Projet « NavSight » – *SLAM Collaboratif*

Projet centenaire ESTACA – 2025

Sasha Contrepois, Quentin Landon, Maxime Leclair, Clémence Sabot – SEN 2022-2025

I. Contexte

Le projet NavSight développe **une solution de navigation collaborative pour flottes de drones autonomes**, répondant aux défis majeurs de l'industrie des transports du futur. Notre technologie de localisation par SLAM s'impose comme une alternative robuste au GPS, particulièrement dans les environnements confinés ou complexes où la couverture satellite est inexistante.

Figure 1. Grands Paris site en construction.

Les applications sont multiples et stratégiques : dans le secteur ferroviaire, l'inspection des tunnels et des gares souterraines nécessite une flotte de drones coordonnés pour surveiller efficacement l'état des voies et des infrastructures critiques. Sur Mars ou la Lune, où le GPS n'existe pas, notre technologie permet à plusieurs rovers d'établir collectivement une cartographie précise et de se localiser dans des environnements inconnus, notamment lors de l'exploration des cratères polaires ou des tubes de lave souterrains. Dans les métros et complexes intermodaux, nos essaims de drones assurent une surveillance continue des installations techniques et des infrastructures.

L'ampleur de ces environnements rend impossible leur couverture par un unique drone. Aujourd'hui, la plupart de ces équipements sont contrôlés par un opérateur, rendant ces systèmes difficiles à étendre en augmentant le nombre de drones en vol simultanément. Avec NavSight, nous souhaitons rendre ces drones autonomes, permettant ainsi le vol simultané d'un nombre important de drones pour couvrir une zone plus grande dans un intervalle de temps plus court.

Pour se faire, nous nous sommes intéressés à l'architecture de ces systèmes, et avons identifié un point critique pour le déploiement à grande échelle de ces systèmes : la localisation.

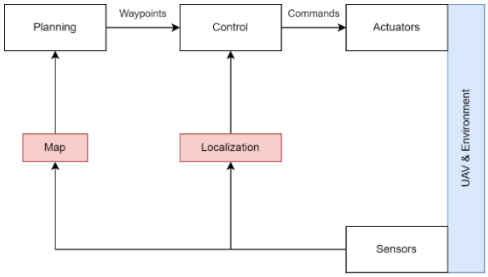


Figure 2. Architecture d'un système de drone non autonome.

Pour se localiser, le drone utilise ses capteurs et utilise une carte (locale ou globale) dans laquelle il se positionne. De plus, un retour vidéo est souvent utilisé pour donner des informations à l'opérateur pour se positionner avec précision. Cependant, dans nos cas d'utilisation, certains capteurs sont limités et nous souhaitons nous affranchir de l'opérateur dans la boucle de contrôle.

Systèmes actuels et limites

Capteurs

Inertial Navigation System (INS)

Une centrale inertielle repose sur le principe d'estimer la position relative du système en mesurant les accélérations et la vitesse angulaire du système.

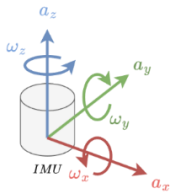


Figure 1. Représentation schématique d'un IMU : couplage d'un accéléromètre et d'un gyroscope.

$$x_{estim}(t) = \int \int \left(a_x(t) + \varepsilon(t)\right)$$

La centrale inertielle possède cependant un défaut : en intégrant l'erreur du capteur (noté $\varepsilon(t)$), l'estimation de position va diverger au fil du temps. On ne peut donc pas utiliser la centrale inertielle seule.

Généralement, la centrale inertielle est utilisée de pair avec un système de positionnement par satellite pour se recalibrer à intervalle régulier [1].

Global Navigation Satellite System (GNSS)

La technologie GNSS permet de se localiser de manière absolue sur Terre, à l'aide de communications radio transmises par au moins 3 satellites. La précision d'un tel système est de l'ordre de 10m pour les services grand public. Cette précision peut cependant être améliorée et atteindre le mètre voire la dizaine de centimètres pour les services de haute précision. Par exemple, dans les meilleures conditions (zone dégagée) le service Galileo HAS atteint une précision de 20cm horizontalement et 40cm verticalement [2].

Cependant, ce système perd toute précision dans les zones denses (ville, sites industriels avec murs/pièces métalliques) et est même inutilisable des zones souterraines ou intérieure. En effet, les ondes sont perturbées par la matière et les réflexions sur les parois [3].

