ÉCOLE NATIONALE SUPÉRIEURE DE TECHNIQUES AVANCÉES

TP4 : Simultaneous Localization and Mapping using Extended Kalman Filter

Auteur :
Vanessa LOPEZ

Encadrants :
M. David FILLIAT

ROB312 - Navigation pour les systèmes autonomes

14 novembre 2023

Table des matières

1 Introduction

On va appliquer le méthode de *"Simultaneous Localization and Mapping"* (SLAM) qui établit une carte d'un environnement inconnu à l'aide d'un filtre de Kalman étendu pour estimer l'état d'un robot et les positions des amers dans un environnement sur la base des observations des capteurs.

Dans un premier temps, on définira les notions *Mapping* qui est estimer la carte la plus probable, par rapport aux données perçues par le robot, dans l'espace de la représentation choisie à l'aide de certains capteurs et *Localization* qui utilise principalement le suivi de la position en découvrant l'environnement.

On met en œuvre cet algorithme sur un programme représentant la trajectoire d'un véhicule, lequel suivre les étapes suivantes. Dans un premier temps, l'estimation de la position est réalisée sur la base de la dynamique du modèle et la prédiction de la covariance. Cette dernière est représentée par l'équation suivante.

$$P_{k+1|k} = \nabla F_x \times P_{k|k} \times \nabla F_x^T + \nabla F_u \times Q_k \times \nabla F_u^T \tag{1}$$

Où, les ∇F_x et ∇F_u désignent les matrices jacobiennes des vecteur d'état par rapport au position et command.

Les estimations des amers sont ensuite calculées avec la distance de Mahanalobis pour associer ces estimations à l'un des amers. Cela permet d'ajouter un nouveau point ou d'effectuer une mise à jour de Kalman.

Si un nouveau point doit être ajouté, le vecteur d'état et la matrice de covariance sont étendus en ajoutant les informations de ce nouveau amer. En revanche, si une mise à jour de kalman est nécessaire, étant donné que l'estimation d'amer n'est pas une nouvelle référence, on modifie les équations de gain de kalman (facteur qui détermine dans quelle mesure la prédiction doit être ajustée en fonction des données observées), le vecteur d'état et la matrice de covariance comme indiqué dans les équations suivantes.

$$K_{k+1} = P_{k+1|k} \times H_{k+1}^T \times \left[H_{k+1} \times P_{k+1|k} \times H_{k+1}^T + R_k \right]^{-1} \tag{2}$$

$$\hat{X}_{k+1} = \hat{X}_k + K_k \left[Z_k - h(\hat{X}_k) \right] \tag{3}$$

$$P_k = [I - K_k \times H_k] \times P_k \tag{4}$$

Où $Z_k - h(\hat{X}_k)$ (eq 2) est l'innovation (la différence entre l'observation réelle Z_k et l'observation prédite $h(\hat{X}_k)$), $H_{k+1} \times P_{k+1|k} \times H_{k+1}^T + R_k$ (eq 1)(également noté S) est la covariance de l'innovation et H_k est la matrice jacobienne du modèle d'observation. Source : [2]

2 Influence de l'environnement

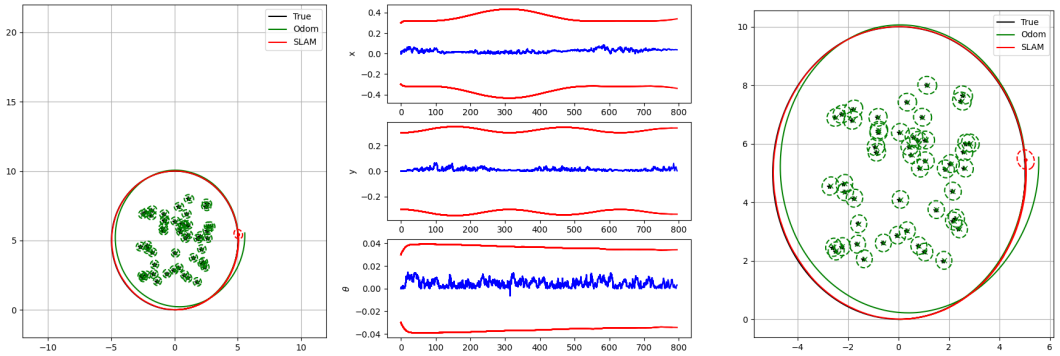
2.1 Le nombre, la position des amers et la trajectoire du robot

