



CARTOGRAPHIE ET LOCALISATION SIMULTANÉES FILTRAGE DE KALMAN

GABRIEL DUPUIS - ALIONA LEJUSTE

SOMMAIRE

Introduction

Problématique

I. Contextualisation

A. Principe du filtre de Kalman

B. Méthodes mathématiques

II. Application dans un cas simplifié

A. Réalisation du filtre de Kalman

B. Résultats

III. Application dans un cas complexe

A. Principe et initiatives

B. Résultats

Conclusion



Problématique

Comment décrire et corriger la trajectoire d'un drone dans un environnement donné à l'aide du filtre de Kalman ?

I. Contextualisation

A. Principe du SLAM et filtrage de Kalman

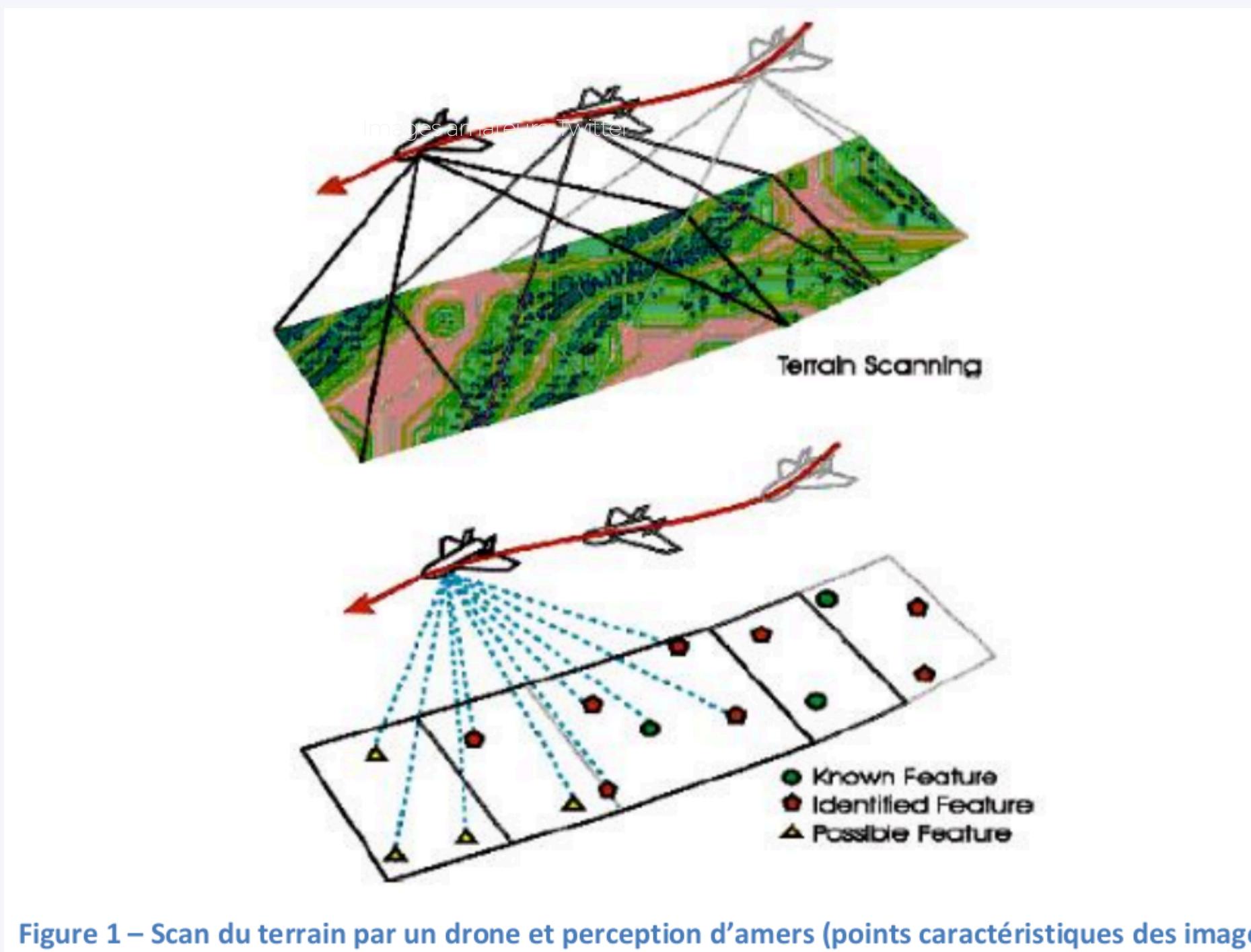


Figure 1 – Scan du terrain par un drone et perception d'amers (points caractéristiques des images)

Amer: point de repère de l'environnement

Odométrie: estimation bruitée déterminée par la centrale inertie de la distance parcourue durant un pas de temps

SLAM: Simultaneous Localization And Mapping, problème d'estimation de la position du drone et des amers à partir de l'odométrie et des perceptions

I. Contextualisation

A. Principe du SLAM et filtrage de Kalman

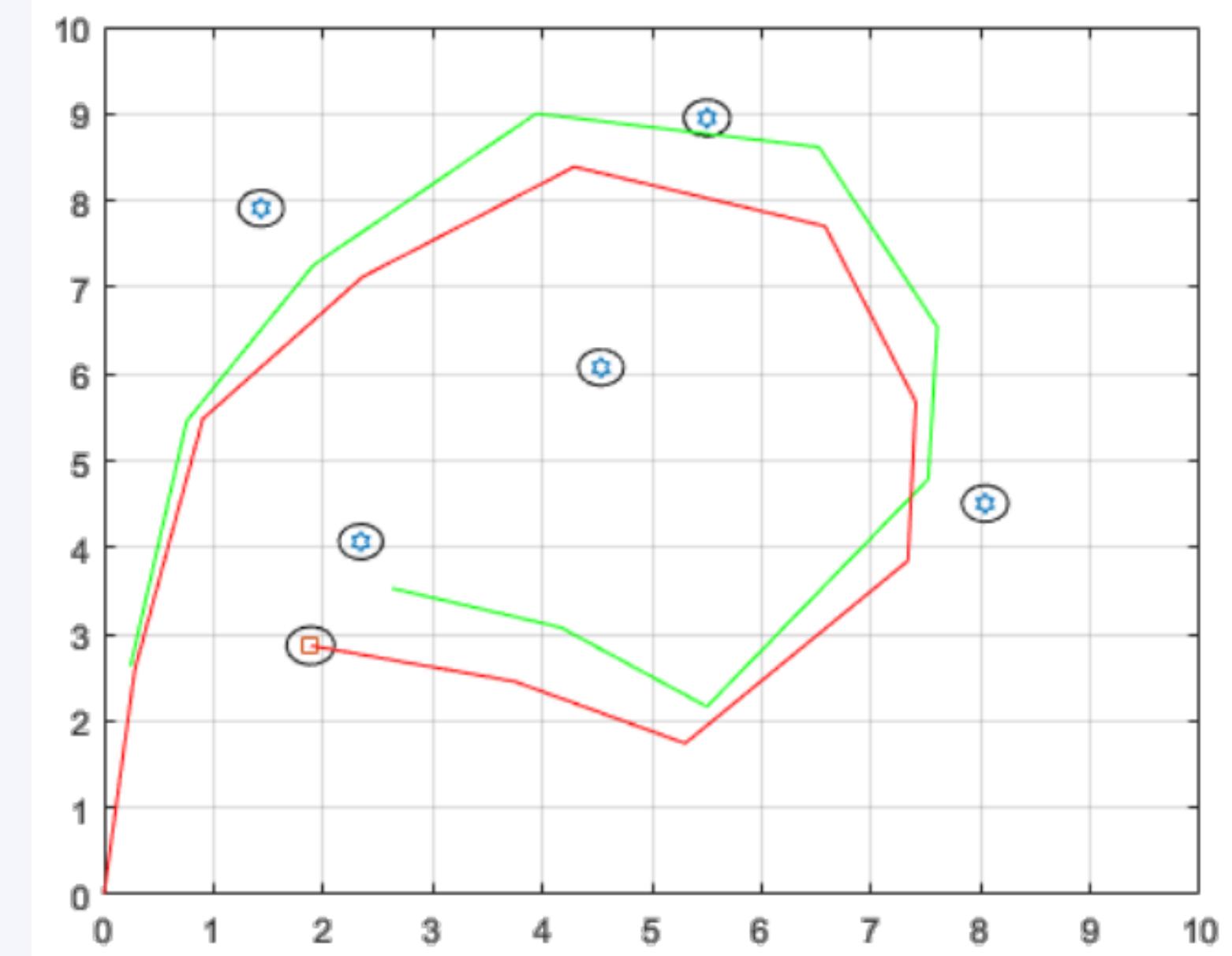
1. INITIALISATION DES DONNÉES x_0, p_0 SELON LES HYPOTHÈSES
2. POUR CHAQUE PAS DE TEMPS:
 - A. LIRE **LES PERCEPTIONS** y ET **L'ODOMÉTRIE** u
 - B. PRÉDIRE L'ÉTAT x^* DEPUIS LE PRÉCÉDENT
 - C. PRÉDIRE LA MATRICE DE COVARIANCE p^*
 - D. PRÉDIRE LES PERCEPTIONS y^* DEPUIS L'ÉTAT PRÉDIT
 - E. CORRIGER L'ÉTAT SELON LES DIFFÉRENCES DE PERCEPTIONS
 - F. CORRIGER LA MATRICE DE COVARIANCE
3. AFFICHER LA TRAJECTOIRE

I. Contextualisation

A. Principe du SLAM et filtrage de Kalman

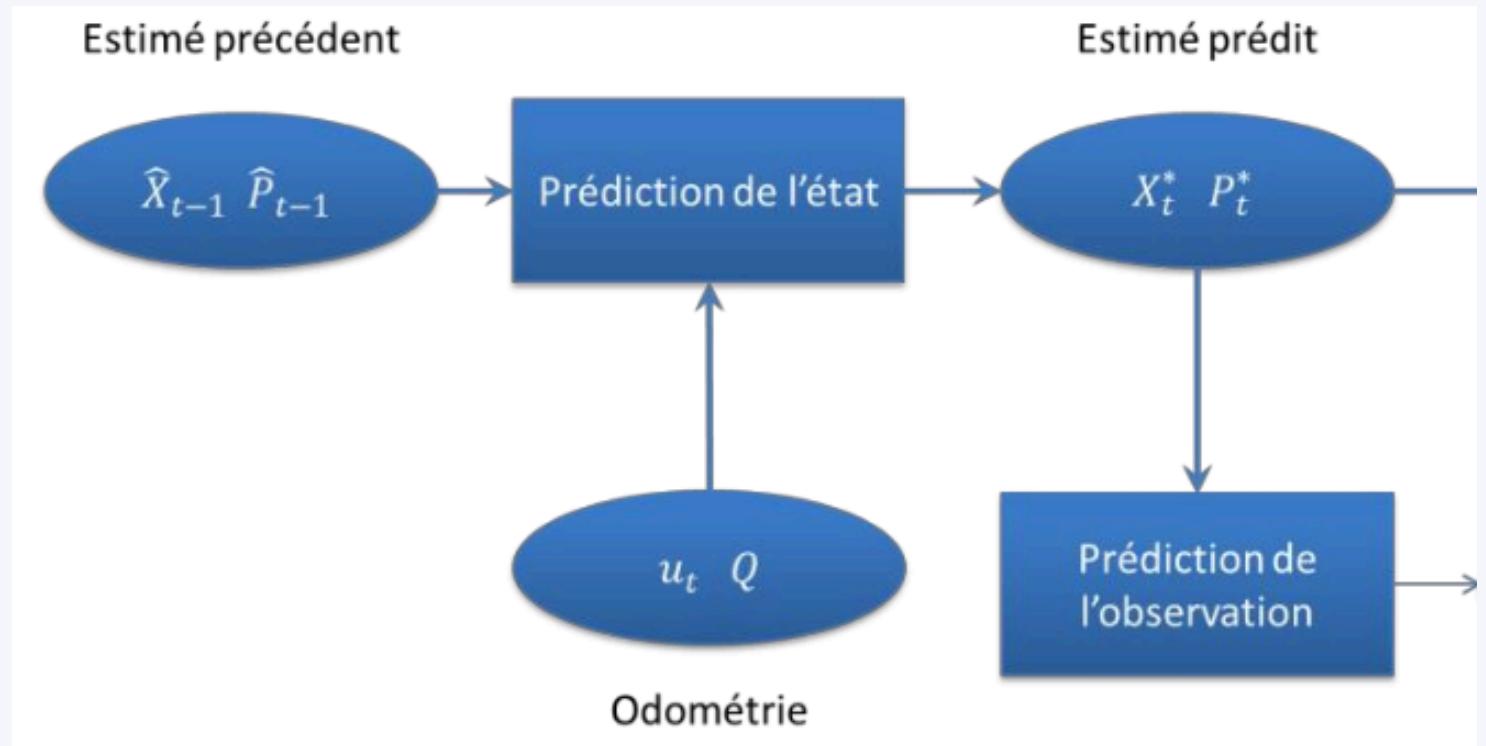
$$X_t = \begin{bmatrix} x \\ y \\ x_1 \\ y_1 \\ \vdots \\ x_N \\ y_N \end{bmatrix}$$

$$Y_t = \begin{bmatrix} x_1 - x \\ y_1 - y \\ \vdots \\ x_N - x \\ y_N - y \end{bmatrix}$$



I. Contextualisation

B. Méthode mathématiques: PREDICTION



X_t Vecteur de positions absolues contenant la position du drone et de chacun des amers à l'instant t.

X_t^* Vecteur de positions prédites non corrigées à l'instant t.

P_t Matrice de covariance absolue donnant l'incertitude sur les positions de X_t .

P_t^* Matrice de covariance prédictive non corrigée donnant l'incertitude sur les positions de X_t^* .

