BE Filtrage Optimal

Introduction

Présentation de la Coupe de France de Robotique

C’est quoi la coupe de France de Robotique ?

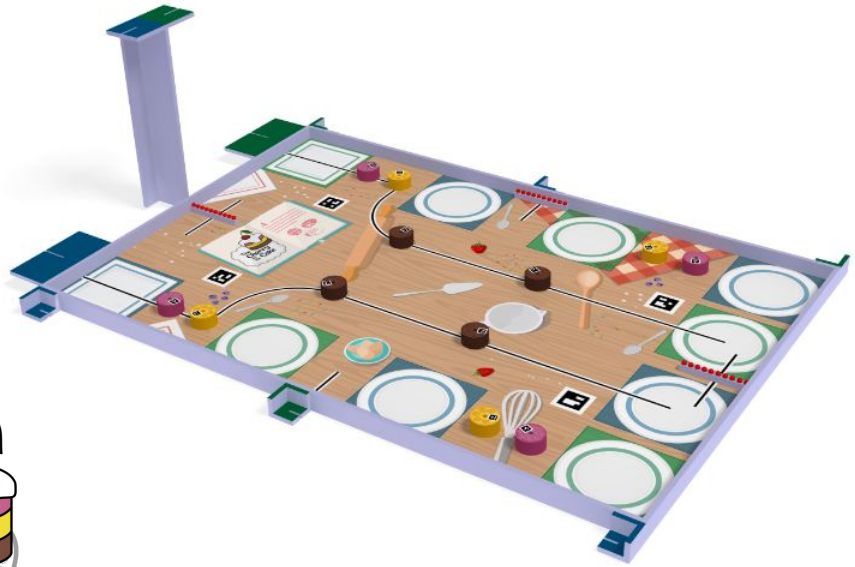
Coupe de france de robotique:

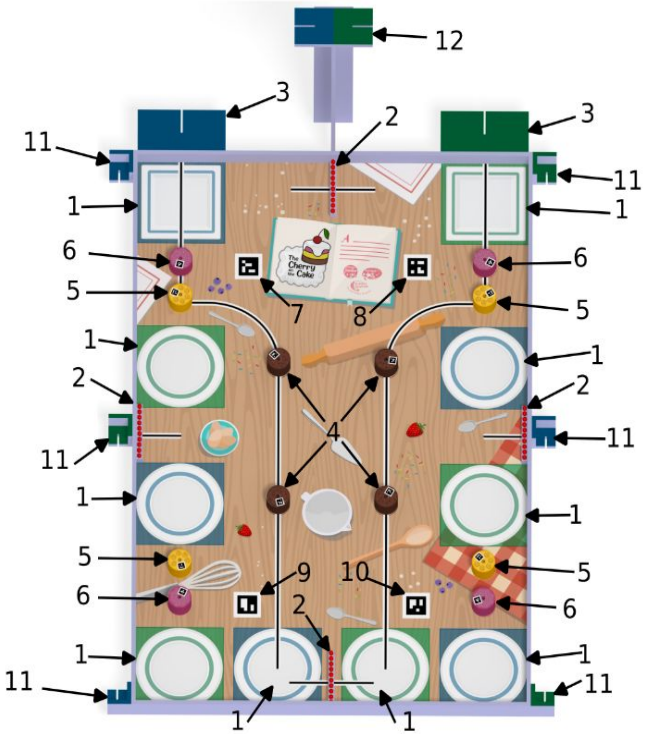
- Lieu : La Roche-Sur-Yon
- Date : du 17 au 21 mai 2023

Plus de 200 équipes inscrites (IUT, écoles d’ingé, universités, etc…)

Des industriels partenaires de la Coupe (STMicroelectronic, Michelin, Exotec, etc)

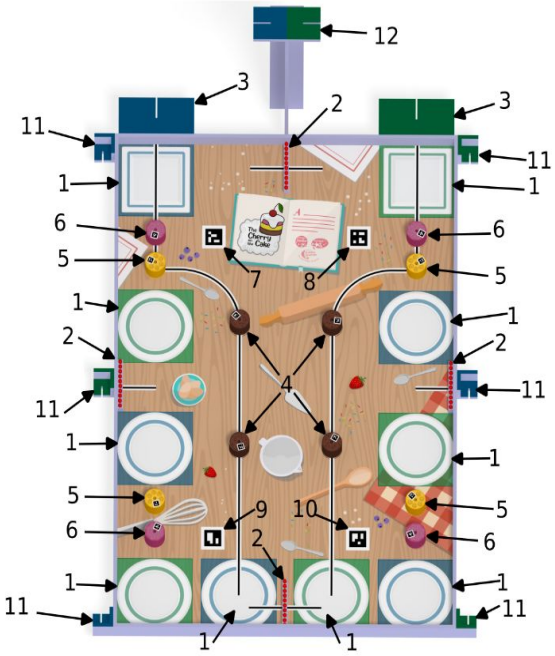
L’édition de cette année



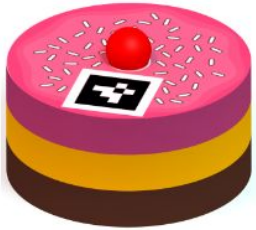


Les Objectifs Principaux

1) Faire des gâteaux



1 points par étage

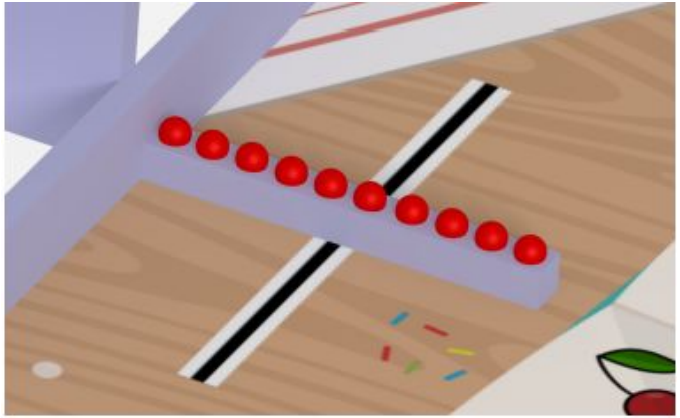


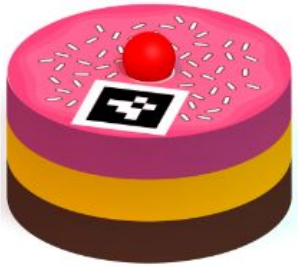
+3 points pour un gâteau complet à 3 étages

Les Objectifs Principaux

2) Mettre la cerise sur le gâteau :

- 10 Cerises par équipes et 40 cerises disponible sur le plateau

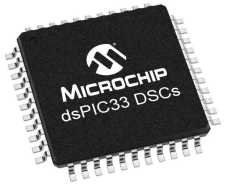




+4 points si la cerise est posée sur le gâteau

Notre Architecture

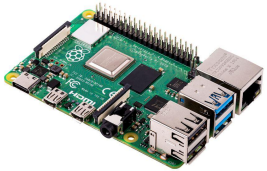
Cartes sous Microcontrôleur PIC :



- Contrôle total sur l'Hardware
- Grande Réactivité car grande proximité avec l'Hardware

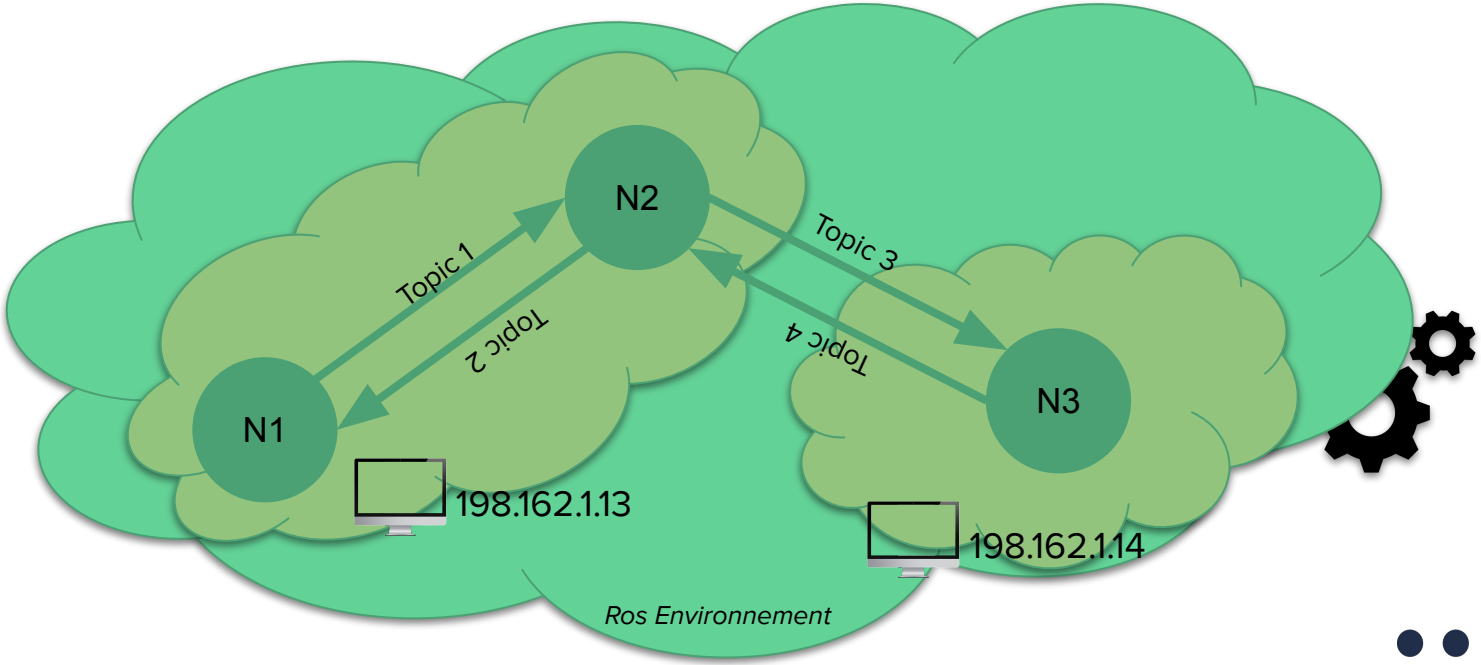
Liaison USB

Carte Raspberry Pi sous Linux :



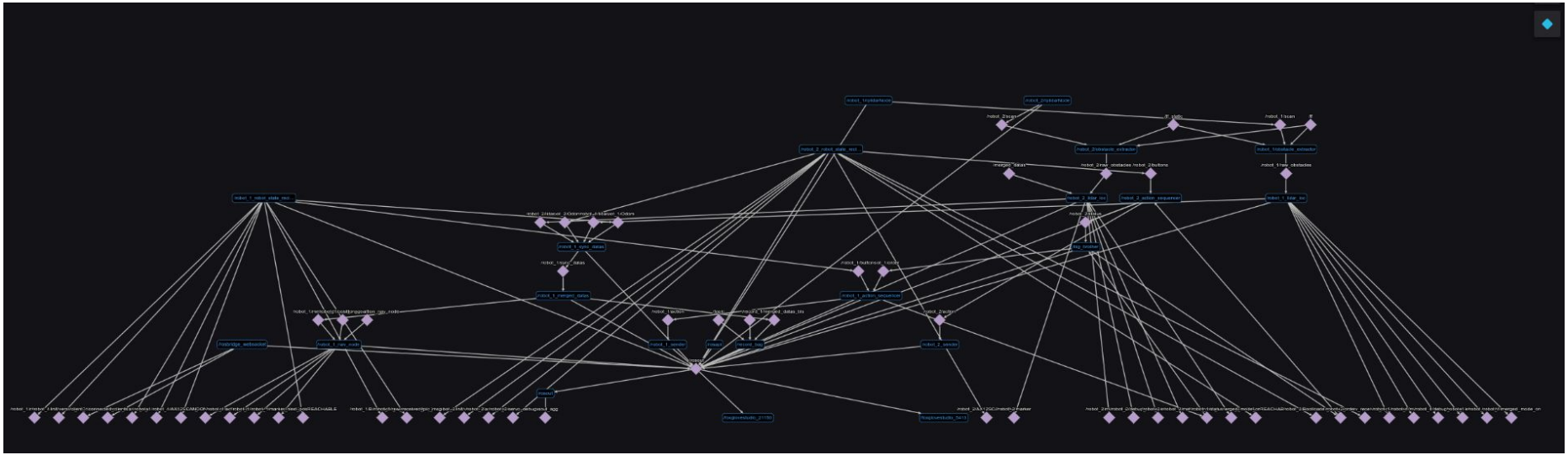
- Réalisation de Calculs Complexes
- Grand panel de librairies disponibles