Étant donné que l'amer correspondant à chaque estimation est connu sans ambiguïté, lorsque les amers sont proches, la trajectoire est bonne et la covariance est faible.

2.1.1 Boucle courte et une carte dense avec de nombreux amers à l'intérieur du rayon de perception du robot

Le rayon de la trajectoire sera réduit en utilisant le signal de commande dont dispose le robot pour modéliser sa dynamique.

On peut constater qu'en contenant un grand nombre d'amers dans la boucle courte, on obtient des amers sont très proches de la trajectoire, ce qui génère un suivi aligné sur la trajectoire réelle indépendamment des perturbations de l'odométrie en montrant une erreur proche de zéro et une faible variance de l'erreur. En plus, les mesures non ambiguës de la position des amers pendant toute la trajectoire fournissent de bonnes estimations, ce qui permet au modèle d'être plus précis.



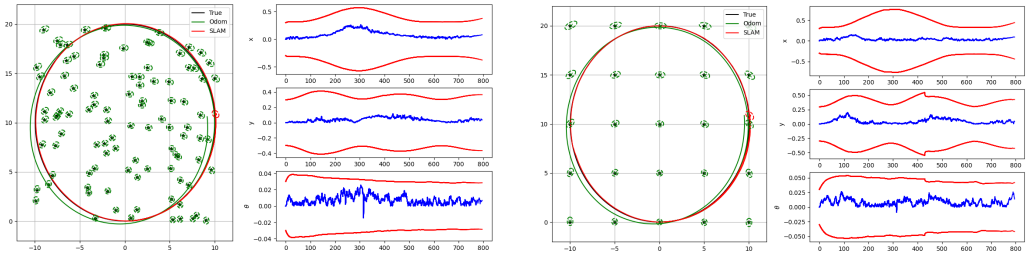
(a) Résultat (b) Zoom sur le boucle

FIGURE 1 – Boucle courte et plusieurs amers dedans le rayon

2.1.2 Longue boucle et une carte dense avec de nombreux amers tout au long de la boucle

Pour cette modification, on a évalué 50 points aléatoires tout autour de la carte (Fig. 2(a)) et quelques points uniformément répartis (Fig. 2(b)) pour illustrer leur comportement par rapport à l'espacement des amers. Dans le cas des amers distribués de manière aléatoire, une erreur proche de zéro et une faible variance ont pu être mises en évidence, étant donné que le robot a rencontré des amers très proches de sa position tout au long de la trajectoire. Au contraire, dans l'illustration des amers uniformément distribués, malgré le fait

qu'il parvienne à suivre la bonne trajectoire, une erreur un peu plus importante et une augmentation de la variance peuvent être mises en évidence parce qu'il est passé par des amers qui n'étaient pas aussi proches que dans l'autre cas au long de la trajectoire.

