussée du robot peut être **supérieure** ou **inférieure** à son propre poids

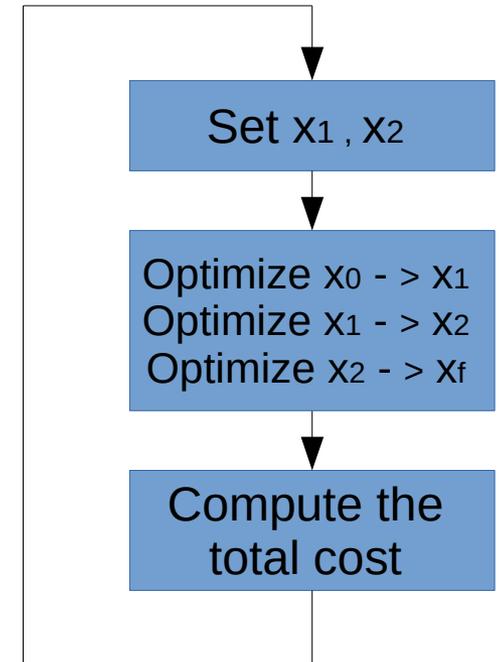
# MonkeyRotor : génération de trajectoires

Algorithme spécifique pour résoudre ce problème



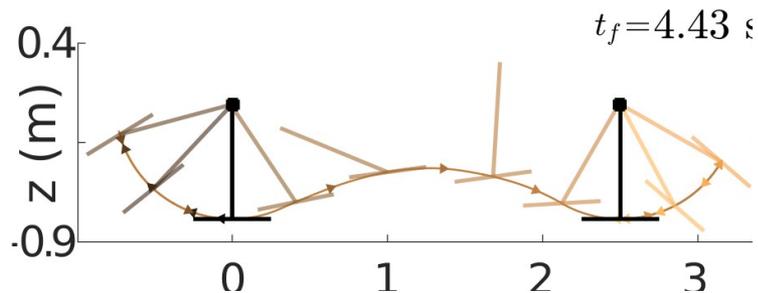
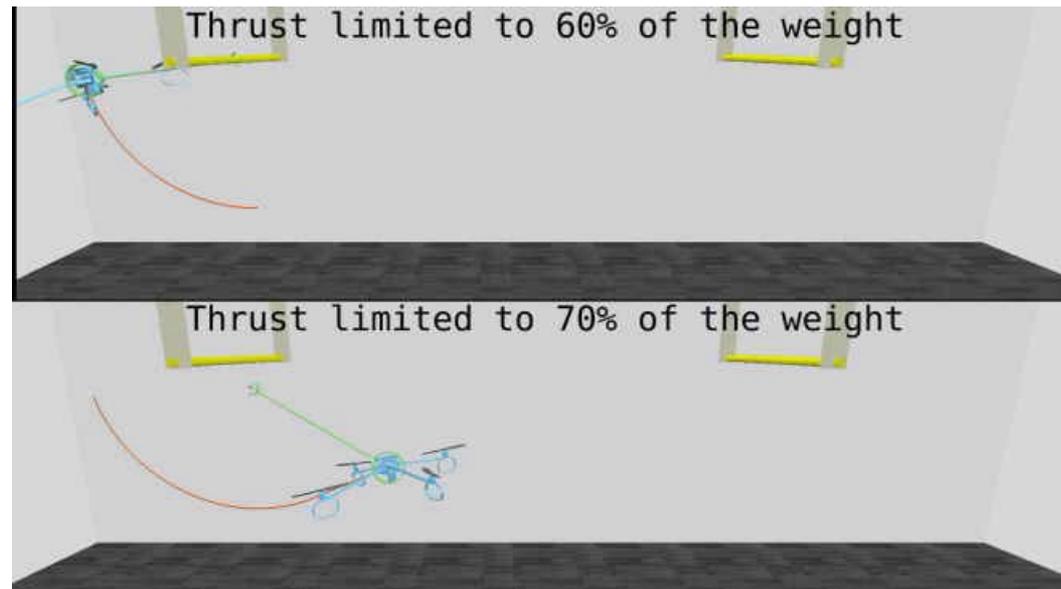
Deux **boucles imbriquées** :