Notre Architecture



ROS

Notre Architecture

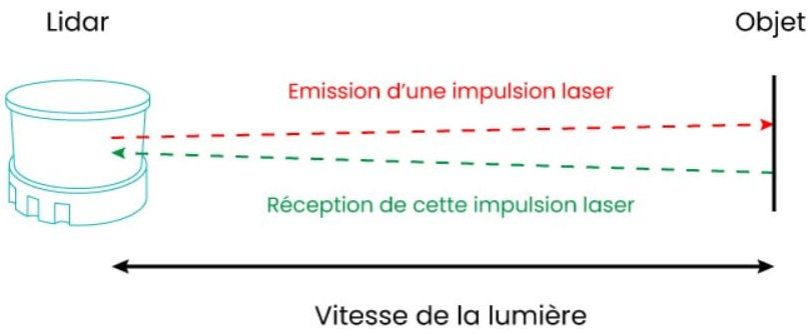


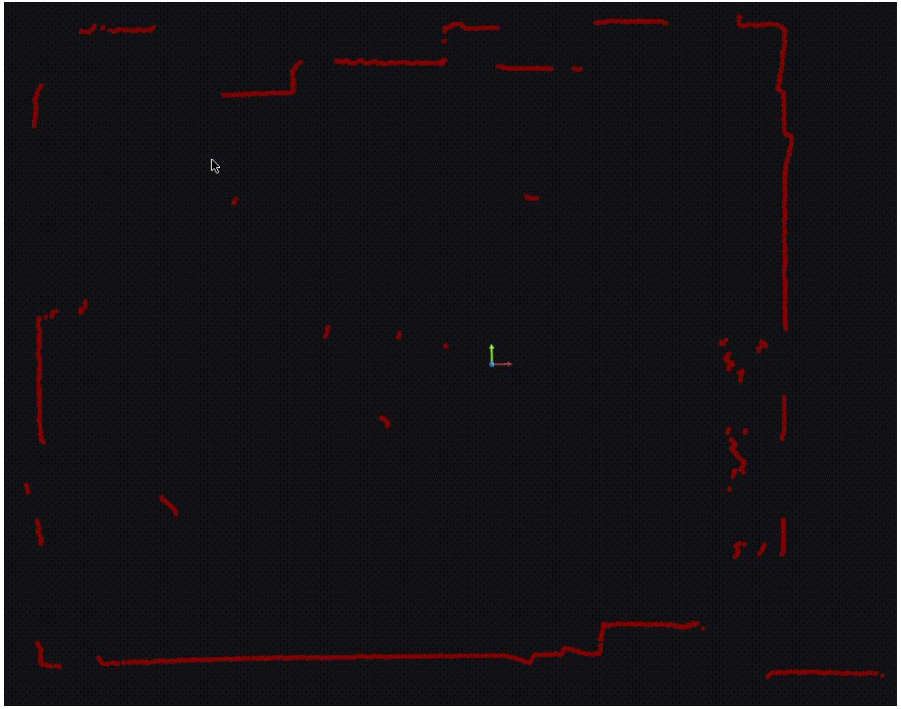
Positionnement Lidar

Les données renvoyées par le lidar

Capteur Infrarouge

Rplidar A3

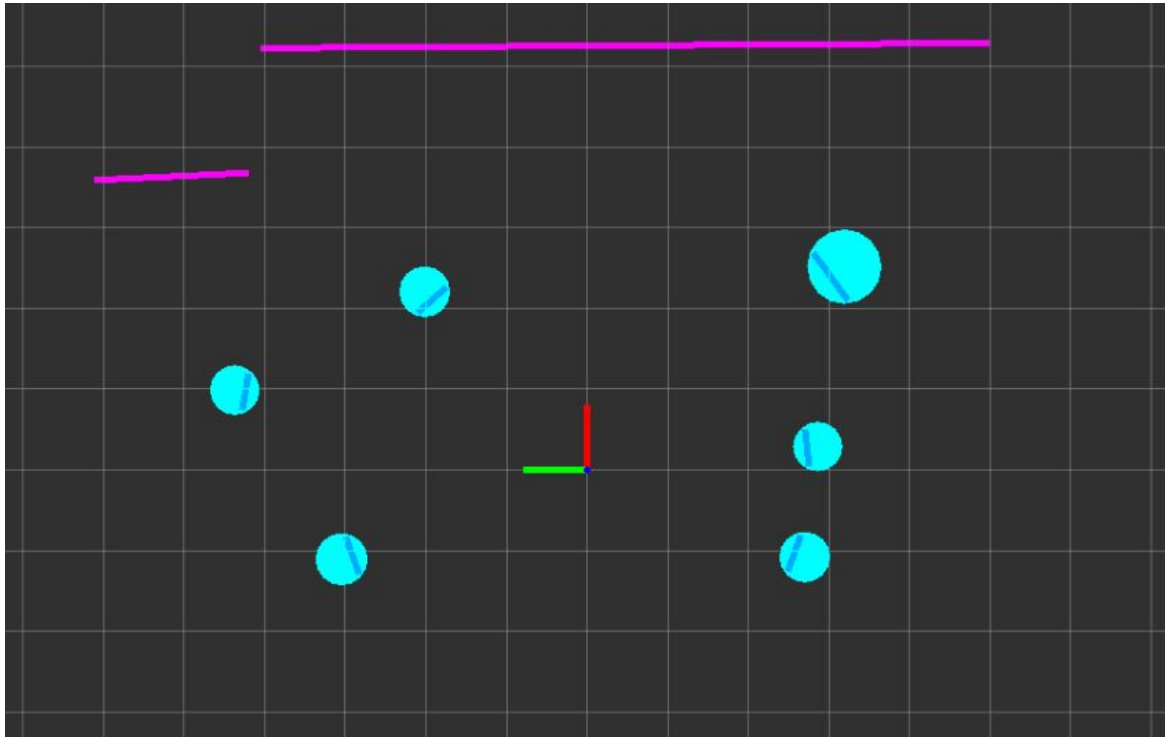




Utilisation d’une librairie

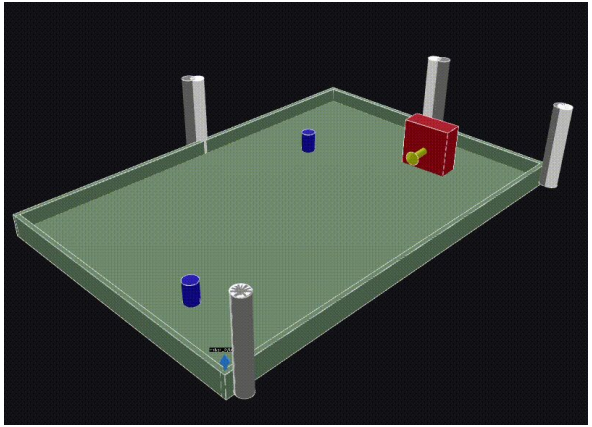
GitHub

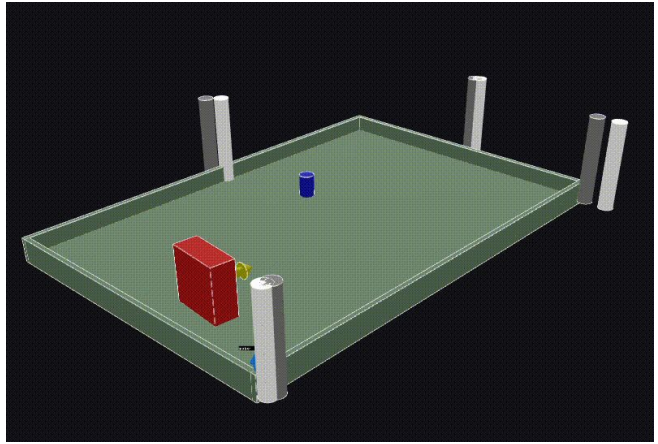
obstacle_detector



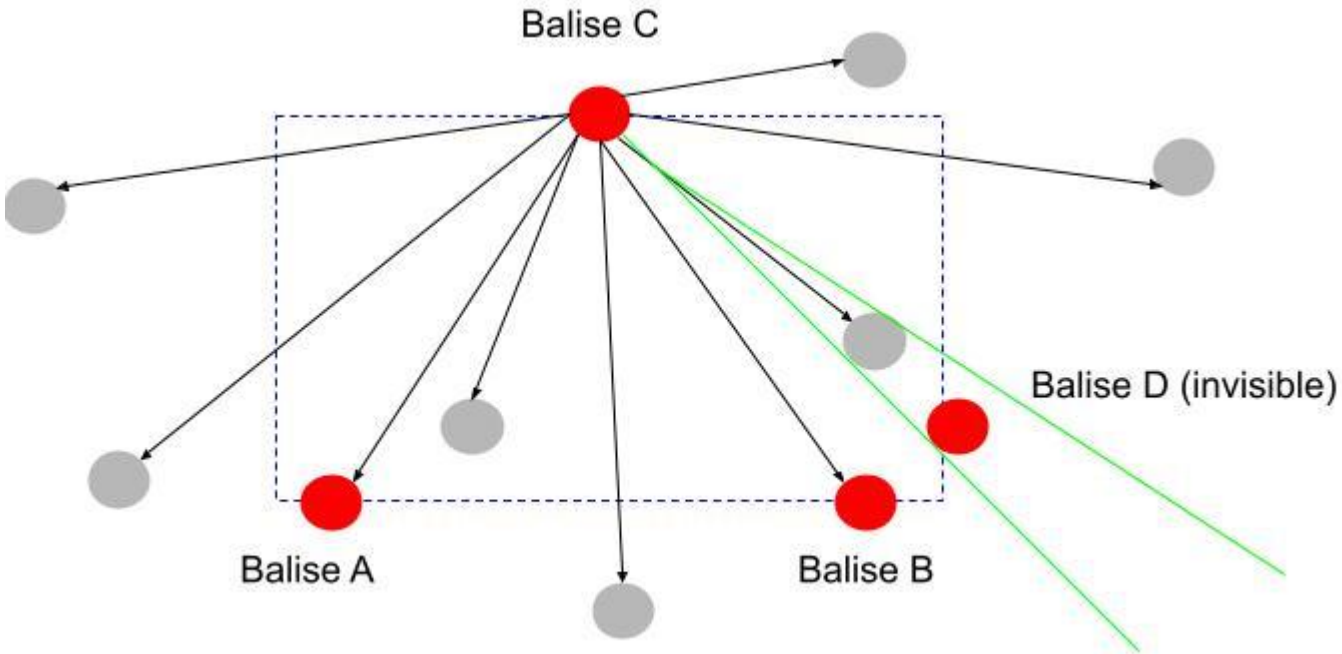
Utilisation d’une librairie

Rendu final après analyse des données

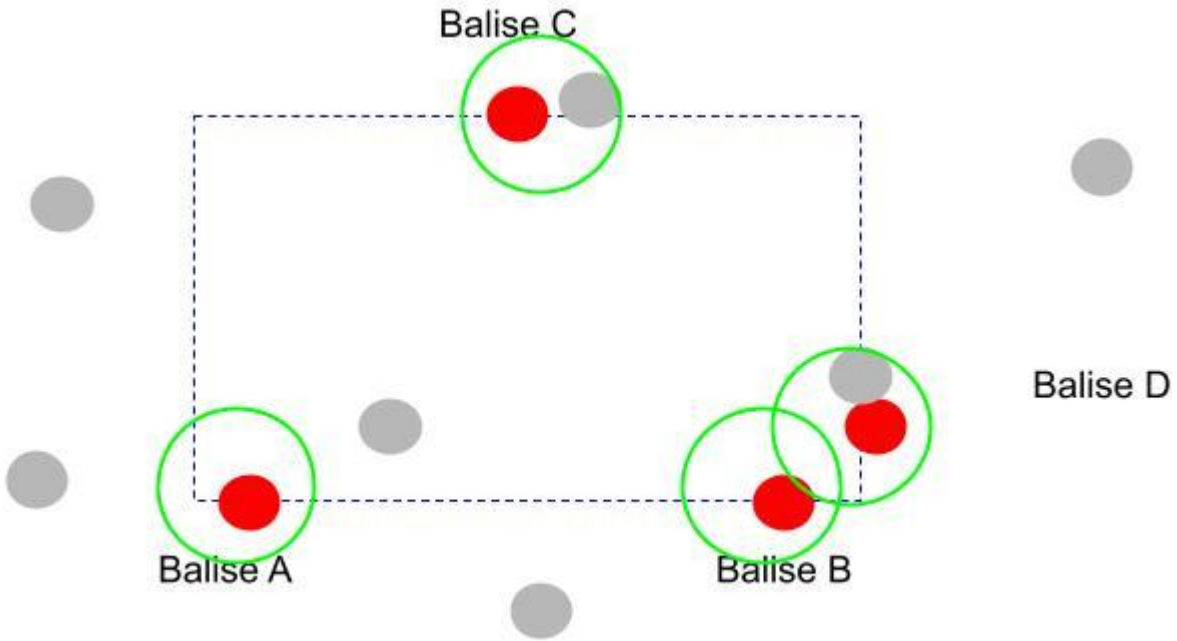




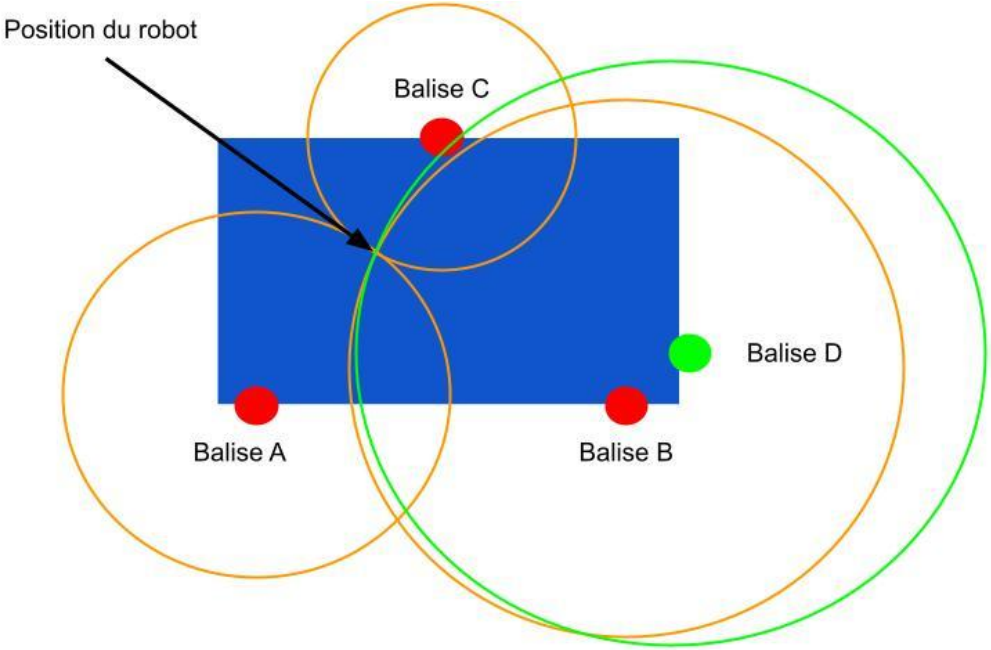
Trouver la localisation des balises



Trouver la localisation des balises

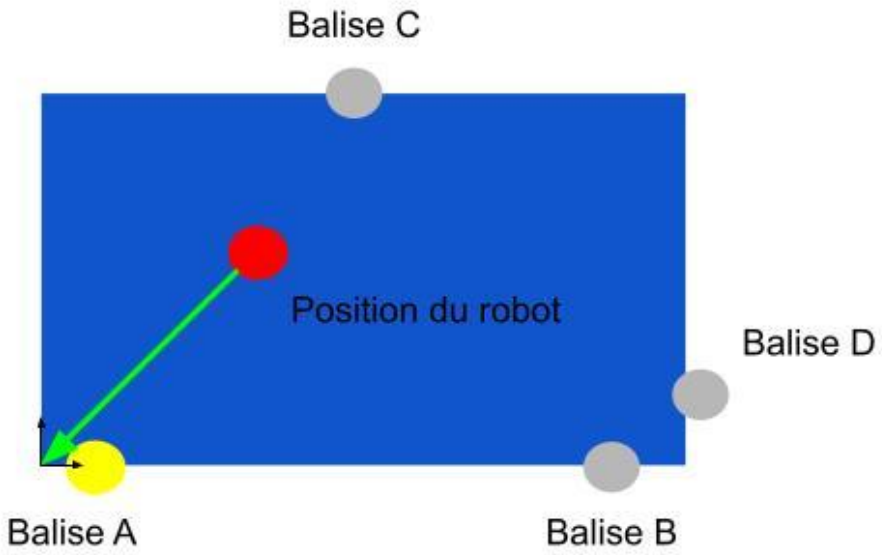


Calcul de la position par triangulation



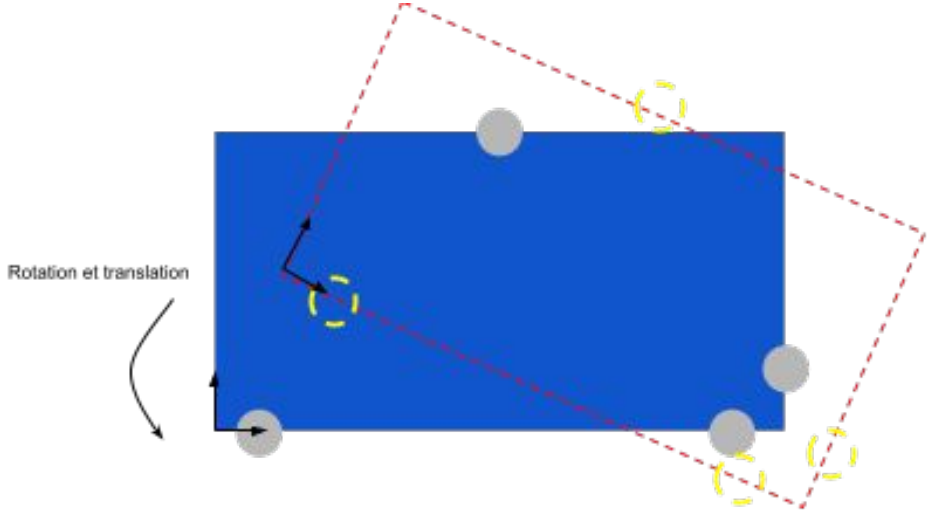
$$d_A^2 = (x - x_A)^2 + (y - y_A)^2$$