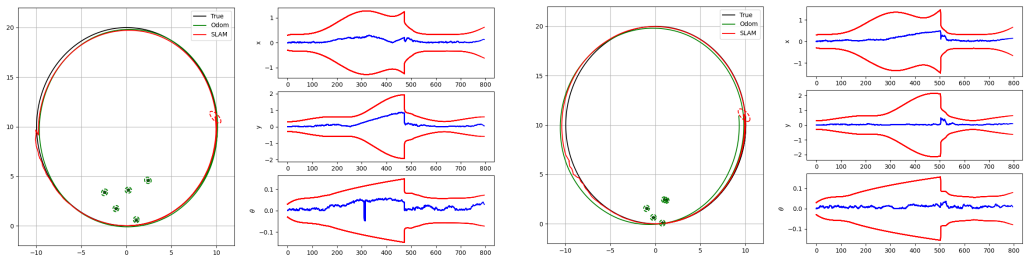


(a) Amers aléatoires (b) Amers uniformément répartis

FIGURE 2 – Longue boucle et plusieurs amers

2.1.3 Longue boucle et une carte éparse avec seulement quelques amers près de la position de départ

Pour cette modification, on a évalué 5 amers légèrement dispersés au début de la trajectoire (Fig. 3(a)) et 5 amers très proches du début (Fig. 3(b)) afin d'observer leur comportement. On peut constater que lorsqu'il n'y a pas suffisamment des amers accompagnant la trajectoire, l'algorithme doit suivre l'odométrie en raison du peu d'informations qu'il reçoit des capteurs, comme le montrent les illustrations en haut de la trajectoire circulaire. Mais lorsqu'il ferme la boucle et reçoit des informations des amers proches de l'origine, il corrige brusquement sa trajectoire, ce qui réduit considérablement l'erreur et la variance. Il est à noter que la figure 3(a) corrige sa trajectoire avant la figure 3(b) en raison de la dispersion des amers.



(a) Amers proches de l'origine (b) Amers plus proches de l'origine

FIGURE 3 – Longue boucle et amers près du départ

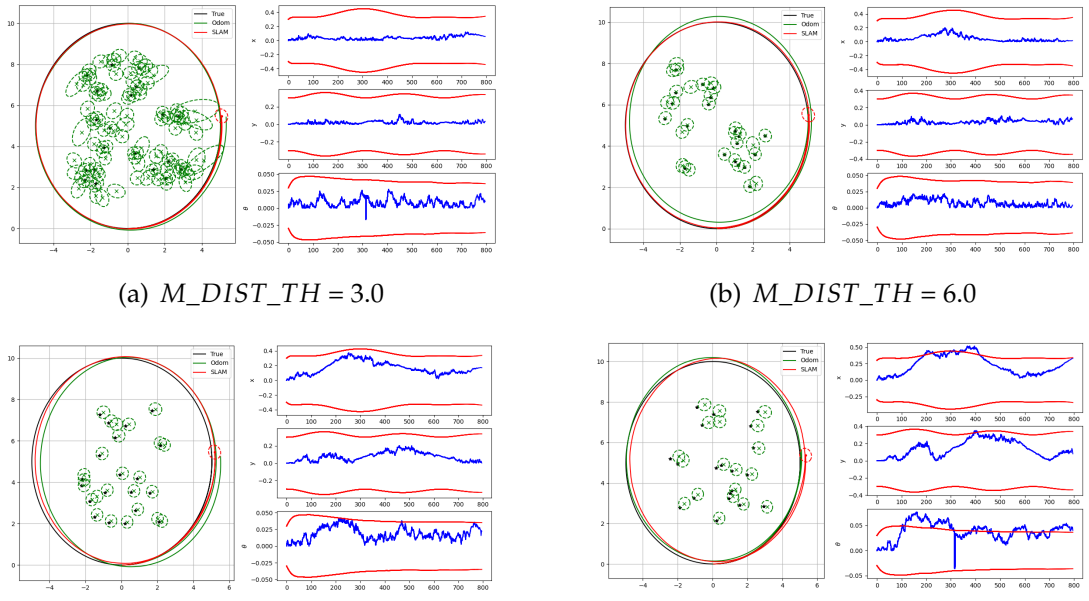
2.2 Mahalanobis distance

Etant donné que nous ne disposons pas d'une association de points amers sans ambiguïté dans la simulation, on a recours à l'utilisation de la distance de Mahalanobis. Cette distance fait référence à l'espacement entre la position d'amer et une prédiction d'amer, qui est filtrée avec le seuil qui définit l'espacement maximal qui peut être trouvé dans ces mesures. Enfin, si cette distance est inférieure au seuil, la prédiction d'amer est associée à un amer sur la carte.