1.
$$X_t^* = \hat{X}_{t-1} + Bu_t$$

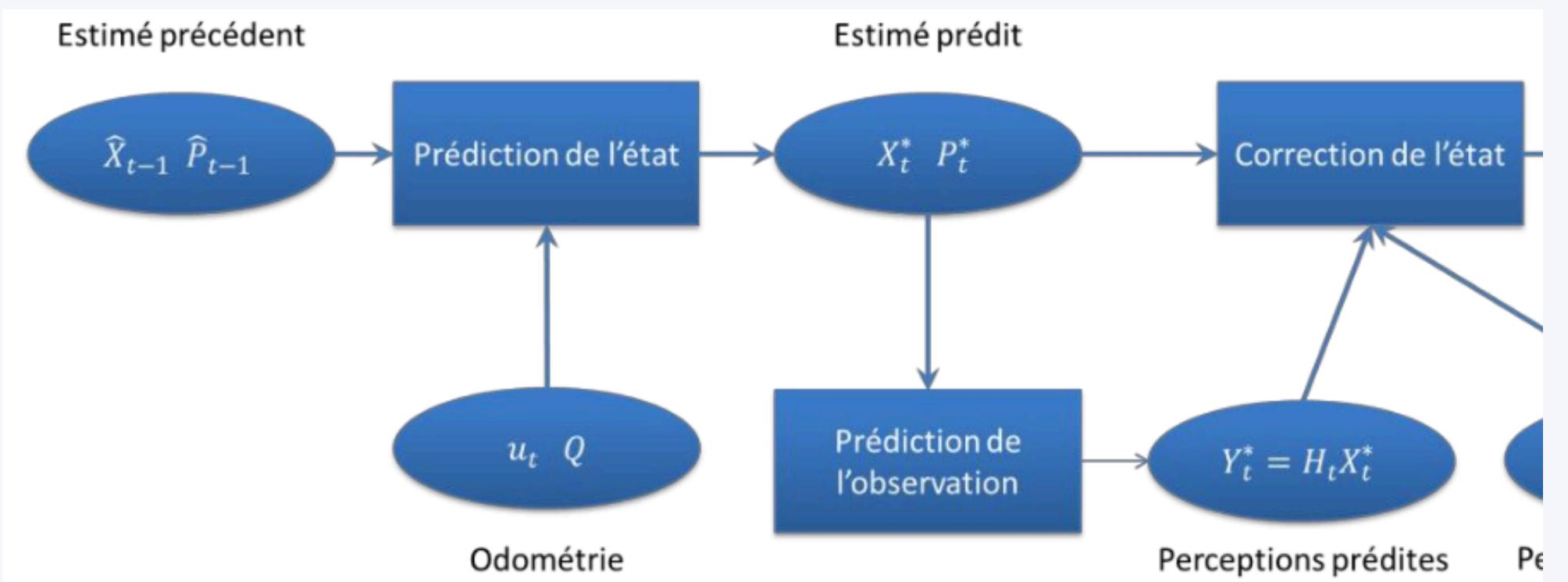
2.
$$P_t^* = \hat{P}_{t-1} + BQ B^T$$

u_t Vecteur d'odométrie à l'instant t.

Q Matrice d'incertitude sur l'odométrie u_t .

I. Contextualisation

B. Méthode mathématiques: PREDICTION



$$1. \quad X_t^* = \hat{X}_{t-1} + Bu_t$$

$$2. \quad P_t^* = \hat{P}_{t-1} + BQB^T$$

$$3. \quad Y_t^* = H_t X_t^*$$

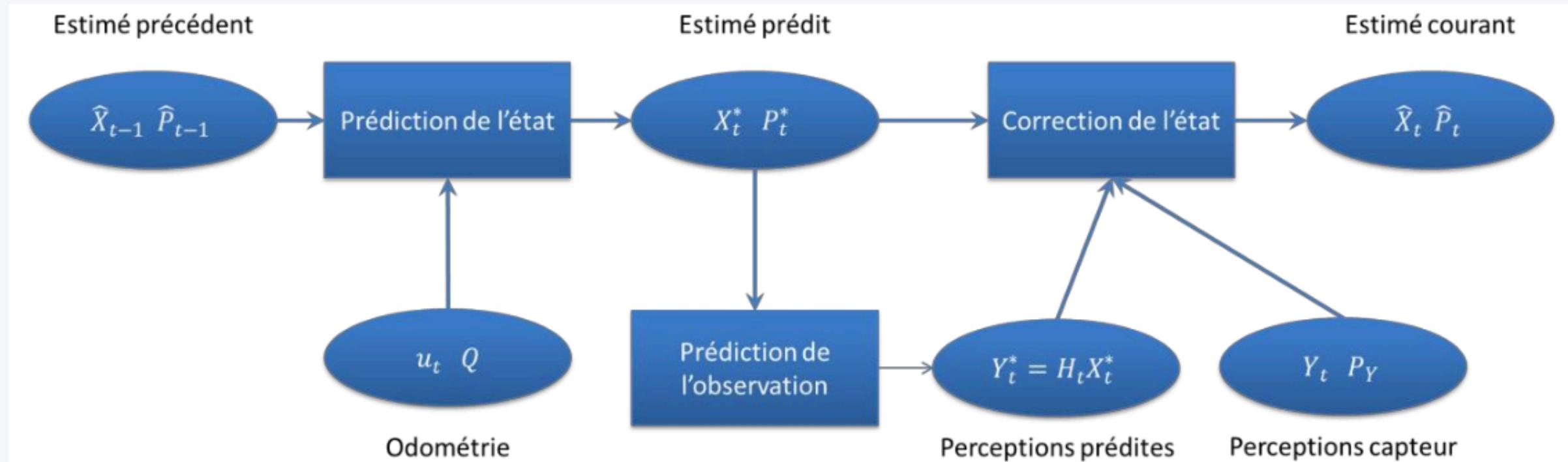
Y_t Vecteur de perceptions absolues contenant la distance relative entre le drone et chaque amer à l'instant t.

P_Y Matrice de covariance sur l'erreur des perceptions du capteur.

Y_t^* Vecteur de perceptions prédictes non corrigées à l'instant t.

I. Contextualisation

B. Méthode mathématiques: CORRECTION



\hat{X}_t Vecteur de positions prédites corrigées à l'instant t.

\hat{Y}_t Vecteur de perceptions absolues prédites corrigées à l'instant t.

\hat{P}_t Matrice de covariance prédite corrigée donnant l'incertitude sur les positions de \hat{X}_t .

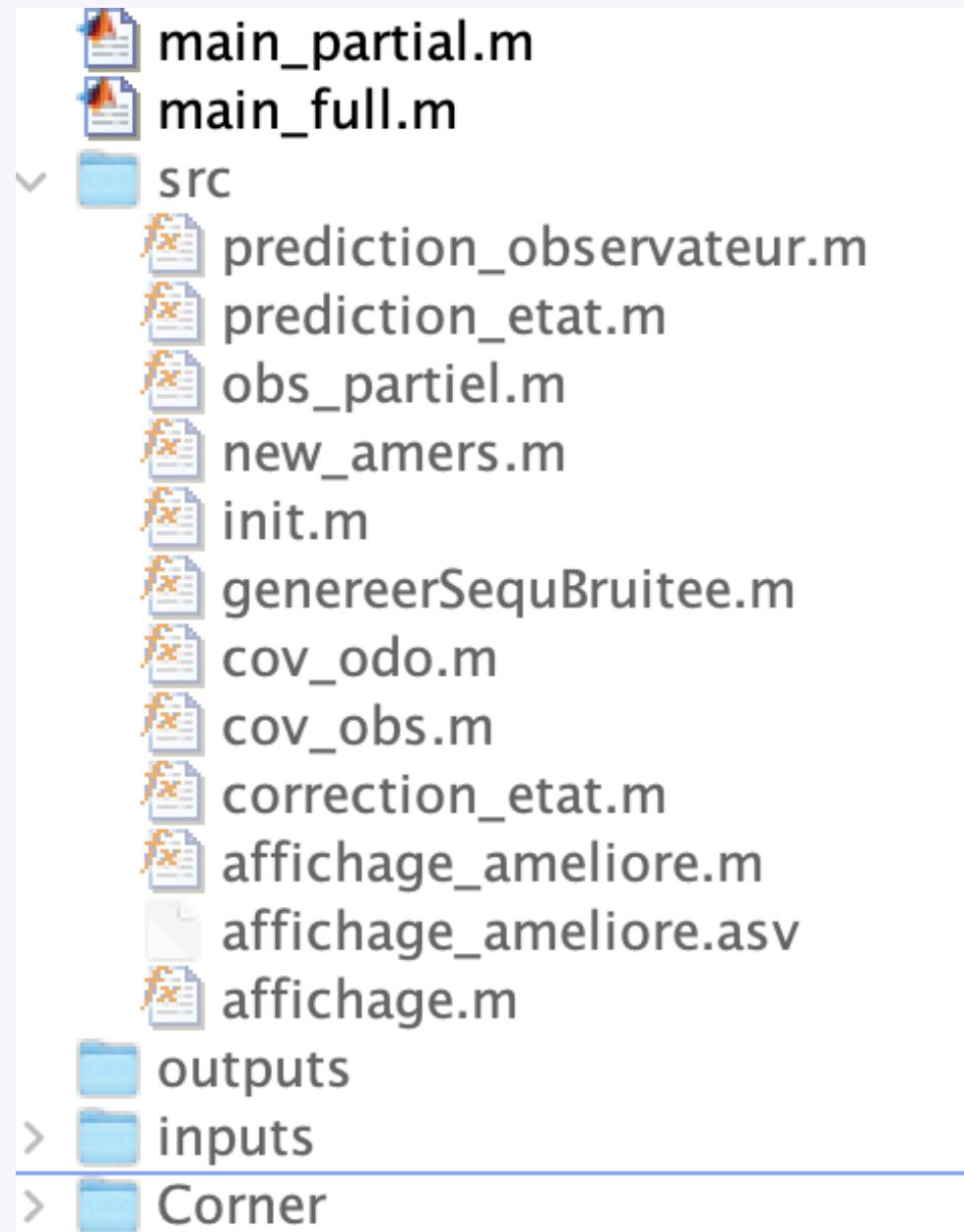
1. $X_t^* = \hat{X}_{t-1} + Bu_t$
2. $P_t^* = \hat{P}_{t-1} + BQB^T$
3. $Y_t^* = H_t X_t^*$
4. $\hat{X}_t = X_t^* + K_t(Y_t - Y_t^*)$
5. $\hat{P}_t = P_t^* - K_t H_t P_t^*$

$$K_t = P_t^* H_t^T (H_t P_t^* H_t^T + P_Y)^{-1}$$

K_t Gain de Kalman à l'instant t.

II. Connaissance totale des amers

A. Réalisation du filtre de Kalman : un peu de code ...



```
function [Xtchap, Ptchap] = correction_etat(Xtstar, Ptstar, Ytstar, Yt, Ht, Py)
%UNTITLED6 Summary of this function goes here
% Detailed explanation goes here
Kt = Ptstar*Ht' / (Ht*Ptstar*Ht' + Py);
Xtchap = Xtstar + Kt * (Yt - Ytstar);
Ptchap = Ptstar - Kt * Ht * Ptstar;
end
```

CORRECTION_ETAT.M

```
% Prediction sur la position
[Xt_star,Pt_star] = prediction_etat(Xt,donnees_odom,A, B,Pt,Q);

% Prediction sur l'observation
[Yt_star] = prediction_observateur(Xt_star, Ht);

% Correction de la prediction
[Xt, Pt] = correction_etat(Xt_star, Pt_star, Yt_star, Yt, Ht, P_Y);

% Affichage de la carte
affichage(Xt,Pt);
```

MAIN_FULL.M

II. Connaissance totale des amers

A. Réalisation du filtre de Kalman

```
percep : 2.059525 1.537290 1.102366 5.364240 5.407859 6.277271 7.850414 1.886459 4.320856  
3.534482  
odom : 0.512546 2.822456  
percep : 1.397940 -1.399142 0.491180 2.388502 4.589120 3.410499 7.197695 -1.019324 3.668032  
0.524368  
odom : 1.155917 1.797266
```

FULLOBSERVATION.DATA

ICI : TOUJOURS LE MÊME NOMBRE DE DONNÉES

```
format_percep = '%s : %f %f %f %f %f %f %f %f';  
data_percep = textscan(ligne, format_percep, 'Delimiter', '');
```

MAIN_FULL.M

II. Connaissance totale des amers

A. Réalisation du filtre de Kalman

```
% Afficher le drone avec son incertitude  
scatter(X_drone, Y_drone, 'filled', 'Marker', 'o', 'DisplayName', 'Drone');  
hold on;  
viscircles([X_drone, Y_drone], sigma_drone, 'EdgeColor', 'b', 'LineWidth', 1, 'LineStyle', '--');  
  
% Afficher les amers avec leur incertitude  
scatter(X_amers, Y_amers, 'filled', 'Marker', 'o', 'DisplayName', 'Amers');  
viscircles([X_amers, Y_amers], sigma_amers, 'EdgeColor', 'r', 'LineWidth', 1, 'LineStyle', '--');
```

AFFICHAGE.M

CHOIX DE REPRESENTER LES INCERTITUDES PAR
DES CERCLES ET NON PAR DES ELLIPSES

DEFINITION DE P_T : MATRICE DE COVARIANCE

```
%Incertitude issue de la distance  
M(i,i) = power(k*norm(Y_t(i:(i+1))), 2);  
M(i+1, i+1) = M(i,i);
```