- **Interne** pour trouver les sous-trajectoires
  - « Transcription directe » (splines par morceaux)
  - $\min t_f$  avec actionnement **limité**
- **Externe** pour trouver les événements de transition
  - Pas de gradient analytique ici



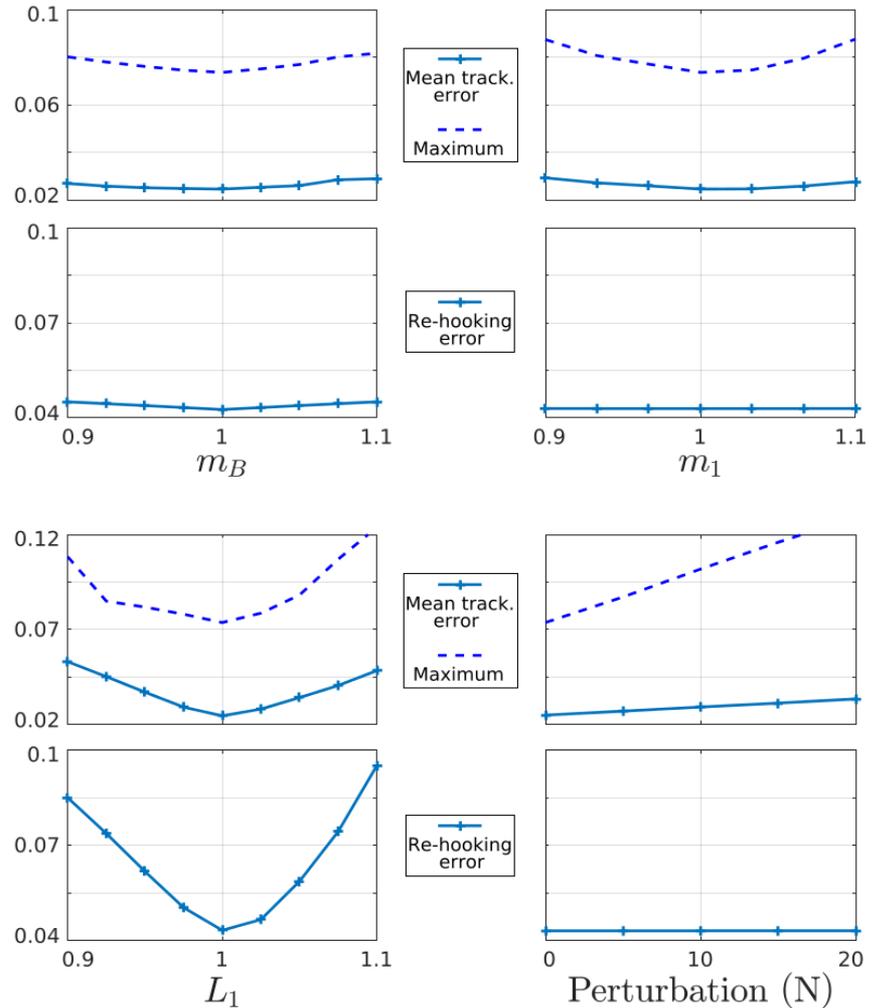
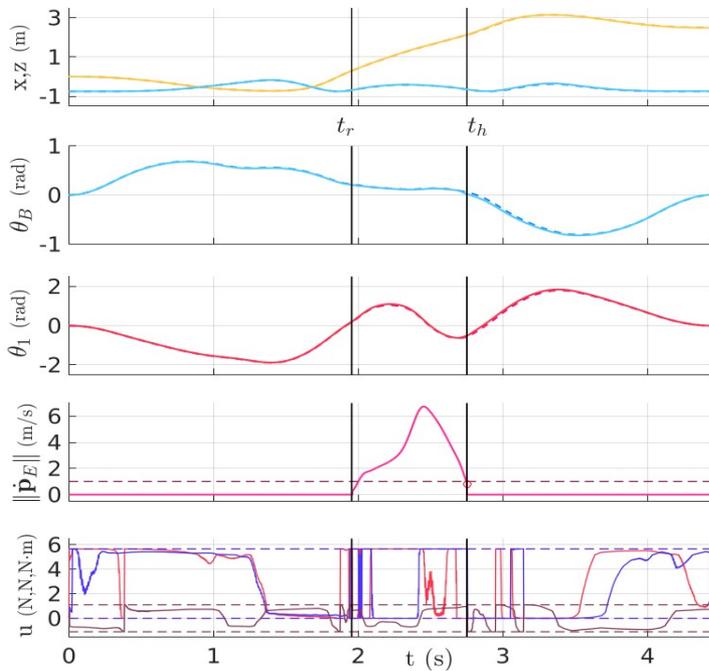
# MonkeyRotor : génération de trajectoires