2.2.1 Boucle courte et une carte dense avec de nombreux amers à l'intérieur du rayon de perception du robot

Le rayon de la trajectoire sera réduit de 0.5 en utilisant le signal de commande dont dispose le robot pour modéliser sa dynamique.

Lorsque le seuil de distance de Mahalanobis est très bas, presque aucune prédiction d'amer ne sera associée à un amer défini sur la carte, ce qui génère plusieurs "points", comme le montre la figure 4(a) produisant un grand nombre de calculs inutiles, ce qui entraîne une simulation très lente. Au contraire, si le seuil est trop élevé, différentes prédictions "éloignées" seront associées à un amer, ce qui faussera la position exacte d'amer dans la carte, comme le montre la figure 4(d).



(a) $M\_DIST\_TH = 3.0$ (b) $M\_DIST\_TH = 6.0$

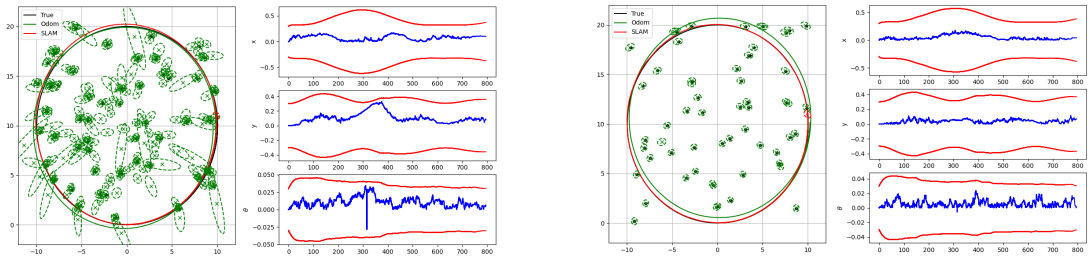
(c) $M\_DIST\_TH = 9.0$ (d) $M\_DIST\_TH = 15.0$

FIGURE 4 – Modification de seuil de distance Mahalanobis sur boucle courte et plusieurs amers dedans le rayon

On peut observer que la précision de la trajectoire diminue à mesure que le seuil de la distance de Mahalanobis augmente, car lorsque le seuil est élevé, des points très éloignés sont associés au amer, ce qui entraîne un décalage de la position sur la carte et fournit des informations erronées à la trajectoire.

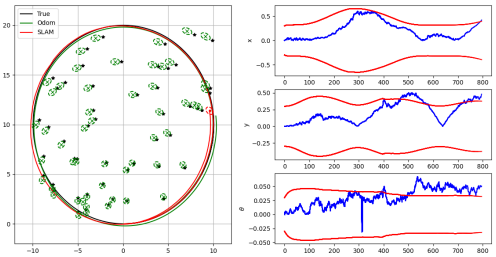
2.2.2 Longue boucle et une carte dense avec de nombreux amers tout au long de la boucle

Dans cette modification, on retrouve le même comportement de faible association à des seuils de distance de Mahalanobis très bas et de forte association à des seuils très élevés, ce qui peut entraîner un décalage des amers.