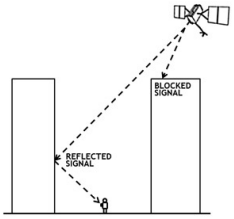


Figure 2. Réflexion du signal GPS sur une paroi dans une zone dense et blocage du signal direct vers le récepteur empêchant une bonne estimation de la position.

De plus, le GNSS est très vulnérable au brouillage et au spoofing, rendant son utilisation pour un usage militaire impossible [4].

Live video feed & Operator in the loop

Afin de contrôler un système, la meilleure manière repose sur l'intégration dans la boucle de contrôle d'un opérateur. A l'aide d'un flux vidéo en direct, l'opérateur est capable de se situer dans l'espace et d'estimer la position du drone par rapport aux obstacles.

Cependant, la transmission de flux de données peut être également perturbée en fonction du lieu d'utilisation : milieu souterrain, industriel avec parois métalliques, brouillages intentionnels …

Type	Avantages	Inconvénients
GNSS	Position absolue *Précision à 1m*	Sensible à l'environnement. Brouillage
IMU	Position relative Précis à court terme *Précis 10m < 1minute*	Drift sur le long terme
Vidéo & Opérateur	Bonne estimation Position absolue et relative Coût du capteur	Transmission sensible à l'environnement Brouillage Risque humain Coût de l'opérateur

Cartographie

La cartographie, élément essentiel de la localisation pour un système repose le plus souvent sur des maps cloud (eHorizon pour l'automobile). Celle-ci ne sont souvent pas actualisé et même indisponible pour des lieux privés, ainsi difficile de s'y localiser en temps réel.

II. Solution NavSight

NavSight s'affranchit des limites évoquées ci-dessus en utilisant un système de *Simultaneous Localisation and Mapping* – ou *SLAM*. Grâce à ce système, le drone est capable à la fois d'établir une carte 3D de son environnement et s'y localiser.

L'un des critères de réussite est d'être capable de fournir une information de localisation en zone dense ou en intérieur avec une précision équivalente au système Galileo HAS : 95% des valeurs à ±20cm de la valeur réelle.

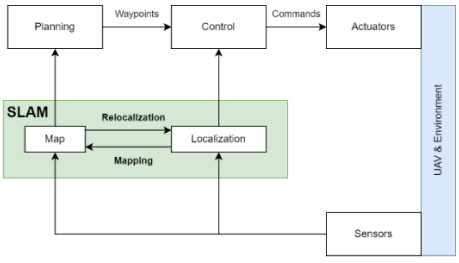


Figure 3. Architecture d'un drone autonome avec le SLAM pour se localiser et cartographier

Il existe divers types de SLAM, reposant sur divers types de capteurs : Lidar, Caméra mono, Caméra stéréo, Caméra RGB avec profondeur …

Nous souhaitons être capable de développer notre solution sur un essaim, avec un nombre élevé de drone dans le but de couvrir une large surface d'opération. Afin de rendre notre système compétitif, il se doit d'être le plus économique possible. Ainsi, nous nous sommes tournés vers le système SLAM le moins cher sur le marché : SLAM Monoculaire.

Hypothèses - Grand Paris Express (200km de tunnels)

- Surface à couvrir : 200km de tunnels
- Autonomie drone : 30 minutes
- Vitesse moyenne d'inspection : 5km/h

Besoin de 20 drones pour une couverture journalière complète.

Solution	Coût capteur/drone	Coût total flotte (20 drones)
Velodyne Puck (VLP-16)	4 000€	80 000€
GPS RTK + Base	2 000€	40 000€ + 10 000€ (bases)
NavSight (Caméra mono)	200€	4 000€

Note : Prix des capteurs seuls, hors coûts d'intégration et infrastructure

Figure 5. Étude de coût capteur de localisation

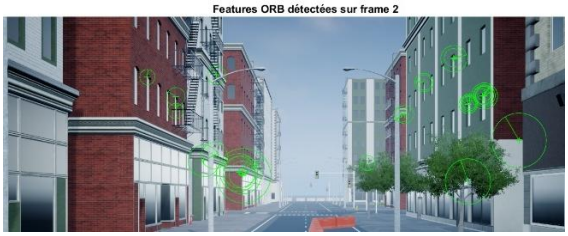
Principe du SLAM visuel monoculaire

Le SLAM construit une carte et se localise à l'intérieur de celle-ci avec une suite d'étapes que l'on appelle « pipeline ». Nous utilisons un ORB-SLAM [5], reconnu pour ses performances temps réel. Le pipeline se décompose de la manière suivante :