Résultat intéressant : **émergence** d'une manoeuvre de balancement



# MonkeyRotor : simulations

- Commande temps-réel commuté
- Boucle fermée robuste aux perturbations



# MonkeyRotor : trajectoire robuste

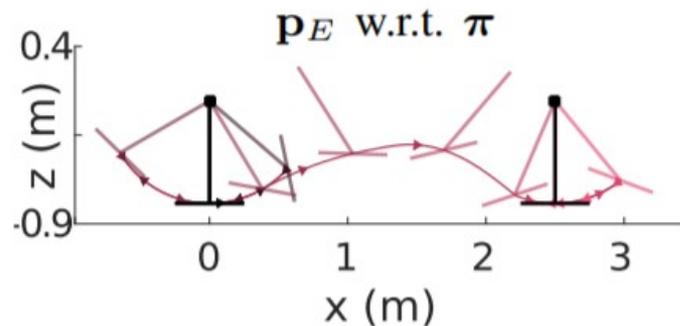
- Le **rattachement** est difficile :
  - Il faut s'assurer que l'effecteur atteint la **branche**
  - Une des causes d'imprécision vient des paramètres  $\mathbf{p}$  du modèle

# MonkeyRotor : trajectoire robuste

- Le **rattachement** est difficile :
  - Il faut s'assurer que l'effecteur atteint la **branche**
  - Une des causes d'imprécision vient des paramètres  $\mathbf{p}$  du modèle
- On construit un coût à minimiser pour réduire cet effet :