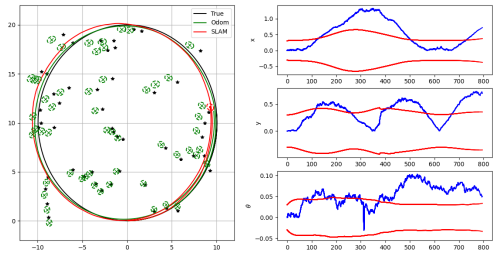


(a) $M\_DIST\_TH = 3.0$

(b) $M\_DIST\_TH = 7.0$



(c) $M\_DIST\_TH = 9.0$



(d) $M\_DIST\_TH = 15.0$

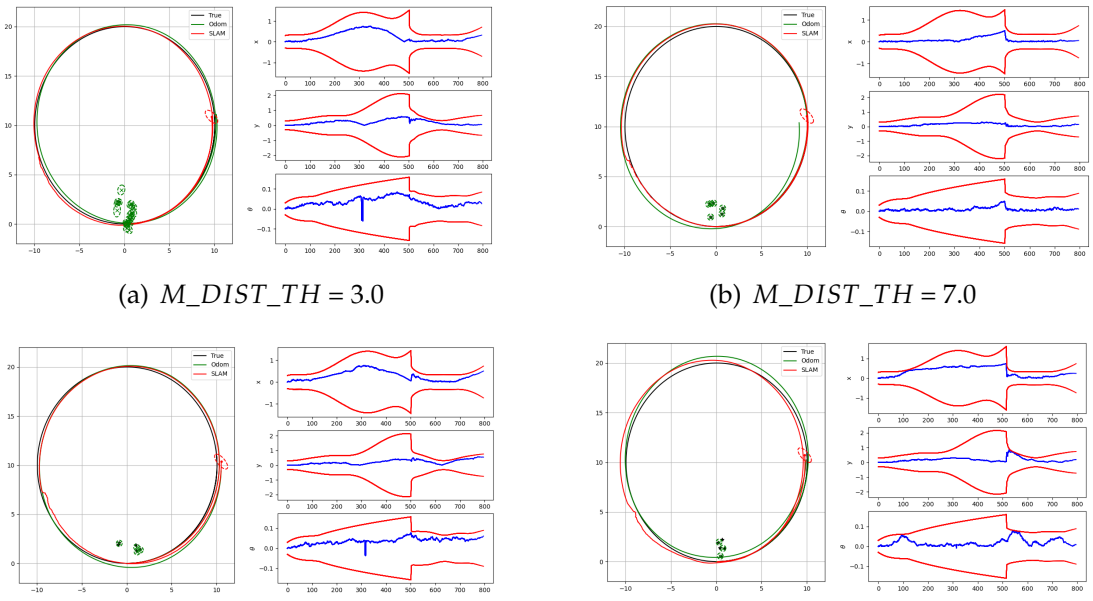
FIGURE 5 – Modification de seuil de distance Mahalanobis sur longue boucle et plusieurs amers

On peut constater que dans cette modification, l'erreur et les décalages des amers sont plus importants que dans la modification précédente, car la boucle étant plus grande, elle a tendance à accumuler les écarts causés par les mauvaises estimations.

2.2.3 Longue boucle et une carte éparse avec seulement quelques amers près de la position de départ

Cette modification permet d'obtenir les mêmes comportements en ce qui concerne l'augmentation ou la diminution du seuil de la distance de Mahalanobis, ainsi que la perte d'informations des amers le long de la trajectoire en suivant uniquement l'odométrie jusqu'à ce qu'ils détectent à nouveau un point proche de l'origine.

Dans la dernière figure 6(d), il ne suit même pas l'odométrie, puisque l'information de référence des amers d'origine est tellement déplacée, qu'il propage ce déplacement le long de la trajectoire.



