Informatique

Newton – Dichotomie

TD – Système 4 barres

A. Système 4 barres

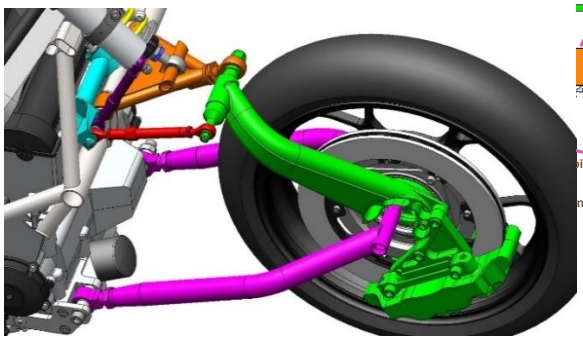
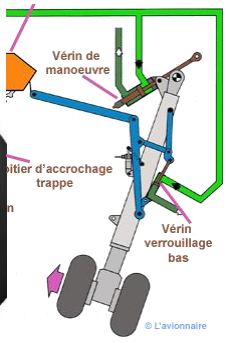
Mise en situation

Un grand nombre de systèmes qui nous entourent présente un modèle cinématique similaire contenant 4 liaisons pivots et 4 « barres », en voici des exemples :





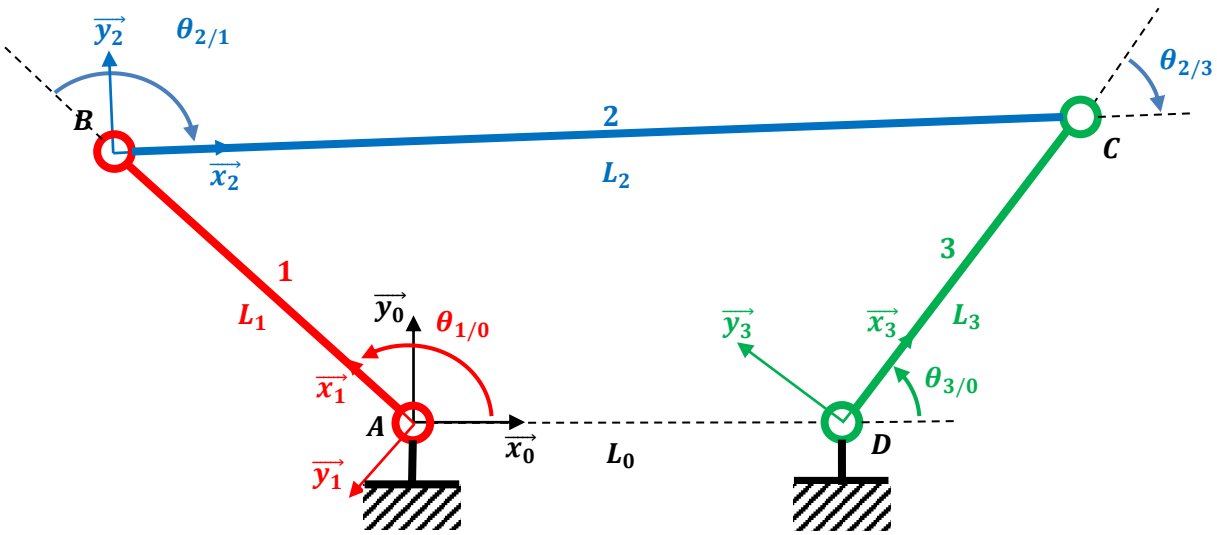


Ce genre de mécanismes plans sont les plus complexes à résoudre à la main du fait de leur équation caractéristique vue plus loin dans ce sujet.

Modèle cinématique

On propose le modèle cinématique général suivant :



Objectifs

Notre objectif consiste à :

- Déterminer la relation liant les angles d'entrée θ_{10} et de sortie θ_{30}
- Résoudre cette relation sous Python avec une méthode itérative par dichotomie
- Mettre en place une animation du schéma cinématique sous Python

Données

Nous étudierons un système dont les dimensions sont les suivantes :

$$L_0 = 100\ mm \quad ; \quad L_1 = 100\ mm \quad ; \quad L_2 = 150\ mm \quad ; \quad L_3 = 100\ mm$$

-Mise en place de l'équation du problème-

On souhaite obtenir l'équation liant θ_{10} (entrée) et θ_{30} (sortie).