$$d_B^2 = (x - x_B)^2 + (y - y_B)^2$$
$$d_C^2 = (x - x_C)^2 + (y - y_C)^2$$

Calcul de la position par triangulation



$$x_R = (x_P - \tilde{x})\, cos(\theta) + (y_P - \tilde{y})\, sin(\theta)$$
$$y_R = -(x_P - \tilde{x})\, sin(\theta) + (y_P - \tilde{y})\, cos(\theta)$$

Calcul de la position par méthode équation normale



$$x_{PA} = \tilde{x} + x_{RA}\,cos(\theta) - y_{RA}\,sin(\theta)$$
$$y_{PA} = \tilde{y} + y_{RA}\,cos(\theta) + x_{RA}\,sin(\theta)$$
$$x_{PB} = \tilde{x} + x_{RB}\,cos(\theta) - y_{RB}\,sin(\theta)$$
$$y_{PB} = \tilde{y} + y_{RB}\,cos(\theta) + x_{RB}\,sin(\theta)$$
$$x_{PC} = \tilde{x} + x_{RC}\,cos(\theta) - y_{RC}\,sin(\theta)$$
$$y_{PC} = \tilde{y} + y_{RC}\,cos(\theta) + x_{RC}\,sin(\theta)$$

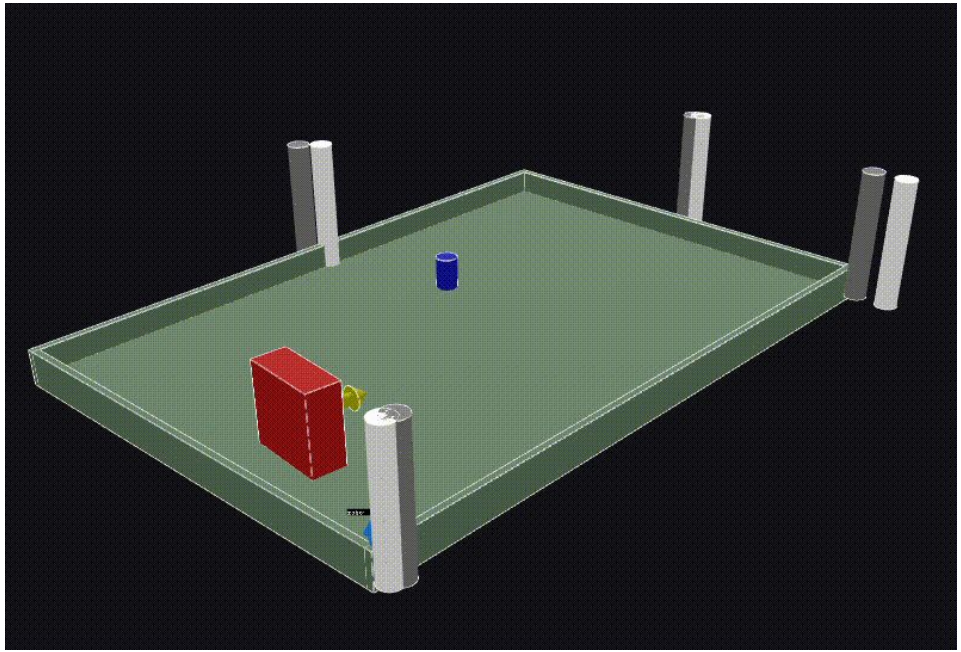
Calcul de la position par méthode équation normale