(a) $M\_DIST\_TH = 3.0$ (b) $M\_DIST\_TH = 7.0$

(c) $M\_DIST\_TH = 9.0$ (d) $M\_DIST\_TH = 15.0$

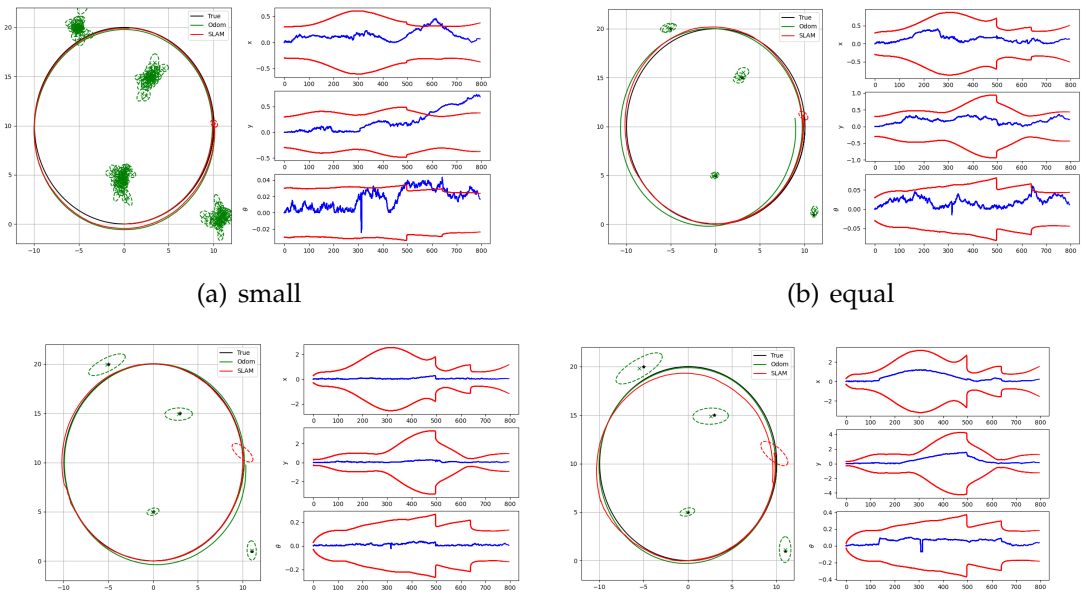
FIGURE 6 – Modification de seuil de distance Mahalanobis sur longue boucle et amers près du départ

3 Modèles probabilistes

Pour la variation de ces paramètres, le bruit estimé a été multiplié par des valeurs de 0,1, 1 et 10 pour être respectivement plus petit, moyen et plus grand que le bruit utilisé par la simulation. La variable Q_k est utilisée dans les calculs de covariance dans l'étape de prédiction de Kalman (eq 1) représentant le bruit dans le signal de commande et la variable R_k est utilisée dans les étapes de prédiction de nouveaux amers et de leurs mises à jour représentant le bruit dans les mesures.

— Modification de R_k : Si ce bruit est très petit, le gain K_k devient très grand (eq. 2), ce qui fait que dans l'eq. 3 cette variable attribue beaucoup d'importance à l'innovation en appréciant les mesures prises par le capteur amers. Par contre, s'il y a un très grand bruit, le gain de Kalman devient très petit, donnant très peu d'importance à l'innovation et ignorant presque les mesures du capteur.

— Modification de Q_k : Si ce bruit est très faible, on attribue plus d'importance au vecteur d'état des positions qu'à la direction de la commande (eq. 1).



(a) small (b) equal

(c) Bruit plus grand (10) (d) Bruit plus grand (20)

FIGURE 7 – Changement des valeurs de bruit estimées Q et Py par rapport aux valeurs utilisées pour la simulation

Small : Le filtre de Kalman sous-estime la quantité de bruit dans le système. Cela peut conduire à une confiance excessive dans l'état et les mesures estimés, ce qui peut à son tour entraîner de mauvaises performances du filtre. Plus précisément, il peut ne pas réagir assez rapidement aux changements de l'état réel, car il suppose que la plupart des changements observés sont dus au bruit. On peut constater que dans la figure 7(a), plusieurs points sont générés autour des amers définis dans la carte et que la trajectoire est centrée sur l'odométrie.

Equal : Le filtre de Kalman sera optimal, ce qui lui permettrait d'équilibrer correctement le compromis entre la confiance dans les mesures les plus récentes et la confiance dans les estimations de l'état antérieur.

Bigger : Le filtre de Kalman surestime la quantité de bruit dans le système. Le filtre peut être moins confiant dans son estimation de l'état actuel et accorder plus de poids aux nouvelles mesures. Cela pourrait permettre au filtre d'être plus réactif aux changements de l'état réel, mais cela pourrait également le rendre plus sensible au bruit dans les mesures (estimations moins stables).

