Présentation : Contrôle de la trajectoire

Table des matières

1 Introduction

1.1 Contexte : Formule ÉTS - voiture autonome

Dans le cadre de la Formule ÉTS, il y a un département driverless, je suis en charge de faire **le contrôle de la trajectoire**, mais aussi de faire l'algorithme de planification de la trajectoire.

Formule ÉTS

On peut voire dans la photo ci-dessous une voiture de l'équipe de Munich de Formula Student en épreuve driverless, ici, c'est du trackdrive, c'est-à-dire un circuit.

Équipe : Formula Student de Munich : épreuve du trackdrive en driverless

1.2 Stratégie driverless

Voici un schéma qui présente la stratégie driverless que la voiture va avoir.

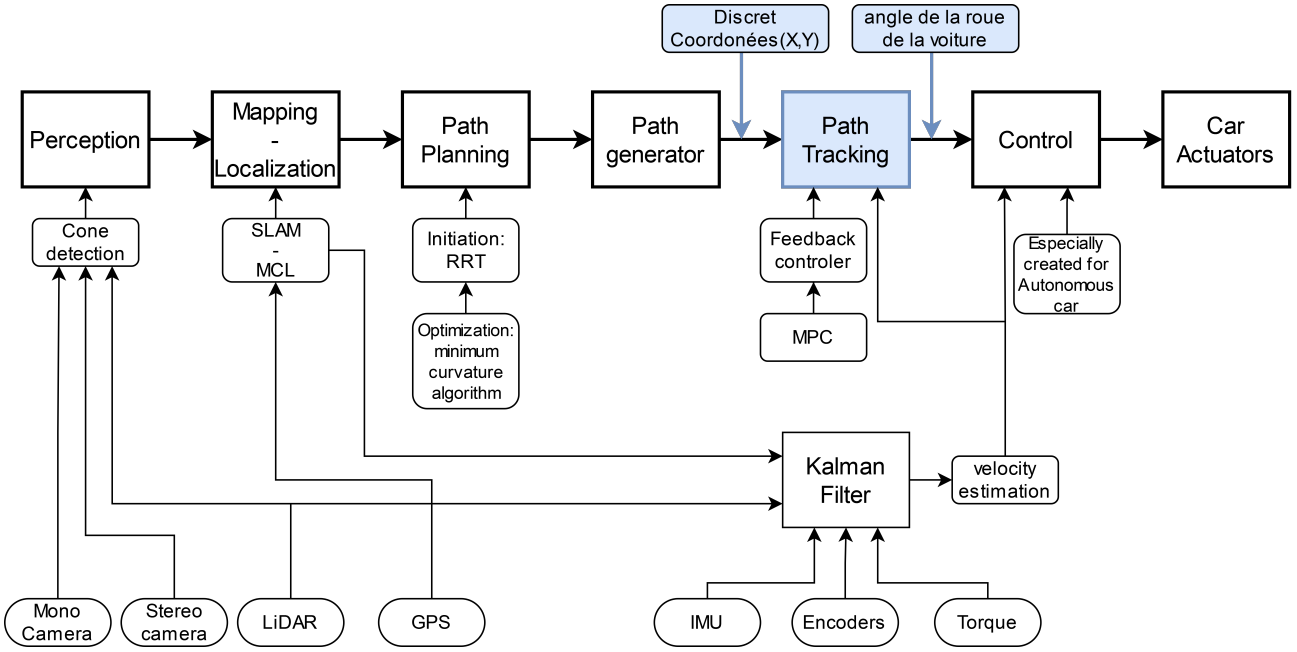


Schéma de la stratégie driverless

1.3 Schéma global : Path-Tracking $\Longleftrightarrow$ Contrôle de la trajectoire

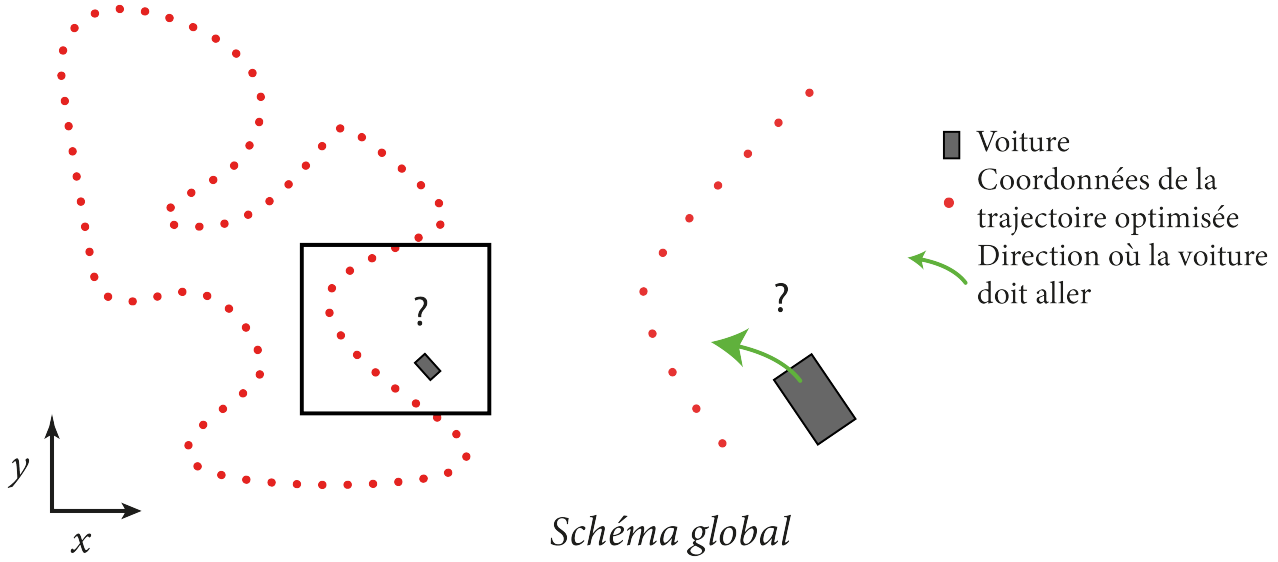
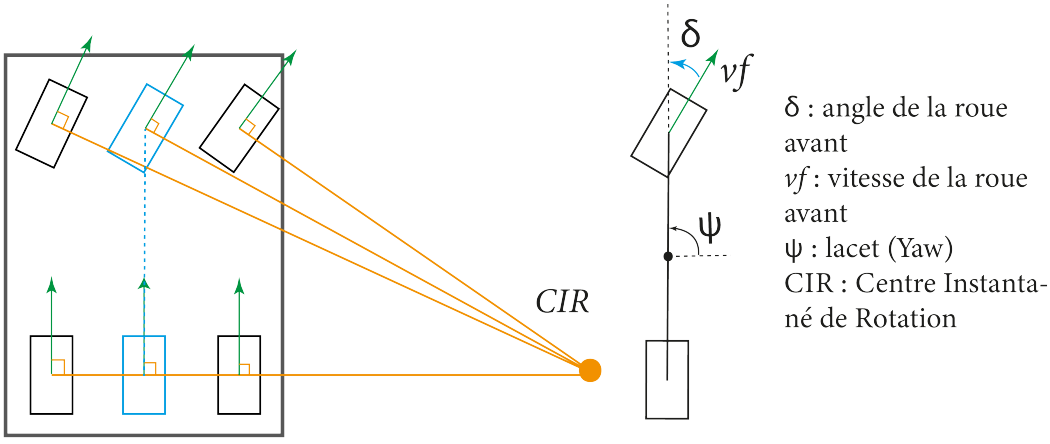


Schéma global

2 Présentation du modèle : bicycle model

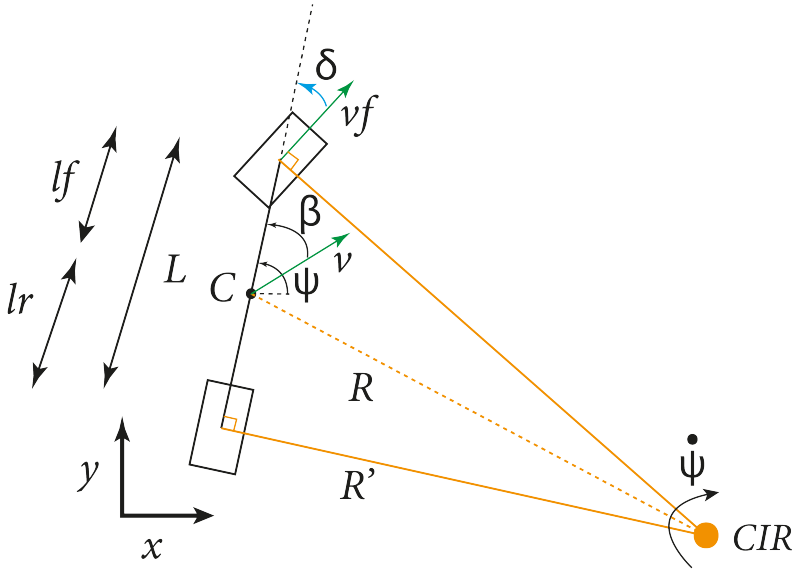
2.1 Présentation du bicycle model

Pour modéliser une voiture, on utilise en général le modèle d'Ackermann. Cependant, nous voulons ici simplifier le model, pour cela une méthode très courante est d'utiliser le bicycle modèle.