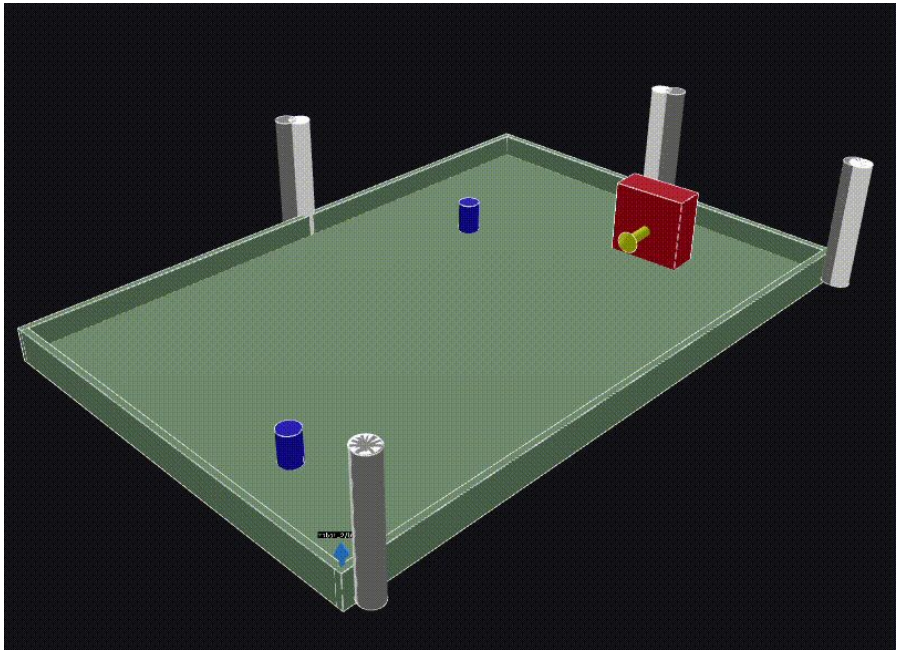
$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \\ cos(\theta) \\ sin(\theta) \end{pmatrix}$$

$$\begin{aligned} & AX = b \\ \Rightarrow \quad & A^T AX = A^T b \\ \Rightarrow \quad & X = (A^T A)^{-1} A^T b \end{aligned}$$

$$\mathrm{A} = \begin{pmatrix} 1 & 0 & x_{RA} & -y_{RA} \\ 0 & 1 & y_{RA} & x_{RA} \\ 1 & 0 & x_{RB} & -y_{RB} \\ 1 & 0 & y_{RB} & x_{RB} \\ 1 & 0 & x_{RC} & -y_{RC} \\ 1 & 0 & y_{RC} & x_{RC} \end{pmatrix} \qquad \mathrm{b} = \begin{pmatrix} x_{RA} \\ y_{RA} \\ x_{RB} \\ y_{RB} \\ x_{RC} \\ y_{RC} \end{pmatrix}$$

Rendu final





Fusion des capteurs et Estimation de la position des Ennemis

Objectifs du Système

1. Fiabiliser l’estimation de la position du robots à partir des données de positionnement Lidar et Odométrique

   Repose sur l’utilisation d’un filtre de Kalman ayant pour but d’évaluer le bias présent entre l’Odométrie et le Lidar et par conséquent d’effectuer une correction

2. Permettre le suivi d’obstacles (robots ennemis) sur le plateau de jeu en fonction des informations collectées par les deux robots

   Utilisation d’un filtre de Kalman afin de faire du suivi d’obstacle en utilisant une information à priori sur la position de ces obstacles

C'est quoi un filtre de Kalman ?

$$\begin{cases} \mathbf{x}_k &= \mathbf{F}\mathbf{x}_{k-1} + \mathbf{B}_k\mathbf{u}_k + \mathbf{w}_k \\ \mathbf{y}_k &= \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k \end{cases}$$

$\mathbf{x}_k$: Vecteur d'etat

$\mathbf{F}_k$: Matrice de Transition

$\mathbf{B}_k$: Matrice de Controle de la Commande

$\mathbf{u}_k$: Vecteur de Commande

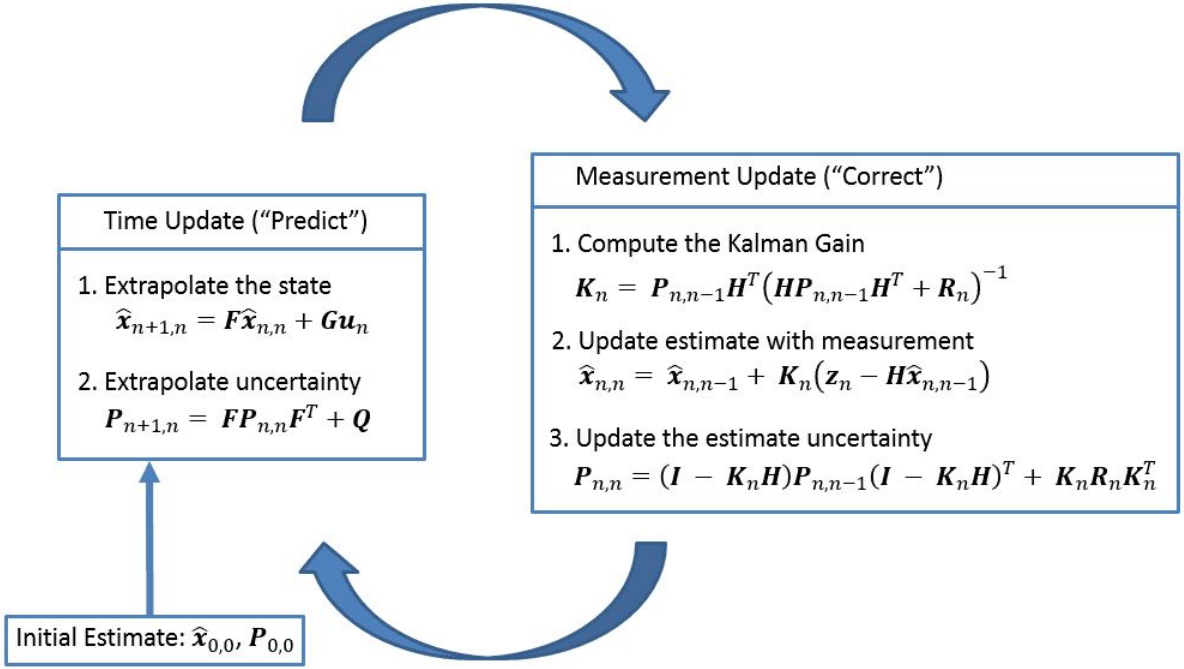
$\mathbf{w}_k$: Vecteur de variable aleatoire Gaussienne de matrice de Covariance $\mathbf{Q}_k$

$\mathbf{y}_k$: Vecteur de mesure

$\mathbf{H}_k$: Matrice d'observation

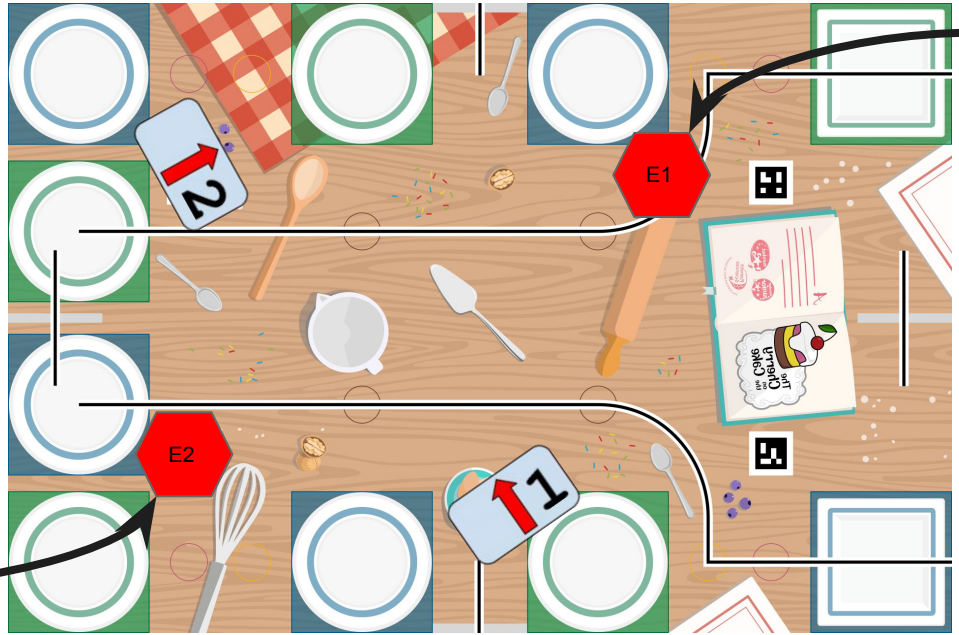
$\mathbf{v}_k$: Vecteur de variable aleatoire Gaussienne de matrice de Covariance $\mathbf{R}_k$

C’est quoi un filtre de Kalman ?



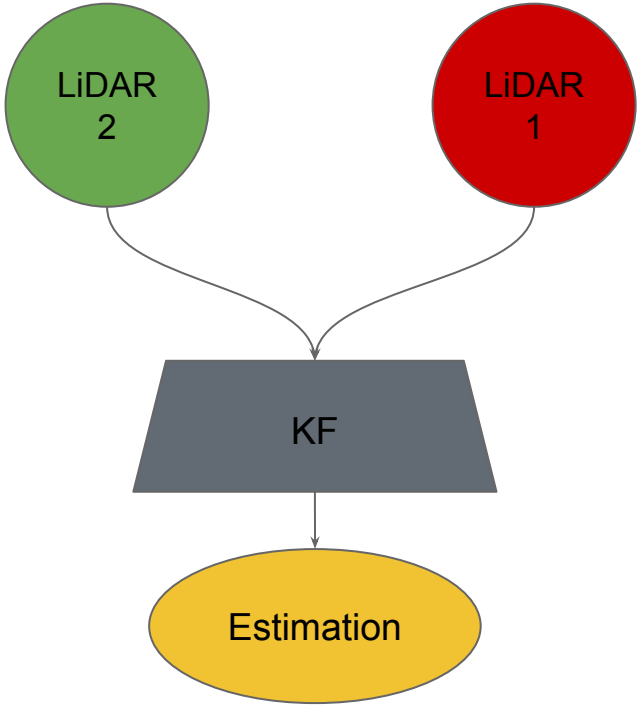
Explicitation du problème

1. Suivi des obstacles



Explicitation du problème

1. Suivi des obstacles



<u>Vecteur d'état :</u> $\mathbf{x} = \begin{bmatrix} x & \dot{x} & y & \dot{y} \end{bmatrix}^{\mathsf{T}}$

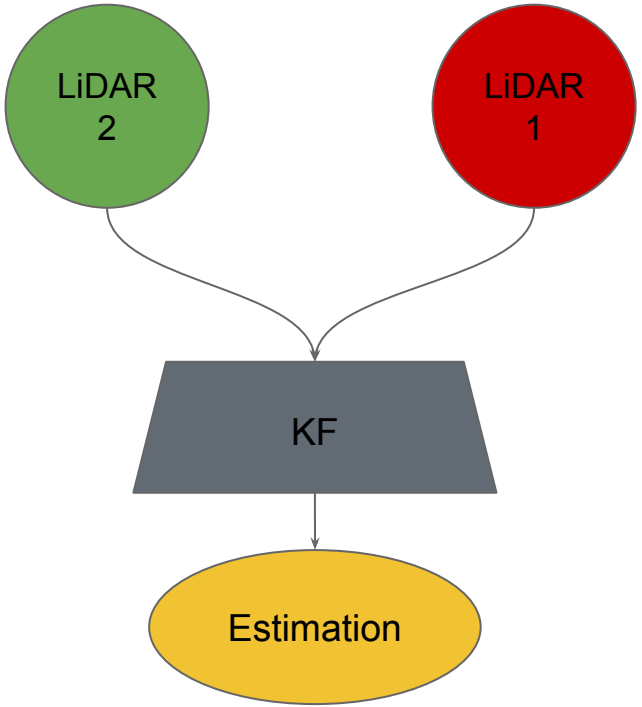
<u>Loi à priori :</u>

$$\begin{aligned} x &= 1x + \Delta t\dot{x} + 0y + 0\dot{y} \\ v_x &= 0x + 1\dot{x} + 0y + 0\dot{y} \\ y &= 0x + 0\dot{x} + 1y + \Delta t\dot{y} \\ v_y &= 0x + 0\dot{x} + 0y + 1\dot{y} \end{aligned} \Rightarrow \mathbf{F} = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}) \Rightarrow \mathbf{Q} = \begin{bmatrix} \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} & 0 & 0 \\ \frac{\Delta t^3}{2} & \Delta t^2 & 0 & 0 \\ 0 & 0 & \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} \\ 0 & 0 & \frac{\Delta t^3}{2} & \Delta t^2 \end{bmatrix} \sigma_{pos}^2$$

