Contrôle de robots unicycle et bicyclette

David FILLIAT

29 janvier 2020

1 Introduction

Dans ce TP, nous allons mettre en oeuvre des méthodes de contrôle très simples, à base de contrôleur PID, pour le contrôle de robots unicycle et bicyclette. Les programmes fournis permettent de simuler des robots se déplaçant sur le plan en suivant des modèles unicycle et bicyclette en exploitant un contrôle que vous devez programmer.

2 Contrôle de modèle unicycle

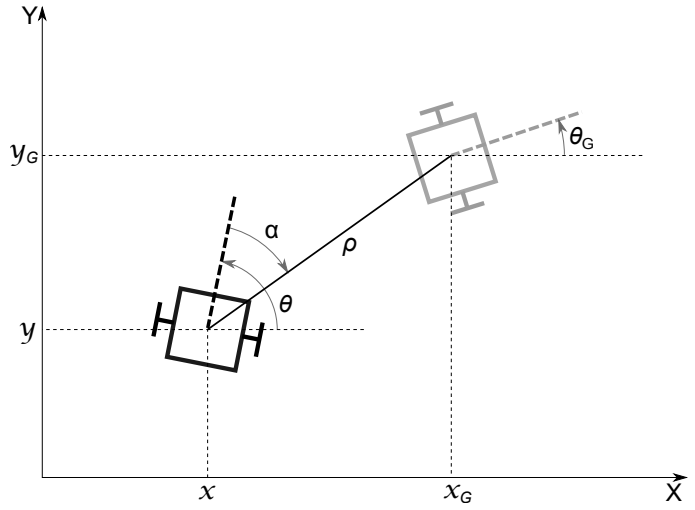

FIGURE 1 – Contrôle d'un unicycle vers une pose.

Dans ce premier exercice, vous devez écrire un contrôleur PID qui guide un unicyle vers une pose fixée (figure 1). Le modèle unicycle a une vitesse de translation limitée ($\|v\| < 1m.s^{-1}$) et une vitesse de rotation limitée ($\|\omega\| < \pi rad.s^{-1}$). De plus l'accélération est elle aussi limitée ($\|\frac{dv}{dt}\| < 10m.s^{-2}$ et $\|\frac{d\omega}{dt}\| < 10rad.s^{-2}$).

2.1 Question 1 - contrôle unicycle

Vous devez implémenter deux contrôleurs Proportionnels, un premier pour amener le robot sur la position du but et un second pour ajuster son orientation :

— Loin du but ($\rho > 0.05$), le contrôleur utilisera les variables ρ et α illustrée dans la figure 1 pour calculer les vitesses de translation et de rotation :

$$
\begin{aligned}
\rho &= \sqrt{(x_G - x)^2 + (y_G - y)^2} \\
\alpha &= \arctan \frac{(y_G - y)}{(x_G - x)} - \theta \\
v &= K_\rho \times \rho \\
\omega &= K_\alpha \times \alpha
\end{aligned}
$$

Pour aller plus vite vers le but, sans faire de grands arcs de cercle, vous pouvez faire tourner le robot sur place si sa direction est trop éloignée de la direction du but :

$$
\begin{aligned}
\text{si } |\alpha| &> \alpha_{max} \\
v &= 0
\end{aligned}
$$

— Lorsque le robot est proche du but ($\rho < 0.05$), le contrôleur utilisera la variable β ci-dessous pour calculer la vitesse de rotation :

$$
\begin{aligned}
\beta &= \theta_G - \theta \\
\omega &= K_\beta \times \beta
\end{aligned}
$$

Vous devez implémenter cette méthode dans la fonction `UnicycleToPoseControl`. Pensez à utiliser la fonction `atan2` (voir doc Python dans NumPy) pour calculer l'arctangente. Utilisez également la fonction fournie `AngleWrap` pour ramener les angles systématiquement entre $-\pi$ et π. Vous pouvez tester votre méthode à partir d'une position aléatoire à l'aide de la fonction `UniclycleToPose`. Réglez les gains du contrôleur et le paramètre α_{max} pour atteindre rapidement le but en limitant les oscillations.

3 Contrôle de modèle bicyclette

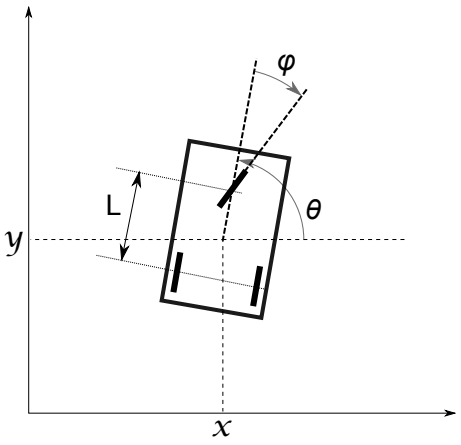

FIGURE 2 – Modèle bicyclette.

Nous allons maintenant réaliser le contrôle d'un modèle de bicyclette (figure 2). Ce modèle bicyclette a une vitesse de translation limitée ($\|v\| < 1m.s^{-1}$) et un angle de volant limité ($\|\phi\| < 1.2rad$). De plus l'accélération est elle aussi limitée ($\|\frac{dv}{dt}\| < 10m.s^{-2}$) ainsi que la vitesse de rotation du volant ($\|\frac{d\phi}{dt}\| < 8rad.s^{-1}$).