2.2 Cinématique

Voici la cinématique d'un bicycle model, j'utilise les notations de la convention SAE (Society of Automotive Engineers)



Bicycle model en cinématique

δ : angle de la roue avant
v : vitesse du véhicule
vf : vitesse de la roue avant
ψ : lacet (Yaw)
ψ : vitesse de rotation
β : slip angle
C : centre de gravité
L : distance inter roue ($lf + lr = L$)
lf : distance avec le Cet la roue avant
lr : distance avec le C et la roue arriere
R : distance en C et le CIR
R' : distance entre la roue arriere et le CIR
CIR : Centre Instantané de Rotation

En prenant $C = (x, y)$ et $w = \dot{\psi}$. On peut ainsi déterminer les 3 états (x, y, ψ) de la voiture :

$$\begin{cases} \dot{x} = v \cdot \cos(\psi + \beta) \\ \dot{y} = v \cdot \sin(\psi + \beta) \\ \dot{\psi} = \frac{v}{R} \end{cases}$$

Nous voyons ici que l'équation est déterminée avec R qui est une inconnue. Nous pouvons déterminer δ par rapport aux caractéristiques géométrique de la voiture.

$$\tan\delta = \frac{l_f + l_r}{R'} \Leftrightarrow R' = \frac{l_f + l_r}{\tan\delta}$$

$$\tan\beta = \frac{l_r}{R'} \Leftrightarrow \tan\beta = \frac{l_r}{l_r + l_f} \cdot \tan\delta$$

$$\cos\beta = \frac{R'}{R} \Leftrightarrow \frac{1}{R} = \frac{\cos\beta}{R} = \frac{\cos\beta \cdot \tan\delta}{l_f + l_r}$$

On trouve ainsi l'équation qui régit la voiture :

$$\boxed{\text{Bicycle model temporel : } \begin{cases} \dot{x} = v \cdot \cos(\psi + \beta) \\ \dot{y} = v \cdot \sin(\psi + \beta) \\ \dot{\psi} = \dfrac{v}{l_r + l_f} \cdot \cos\beta \cdot \tan\delta \end{cases} \quad avec\ \beta = \arctan(\frac{l_r}{l_r + l_f} \cdot \tan\delta)} \tag{1}$$

Avec les simplifications suivantes :

$$\begin{aligned} &l_f = l_r \\ &L = l_r + l_f \\ &petit\ angle \rightarrow \beta \sim 0,\ \tan(\delta) \sim \delta,\ \cos(\delta) \sim 1 \end{aligned}$$

On a alors l'équation suivante :

$$\boxed{\text{Bicycle model temporel simplifié : } \begin{cases} \dot{x} = v \cdot \cos(\psi) \\ \dot{y} = v \cdot \sin(\psi) \\ \dot{\psi} = \dfrac{v}{L} \cdot \delta \end{cases}} \tag{2}$$

Le Path planning envoie un nouveau point objectif tous les $T_e = \mathbf{0.1\ seconde}$, il faut donc **discrétiser** les équations :

$$\begin{cases} X(k+1) = X(k) + T_e \cdot v \cdot \cos(\Psi(k)) \\ Y(k+1) = Y(k) + T_e \cdot v \cdot \sin(\Psi(k)) \\ \Psi(k+1) = \Psi(k) + \frac{T_e \cdot v}{L} \cdot \delta(k) \end{cases}$$

$$\Rightarrow \boxed{\text{Bicycle model discret : } \begin{cases} x(z) = \dfrac{T_e \cdot v \cdot \cos(\psi(z))}{1 - z^{-}1} \\ y(z) = \dfrac{T_e \cdot v \cdot \sin(\psi(z))}{1 - z^{-}1} \\ \psi(z) = \dfrac{T_e \cdot v}{L(1 - z^{1})} \cdot \delta(z) \end{cases}} \tag{3}$$