COV_OBS.M

II. Connaissance totale des amers

A. Réalisation du filtre de Kalman

```
% Afficher la trajectoire prédict par l'odométrie
subplot(1, 2, 1);
plot(X_odom, Y_odom, '-o', 'DisplayName', 'Odométrie', 'Color', 'green', 'LineWidth', 2);
hold on;images amateurs Twitter

% Afficher son incertitude
viscircles([X_odom, Y_odom], sigma_drone, 'EdgeColor', 'g', 'LineWidth', 1, 'LineStyle', '--');

% Afficher les amers avec leur incertitude
plot(X_amers, Y_amers, 's', 'DisplayName', 'Amers', 'Color', 'blue');
viscircles([X_amers, Y_amers], sigma_amers, 'EdgeColor', 'b', 'LineWidth', 1, 'LineStyle', '--');

% Afficher la trajectoire corrigée par le filtre de Kalman

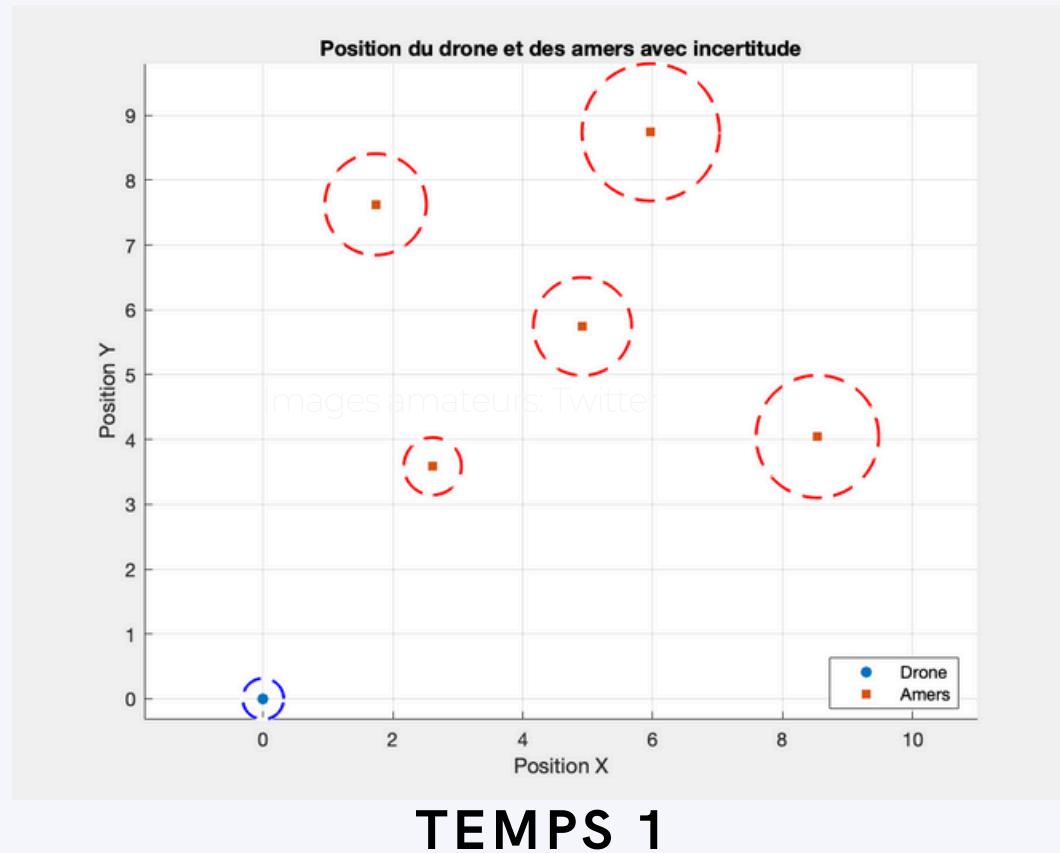
plot(X_kalman, Y_kalman, '-o', 'DisplayName', 'Filtre de Kalman', 'Color', 'red', 'LineWidth', 2);
```

```
% Afficher la matrice de covariance
subplot(1, 2, 2);
imagesc(mat_cov);
title('Matrice de Covariance');
colorbar;
```

AFFICHAGE_AMELIORE.M

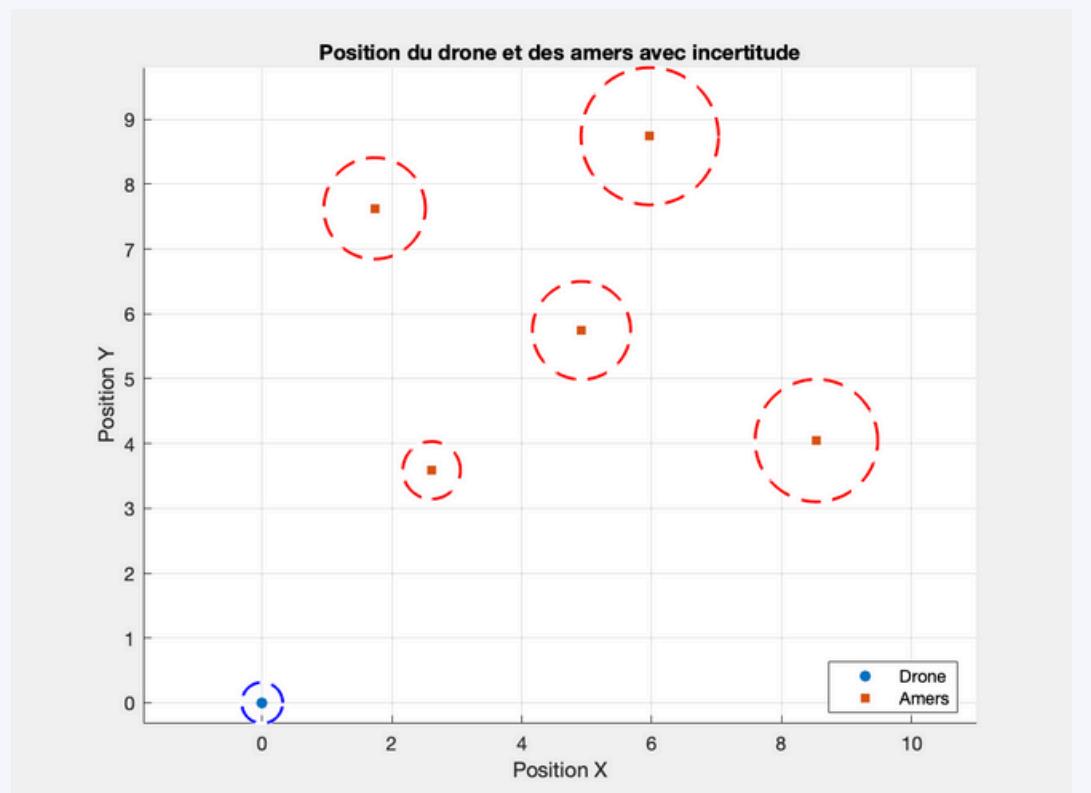
II. Connaissance totale des amers

B. Résultats



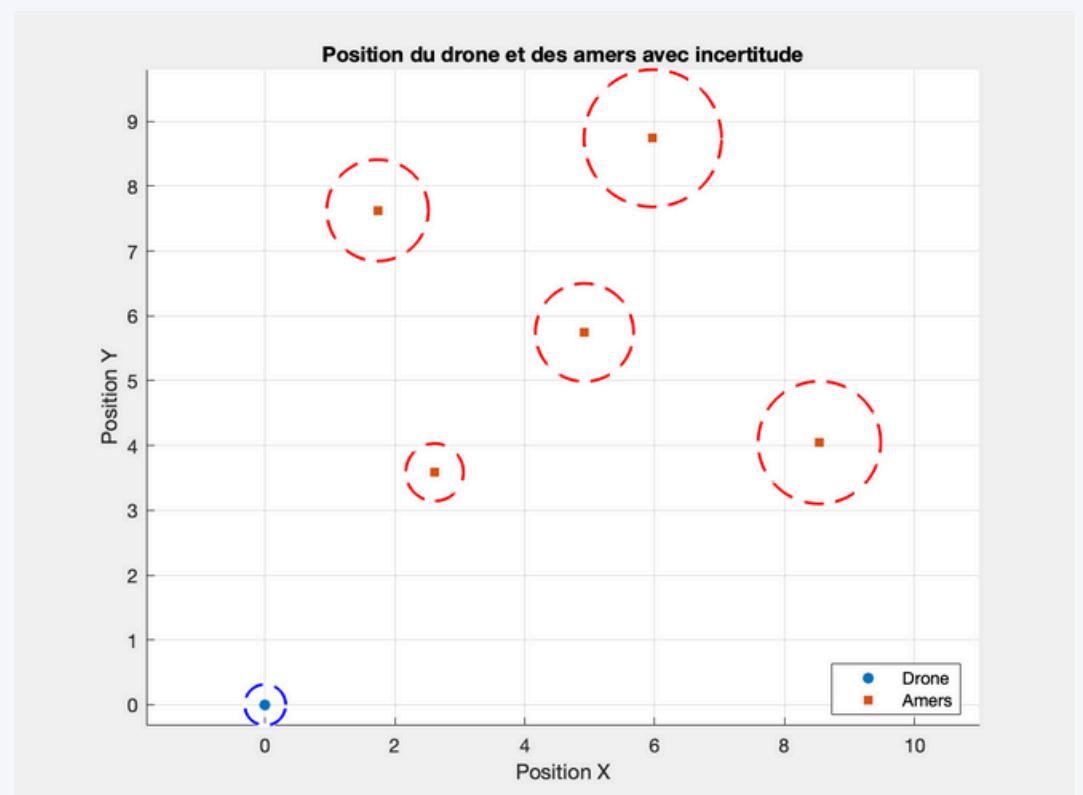
AFFICHAGE DES POSITIONS DU DRONE, DES AMERS ET DE LEUR INCERTITUDE AU COURS DU TEMPS