1. Acquisition de l'image par le drone
2. Feature extraction (landmarks)
3. Tracking (Perspective-n-Point, Bundle adjustment)
4. Local mapping
5. Loop closure

Feature extraction

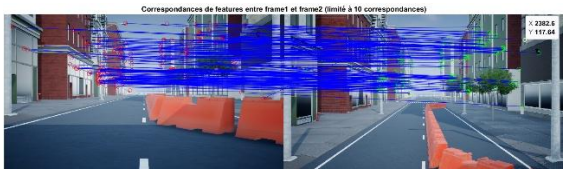
L'extraction des caractéristiques est la première étape du pipeline SLAM monoculaire. Elle consiste à identifier des points clés dans l'image capturée par la caméra monoculaire. Ces points, appelés « features », doivent être suffisamment distinctifs pour être facilement reconnus d'une image à l'autre (corner) Cette étape est cruciale pour garantir une bonne précision dans les étapes suivantes.



Des algorithmes tels que SIFT (Scale-Invariant Feature Transform) ou ORB [6] (Oriented FAST and Rotated BRIEF) sont souvent utilisés. Les features extraites doivent être robustes face aux changements d'éclairage, d'angle de vue ou d'échelle. Cela permet de nous assurer par exemple qu'une feature observée par un drone en milieu de journée (forte luminosité) sera reconnue par un autre drone en fin de journée (faible luminosité).

Tracking

Le tracking est une opération qui s'effectue entre deux images pour estimer le déplacement de la caméra entre ces dernières. Tout d'abord, on identifie les features communes aux deux images.



Le tracking suit l'évolution des points clés identifiés dans plusieurs images successives. L'algorithme Perspective-n-Point (PnP) permet d'estimer la position et l'orientation de la caméra en utilisant les coordonnées des features dans le référentiel global.

Pour affiner cette estimation, le Bundle Adjustment est utilisé. Cet algorithme d'optimisation ajuste simultanément les poses de la caméra et les positions des points clés afin de minimiser les erreurs de reprojection. Cela permet de stabiliser la trajectoire estimée du drone et de renforcer la précision du SLAM monoculaire.

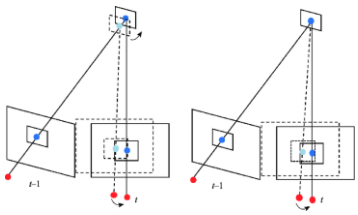


Figure 4. Exemple de bundle adjustment, dans la seconde image, la projection du point dans l'espace est utilisée pour optimiser l'estimation du déplacement entre les deux images.

Local mapping

Cette étape vise à construire une carte locale en trois dimensions autour de la position actuelle du drone. Les features suivies dans les images précédentes sont utilisées pour créer une représentation locale de l'environnement. Cette carte est souvent mise à jour en temps réel, permettant au drone de s'adapter aux nouvelles informations capturées par la caméra. La cartographie locale est essentielle pour détecter les changements de l'environnement et éviter les obstacles.

Loop closure

La fermeture de boucle est une étape cruciale pour corriger les dérives accumulées dans les estimations de position. Lorsqu'un drone reconnaît une zone précédemment visitée (grâce à la détection de features déjà cartographiées), il peut recalibrer sa position en comparant les poses actuelles et passées. Cette étape permet de minimiser les erreurs accumulées au fil du temps et d'améliorer la précision globale de la carte.

Elle est particulièrement importante dans les environnements complexes ou les trajectoires en boucle.

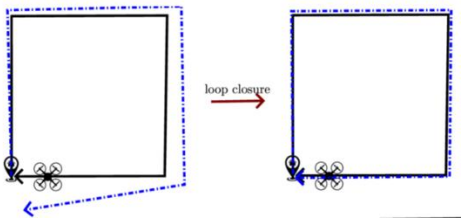


Figure 6. Exemple de Loop closure, Groundtruth (e ıated path (en bleu).

Mise en œuvre

L'équipe NavSight est composée de 4 personnes, divisée en deux équipes :

La première équipe est chargée de mettre en place un environnement numérique sur lequel l'ensemble des algorithmes seront testés. Cela comprend la conception d'un jumeau numérique d'un quadricoptère, la simulation de l'environnement en 3D, l'algorithme d'asservissement et la simulation de l'ensemble des capteurs.

La seconde équipe est chargée de mettre en place un algorithme de SLAM monoculaire collaboratif en premier lieu sur le système simulé puis le déployer sur un système réel. L'objectif principal est d'estimer la précision du système et analyser les cas limites.