On remarque que l'équation n'est pas linéaire à cause du cos et du sin. J'ai donc cherché à les linéariser. Pour cela, j'ai utilisé un développement limité à l'ordre 1. Ainsi, nous obtenons :

$$\Rightarrow \boxed{\text{Bicycle model discret linéarisé : } \begin{cases} x(z) = \dfrac{T_e \cdot v}{1 - z^{-}1} \\ y(z) = \dfrac{T_e \cdot v \cdot \psi(z)}{1 - z^{-}1} \\ \psi(z) = \dfrac{T_e \cdot v}{L(1 - z^{1})} \cdot \delta(z) \end{cases}} \tag{4}$$

J'ai les matrices du jacobien, ce n'est pas concluant pour l'instant :

$$A = \begin{pmatrix} 0 & 0 & -v\,\sin(\psi(t)) & 0 \\ 0 & 0 & v\,\cos(\psi(t)) & 0 \\ 0 & 0 & 0 & \frac{v}{L} \\ 0 & 0 & 0 & 0 \end{pmatrix} \sim \begin{pmatrix} 0 & 0 & -v\,\psi(t) & 0 \\ 0 & 0 & v & 0 \\ 0 & 0 & 0 & \frac{v}{L} \\ 0 & 0 & 0 & 0 \end{pmatrix} \text{ et } B = \begin{pmatrix} 0 \\ 0 \\ \frac{v}{L} \\ 0 \end{pmatrix}$$

Ainsi, grâce au jacobien on trouve que $\dot{X}$:

$$\dot{X} = \dot{X}_0 + \Delta\dot{X} = \begin{pmatrix} x_0 & x_0 & x_0 - \Delta x \cdot v\sin(\psi) & x_0 \\ y_0 & y_0 & y_0 + \Delta y \cdot v\cos(\psi) & y_0 \\ \psi_0 + \frac{\Delta\delta \cdot v}{L} & \psi_0 + \frac{\Delta\delta \cdot v}{L} & \psi_0 + \frac{\Delta\delta \cdot v}{L} & \psi_0 + \frac{\Delta\delta \cdot v}{L} + \frac{\Delta\psi \cdot v}{L} \\ \delta_0 & \delta_0 & \delta_0 & \delta_0 \end{pmatrix}$$

2.3 Dynamique

Je ne vais pas traiter la dynamique du véhicule pour l'instant. Je vais d'abord essayer d'avoir un modèle simple qui fonctionne et que je comprends. J'étudierai le contrôle du véhicule avec un modèle dynamique plus tard.
Traiter la dynamique du système.

3 Méthode de contrôle

3.1 Méthode géométrique : Pure Pursuite

La méthode pure pursuite est la méthode humaine de piloter une voiture : on regarde, essaie en temps réel d'aller à un endroit situer devant nous. On définit cette distance grâce à d la *lookahead distance*.

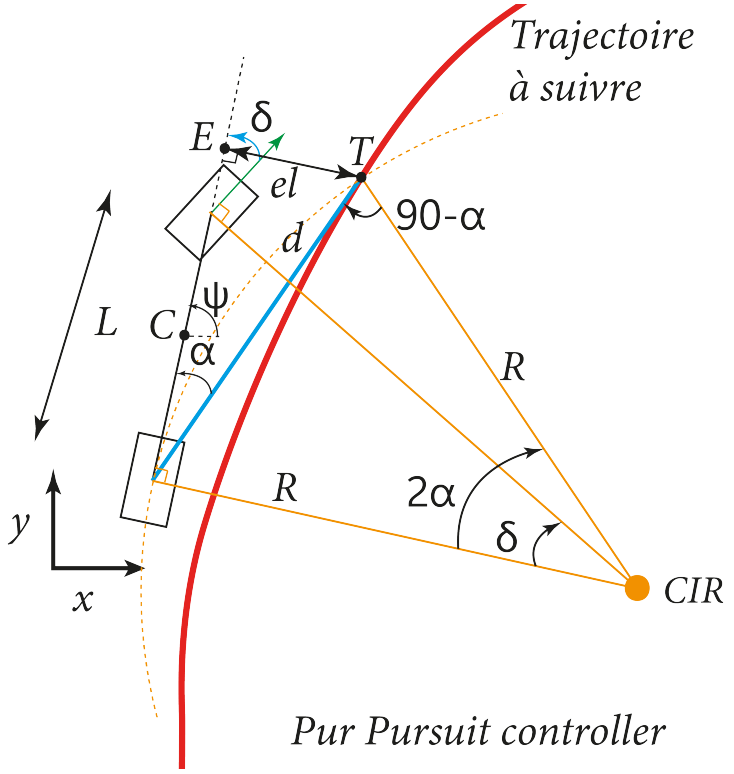
On remarque que

$$\frac{d}{\sin(2\alpha)} = \frac{R}{\sin(\frac{\pi}{2} - \alpha)} \iff \frac{d}{2\sin\alpha\cos\alpha} = \frac{R}{\cos\alpha} \iff \frac{1}{R} = \frac{2\sin\alpha}{d}$$