II. Connaissance totale des amers

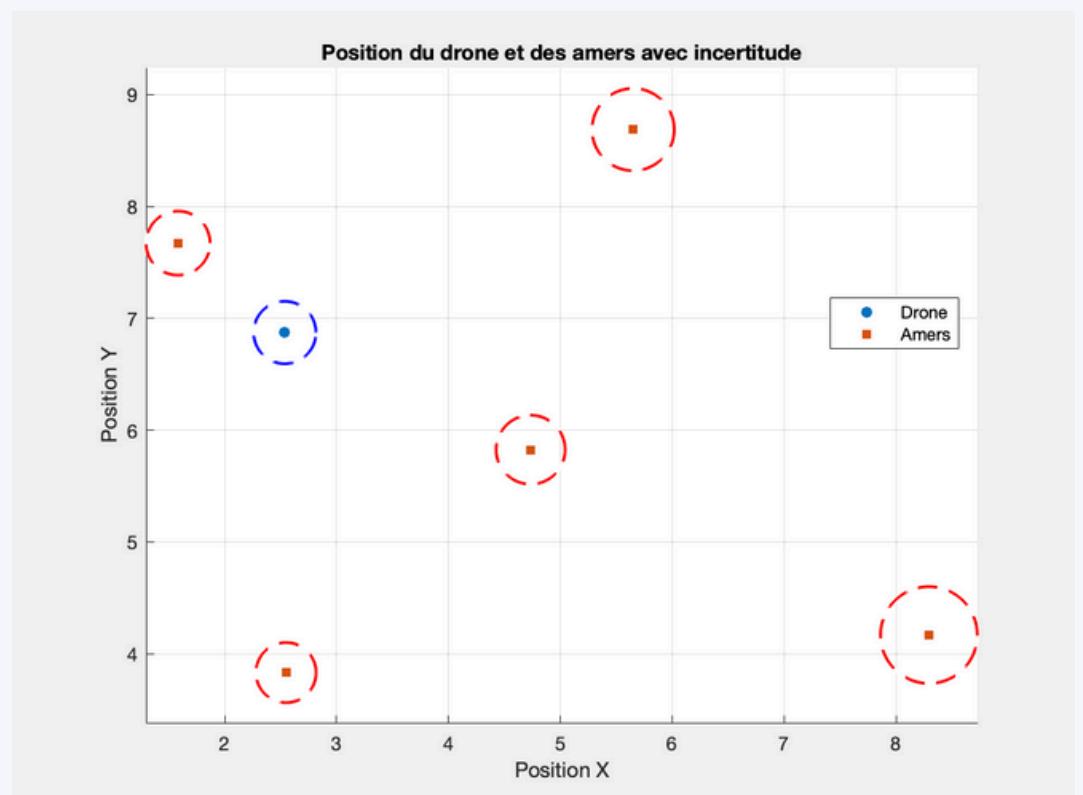


TEMPS 1

II. Connaissance totale des amers

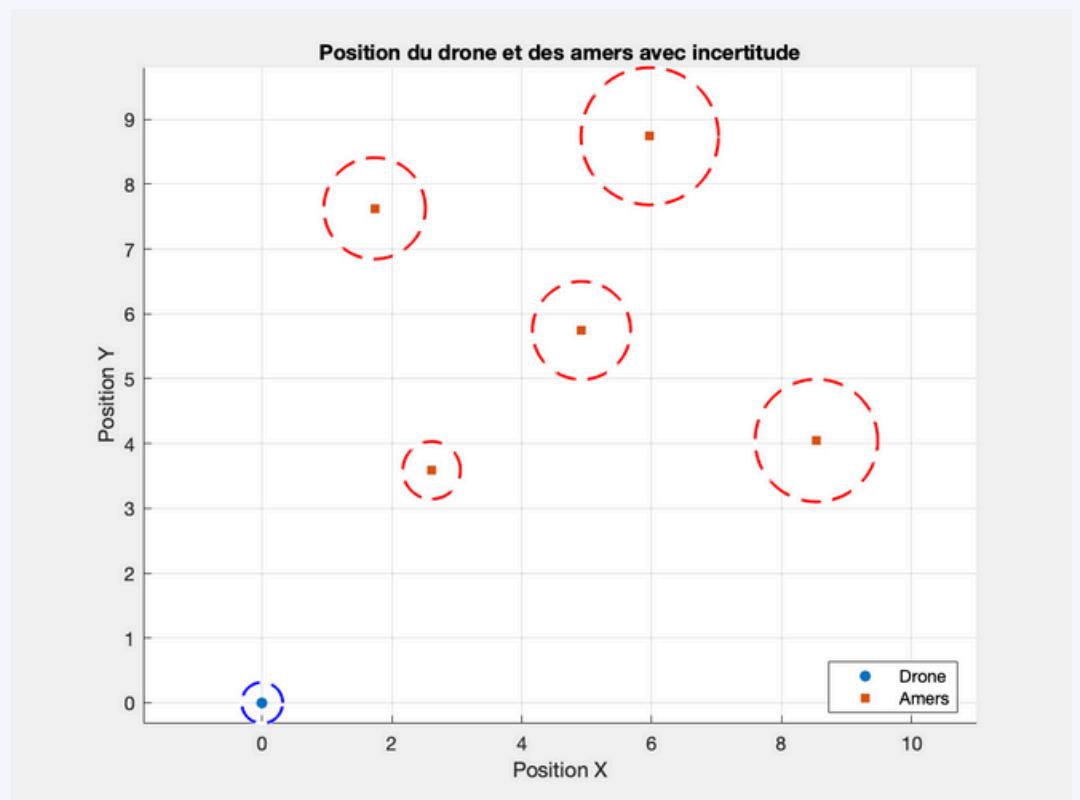


TEMPS 1

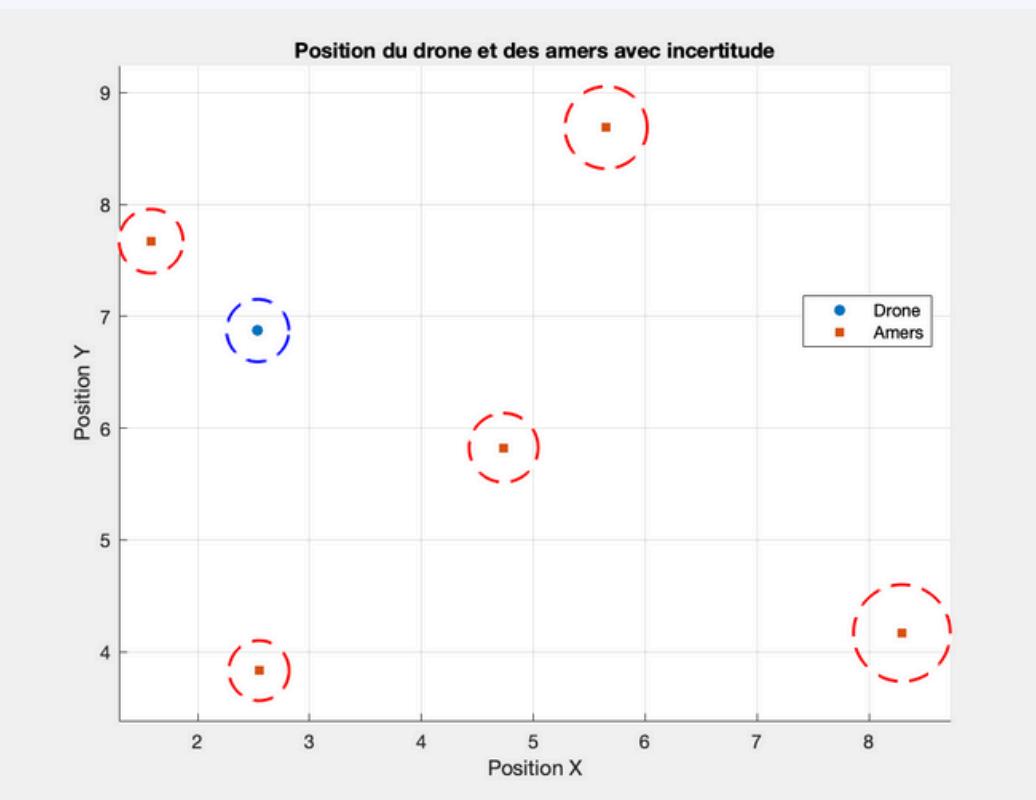


TEMPS 2

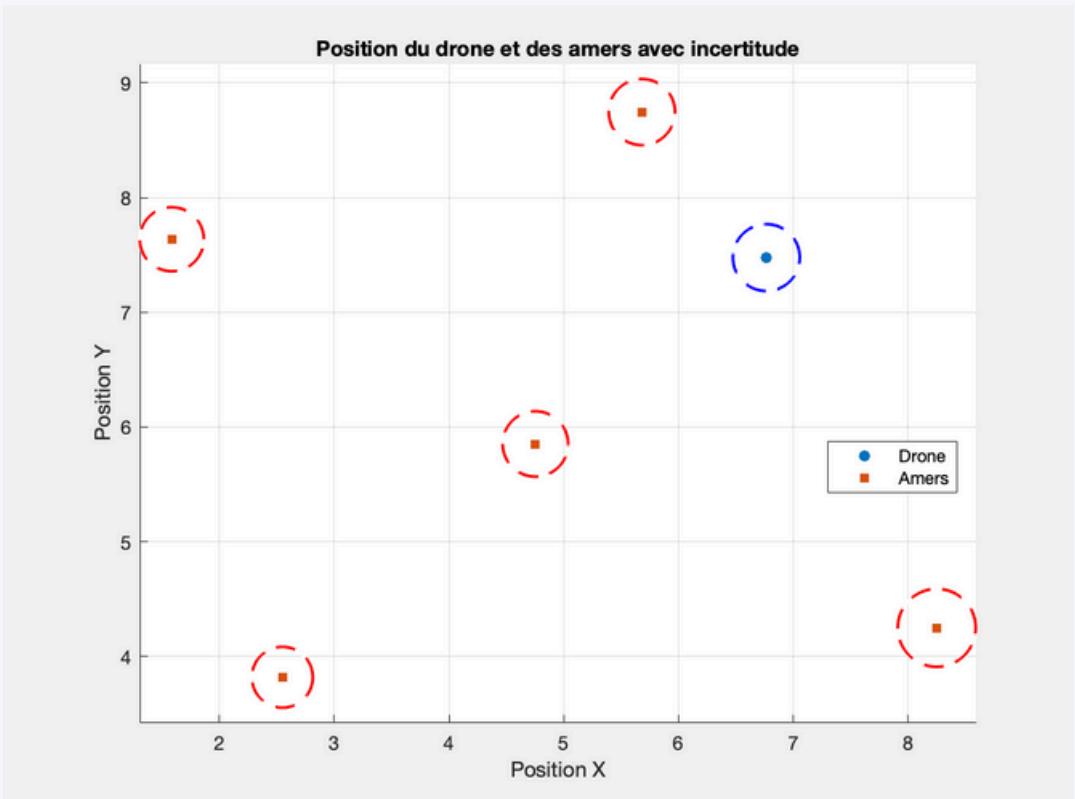
II. Connaissance totale des amers



TEMPS 1

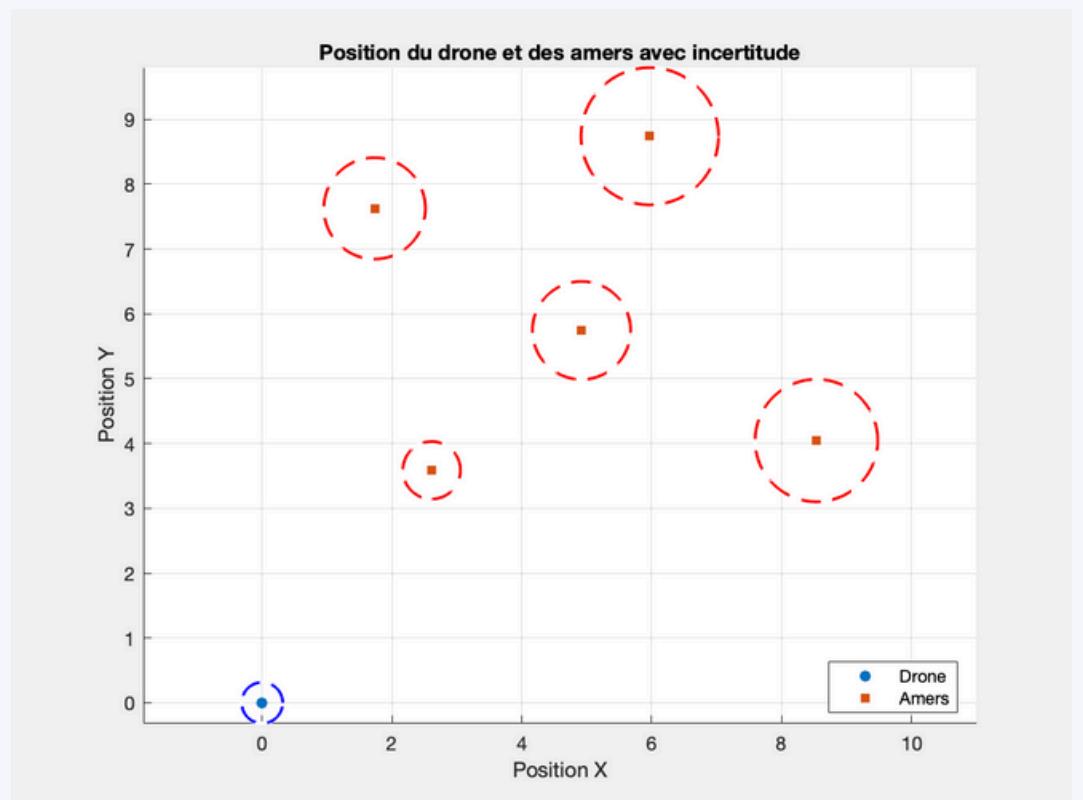


TEMPS 2

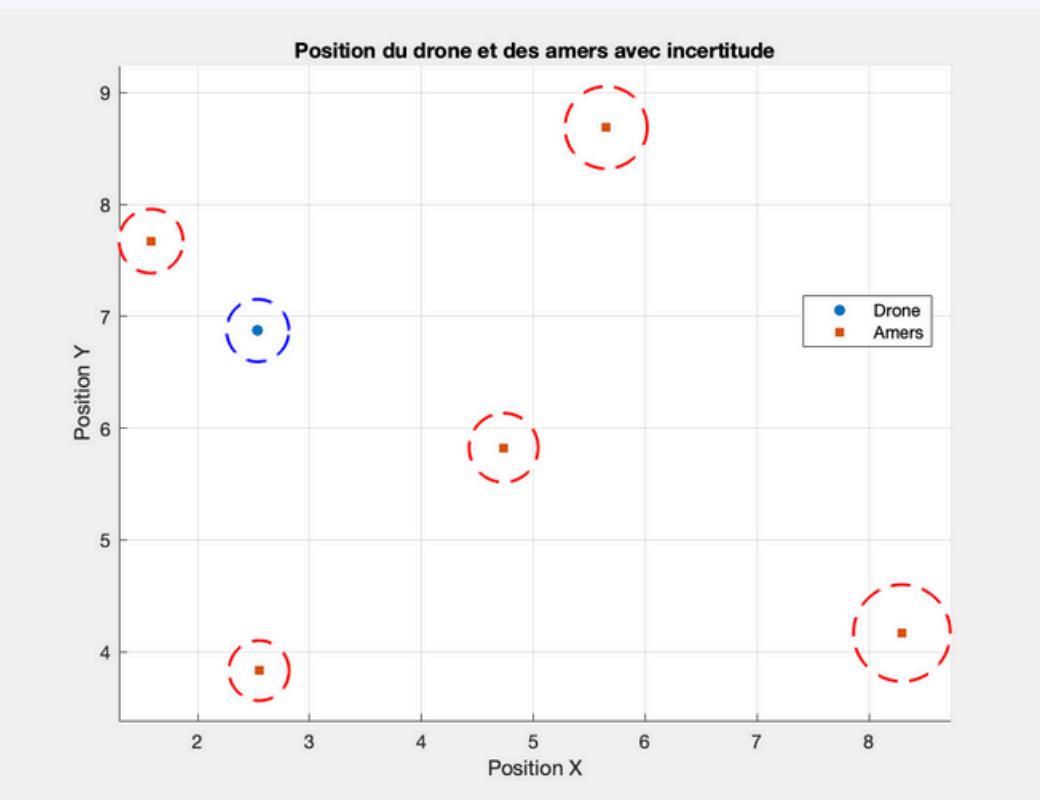


TEMPS 3

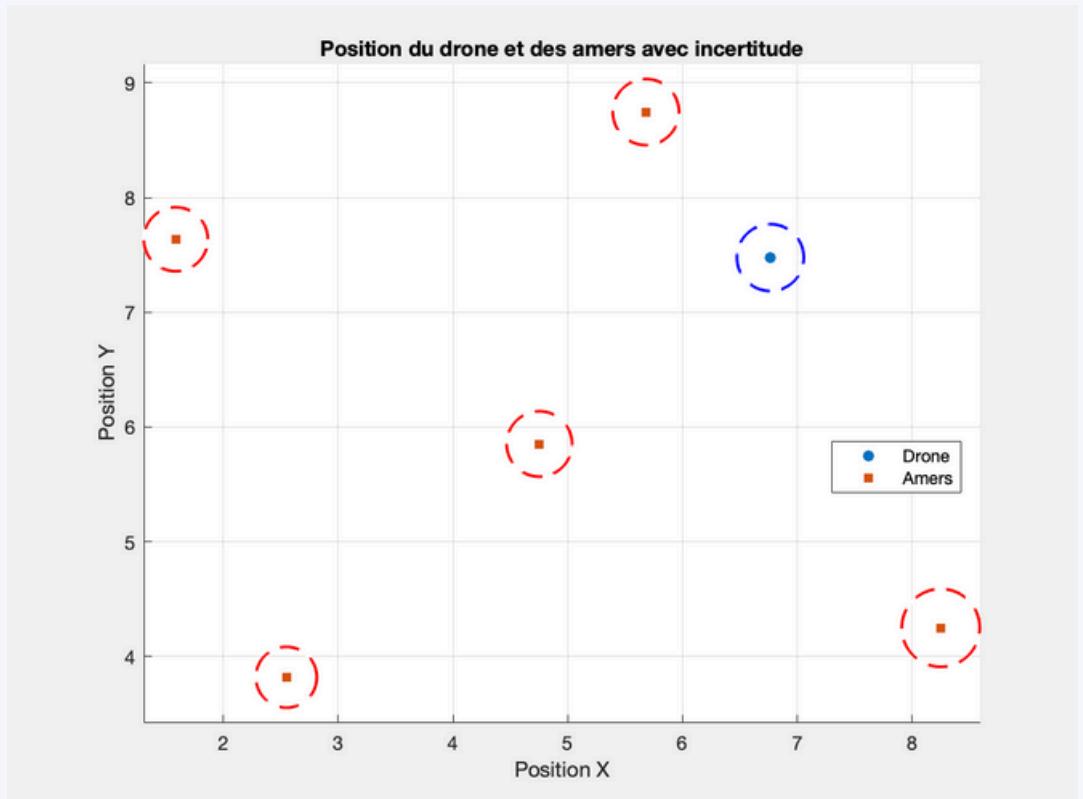
II. Connaissance totale des amers



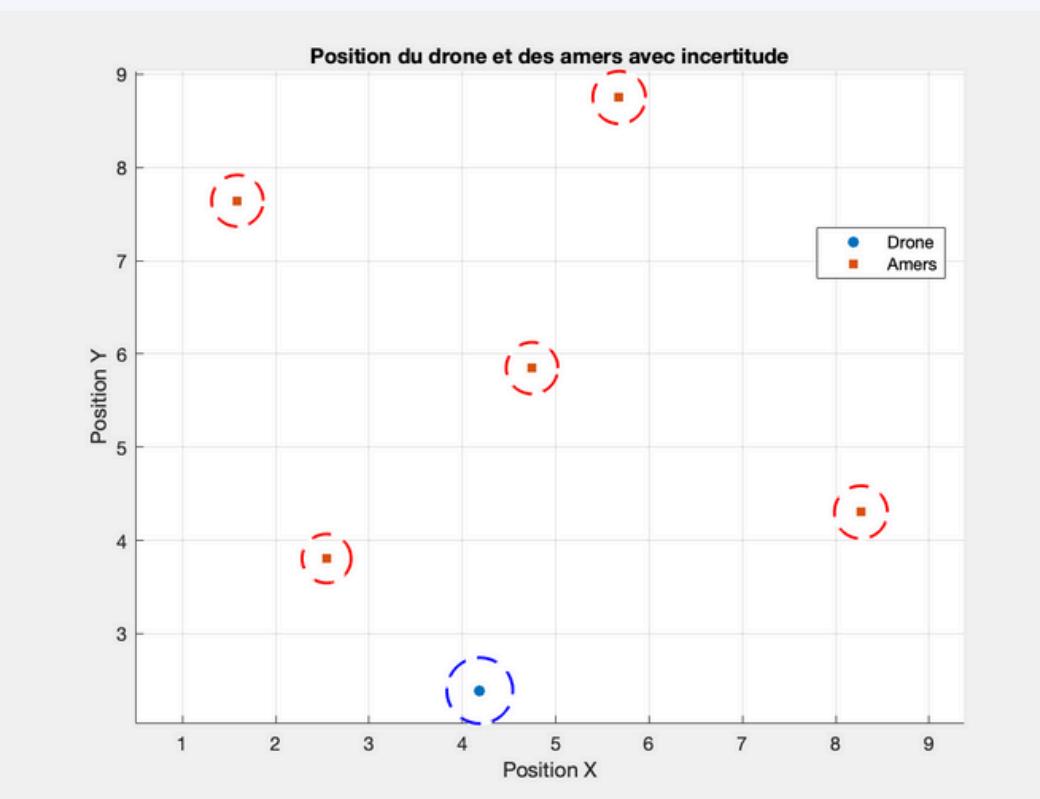
TEMPS 1



TEMPS 2



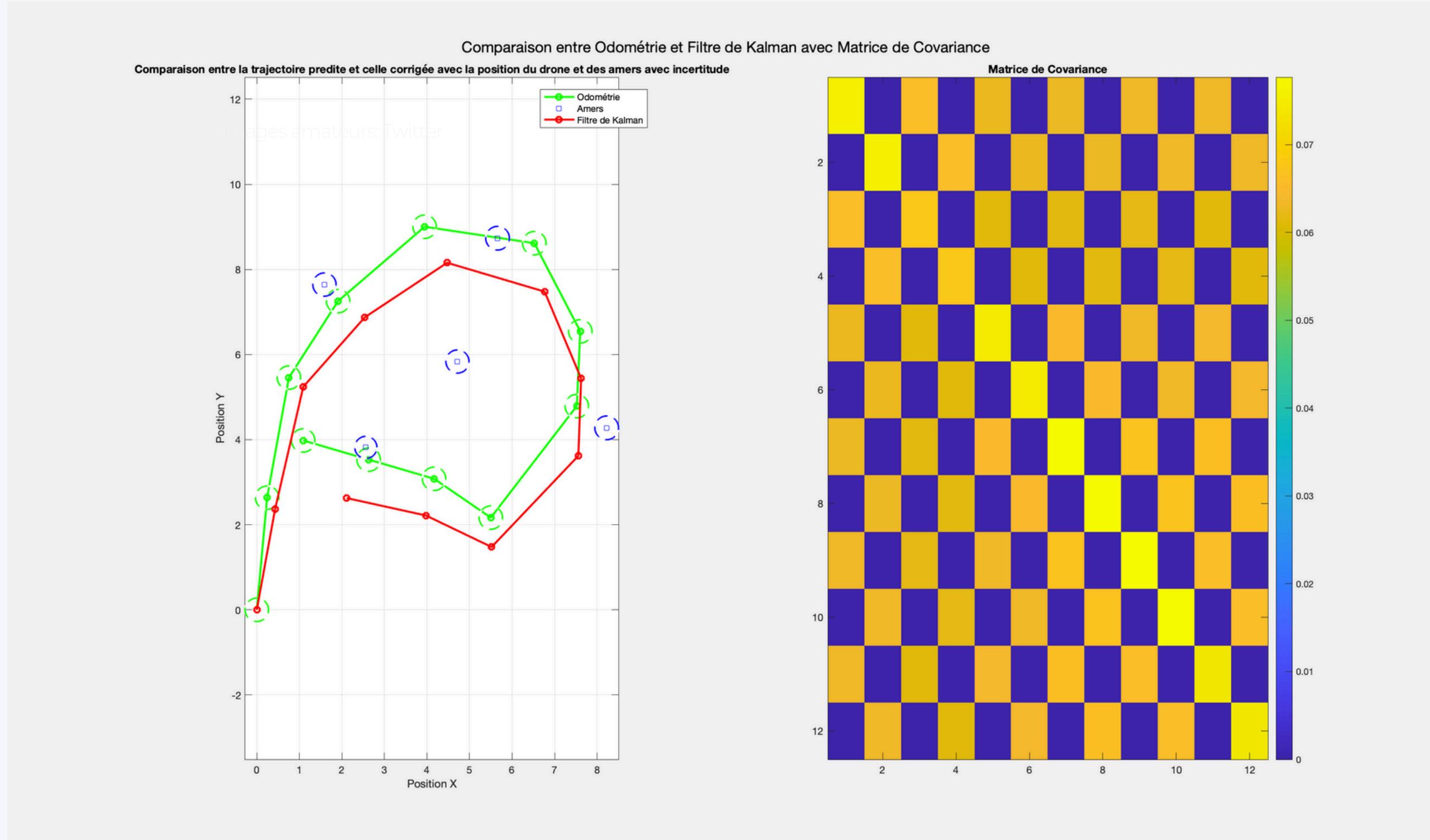
TEMPS 3



TEMPS 4

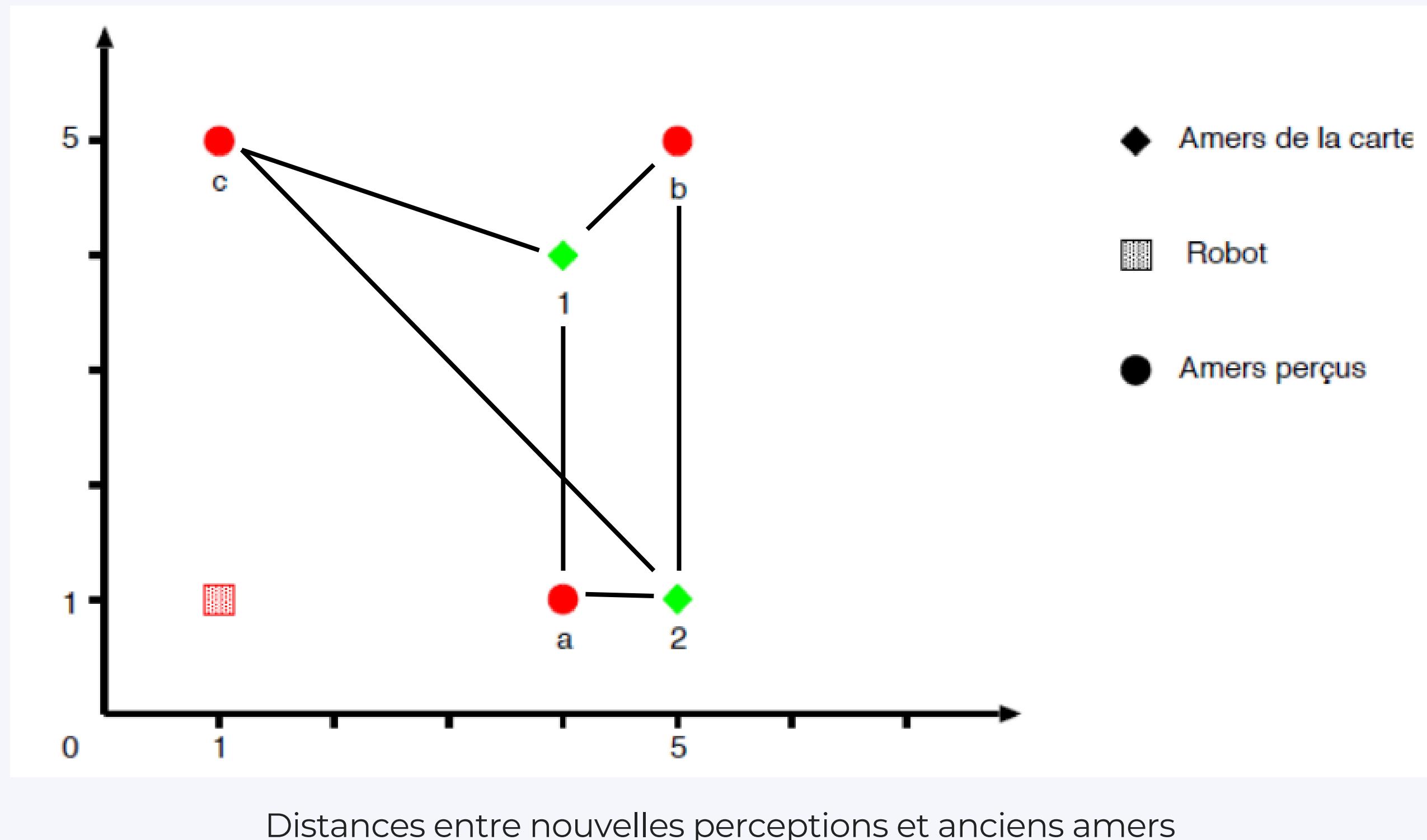
II. Connaissance totale des amers

B. Résultats



III. Connaissance partielle des amers

A. Principe et initiatives



III. Connaissance partielle des amers

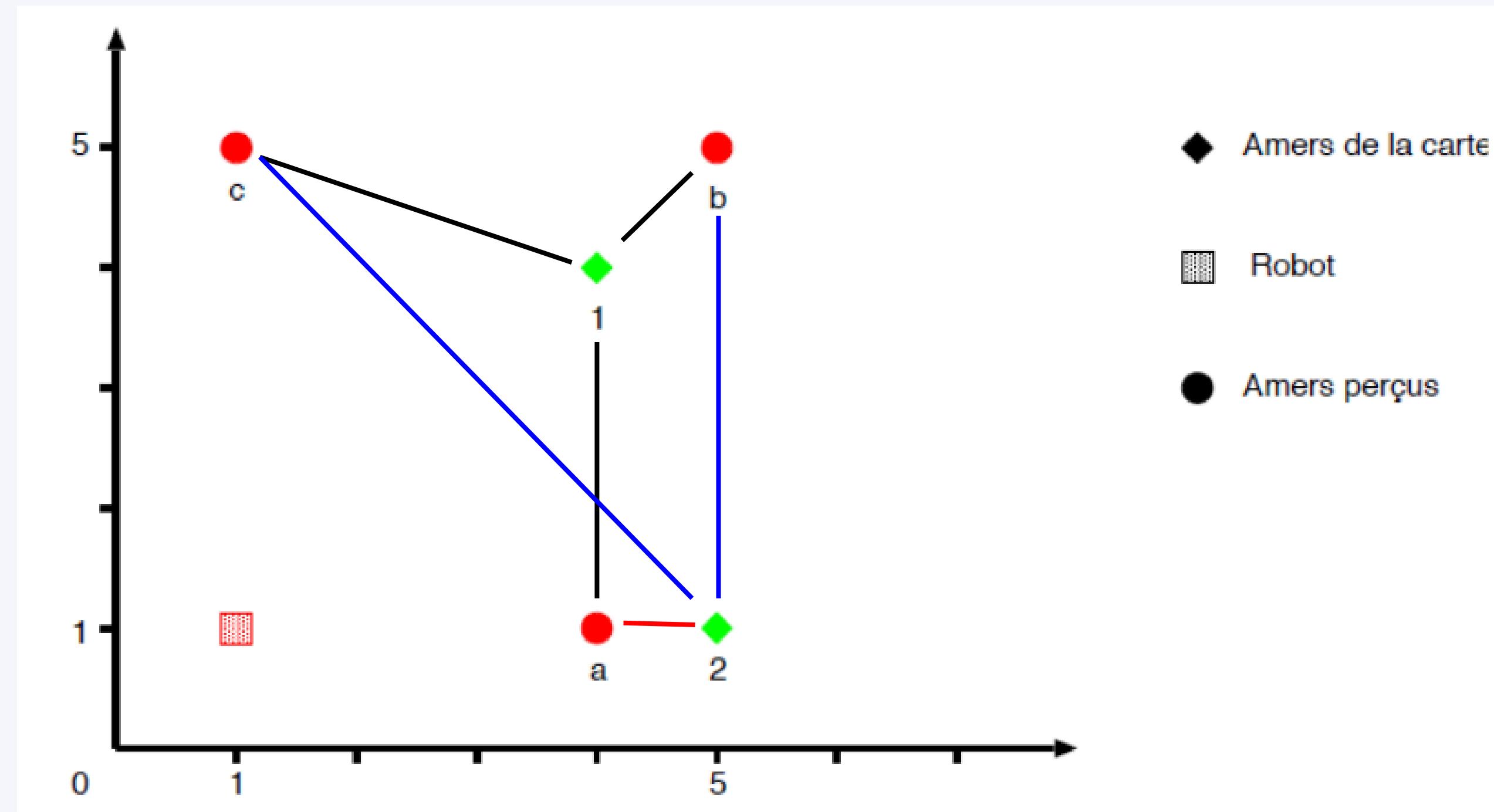
A. Principe et initiatives



CONNEXION ENTRE PERCEPTION ET AMER CONNU



CONNEXION IGNORÉE OU SUPÉRIEURE AU SEUIL