Explicitation du problème

1. Suivi des obstacles



<u>Vecteur de Mesure :</u> $\mathbf{y} = \begin{bmatrix} x_{lidar1} & x_{lidar2} & y_{lidar1} & y_{lidar2} \end{bmatrix}^{\mathsf{T}}$

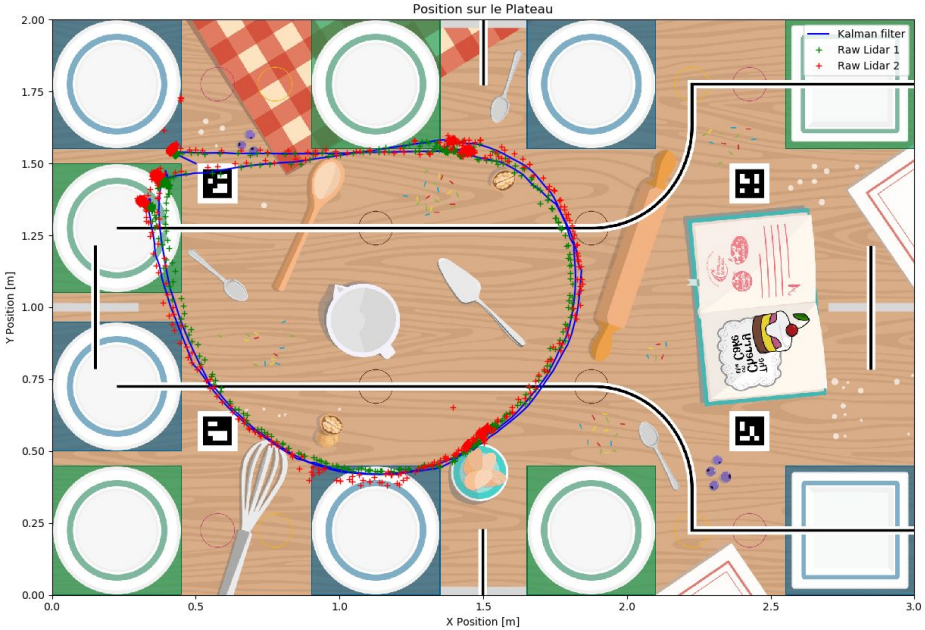
$$\mathbf{H}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

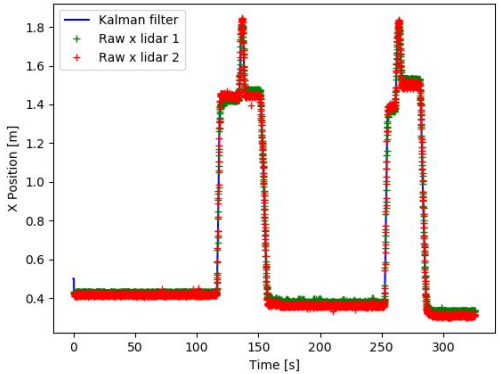
Fonction de la disponibilité des mesures

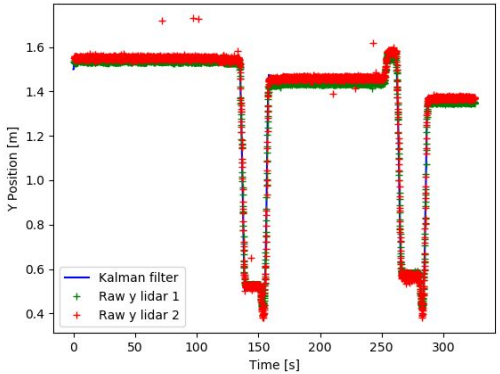
$$\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_{(k)}) \quad \Rightarrow \quad \mathbf{R}_{(k)} = \begin{bmatrix} \sigma^2_{lidar1} & 0 & 0 & 0 \\ 0 & \sigma^2_{lidar2} & 0 & 0 \\ 0 & 0 & \sigma^2_{lidar1} & 0 \\ 0 & 0 & 0 & \sigma^2_{lidar2} \end{bmatrix}$$

Résultats

1. Suivi des obstacles

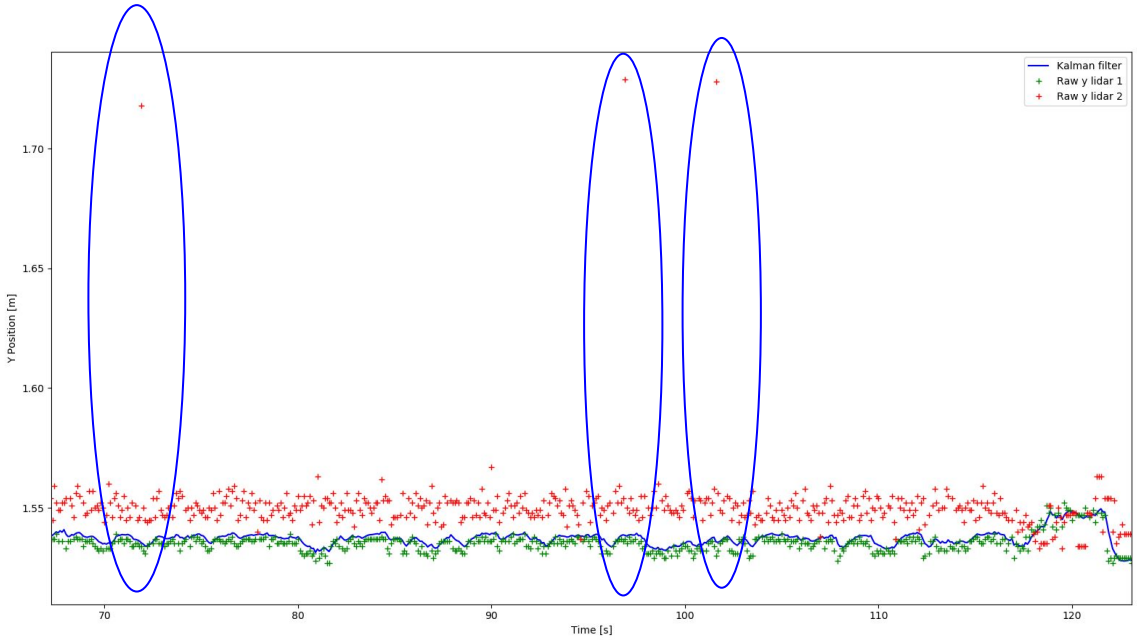






Résultats

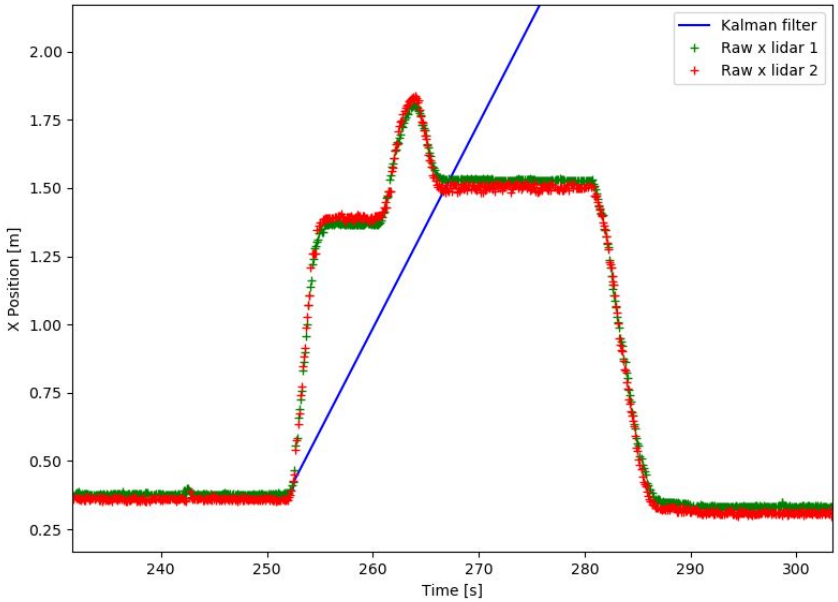
1. Suivi des obstacles



La stabilité n’est plus garantie

Résultats

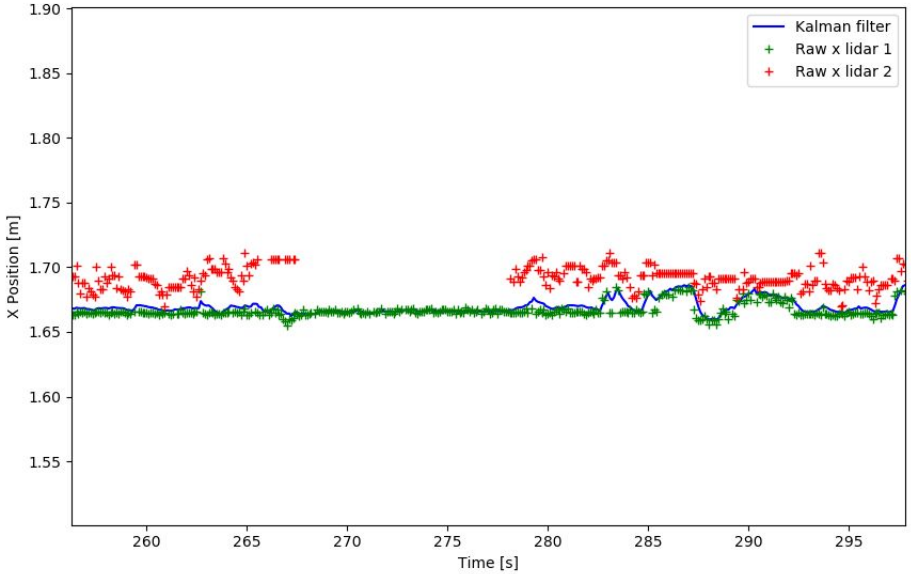
1. Suivi des obstacles



La stabilité n’est plus garantie

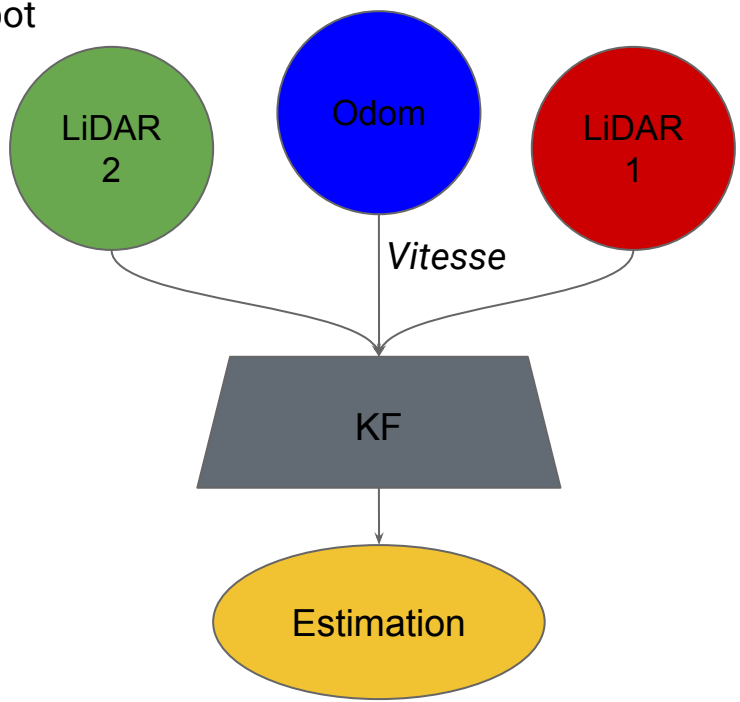
Résultats

1. Suivi des obstacles



Autre solution étudiée avec vitesse odométrique (1)