$$\tan\delta = \frac{L}{R} = \frac{2L\sin\alpha}{d}$$

$$\delta = \arctan(\frac{L}{R} = \frac{2L\sin\alpha}{d}) \rightarrow \textit{petit angle } \delta = \frac{L}{R} = \frac{2L\sin\alpha}{d}$$

$$\sin\alpha = \frac{e_l}{d} \rightarrow \textit{petit angle } \alpha = \frac{e_l}{d}$$



δ : angle de la roue avant
L : distance inter roue
R : rayon de braquage de la voiture
CIR : Centre Instantané de Rotation
T : point objectif (Target)
E : point erreur
el : erreur lateral entre la voiture et l'objectif
d : distance entre la roue arriere et l'objectif (look ahead distance)
ψ : lacet (Yaw)
α : angle entre la voiture et l'objectif

$$\text{Pure Pursuite : } \delta = \frac{2L}{d^2} \cdot e_l$$

On cherche la distance e_l entre E et T.
$T = (x_T, y_T)$
$E = (x_C + (\frac{L}{2} - d\cos\alpha)\sin(\psi), y_C + (d\cos\alpha - \frac{L}{2})\cos(\psi))$

En approximant que $\cos\alpha \sim 1$, on trouve :
$E = (x_C + (\frac{L}{2} - d)\sin(\psi), y_C + (d - \frac{L}{2})\cos(\psi))$

On peut déterminer la distance entre E et G :

$$d(E,T) = e_l = \sqrt{[x_T - x_C + (\frac{L}{2} - d)\sin(\psi)]^2 + [y_T - y_C + (d - \frac{L}{2})\cos(\psi)]^2}$$

En discret, nous faisons la différence latérale en le nouvel objectif par rapport au point E de la voiture actuelle :

$$d(E(k-1), T(k)) = e_l(k)$$
$$= \sqrt{[x_T(k) - x_C(k-1) + (\frac{L}{2} - d)\sin(\psi(k-1))]^2 + [y_T(k) - y_C(k-1) + (d - \frac{L}{2})\cos(\psi(k-1))]^2}$$

On cherche à linéariser cette fonction :
Pour cela nous allons faire des développements limités d'ordre 1 sur el en temporelle car plus simple :

$$el^2 = (x_g - x_c + (\frac{L}{2} - d)\sin(\psi))^2 + (y_g - y_c + (d - \frac{L}{2})\cos(\psi))^2$$

On veut faire un développement limité de $(1+X)^\alpha = 1+\alpha x$. Il faut donc que les termes aux carrées soient sous la forme $(1+x)$. On cherche donc que $(x_g - x_c + (\frac{L}{2} - d)\sin(\psi))^2 = (1+X)^2 \iff X = x_g - x_c + (\frac{L}{2} - d)\sin(\psi) - 1$. De même $Y = y_g - y_c + (d - \frac{L}{2})\cos(\psi) - 1$ et $E_l = e_l - 1$.
Ainsi, nous avons l'équation suivante :

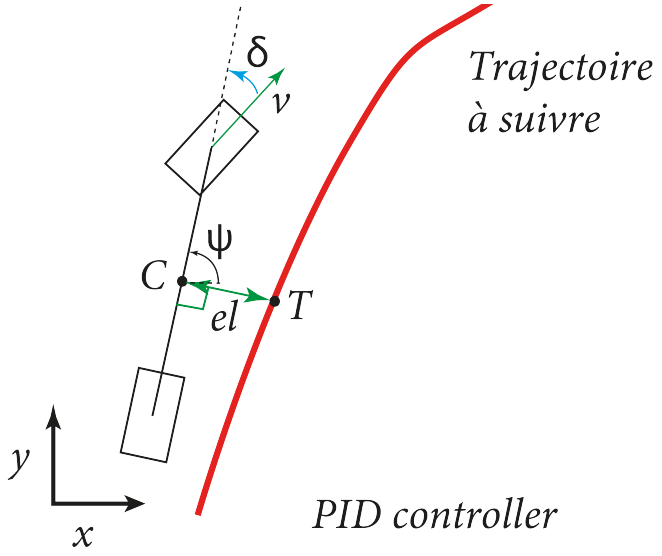
$$(E_l+1)^2 = (X+1)^2 + (Y+1)^2$$
$$\iff 1 + 2(e_l - 1) = 1 + 2(x_g - x_c + (\frac{L}{2} - d)\sin(\psi) - 1) + 1 + 2(y_g - y_c + (d - \frac{L}{2})\cos(\psi) - 1)$$
$$\iff e_l = x_g - x_c + y_g - y_c + (\frac{L}{2} - d)\cdot(\psi - 1) - 0.5$$

