Acquisition photogrammétrique autonome avec le robot Warthog

Rapport de stage assistant ingénieur à l'ENSTA Bretagne

Maxime LEGEAY

Remerciements

J'aimerais remercier dans un premier temps Fabrice Le Bars et Benoit Zerr qui m'ont accueilli au sein du laboratoire de recherche de l'ENSTA Bretagne pour ce stage, et pour un suivi régulier de notre avancement sur le projet. Je remercie également Marie Ponchart pour son aide en photogrammétrie et Robin Sanchez pour son partage d'expérience avec le robot Warthog, qui nous ont tous les deux permis de nous lancer rapidement dans le projet.

Table des matières

Introduction

J'ai pu effectuer un stage au laboratoire de robotique de l'ENSTA Bretagne (2 rue François Verny, www.ensta-bretagne.fr), dont l'objectif était de faire une acquisition photogrammétrique autonome des bâtiments de l'établissement. Nous nous sommes penchés sur cette problématique pendant 12 semaines avec Isaac-Andreï WITT et Enzo-Loid Mohamed ESSONO AUBAME.

La photogrammétrie consiste à reproduire un modèle numérique 3D d'une scène à partir de photographies prises sous plusieurs angles. Les chercheurs et étudiants de l'ENSTA Bretagne se penchent régulièrement sur ce sujet, notamment ceux des branches hydrographie et robotique autonome. Ainsi nous avons pu échanger en début de stage avec Marie Ponchart, diplômée de l'ENSTA Bretagne en 2018 et poursuivant ses études après une spécialisation en hydrographie, qui nous a transmis ses connaissances en photogrammétrie et nous a fait comprendre les enjeux de notre stage. En parallèle, nous avons dû prendre en main le robot Warthog, un UGV (Unmanned Ground Vehicle ou véhicule terrestre inhabité) autonome polyvalent qui nous a servis de support pour nos caméras. Ainsi nous avons beaucoup échangé avec Robin Sanchez, étudiant en robotique qui effectuait son projet de fin d'étude au laboratoire de l'ENSTA Bretagne. Il nous a également transmis ses connaissances concernant le robot Warthog, et nous sommes restés en contact avec lui tout au long du stage puisqu'il connait parfaitement ce robot, dont il a refait l'électronique embarquée avec l'aide de Fabrice Le Bars.

Photogrammétrie

L'importance d'un bon logiciel

Nous avons commencé, en suivant les conseils de Marie Ponchart, par utiliser le logiciel MicMac développé par l'ENSG (https://micmac.ensg.eu/index.php/Accueil). Nous avons avancé avec Enzo-Loïd et Isaac à tâtons, puis avons très vite abandonné ce logiciel pour plusieurs raisons : il est très peu pratique et peu ergonomique, comporte des bugs et demande des ressources importantes pour effectuer des calculs particulièrement longs. Nous avons ensuite assez rapidement basculé sur Colmap (https://colmap.github.io/) après avoir essayé une dizaine de logiciels de photogrammétrie. Nous nous sommes alors penchés sur la théorie qu'on peut trouver derrière ces logiciels et pourquoi c'est si difficile à mettre en place. [1] L'algorithme procède en plusieurs étapes :

- La recherche d'un (ou plusieurs) point d'intérêt dans une photographie qui peut se retrouver facilement sur d'autres photos du même objet prises sous différents angles. Cela peut être un changement de texture, une couleur vive, un mouvement en particulier, etc.
- Le calcul de la parallaxe, c'est-à-dire des angles et distances qui mesurent le changement du point d'observation de l'image. On peut ensuite donner l'orientation des images capturés

- La reconstitution de l’objet 3D à partir des images et des angles de vues.