Connexions entre nouvelles perceptions et anciens amers pour **seuil = 10**

III. Connaissance partielle des amers

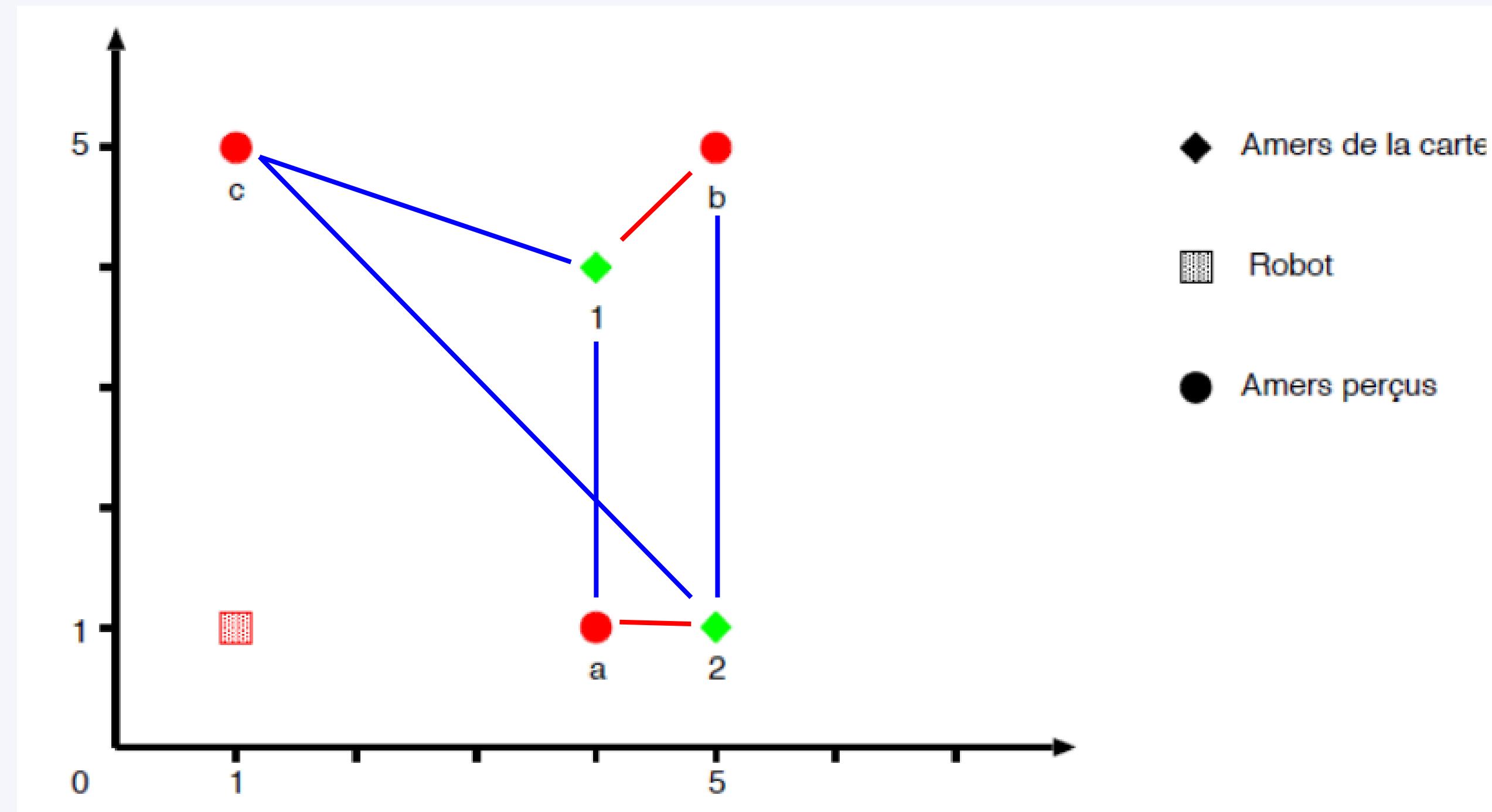
A. Principe et initiatives



CONNEXION ENTRE PERCEPTION ET AMER CONNU



CONNEXION IGNORÉE OU SUPÉRIEURE AU SEUIL



Connexions entre nouvelles perceptions et anciens amers pour **seuil = 10**

III. Connaissance partielle des amers

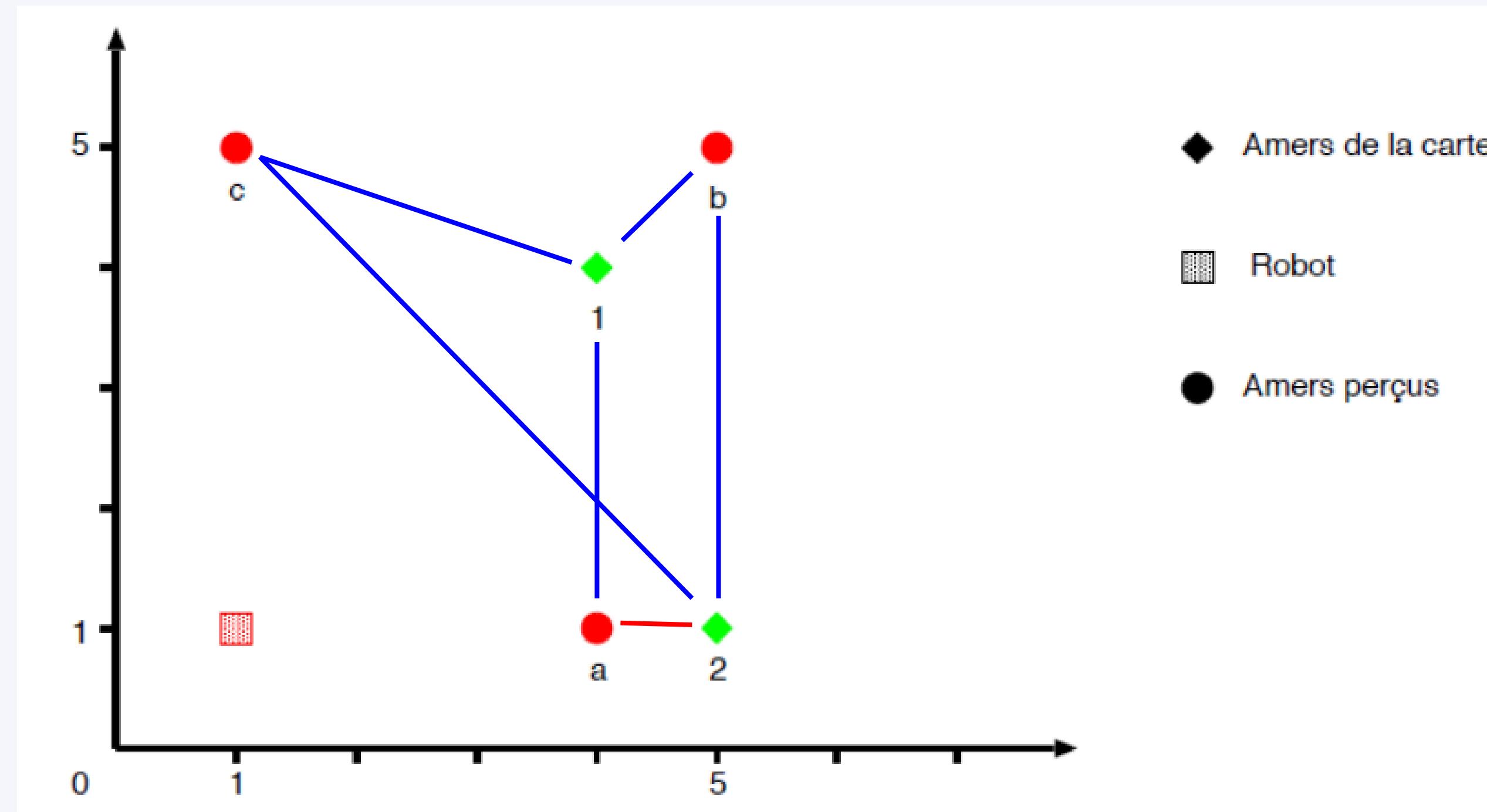
A. Principe et initiatives



CONNEXION ENTRE PERCEPTION ET AMER CONNU



CONNEXION IGNORÉE OU SUPÉRIEURE AU SEUIL



Connexions entre nouvelles perceptions et anciens amers pour **seuil = 1**

III. Connaissance partielle des amers

A. Principe et initiatives: un peu de code ...

```
minimum = min(dist_matrix(:)); % On récupère min pour extraire  
amer_min = 0;  
perc_min = 0;  
while (minimum < seuil) % Tant qu'il y a des distances < seuil  
    for i = 1:N_amers  
        for j = 1:N_perceptions  
            if dist_matrix(i,j) == minimum % On choisit la connexion  
                amer_min = i;  