Question 1: Ecrire la fermeture de chaîne géométrique du système et en déduire les 3 équations scalaires associées par projection dans la base 0
Question 2: Justifier le fait que le choix de la base 0 est propice à la détermination de la relation liant $\boldsymbol{\theta_{10}}$ et $\boldsymbol{\theta_{30}}$
Question 3: Mettre en place la relation entre $\boldsymbol{\theta_{10}}$ et $\boldsymbol{\theta_{30}}$ et les longueurs du système

On note

$$x = \theta_{10} \quad ; \quad y = \theta_{30}$$

Question 4: Mettre cette relation sous la forme $-\boldsymbol{a}\cos\boldsymbol{x} + \boldsymbol{b}\cos\boldsymbol{y} - \boldsymbol{c}\cos(\boldsymbol{y}-\boldsymbol{x}) + \boldsymbol{d} = \boldsymbol{0}$ en précisant ce que valent les constantes $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}$

On souhaite finalement résoudre l'équation suivante :

$$f(x,y) = -a\cos x + b\cos y - c\cos(y-x) + d = 0$$
$$\begin{cases} a = 2L_0L_1 \\ b = 2L_0L_3 \\ c = 2L_1L_3 \\ d = {L_0}^2 + {L_1}^2 - {L_2}^2 + {L_3}^2 \end{cases}$$

Il est possible de résoudre cette équation par le calcul, vous pouvez jeter un œil à ma démonstration jointe à ce sujet. Regardez la solution à la page 10 😊 et vous verrez que savoir procéder par dichotomie, c'est chouette !

-Code élève et modèle SW-

Vous avez à disposition un dossier élèves contenant le code élèves à compléter (utile uniquement pour l'affichage du schéma cinématique en fin de TD) et un modèle SolidWorks du mécanisme.

Question 5: En vous aidant du modèle SolidWorks fourni, justifier le fait que cette équation admet, dans le cas d'un système « qui peut fonctionner », deux solutions $\boldsymbol{y}$ dans l'intervalle $[\boldsymbol{0}, \boldsymbol{2\pi}]$ pour une valeur de $\boldsymbol{x}$ donnée

On supposera que les pièces ne se croisent pas dans la suite.

+Dichotomie+

Question 6: Créer la fonction $dichotomie(fy, Crit, y_1, y_2)$ déterminant la solution par dichotomie de l'équation $fy(y) = 0$ en partant de l'intervalle $[y_1, y_2]$ avec un critère $Crit$ sur la solution y obtenue

Remarque : On utilisera une assertion pour vérifier que la solution existe dans l'intervalle de départ

Question 7: Dans le code, définir $L0$, $L1$, $L2$ et $L3$ en mm et créer a,b,c et d

Question 8: Créer la fonction $f(x, y)$ afin qu'elle renvoie le résultat de l'équation issue de la fermeture géométrique pour x et y quelconques

Vérifier :

```
>>> f(1,2)
-22435.028965668433
```

Question 9: Définir la fonction $fy(y) = f(x, y)$ dont la valeur x sera une variable globale pour cette fonction

```
>>> x=1
```

Vérifier :

```
>>> fy(2)
-22435.02896566844
```

Sur un ordinateur en 64 bits, vérifier pour x=1 : `>>> dichotomie(fy,0.001,0,pi)`
`0.3275048982135844`

Remarque : Pour trouver la même solution que moi, il faut renvoyer le milieu du dernier intervalle en veillant à bien être à une solution à Crit près.

Question 10: Créer la fonction Affiche_Crb(fig,L1,L2) qui trace sur la figure fig la courbe de L2 en fonction de L1

Question 11: Tracer et observer la courbe $fy(y)$ pour $x = 240°$ et $y \in [-\pi, \pi]$ avec un pas de 1°

Nous souhaitons résoudre l'équation de la fermeture géométrique itérativement, c'est-à-dire :

- Partir d'une position initiale de la pièce 1 (entrée) valant $x = 240°$ incluse
- Réaliser des itérations tant que la position de la pièce 3 (sortie) est supérieure à $y = -45°$:
  - Diminuer x de 2°
  - Recherche la solution y dans l'intervalle $\left[-\frac{\pi}{3}, pi\right]$
  - Stocke la valeur x (rd) dans la liste Lx_r et la valeur y (rd) dans la liste Ly_r

ATTENTION : Tout doit être programmé en radians

Remarque : Ces valeurs sont valables dans le cas de notre étude et ont été déterminées à tâtons avec le code fini en parallèle de SolidWorks afin de n'avoir qu'une solution à l'équation résolue.