Ainsi, mettons cela en discret :

$$e_l(k) = x_g(k) - x_c(k-1) + y_g(k) - y_c(k-1) + (\frac{L}{2} - d)\cdot(\psi(k-1) - 1) - 0.5$$
$$\iff \boxed{e_l(z) = x_g(z) - z^-1 \cdot x_c(z) + y_g(z) - z^-1 \cdot y_c(z) + z^-1 \cdot (\frac{L}{2} - d)\cdot\psi(z) - (0.5 + \frac{L}{2} - d)}$$

3.2 Méthode black box : PID

La méthode de la Black-box PID consiste à contrôler son véhicule sans avoir besoin de comprendre la cinématique, la dynamique, le comportement du véhicule avec un PID.



δ : angle de la roue avant
v : vitesse du véhicule
C : Centre de gravité
e : distance entre C et la trajectoire à suivre
ψ : lacet (Yaw)
T : point objectif (Target)

On cherche la distance entre C et G :

$$d(T,C) = e_l = \sqrt{(x_T - x_C)^2 + (y_T - y_C)^2}$$

En discret :

$$\boxed{d(T(k),C(k)) = e_l(k) = \sqrt{(x_T(k) - x_C(k-1))^2 + (y_T(k) - y_C(k-1))^2}}$$

Mettre cette équation en **Z**

Il faut aussi transformer le correcteur PID en discret :
Le PID en continu :

$$\delta(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(t)\,dt + K_d \cdot \frac{de(t)}{dt}$$

Le PID en discret :

$$\delta(k) = K_p \cdot e(k) + K_i \cdot \sum_{i=0}^{k} e_i + K_d \cdot \frac{e(k) - e(k-1)}{Te}$$

Le PID en z :

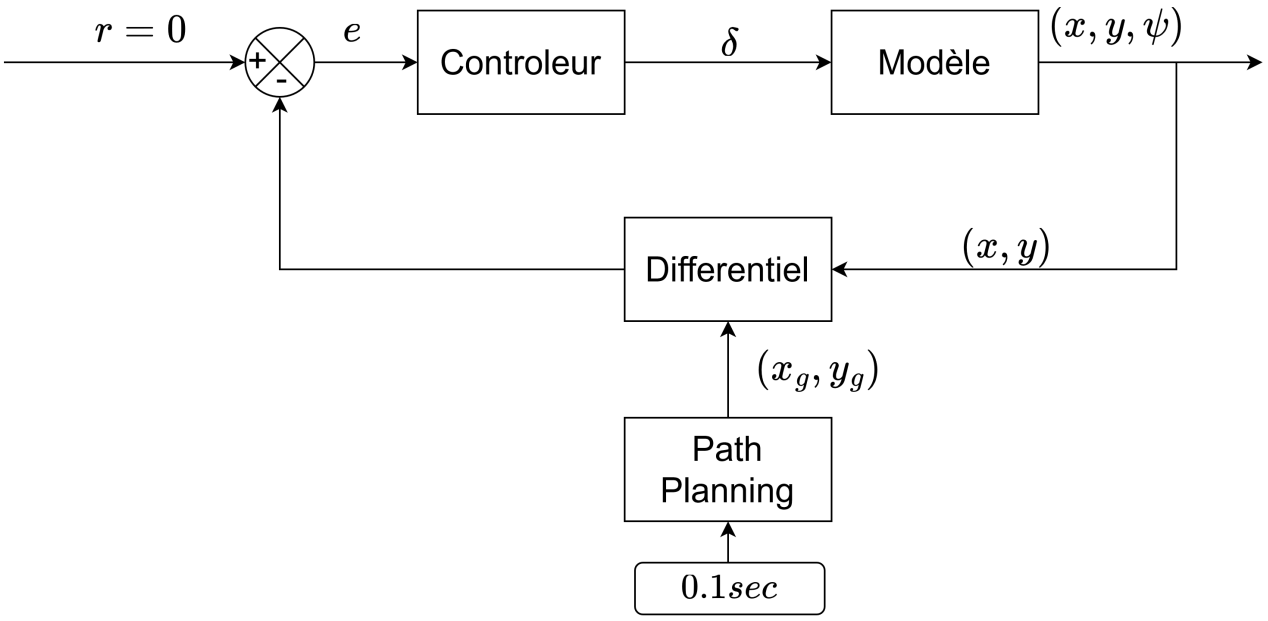
$$\delta(z) = K_p \cdot e(z) + K_i \cdot \frac{Te \cdot z}{z-1} + K_d \cdot \frac{z-1}{Te \cdot z}$$