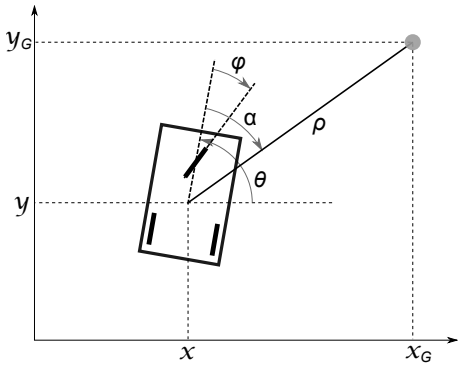

FIGURE 3 – Contrôle d'un modèle bicyclette vers un point.

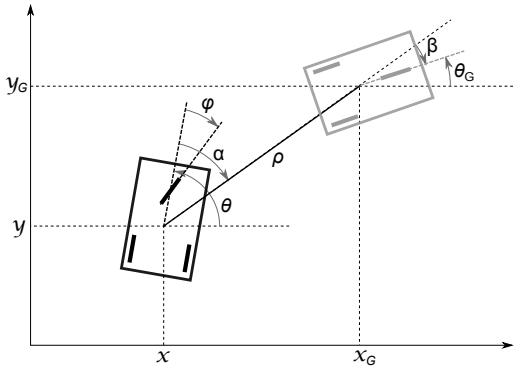

FIGURE 4 – Contrôle d'un modèle bicyclette vers une position.

3.1 Question 2 - contrôle de bicyclette vers un point

Tout d'abord, écrivez un contrôleur proportionnel qui guide le robot vers un point (figure 3). Pour cela, vous écrirez un contrôleur à partir des formules suivantes :

$$\begin{aligned} \rho &= \sqrt{(x_G - x)^2 + (y_G - y)^2} \\ \alpha &= \arctan \frac{(y_G - y)}{(x_G - x)} - \theta \\ v &= K_\rho \times \rho \\ \varphi &= K_\alpha \times \alpha \end{aligned}$$

Vous devez implémenter cette méthode dans la fonction `BicycleToPointControl`. Vous pouvez la tester à partir d'une position aléatoire à l'aide de la fonction `BiclycleToPoint`. Réglez les gains du contrôleur pour atteindre rapidement le but en limitant les oscillations.

3.2 Question 3 - contrôle de bicyclette vers une position

Ecrivez ensuite un contrôleur proportionnel qui guide le robot vers une position avec une orientation finale fixée (figure 4). Pour cela, vous écrirez un contrôleur à partir des formules suivantes :

$$\begin{aligned} \rho &= \sqrt{(x_G - x)^2 + (y_G - y)^2} \\ \alpha &= \arctan \frac{(y_G - y)}{(x_G - x)} - \theta \\ \beta &= \theta_G - \arctan \frac{(y_G - y)}{(x_G - x)} \\ v &= K_\rho \times \rho \\ \varphi &= K_\alpha \times \alpha + K_\beta \times \beta \\ &\textit{avec} \quad K_\beta < 0 \end{aligned}$$

Vous devez implémenter cette méthode dans la fonction `BicycleToPoseControl`. Vous pouvez la tester à partir d'une position aléatoire à l'aide de la fonction `BiclycleToPose`. Réglez les gains du contrôleur pour atteindre rapidement le but en limitant les oscillations.

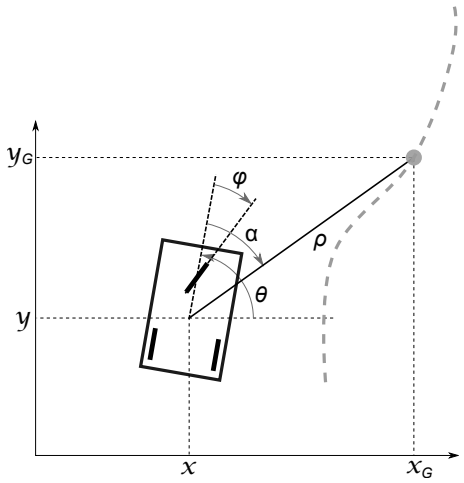

FIGURE 5 – Contrôle d'un modèle bicyclette selon un chemin.

3.3 Question 4 - contrôle de bicyclette selon un chemin

Ecrivez ensuite un contrôleur proportionnel qui guide le robot pour suivre un chemin (figure 5). Pour cela, vous devez à chaque instant déterminer un point sur la trajectoire qui soit un peu devant le robot, à une distance ρ (par exemple $\rho = 0.5$). Une méthode simple est de prendre un point sur la trajectoire et de le faire avancer d'un pas fixe lorsque la distance du robot à ce point est inférieure à ρ. A partir de ce point, écrirez un contrôleur selon les formules suivantes :

$$\begin{aligned} \rho &= \sqrt{(x_G - x)^2 + (y_G - y)^2} \\ \alpha &= \arctan \frac{(y_G - y)}{(x_G - x)} - \theta \\ v &= K_\rho \times \rho \\ \varphi &= K_\alpha \times \alpha \end{aligned}$$

Vous devez implémenter cette méthode dans la fonction `BicycleToPathControl`.