Roadmap

Phase 1 : Simulation

- Environnement de test avec drone numérique
- Simulation 3D et capteurs
- Tests des algorithmes d'asservissement

Phase 2 : SLAM Collaboratif Initial (v0)

- Agent Master : Cartographie d'une Zone.
- Serveur : Génération de la map via Visual inertial SLAM (offline).
- Agent 2 : Navigation avec relocalisation en se basant sur la carte générée. (real time)
- Validation sur divers scénarios (luminosité, trajectoires, obstacles)

Phase 3 : SLAM Collaboratif Avancé (v0.1)

- Positionnement inter-drones (Aruco)
- Relocalisation améliorée

Objectif : Précision ±40cm (95% du temps)

III. Simulation

Nous développons un jumeau numérique du quadrirotor sous MATLAB/Simulink pour analyser ses performances avant expérimentation. Ce modèle simule sa dynamique, teste divers scénarios et optimise le contrôleur.

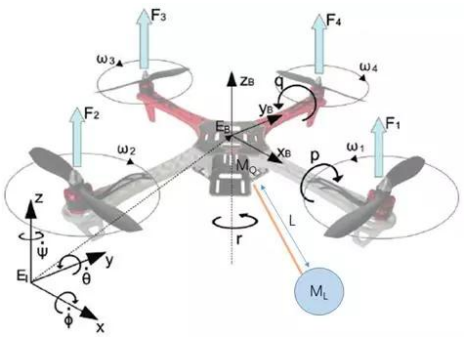
La modélisation en espace d'état permet un contrôle LQR pour la stabilisation et le suivi de trajectoire. Un filtre de Kalman fusionne les données capteurs et modèle dynamique pour affiner l'estimation des états. Enfin, la visualisation 3D via Simulink 3D Animation facilite l'interprétation des résultats et la validation du modèle.

Quadrirotor et Contrôle

Un quadrirotor, drone à quatre rotors, assure un contrôle précis grâce à ses moteurs et capteurs embarqués ajustant orientation et trajectoire en temps réel. Sa dynamique instable nécessite un contrôle avancé, d'où l'utilisation d'un régulateur LQR basé sur un modèle linéarisé pour optimiser la stabilisation et le suivi de trajectoire.

Modèle Mathématique

Le quadrirotor possède six degrés de liberté : trois translations et trois rotations. Sa stabilité repose sur la rotation opposée des paires de rotors, compensant les couples réactifs. Il est modélisé comme un corps rigide et symétrique, sans effets de sol, facilitant la conception des contrôleurs pour ses déplacements et rotations.



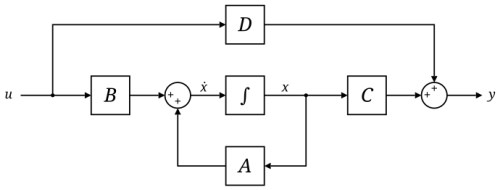
Modèle d'État et Filtrage de Kalman

La modélisation par systèmes d'états permet d'anticiper les mouvements du quadrirotor tout en compensant perturbations et incertitudes pour un contrôle fiable. Le retour d'état assure une stabilité optimale face aux aléas du vol.

Nous utilisons un régulateur quadratique linéaire (LQR) pour stabiliser et suivre la trajectoire du drone, ainsi qu'un filtre de Kalman pour améliorer l'estimation des états. Ce filtre fusionne les mesures bruitées des capteurs avec la prédiction du modèle, garantissant une estimation plus précise des variables du système.

Modélisation en espace d'état

Un système d'état est une représentation mathématique d'un système dynamique qui décrit son évolution à l'aide d'un vecteur d'état regroupant les variables internes du système. Il est modélisé sous la forme d'un ensemble d'équations différentielles.



Le vecteur d'état est défini comme :

$$x = [\, x\ y\ z\ \phi\ \theta\ \psi\ u\ v\ w\ p\ q\ r\ ]^T$$

- $\phi\ \theta\ \psi$: angles de roulis, de tangage et de lacet.
- $p\ q\ r$: vitesses angulaires autour des axes.
- $u\ v\ w$: vitesses linéaires dans le référentiel du drone.
- $x\ y\ z$: positions dans le référentiel inertiel.