4 Schéma-bloc

4.1 Aperçu

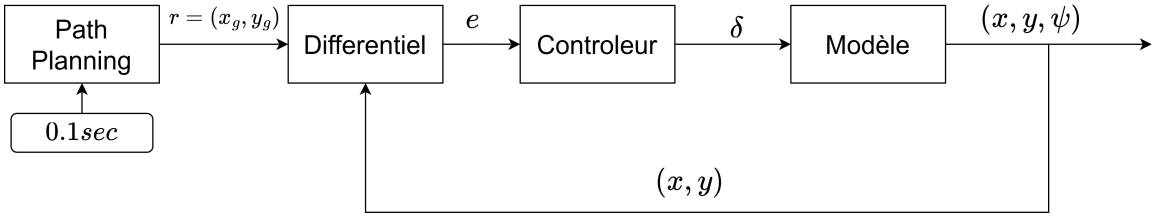
4.1.1 Distance entre la voiture et la trajectoire – Problème de régulation

On peut voir le contrôle de la trajectoire comme un problème de régulation.



4.1.2 Distance entre la voiture et la trajectoire – Problème d'asservissement

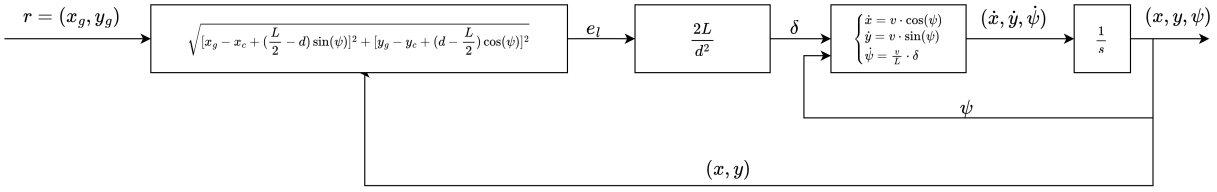
On peut aussi voir le contrôle de la trajectoire comme un problème d'asservissement.



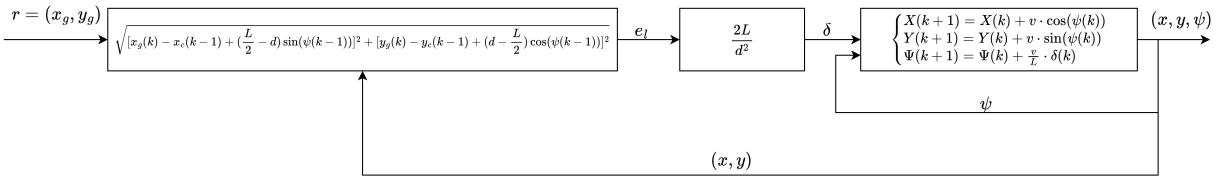
On choisit de prendre ça comme un problème d'asservissement.