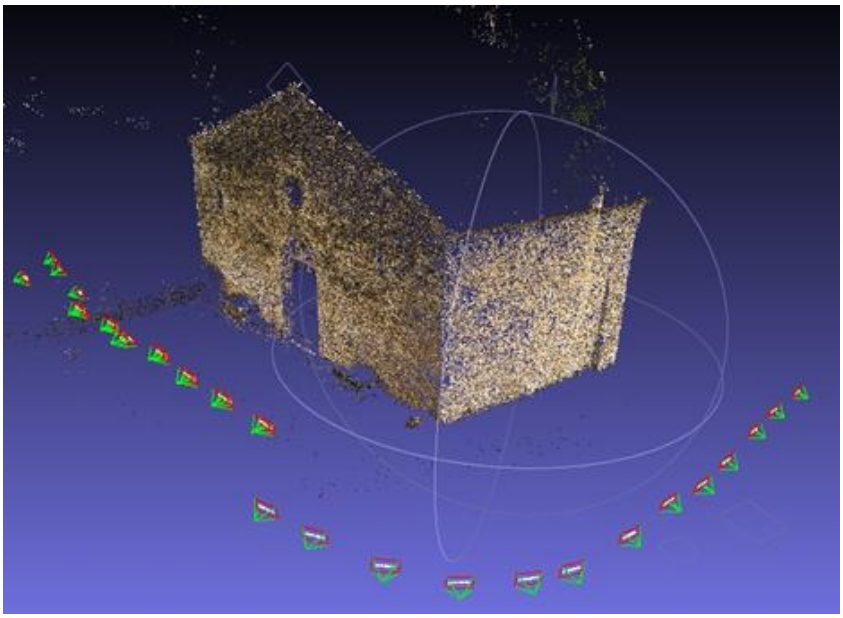

Figure 1 - Exemple d'une reconstitution photogrammétrique 3D d'une maison

On peut voir sur cet exemple la position réelle des caméras (repères verts) et la reconstitution de deux murs d’une maison dont on a pris plusieurs photographies. Il s’est dégagé de cet exemple une contrainte très forte et déterminante pour toute la suite de ce stage : les angles de vus des photos doivent être suffisamment riches pour pouvoir reconstituer un bâtiment en 3D et donc la problématique suivante a naturellement fait son apparition : comment obtenir des photographies satisfaisantes pour réaliser une maquette en photogrammétrie de façon autonome ?

Les contraintes photographiques

Nous nous sommes retrouvés assez rapidement confrontés à des contraintes importantes sur le protocole de prise des photos. Le premier fut la nécessité d'une grande richesse des angles de prise de vue. Lorsque le robot avance en ligne droite et prend des photos sur les trois caméras GoPro, même si celles-ci sont orientées différemment (avec un angle constant), le logiciel ne trouve pas forcément de points d'intérêts exploitables et ne fait parfois pas le lien entre deux photos d'une même façade. Ce problème laissait alors des « trous » dans la reconstitution 3D.

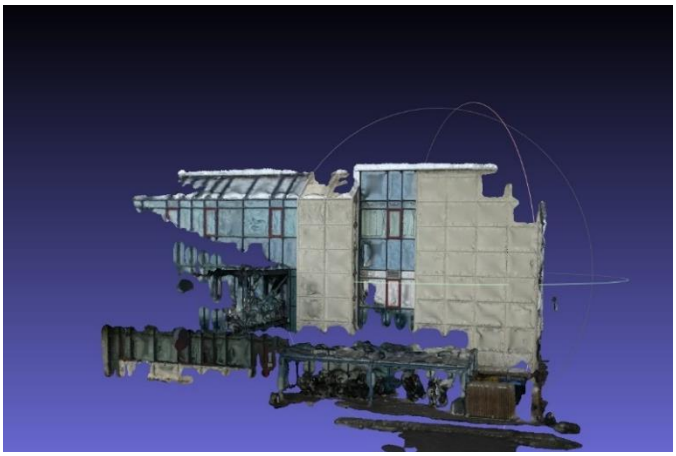

Figure 2 - Reconstitution de la façade sud du batiment E, avec manque de photos

On se rend compte sur cette image que la richesse des angles de prises de vues est fondamentale pour pouvoir avoir une reconstitution fidèle des bâtiments.