2. Fiabilisation de l'estimation de la position du robot



<u>Vecteur d'état :</u> $\mathbf{x} = \begin{bmatrix} x & \dot{x} & y & \dot{y} \end{bmatrix}^{\mathsf{T}}$

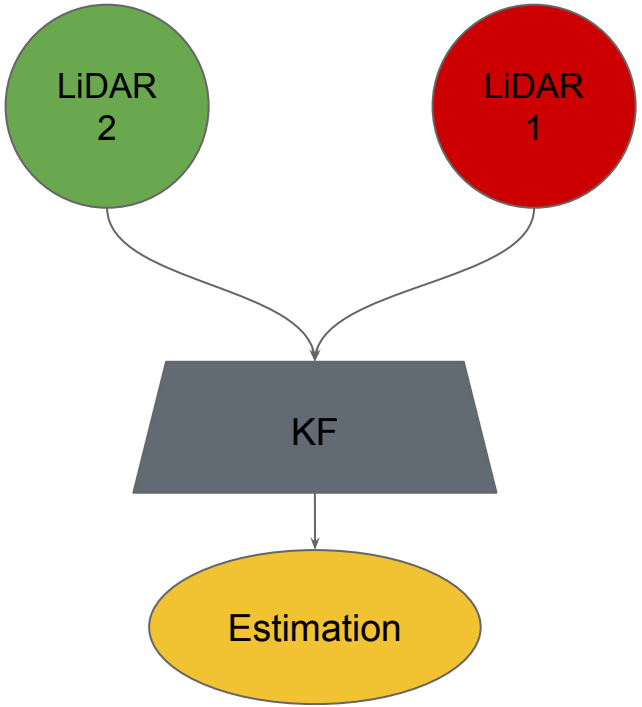
<u>Loi à priori :</u>

$$\begin{aligned} x &= 1x + \Delta t\dot{x} + 0y + 0\dot{y} \\ v_x &= 0x + 1\dot{x} + 0y + 0\dot{y} \\ y &= 0x + 0\dot{x} + 1y + \Delta t\dot{y} \\ v_y &= 0x + 0\dot{x} + 0y + 1\dot{y} \end{aligned} \Rightarrow \mathbf{F} = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

<u>Mais</u> les vitesses sont les vitesses mesurées par l'odométrie au temps t en non pas la vitesse calculée au temps précédent

$$\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}) \Rightarrow \mathbf{Q} = \begin{bmatrix} \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} & 0 & 0 \\ \frac{\Delta t^3}{2} & \Delta t^2 & 0 & 0 \\ 0 & 0 & \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} \\ 0 & 0 & \frac{\Delta t^3}{2} & \Delta t^2 \end{bmatrix} \sigma_{pos}^2$$

Explicitation du problème



<u>Vecteur de Mesure :</u> $\mathbf{y} = \begin{bmatrix} x_{lidar1} & x_{lidar2} & y_{lidar1} & y_{lidar2} \end{bmatrix}^{\mathsf{T}}$

$$\mathbf{H}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

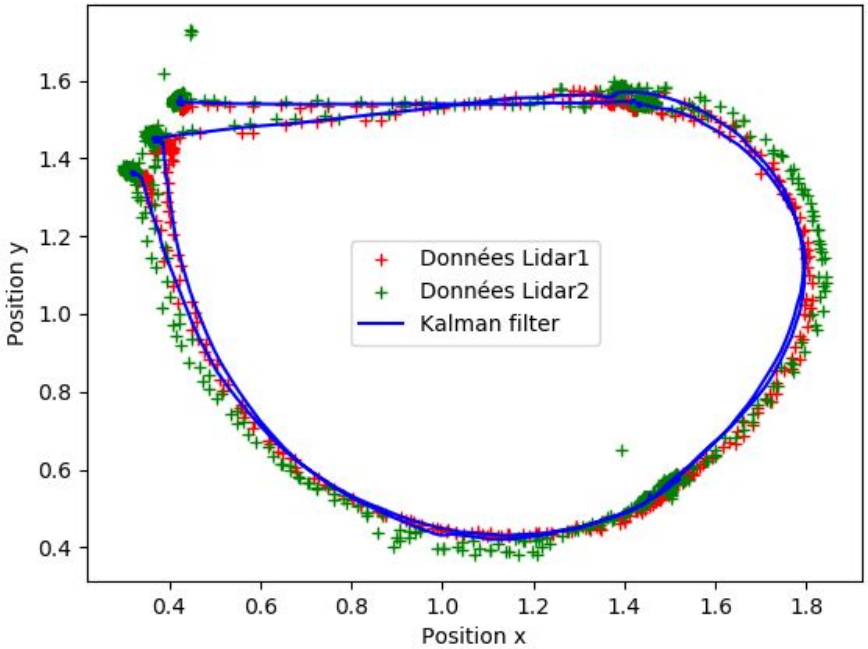
Fonction de la disponibilité des mesures

<u>Uniquement</u> les données lidar dans l'étape de récupération des mesures

$$\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_{(k)}) \quad \Rightarrow \quad \mathbf{R}_{(k)} = \begin{bmatrix} \sigma^2_{lidar1} & 0 & 0 & 0 \\ 0 & \sigma^2_{lidar2} & 0 & 0 \\ 0 & 0 & \sigma^2_{lidar1} & 0 \\ 0 & 0 & 0 & \sigma^2_{lidar2} \end{bmatrix}$$

Résultats

2. Fiabilisation de l'estimation de la position du robot

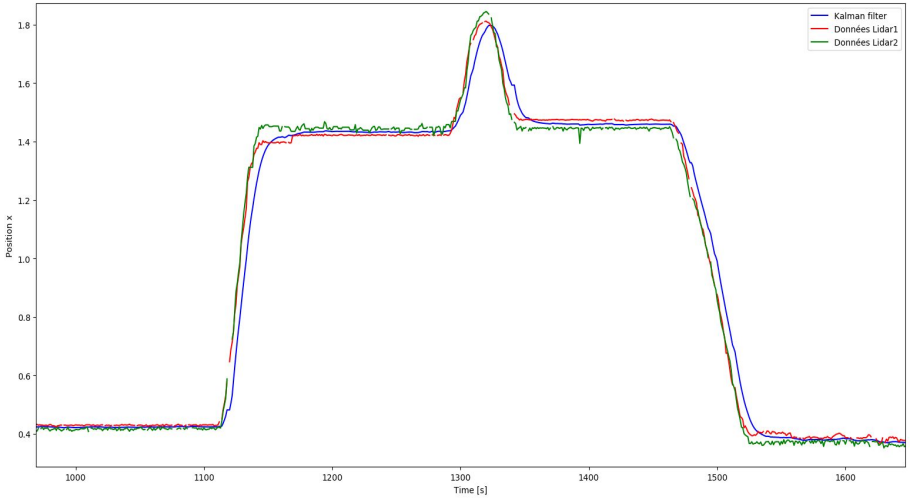


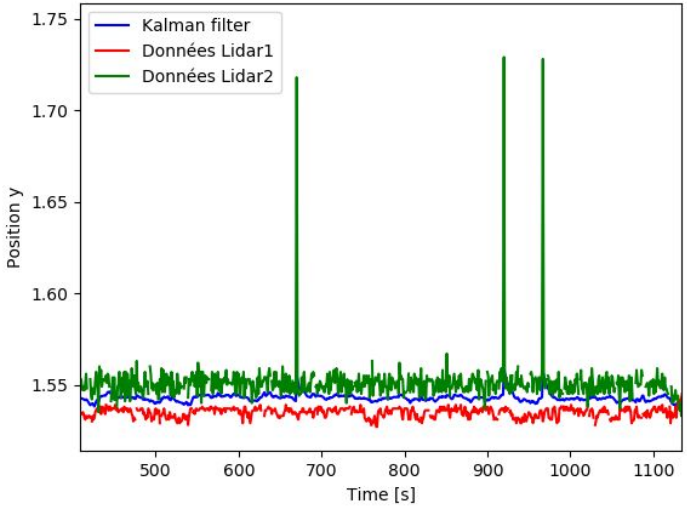
Bonne précision et correction des imperfections

Sur une échelle x-y on ne voit pas le retard crée, bien regarder toutes les représentations

Résultats

2. Fiabilisation de l’estimation de la position du robot

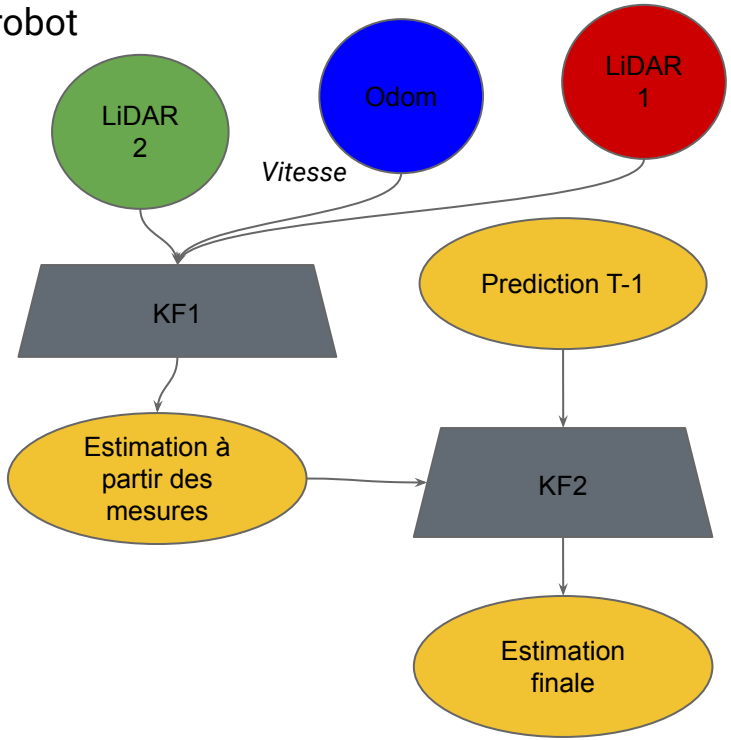




Mais léger retard et léger bruit!