Question 12: Mettre en place cette résolution conduisant à la création des listes Lx_r et Ly_r des angles x et y en radian

Remarque : Aucune solution ? J'ai dit 2°, pas 2 rd

Question 13: Tracer la loi entrée/sortie y=f(x) en degrés

+Et Newton ?+

Question 14: Proposer la fonction fyp(fy,y) renvoyant une approximation de la dérivée de fy en y avec dy=0.00001

Vérifier :

```
>>> x = 1

>>> fyp(fy,0)
-16829.465665432508
```

Question 15: Proposer une fonction newton(fy,fyp,y0,eps) réalisant la résolution de l'équation fy(y)=0 avec la méthode de newton

Vérifier :

```
>>> x = 1

>>> newton(fy,fyp,150*pi/180,0.001)
3.814854454652682
```

Il ne reste plus qu'à réaliser la même procédure que précédemment en dichotomie afin d'obtenir les solutions de l'équation $fy(y) = 0$ pour $x = 240°$ en le faisant évoluer de $-1°$ jusqu'à ce que y soit inférieur à $-45°$.

Question 16: Mettre en place le code réalisant traçant la loi entrée/sortie y=f(x) en degrés

Remarques :

- A la première itération, et pour converger vers la bonne solution, partez d'une valeur d'abscisses y_0 proche de la solution attendue, que vous connaissez de la partie précédente 😉
- Aux suivantes : comme la position de sortie y évolue peu si l'entrée x évolue peu, prenez pour valeur initiale la solution de l'itération précédente 😉

+Bisect et Newton+

Je vous recommande d'essayer de réaliser les mêmes résolutions que précédemment en utilisant les fonction de Python « bisect » et « newton ».

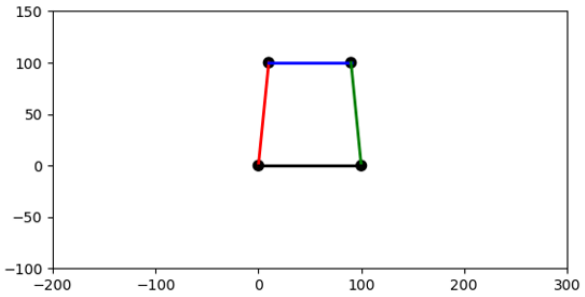
-Animation du schéma cinématique-

Vous avez à disposition dans le dossier élève chargé précédemment un code contenant une fonction d'affichage Affiche_Schema(fig,Pivots,Pieces,Chemin) qui permet d'afficher le schéma cinématique du système et d'en enregistrer l'image dans le dossier Images (chemin de la forme `"Images\\" + str(nim) + ".png"` où `nim` est un entier à partir de 1.

Quelques précisions :

- fig : numéro de la figure d'affichage
- Pivots : Liste de listes des coordonnées des liaisons pivots [Xi,Yi]
- Pieces : Liste de listes de 3 valeurs [Pivot1, Pivot2, Couleur] telles que la pièce de couleur Couleur est reliée à Pivot 1 et Pivot 2 (indices des pivots dans Pivots)

Un code est préétabli dans le fichier élèves afin d'afficher l'image ci-dessous et de l'enregistrer dans le dossier « Images » sous le nom « Essai.png ».



Attention : comme précisé lors de la définition des longueurs, tout cela fonctionnera si vos dimensions sont définies en mm.

Question 17: En utilisant la fonction proposée, afficher le schéma cinématique du mécanisme dans le cas où $\theta_{10} = x = 240°$

-Création d'une vidéo-

Vous avez maintenant une liste d'images à votre disposition, il ne reste plus qu'à créer une vidéo d'animation si cela vous amuse. Consultez mon cours sur Matplotlib pour savoir comment la réaliser.

Et voici ma vidéo : LIEN