4 Initialisation non retardée

Dans cette section, nous utiliserons l'algorithme SLAM avec le filtre de Kalman en utilisant uniquement la direction (angle) du capteur amers. Le principal problème est l'initialisation de nouveaux amers, car il n'est pas possible d'estimer leur position à partir d'une seule perception. Une solution possible est

l'initialisation non différée comme le montre la figure 8 qui initialise plusieurs amers dans un cône correspondant à la direction perçue, la mise à jour du point d'amer le plus proche de la vraie position après autre perception et supprime ensuite les amers inutiles au fil du temps.

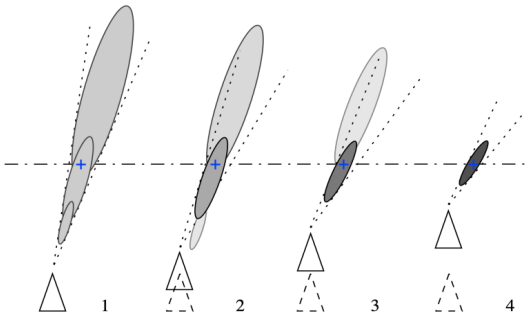
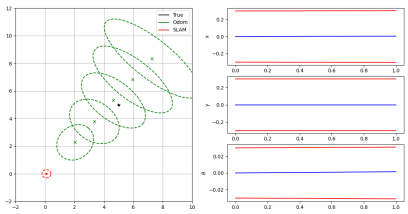
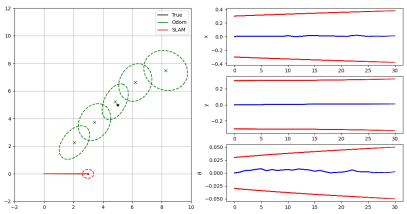


FIGURE 8 – Undelayed initialization, Source [1]

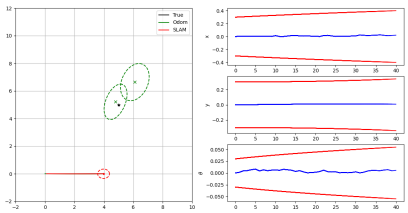
Pour ce faire, la mise à jour du filtre de Kalman est d'abord modifiée de manière à ce que le filtre de Kalman soit effectué avec des calculs de direction uniquement. Ensuite, dans le calcul de l'algorithme SLAM, lorsqu'il détecte un nouveau amer, en plus des estimations de la position du robot, la position estimée et la covariance des amers hypothétiques sont calculées. Dans les itérations suivantes, on compte le nombre de fois où chaque amer est trouvé en dessous du seuil fixé pour la distance de Mahalanobis (distance entre l'amer et chacune des prédictions) et on met à jour l'hypothèse la plus probable en supprimant les amers dont la répétabilité est inférieure à 50 % du amer le plus répété afin d'obtenir une estimation plus proche de la position réelle d'amer sur la carte.



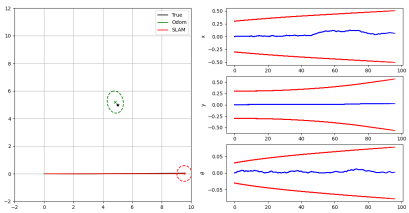
(a) Itération 1



(b) Itération 30



(c) Itération 40



(d) Itération 96

FIGURE 9 – Initialisation non retardée pour les amers

Références

[1] J. SOLA, A. MONIN, M. DEVY et T. LEMAIRE : Undelayed initialization in bearing only slam. *In 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2499–2504, 2005.

[2] Yin ZHANG, Inam ULLAH, Xin SU, Xuewu ZHANG et Dongmin CHOI : Simultaneous localization and mapping based on kalman filter and extended kalman filter. *Wireless Communications and Mobile Computing*, 2020:2138643, 2020.