Autre solution étudiée avec vitesse odométrique (2)

2. Fiabilisation de l'estimation de la position du robot



<u>Vecteur d'état :</u> $\mathbf{x} = \begin{bmatrix} x & \dot{x} & y & \dot{y} \end{bmatrix}^{\mathsf{T}}$

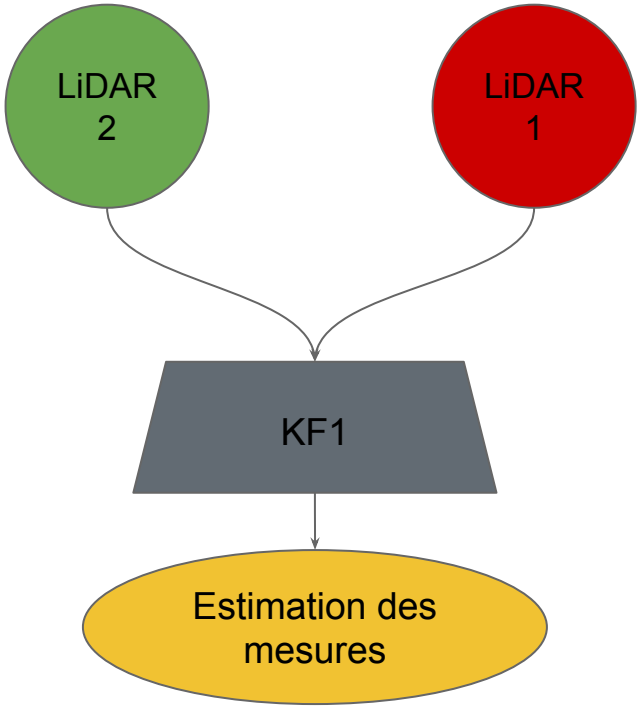
<u>Loi à priori :</u>

$$\begin{aligned} x &= 1x + \Delta t\dot{x} + 0y + 0\dot{y} \\ v_x &= 0x + 1\dot{x} + 0y + 0\dot{y} \\ y &= 0x + 0\dot{x} + 1y + \Delta t\dot{y} \\ v_y &= 0x + 0\dot{x} + 0y + 1\dot{y} \end{aligned} \Rightarrow \mathbf{F} = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

On réalise 2 fois le calcul du filtre, un avec la vitesse odométrique réelle, un autre avec la prédiction

$$\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}) \Rightarrow \mathbf{Q} = \begin{bmatrix} \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} & 0 & 0 \\ \frac{\Delta t^3}{2} & \Delta t^2 & 0 & 0 \\ 0 & 0 & \frac{\Delta t^4}{4} & \frac{\Delta t^3}{2} \\ 0 & 0 & \frac{\Delta t^3}{2} & \Delta t^2 \end{bmatrix} \sigma_{pos}^2$$

Explicitation du problème (relevé de mesures)



<u>Vecteur de Mesure :</u> $\mathbf{y} = \begin{bmatrix} x_{lidar1} & x_{lidar2} & y_{lidar1} & y_{lidar2} \end{bmatrix}^{\mathsf{T}}$

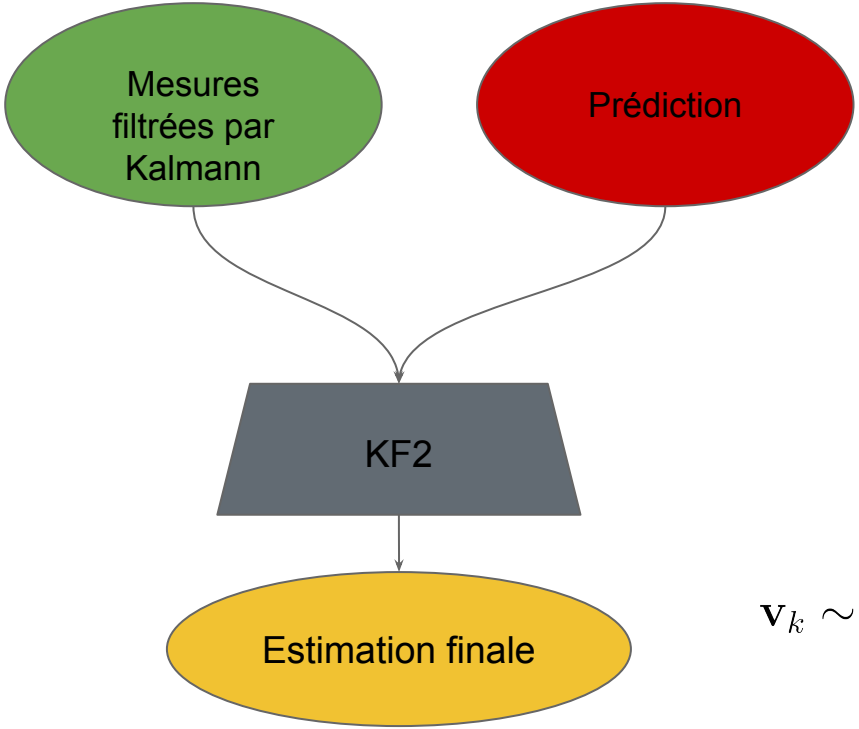
$$\mathbf{H}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Fonction de la disponibilité des mesures

<u>Uniquement</u> les données lidar dans l'étape de récupération des mesures, les vitesses odométriques sont dans l'équation d'état

$$\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_{(k)}) \quad \Rightarrow \quad \mathbf{R}_{(k)} = \begin{bmatrix} \sigma^2_{lidar1} & 0 & 0 & 0 \\ 0 & \sigma^2_{lidar2} & 0 & 0 \\ 0 & 0 & \sigma^2_{lidar1} & 0 \\ 0 & 0 & 0 & \sigma^2_{lidar2} \end{bmatrix}$$

Explicitation du problème (relevé de mesures)



<u>Vecteur de Mesure passé dans Kalmann :</u>

$$\mathbf{y} = \begin{bmatrix} x_{filtre} & \dot{x}_{filtre} & y_{filtre} & \dot{y}_{filtre} \end{bmatrix}^{\mathsf{T}}$$

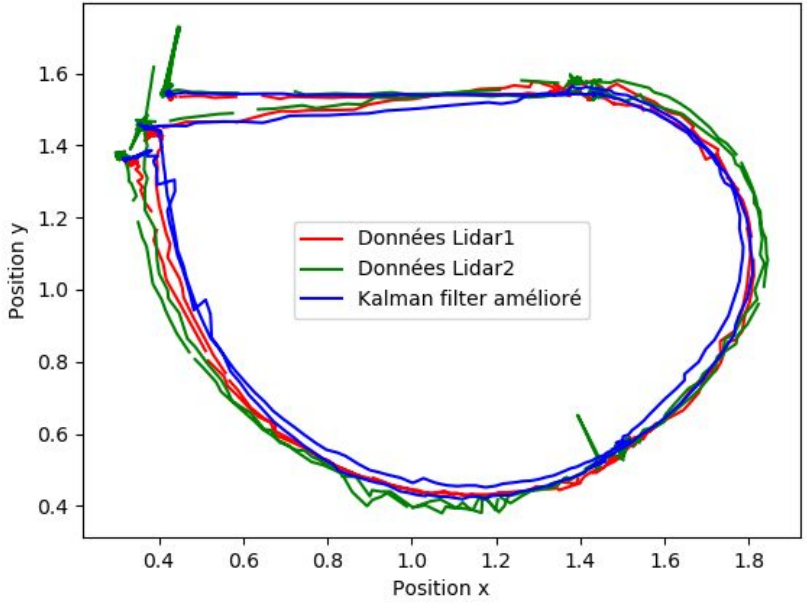
$$\mathbf{H}_k = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Fonction de la disponibilité des mesures une fois filtrée

$$\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_{(k)}) \quad \Rightarrow \quad \mathbf{R}_{(k)} = \begin{bmatrix} \sigma_x^2 & 0 & 0 & 0 \\ 0 & \sigma_{\dot{x}}^2 & 0 & 0 \\ 0 & 0 & \sigma_y^2 & 0 \\ 0 & 0 & 0 & \sigma_{\dot{y}}^2 \end{bmatrix}$$

Résultats

2. Fiabilisation de l’estimation de la position du robot

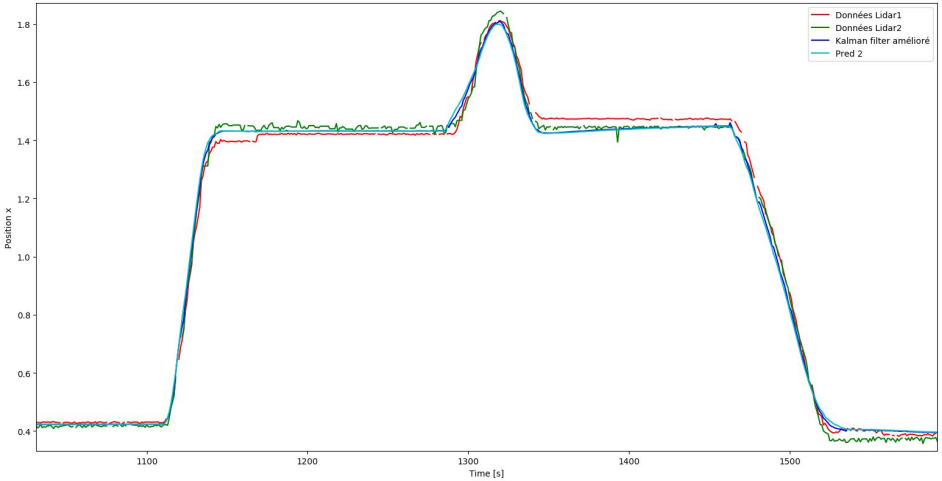


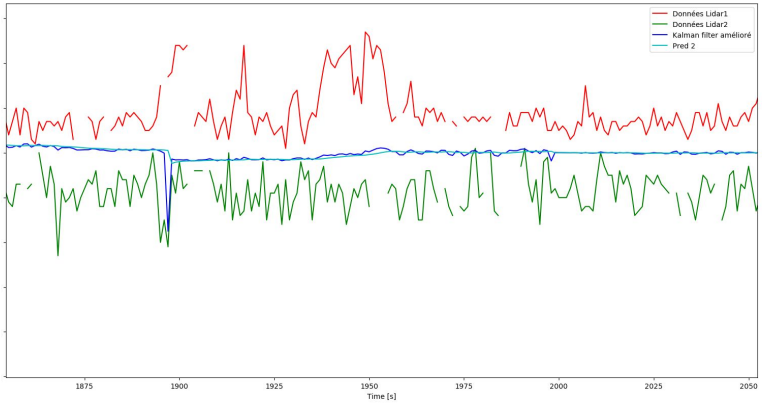
Moins bonne précision à première vue que précédemment

Toujours précis aux positions fixe, réduction du bruit

Résultats

2. Fiabilisation de l’estimation de la position du robot





Correction du retard et réduction du bruit comparé aux précédentes méthodes!

Noeud de Navigation

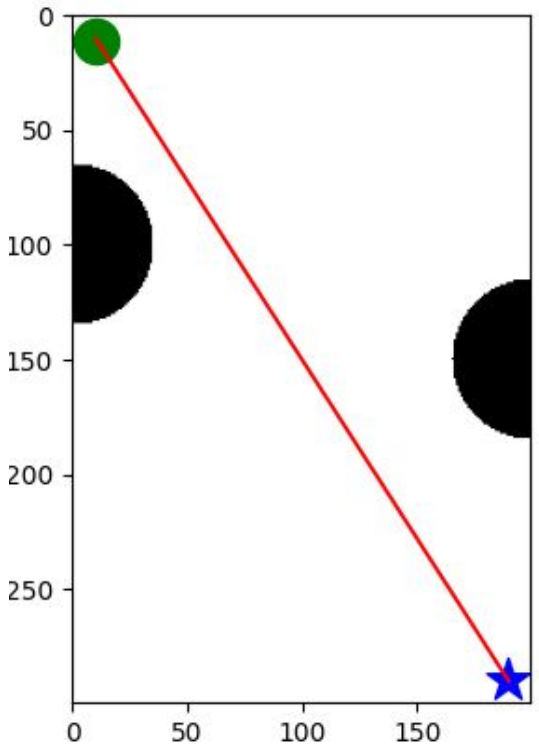
Noeud de navigation

Objectif: -Aller d'un point A à un point B

-Éviter les obstacles

-Adapter le chemin en permanence

-Sortir d'un obstacle si celui-ci nous percute



Carte : 300×200

- Point vert : robot
- Tâches noires : obstacles
- Étoile bleue : objectif

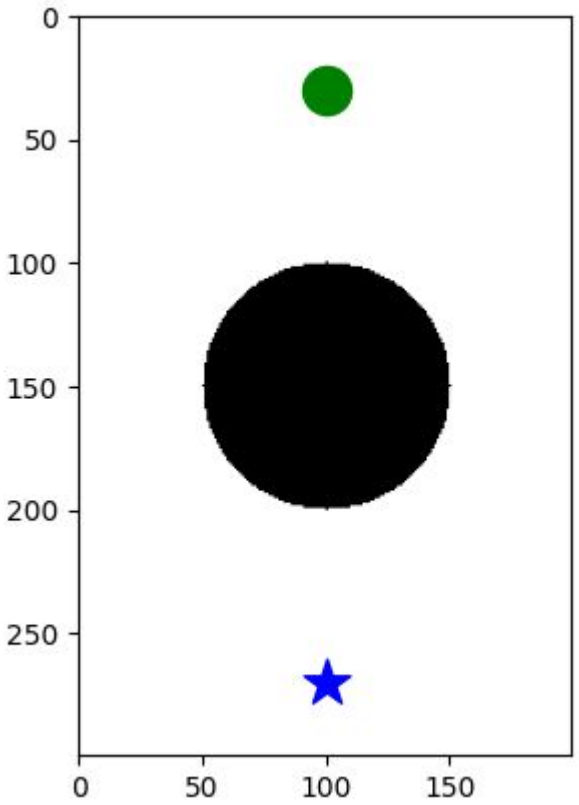
Représentation des données

Cartes des obstacles : Matrice booléenne 300×200.