                perc_min = j;  
            end  
        end  
    end  
    associations(perc_min) = amer_min; % On enregistre la connexion  
  
    dist_matrix(amer_min, :) = Inf; % On rend inutile toutes les autres  
    dist_matrix(:, perc_min) = Inf;  
  
    minimum = min(dist_matrix(:));% On considère la connexion minimale  
  
    N_known = N_known + 1;  
end
```

III. Connaissance partielle des amers

A. Principe et initiatives: un peu de code ...

```
% Lecture des premières perceptions  
data = textscan(fid, "%s");  
data = data{1};
```

Lecture du fichier original

```
% Indice de lecture dans l'ensemble des données parsées  
i_match = 3; % On commence à 3 pour aller chercher directement les percepts  
  
Yt = [];  
while ~strcmp(data{i_match}, 'odom')  
    Yt = [Yt; str2double(data{i_match})];  
    i_match = i_match+1;  
end
```

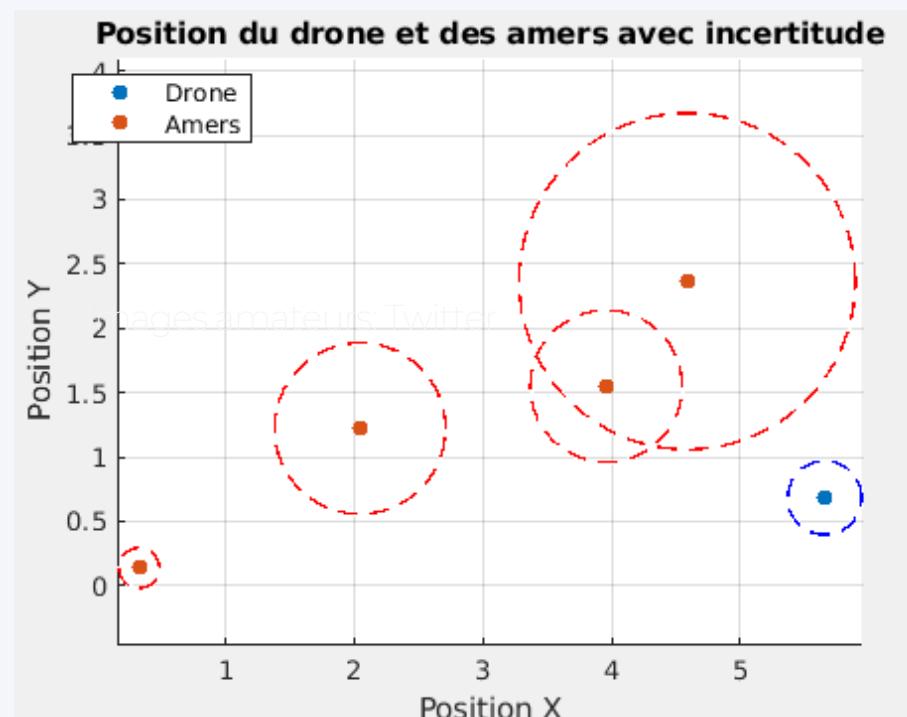
Exemple de parsing au début du fichier

```
odom : 1.719368 0.481953  
percep : -1.525463 -0.415648 1.334917 0.153515
```