Pour faciliter la conception du contrôle et du filtre de Kalman, le système est linéarisé autour d'un point d'équilibre. Le modèle d'espace d'état s'écrit :

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases}$$

- A décrit la dynamique du système.
- B relie les entrées de contrôle aux dérivées d'état.
- C=I (identité), supposant tous les états mesurables.
- D=0, absence de lien direct entre entrées et sorties.

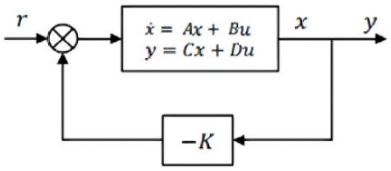
Les matrices A et B sont définies comme suit :

$$A = \begin{bmatrix} 0&0&0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&-g&0&0&0&0&0&0&0 \\ 0&0&0&g&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix} \text{ et } B = \begin{bmatrix} 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \\ 0&0&0&0 \\ \frac{1}{m}&0&0&0 \\ 0&\frac{1}{I_x}&0&0 \\ 0&0&\frac{1}{I_y}&0 \\ 0&0&0&\frac{1}{I_z} \end{bmatrix}$$

Ix, Iy, Iz sont les moments d'inerties.

Régulateur Quadratique Linéaire

Le contrôleur LQR optimise le contrôle des systèmes linéarisés, comme le quadrirotor, en minimisant une fonction de coût prédéfinie. Le contrôleur LQR est calculé en déterminant un gain optimal K tel que : $u = -Kx$



L'approche LQR stabilise efficacement le quadrirotor en vol stationnaire en minimisant une fonction de coût. Elle assure une stabilité optimale avec une consommation énergétique réduite et convient aux systèmes MIMO en temps réel grâce à son efficacité de calcul.

Filtrage de Kalman

Le filtre de Kalman [7] fusionne les données de l'IMU et du SLAM pour améliorer l'estimation des états du quadrirotor, compensant les dérives de l'IMU et corrigeant les erreurs du SLAM.

Prédiction

À chaque itération, une prédiction de l'état futur est effectuée à partir du modèle :

$$x_{k|k-1} = A\hat{x}_{k-1|k-1} + Bu_k$$

- $x_{k|k-1}$ Est la prédiction de l'état au pas k, basée sur l'état estimé au pas k−1.
- $A\hat{x}_{k-1|k-1}$ Représente l'évolution de l'état par le modèle.
- Bu_k Est l'effet des commandes appliquées.

Correction

Lorsque de nouvelles mesures y_k sont disponibles, elles sont utilisées pour corriger l'estimation avec la mise à jour suivante :

Calcul du gain de Kalman :

$$K_k = P_{k|k-1}C^T(CP_{k|k-1}C^T + R)^{-1}$$

Où K_k est le gain de Kalman et P est la matrice de covariance de l'incertitude sur l'état.

Correction de l'état estimé :

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - C\hat{x}_{k|k-1})$$

$(y_k - C\hat{x}_{k|k-1})$ représente la différence entre la mesure réelle et la prédiction du modèle.

IV. SLAM v0 – Collaboration asynchrone

Dans cette première version, nous implémentons une collaboration asynchrone entre deux agents :

Le premier agent (A) effectue une « reconnaissance » de la zone à l'aide d'une caméra monoculaire. Cet agent est contrôlé par un opérateur à distance.

Le flux vidéo est transféré au serveur. Ce dernier, plus puissant, effectue la partie back-end du SLAM de manière hors-ligne (bundle-adjustment, loop-closures, optimisation de la carte). Une fois cette opération terminée, le serveur possède alors une carte de la zone patrouillée par l'agent A, avec les keypoints et leur position estimée dans l'espace.

Cette carte est ensuite envoyée à l'agent B. Lors de son vol, il peut alors se positionner avec de l'odométrie visuelle (tracking) mais il est également capable de repérer des key points vu au préalable par l'agent A pour se repositionner (Relocalisation globale ou locale).

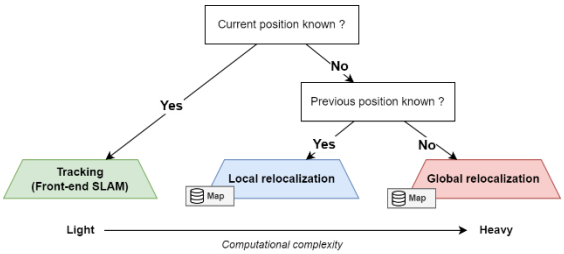


Figure 7. Relocalisation d'un agent à partir de la map générée.