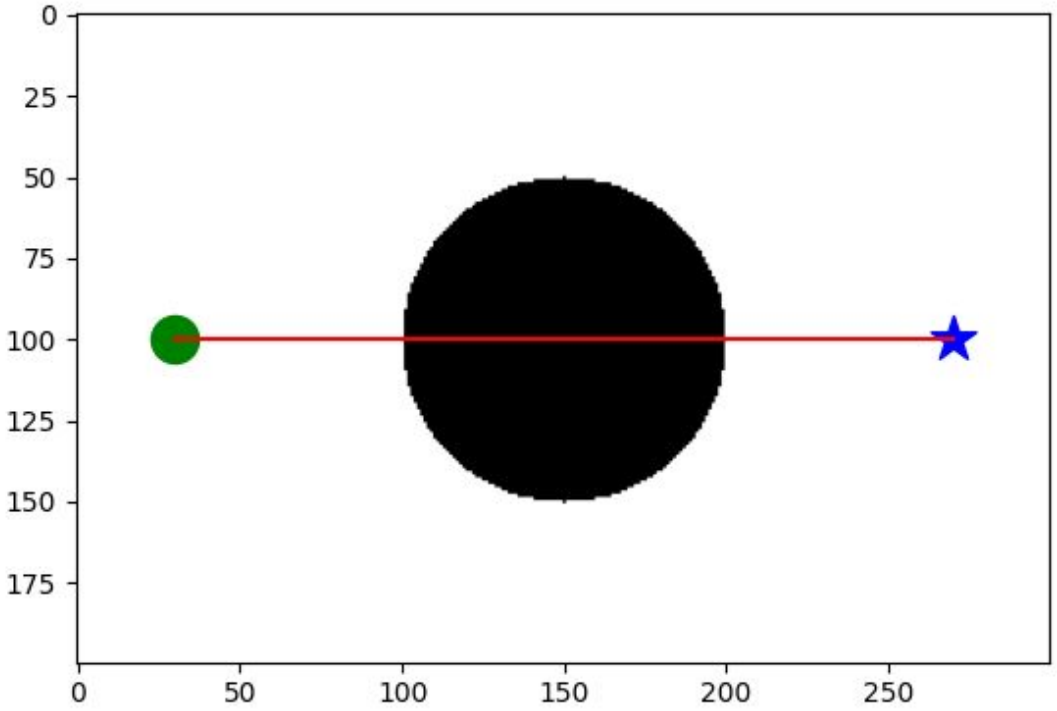
Chemin : Liste de point à rejoindre jusqu'à l'objectif

Point : 2-uplet de la forme (x,y)

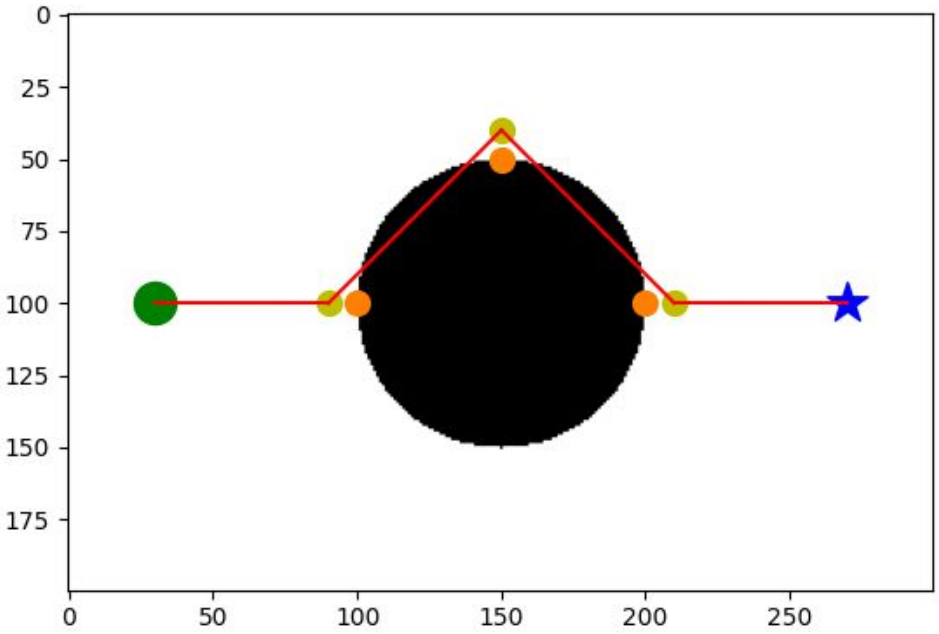
Résolution du problème



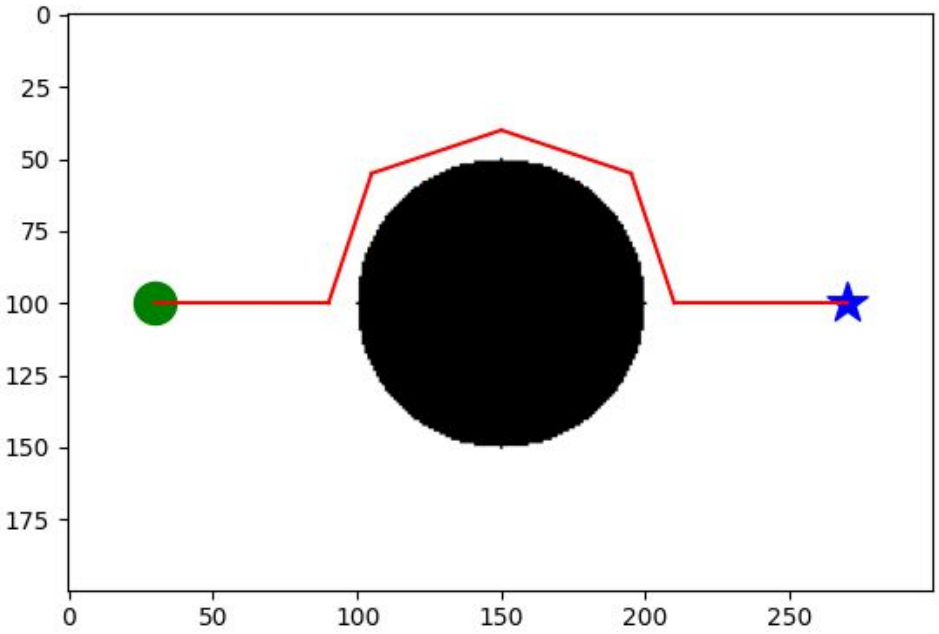
Résolution naïve



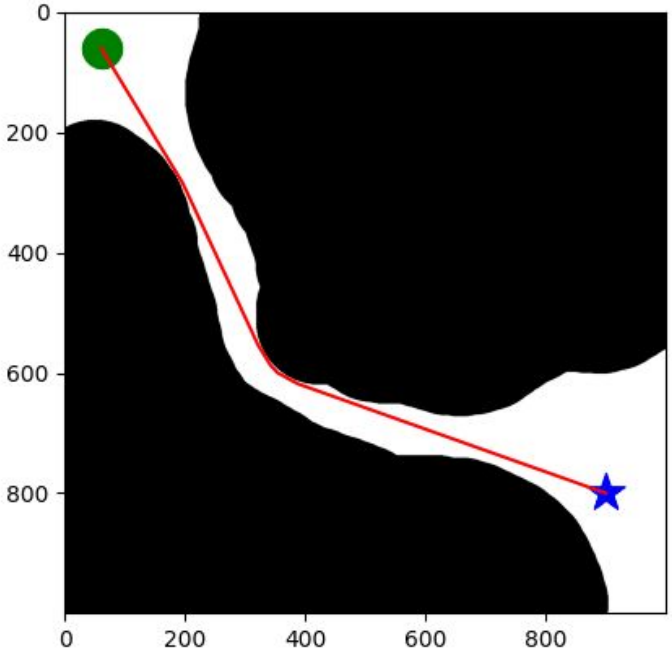
Résolution par segment



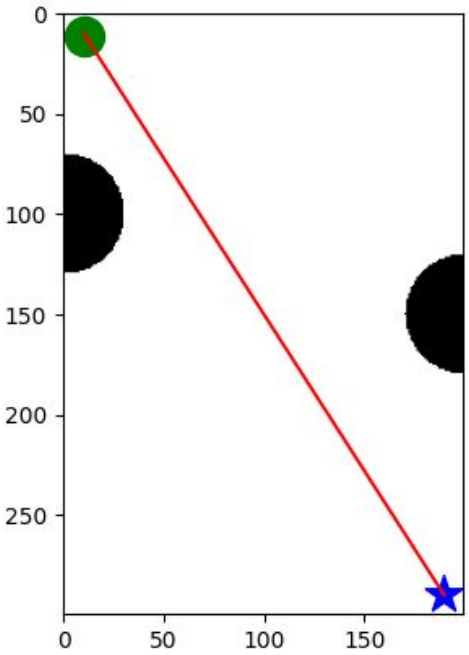
Résolution récursive



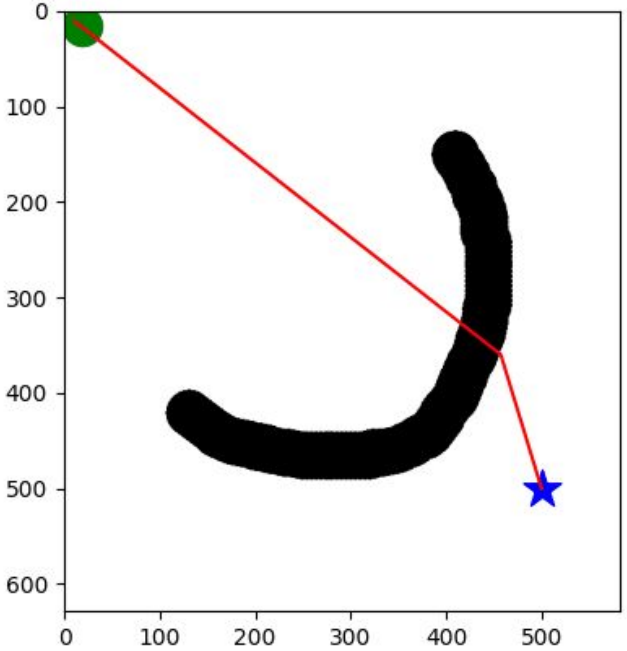
Application



Application dynamique



Cas d’erreur



Avantages et inconvénients

Avantages

- Rapide
- Optimal dans le cas d'obstacles multiples circulaires
- "Fait main"

Inconvénient

- Ne converge pas dans certains cas

Voies d’améliorations

- Écrire un algorithme pour modifier partiellement le chemin
- Diminuer le nombre de cas d’erreur
- Améliorer la trajectoire finale à partir des points donnés