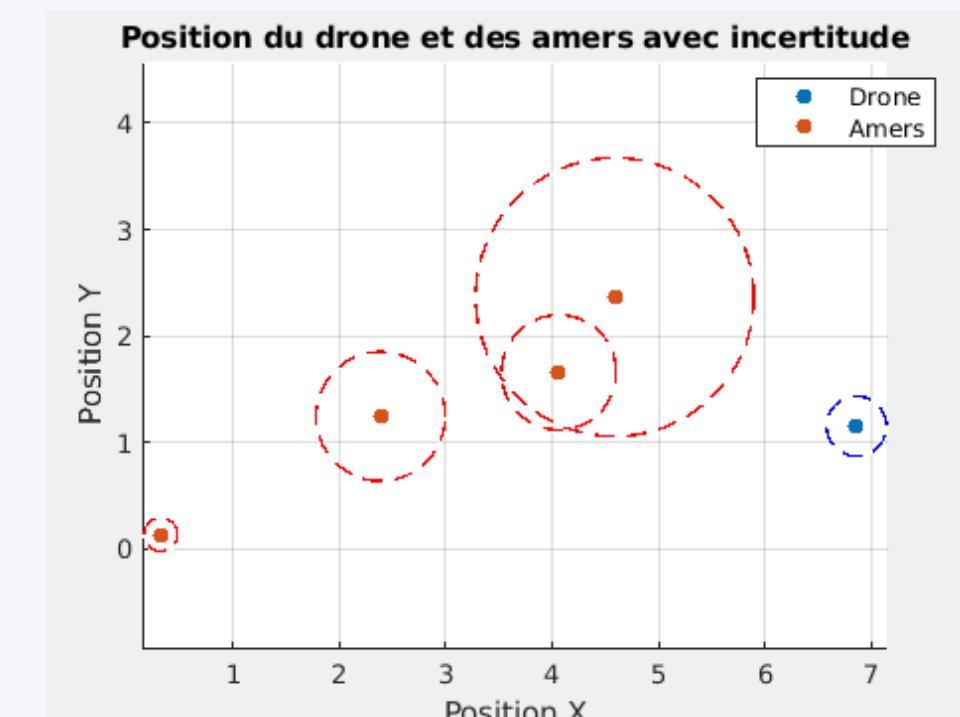
Exemple d'inputs

III. Connaissance partielle des amers

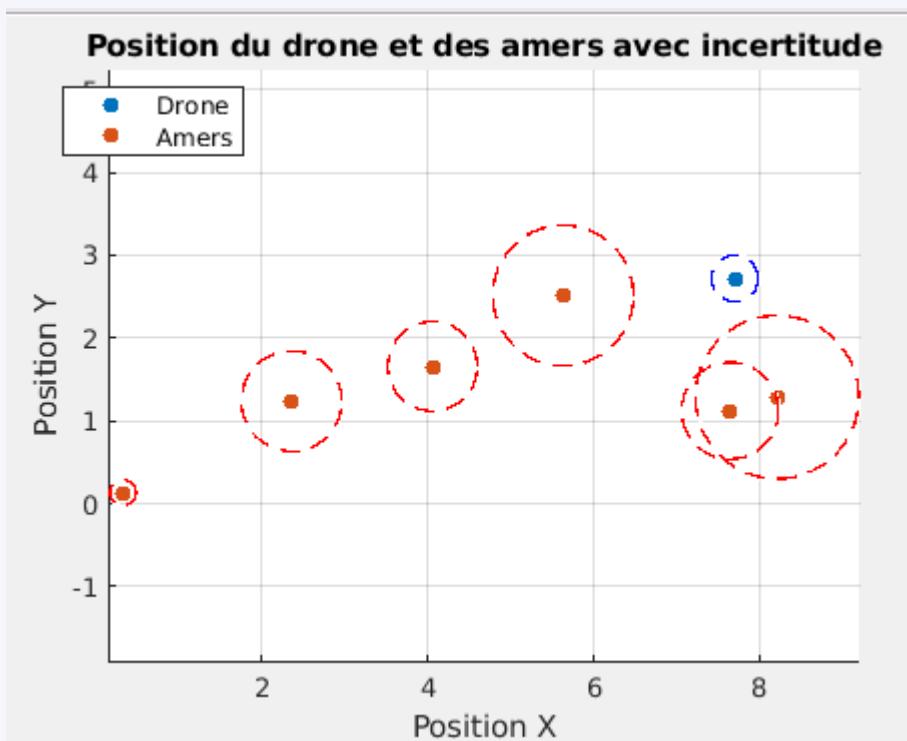
B. Résultats



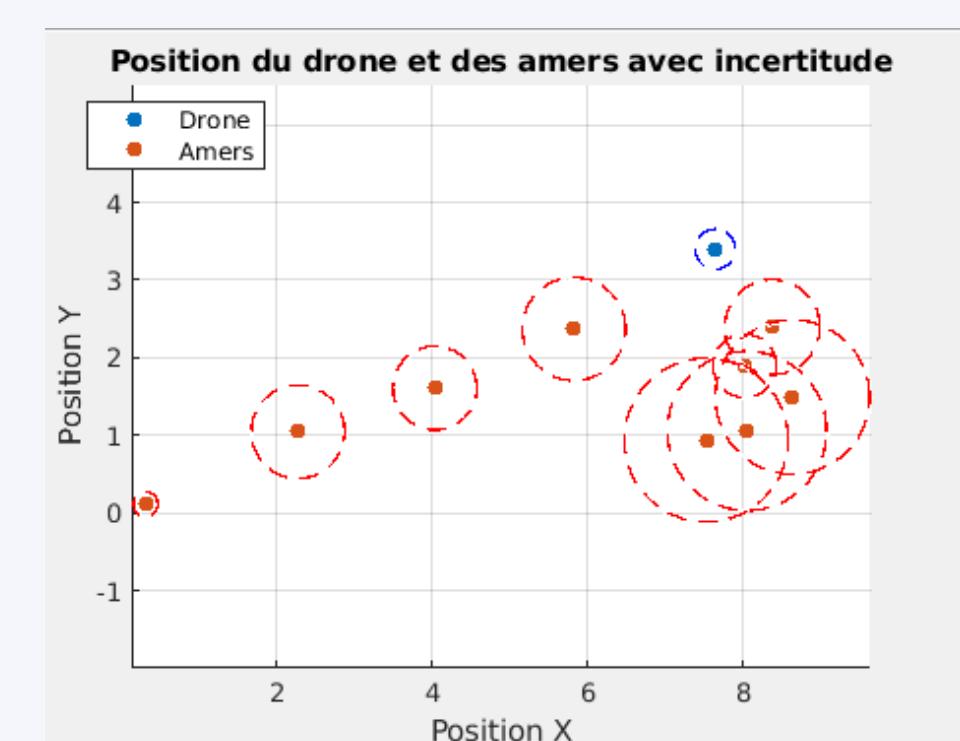
TEMPS 1



TEMPS 2



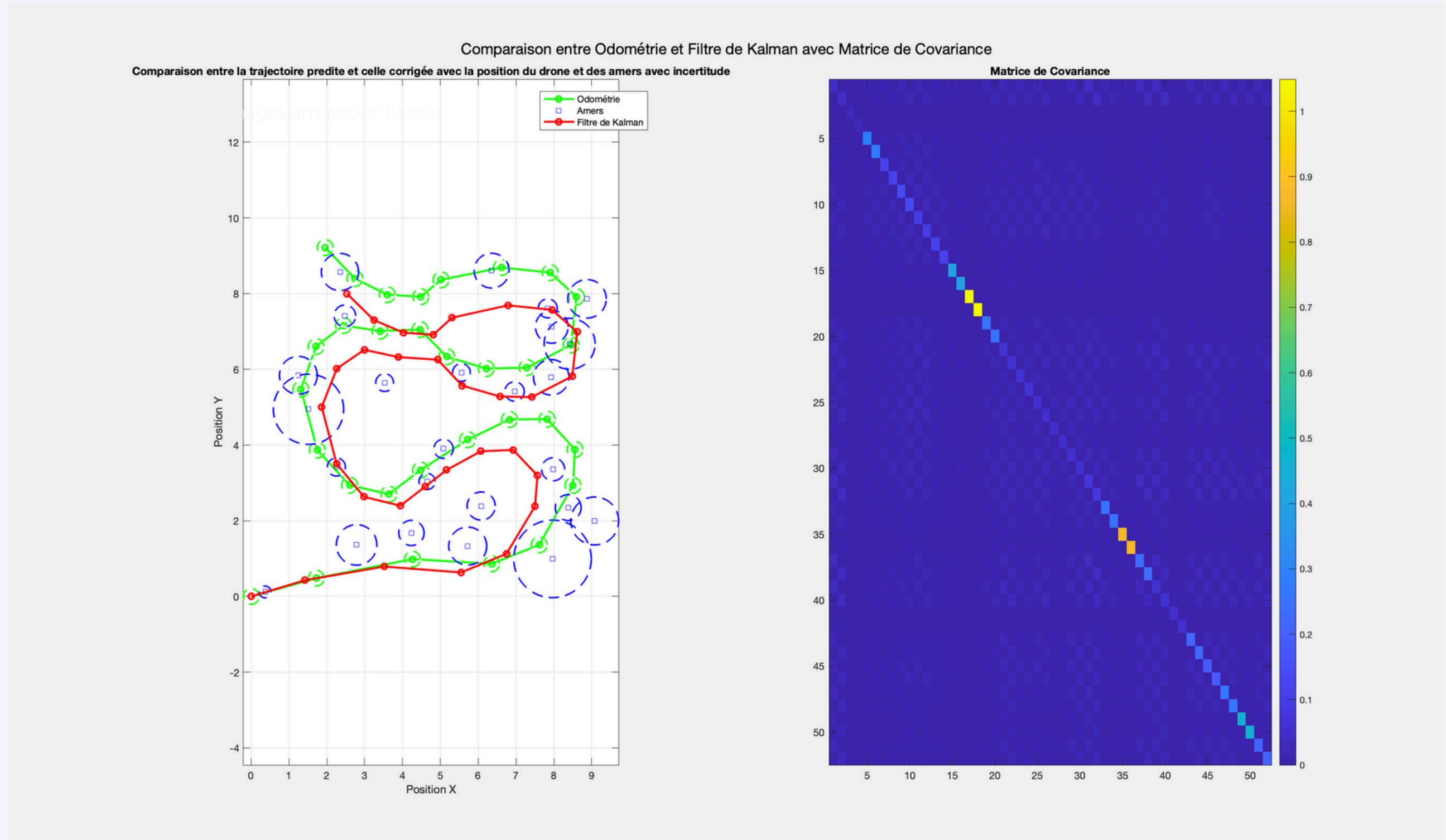
TEMPS 3



TEMPS 4

III. Connaissance partielle des amers

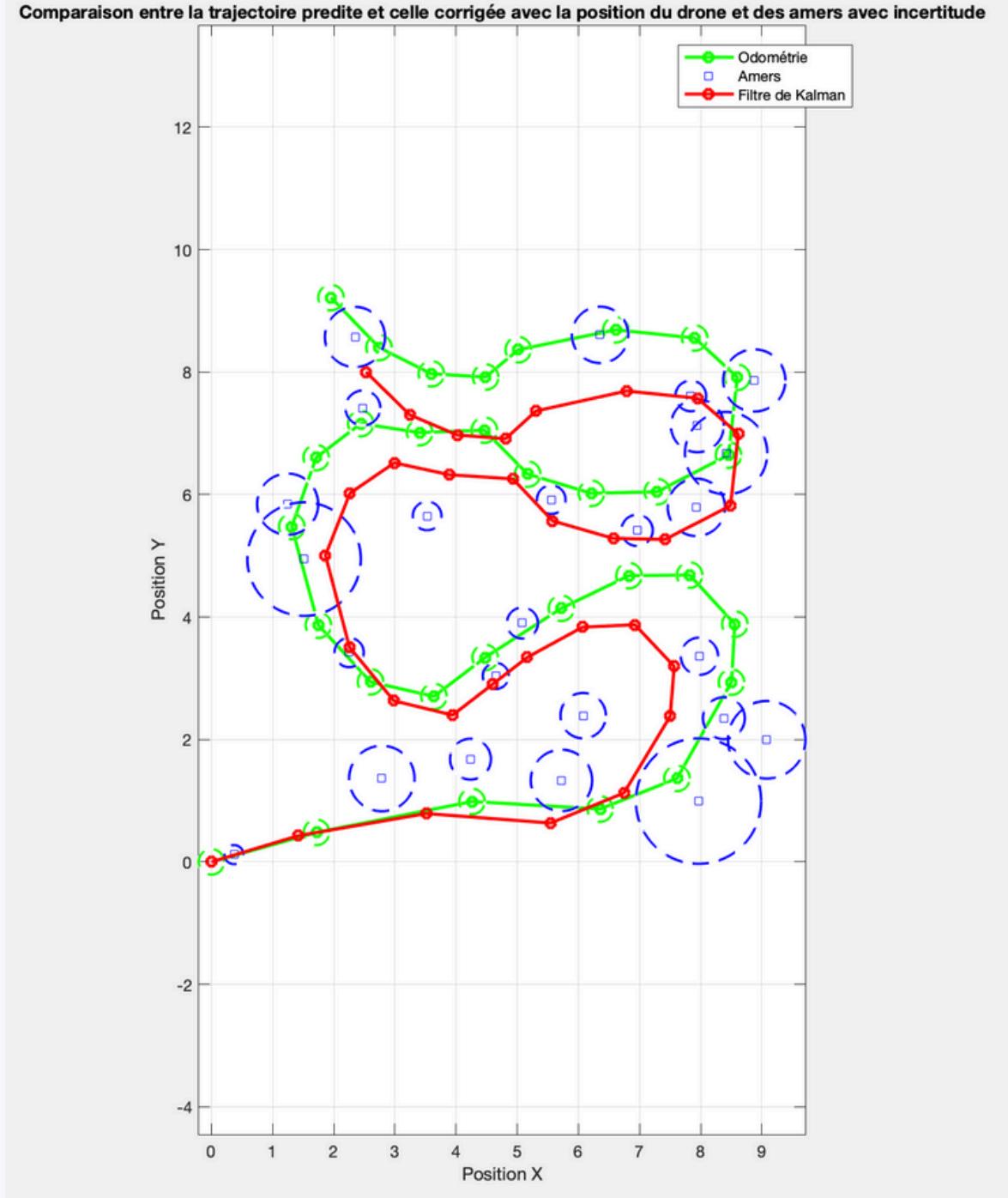
B. Résultats



À GAUCHE :
COMPARAISON ENTRE
• TRAJECTOIRE VERTE :
ODOMÉTRIE
• TRAJECTOIRE ROUGE :
CORRIGÉE PAR LE
FILTRE

À DROITE :
MATRICE DE COVARIANCE

Conclusion

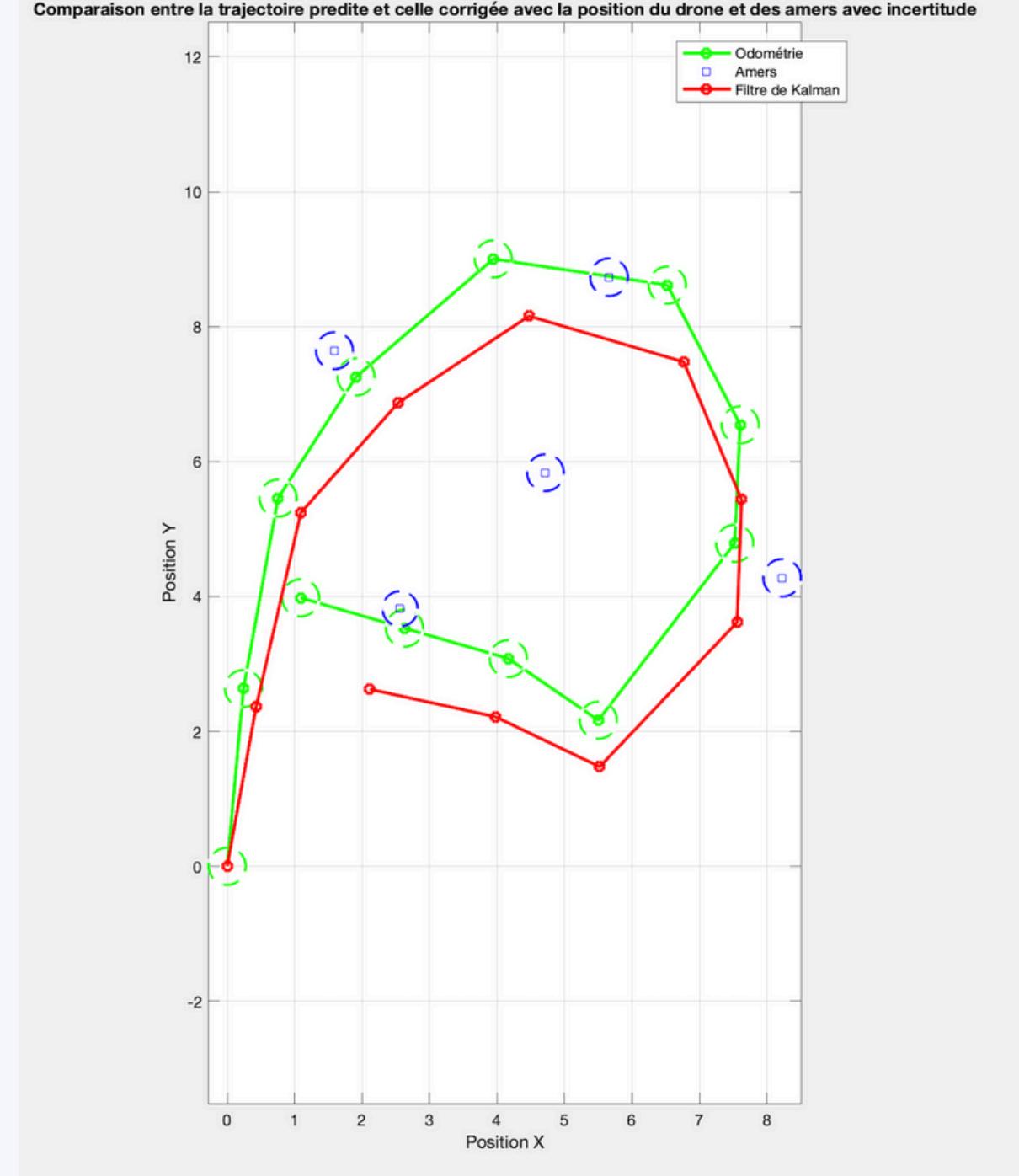


DIFFICULTÉS RENCONTRÉES :