Cette technique permet à des agents avec une faible puissance de calcul et uniquement une caméra de pouvoir se localiser en temps réel. Ils peuvent effectuer des missions dans cette map connu généré par le master.

V. Résultats

On veut vérifier les impacts qu'ont différents changements entre le passage du premier agent et du second. Nous utilisons comme métriques de performances la RMSE (Root Mean Square Error):

$$RMSE\ (Root\ mean\ square\ error)\ = \sqrt{(\Sigma(\hat{x_i} - x_i)^2 / n)}$$

Résultat en simulation

Dans la simulation, on partage une carte réalisée au préalable à deux drones de l'essaim. On leur fait suivre une trajectoire et on compare l'estimation de position avec la position réelle.

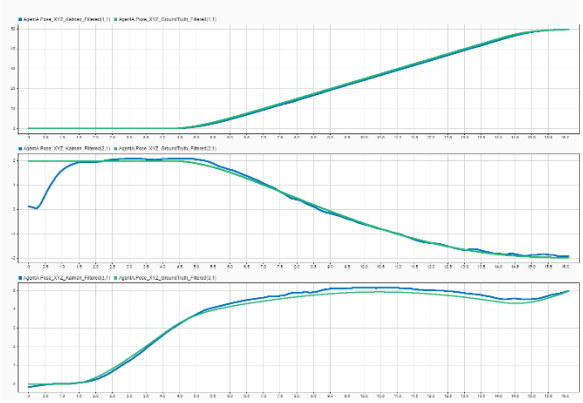


Figure 5. Position estimée (Bleu) et Position réelle (Vert) en simulation sur un trajet de 50 mètres, avec deux drones.

Métriques de performances :

Métrique	Global	X	Y	Z
RMSE (m)	0.33	0.54	0.08	0.18

Drone réel

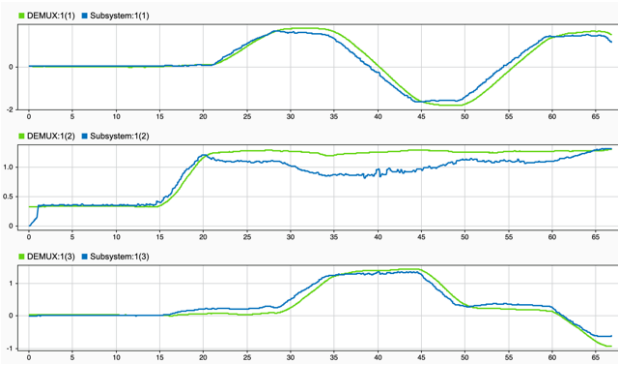


Figure 9. Position estimée (Bleu) et Position réelle (Vert) dans la volière

Métriques de performances :

Métrique	Global	X	Y	Z
RMSE (m)	0.20	0.20	0.24	0.16

Conclusion et améliorations

Les résultats sont très satisfaisants. En effet, grâce à cette solution de SLAM visuelle, nous arrivons à estimer la position des drones avec une erreur globale inférieur à 40cm.

Si l'environnement change ou la luminosité change, on observe une dégradation de l'estimation de position.

D'après ces résultats sur la première version, nous allons continuer à développer une solution plus performante :

- Construction de la carte en temps réel
- Fusion de carte en temps réel
- SLAM synchrone
- Comportement d'essaim autonome

VI. Références

[1] . M. S. Grewal , L. R. Weill et A. P. Andrews, Global Positioning Systems, Inertial Navigation, and Integration, 2007.

[2] GALILEO GNSS, «Galileo High Accuracy Service (HAS) - Info Note,» 2020.

[3] GPS.gov, «GPS Accuracy,» [En ligne]. Available: https://www.gps.gov/systems/gps/performance/accuracy/.

[4] K. Rados, M. Brkić et D. Begušić, «Recent Advances on Jamming and Spoofing Detection in GNSS».

[5] R. Mur-Artal, J. M. M. Montiel et J. D. Tardós, «ORB-SLAM: A Versatile and Accurate Monocular SLAM System,» *IEEE Transactions on Robotics,* vol. 31, n° %15, pp. 1147 - 1163, 2015.

[6] E. Rublee, V. Rabaud, K. Konolige et G. Bradski, «ORB: an efficient alternative to SIFT or SURF,» chez *2011 International Conference on Computer Vision*, Barcelona, Spain, 2011.

[7] R. E. Kalman, «A New Approach to Linear Filtering and Prediction Problems,» *ASME Journal of Basic Engineering,* vol. 82, pp. 35-45, 1960.