4.2 Schéma-bloc : Pure Pursuite

4.2.1 Temporel

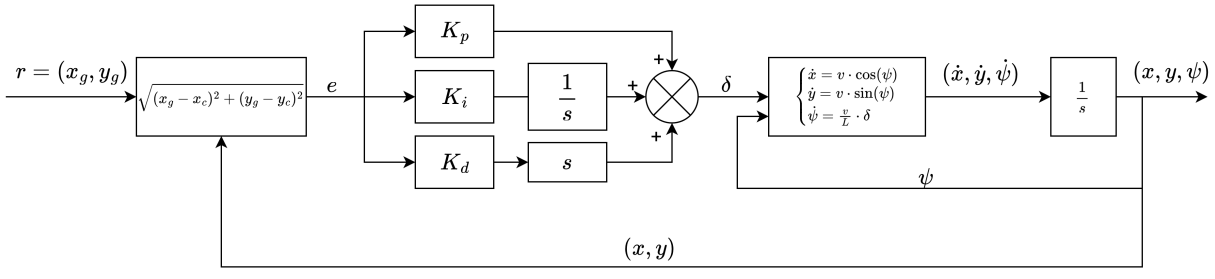


4.2.2 Discret

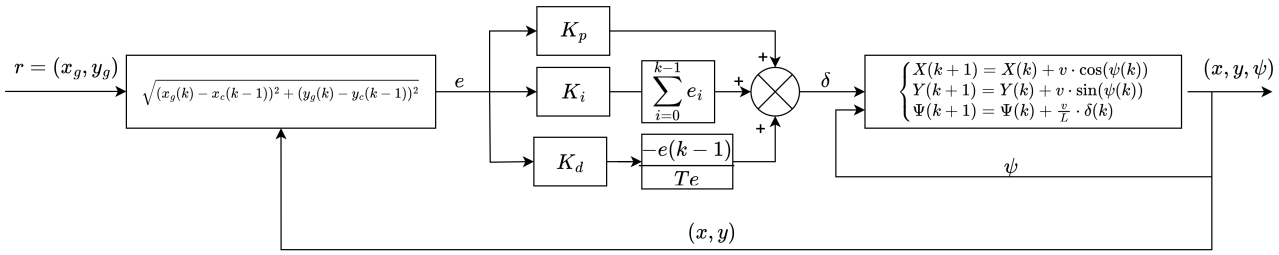


4.3 Schéma-bloc : PID

4.3.1 Temporel

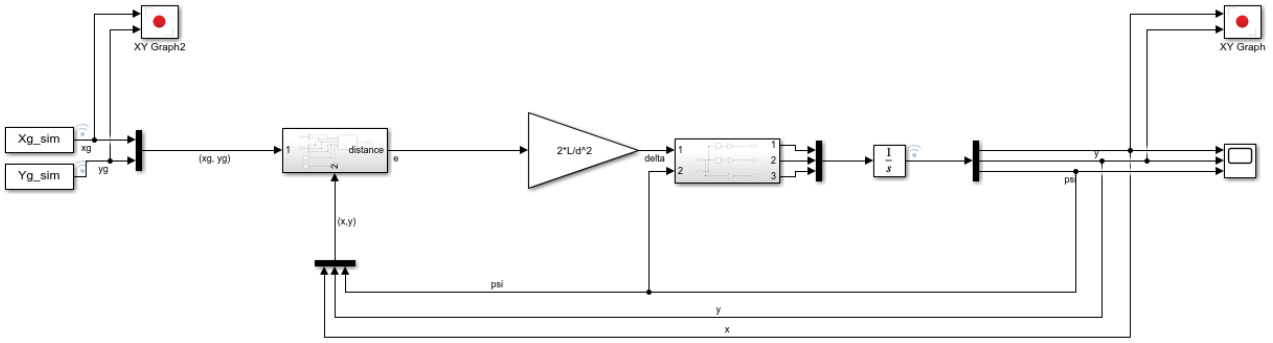


4.3.2 Discret



5 Matlab

5.1 Simulation du Pure Pursuit sur Simulink



5.2 Simulation du PID sur Simulink

À faire en suivant la même méthode que le Pure Pursuit

6 TO DO

- ☑ Discrétiser en z les équations de e_l pour le Pure Pursuite et le PID en utlisant e_l^2 pour simplifier la racine
- ☑ Passer par le jacobien pour simplifier le ψ et ainsi faire les petit angle dans e_l du Pur Pursuite
- ☐ Faire la dynamique du bicycle model
- ☒ Décomposer le modèle du système en $\frac{\psi}{\delta} \rightarrow \frac{(\dot{x},\dot{y})}{\psi} \Rightarrow (\cos\psi, \sin\psi) \Rightarrow$ non linéaire !
- ☐ Lire et résumer en MarkDown sur Obsidian
  - ☑ AMZ Driverless : The Full Autonomous Racing System
  - ☑ Automatic Steering Methods for Autonomous Automobile Path Tracking
  - ☐ Implementation of path tracking algorithms and trajectory optimization based on the extended kalman filter
  - ☐ Robotics, Vision and Control - Chap 4, 5, 6
- ☑ Faire un encadrer dans le schéma de stratégie sur path tracking / planning et indiqué que c'est du discret
- ☑ matlab symbolic pour vérifier linéarisation
- ☑ matlab symbolic pour jacobien
- ☐ Avec les formules linéarisées du bicycle model linéarisé
  - ☐ Faire simulink avec les nouvelles formules linéarisés
  - ☐ Faire matlab log avec les nouvelles formules linéarisés
  - ☐ Faire schema-bloc