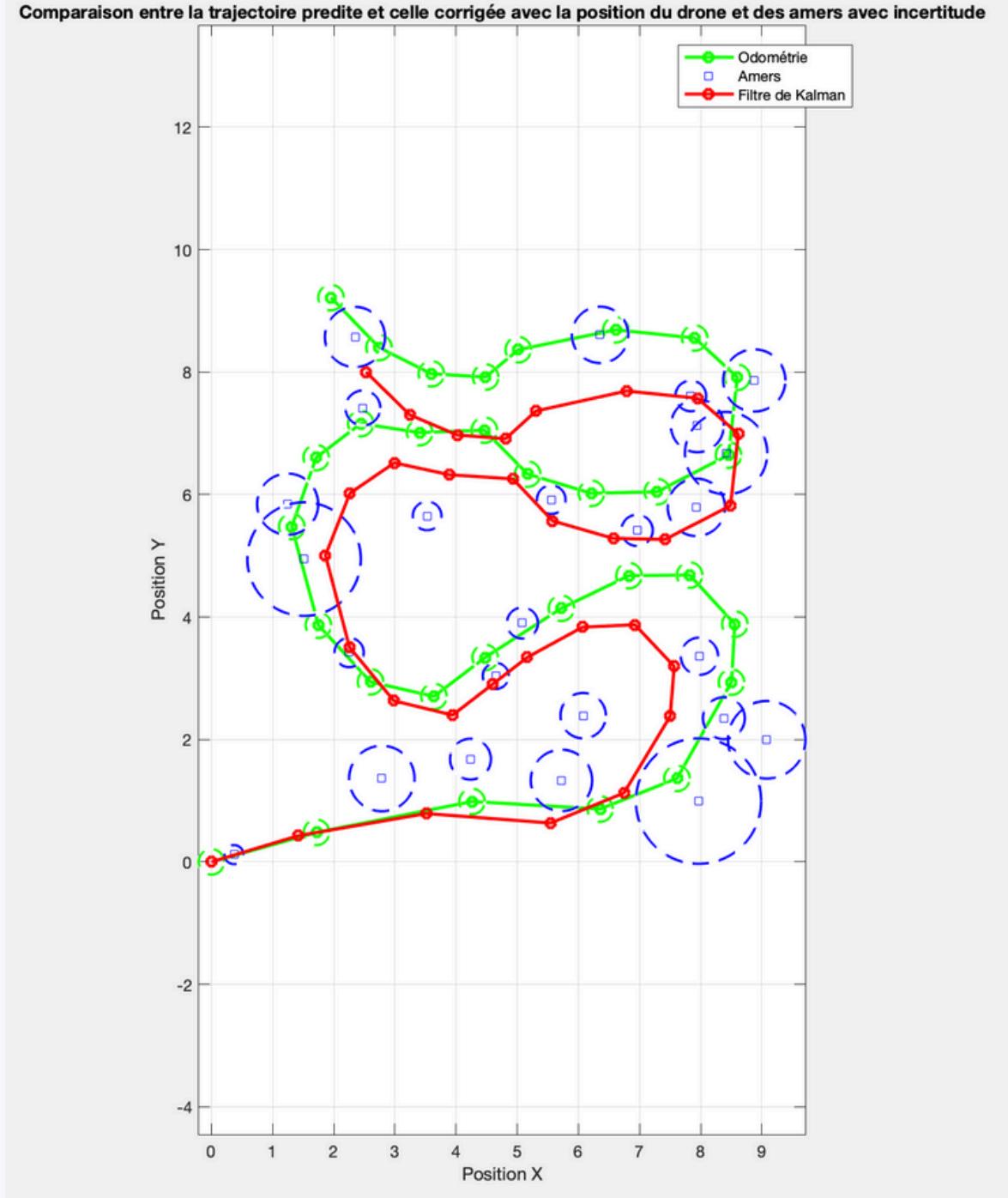
- Assimilation des notations et des différentes dimensions
- Interprétation d'un fichier de taille variable
- Nécessité de calibrage

SOLUTIONS APPORTÉES :

- Mise en place d'un système de seuil
- Calibrage selon des trajectoires de référence



Conclusion



DIFFICULTÉS RENCONTRÉES :

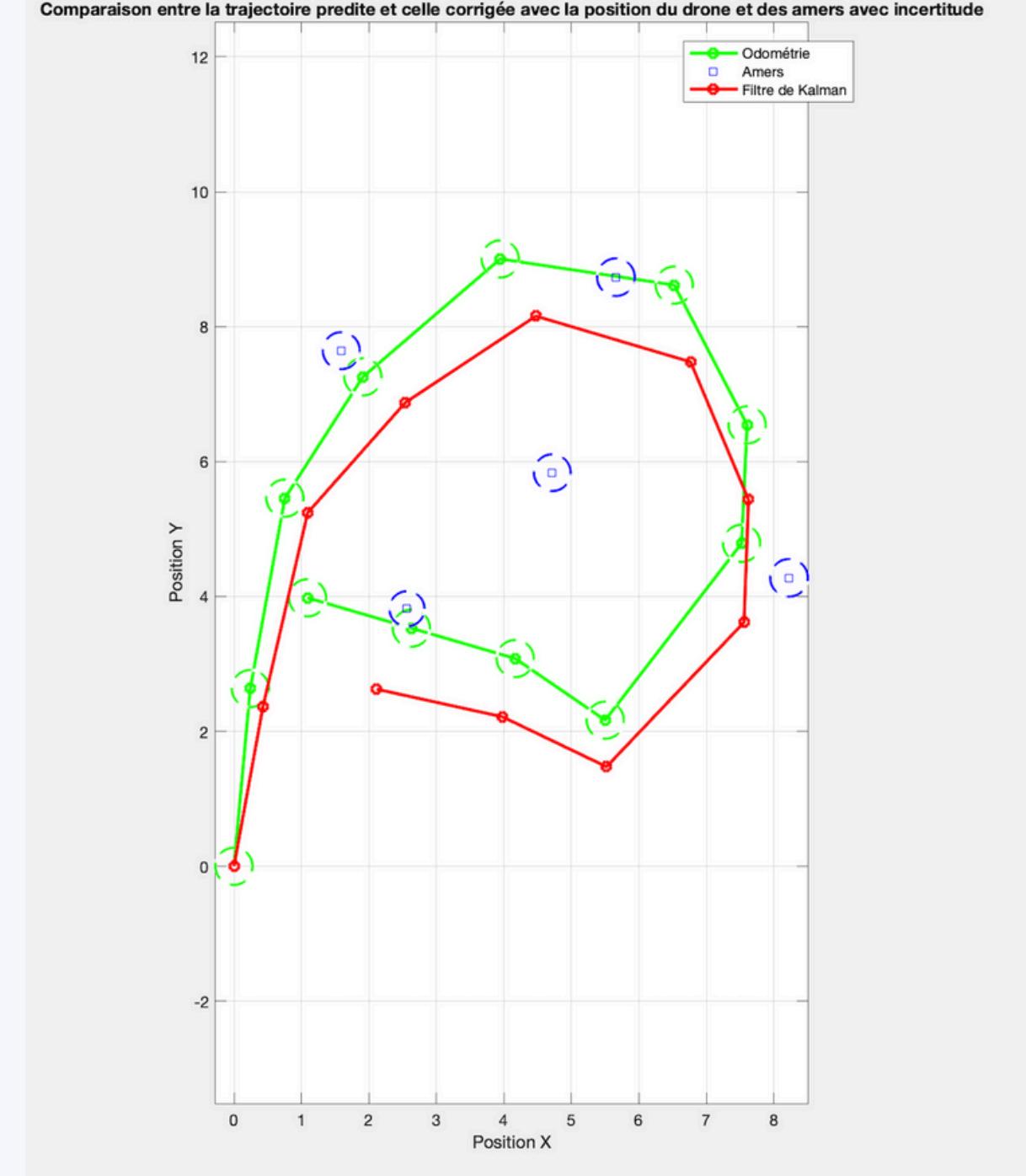
- Assimilation des notations et des différentes dimensions
- Interprétation d'un fichier de taille variable
- Nécessité de calibrage

SOLUTIONS APPORTÉES :

- Mise en place d'un système de seuil
- Calibrage selon des trajectoires de référence

POUR ALLER PLUS LOIN :

- Faire du traitement d'images
- Appliquer la méthode sur les données extraites



GLOSSAIRE

X_t Vecteur de positions absolues contenant la position du drone et de chacun des amers à l'instant t.

X_t^* Vecteur de positions prédictes non corrigées à l'instant t.

\hat{X}_t Vecteur de positions prédictes corrigées à l'instant t.

Y_t Vecteur de perceptions absolues contenant la distance relative entre le drone et chaque amer à l'instant t.

P_Y Matrice de covariance sur l'erreur des perceptions du capteur.

Y_t^* Vecteur de perceptions prédictes non corrigées à l'instant t.

\hat{Y}_t Vecteur de perceptions absolues prédictes corrigées à l'instant t.

GLOSSAIRE

u_t Vecteur d'odométrie à l'instant t.

Q Matrice d'incertitude sur l'odométrie u_t .

P_t Matrice de covariance absolue donnant l'incertitude sur les positions de X_t .

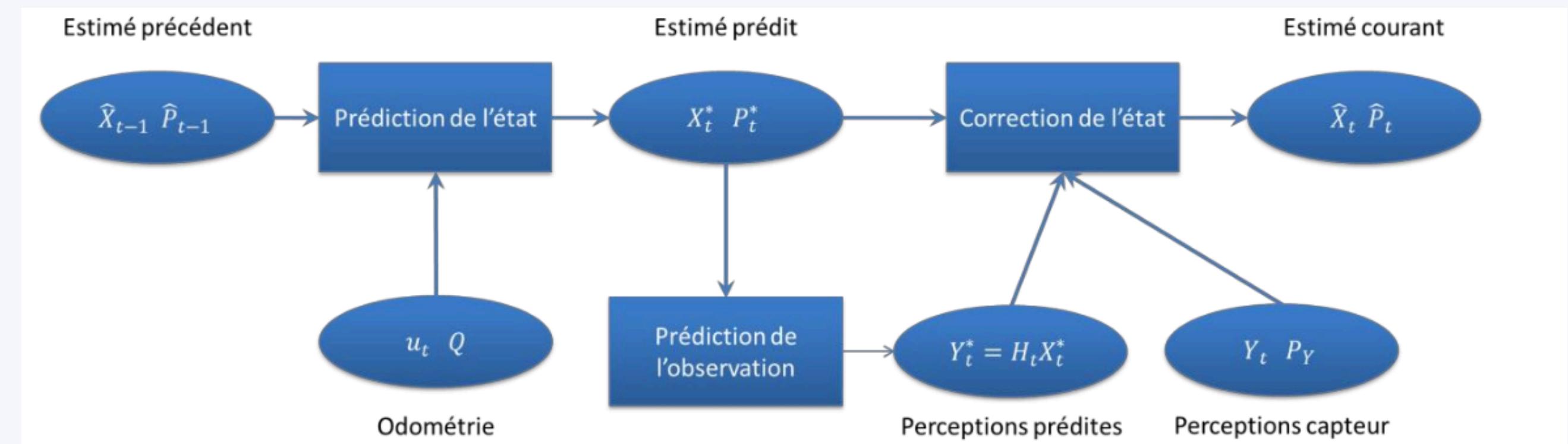
P_t^* Matrice de covariance prédictive non corrigée donnant l'incertitude sur les positions de X_t^* .

\hat{P}_t Matrice de covariance prédictive corrigée donnant l'incertitude sur les positions de \hat{X}_t .

K_t Gain de Kalman à l'instant t.

B Matrice complétant u_t par des zéros pour le calcul de X_t^* ..

Annexe



$$1. \quad X_t^* = \hat{X}_{t-1} + Bu_t$$

$$2. \quad P_t^* = \hat{P}_{t-1} + BQB^T$$

$$3. \quad Y_t^* = H_t X_t^*$$

$$4. \quad \hat{X}_t = X_t^* + K_t(Y_t - Y_t^*)$$

$$5. \quad \hat{P}_t = P_t^* - K_t H_t P_t^*$$

$$K_t = P_t^* H_t^T (H_t P_t^* H_t^T + P_Y)^{-1}$$