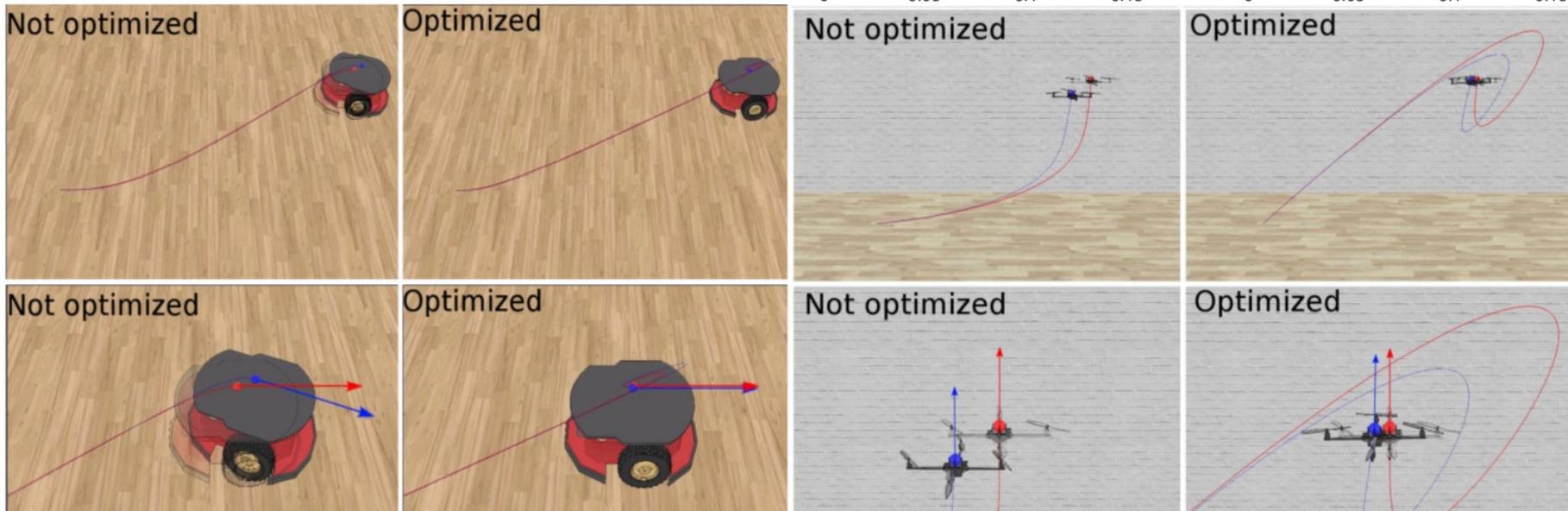
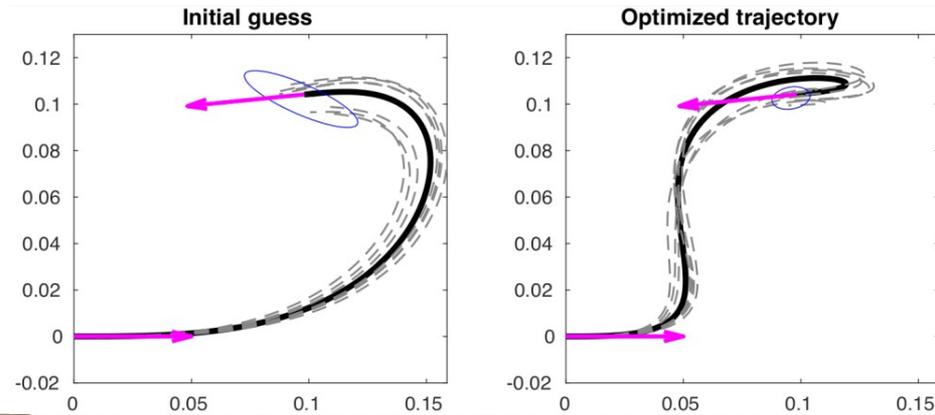
$$J_{sens} := \|\Psi(\mathbf{t}_2)\| = \left\| \frac{\partial \mathbf{x}}{\partial \mathbf{p}}(\mathbf{t}_2) \right\|$$

$$J_{tot} := t_f + \gamma J_{sens}$$



# Trajectoires robustes aux incertitudes

- Méthode testée sur robots unicycle et quadricopter
- Comparaison de la performance en suivi de trajectoire avec des **erreurs paramétriques aléatoires**
- Les trajectoires optimisées donnent bien lieu à de plus **petites erreurs**



# Conclusion

## Fait :

- Premier développement pour la locomotion aérienne
  - Génération de trajectoires pour le MonkeyRotor
  - Simulation du système asservi
  - Réalisation d'un système de rattachement pour le proto
- Framework pour la génération de trajectoires « minimal-sensitive »
  - Validation du concept
  - Généralisation de la méthode
  - Validation expérimentale

## A faire :

- Faire fonctionner un proto complet du MonkeyRotor
- Appliquer la méthode « sensitivity » à des robots plus complexes



Merci

**Questions ?**