Présentation du robot Warthog

Nous avons utilisé le robot Warthog, un véhicule autonome terrestre de 280kg, fabriqué par Clearpath Robotics mais dont l'électronique embarquée a été refaite par des étudiants roboticiens FISE 2021 et leurs encadrants. Il possède deux moteurs électriques contrôlant respectivement les roues droites et gauches du robot. Il est alimenté par douze batteries 12V en série et il est équipé de plusieurs capteurs montés sur des profilés métalliques et facilement modulables :

- Trois sonars acoustiques à l'avant
- Un Lidar
- Un GPS
- Une centrale inertielle
- Deux encodeurs respectivement sur les roues droites et gauches.

Nous avons rajouté à cette installation des composants nécessaires pour notre projet :

- Trois caméras GoPro
- Une carte Raspberry Pi 3 alimentée par une batterie
- Deux GPS reliés une carte ArduSimple pour l'acquisition des données.

Figure 3 - Installation des caméras et des GPS pour la photogrammétrie

Le robot doit être manœuvré par deux utilisateurs : un chargé de la sécurité, déverrouille les quatre coupe-circuits présents sur le robot et actionne la télécommande coupe-circuit qu'il garde en main en cas de problème. Le deuxième utilisateur peut contrôler le robot manuellement à l'aide d'une télécommande dont les données sont réceptionnées par une carte arduino sur le robot, ou bien se connecter à ce dernier et lui donner une mission à accomplir en mode autonome.

Acquisition autonome

La trajectoire à suivre pour l'acquisition des photos

Le contrôle du robot Warthog se fait sous ROS, et tous les codes sont écrits en C++ alors nous avons dû prendre du temps pour nous familiariser avec les algorithmes qui sont déjà dessus. Ce robot se contrôle en mode « char », c'est-à-dire qu'un différentiel de vitesse entre les roues de droite et de gauche lui permet de se diriger, il ne possède pas d'essieu permettant de contrôler l'orientation des roues. Le programme de contrôle mis en place dans le robot suit exactement le modèle de la voiture de Dubins que nous connaissons bien. Nous n'avons pas touché aux nœuds ROS du robot qui permettent entre autres de récupérer les données capteurs et d'envoyer la commande aux moteurs. Nous avons principalement touché aux algorithmes de calcul de la trajectoire afin de faire bouger le robot comme on le souhaite.

Afin d'obtenir des photos sous tous les angles, nous avons pensé que le plus simple était de faire suivre au robot une trajectoire en « zig-zag », représentée par une sinusoïde. Ainsi le robot devait suivre une trajectoire de la forme $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} t \\ A*\sin(\omega*t+\varphi) \end{pmatrix}$ avec A, ω et φ des coefficients à définir et t un paramètre. Il s'agissait ensuite de « tourner » cette sinusoïde dans l'espace pour qu'elle puisse relier deux points, un point de départ et un point d'arrivée.

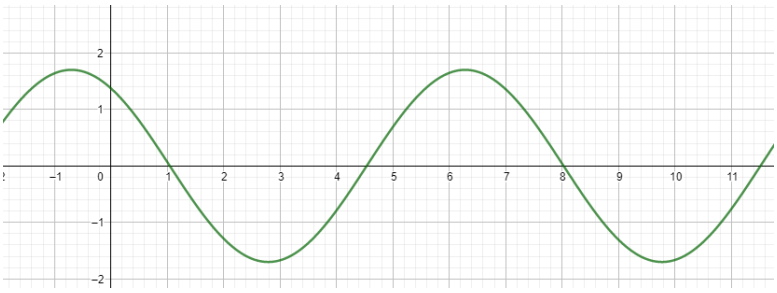

Figure 4 - Fonction sinus avec des paramètres arbitraires

Ensuite, pour obtenir cette trajectoire en sinus entre deux points, il a suffi de le tourner dans l'espace, puis d'ajouter les coordonnées du point de départ. Prenons un départ au point $A\begin{pmatrix} x_A \\ y_A \end{pmatrix}$, et une arrivée au point $B\begin{pmatrix} x_B \\ y_B \end{pmatrix}$. Alors on peut noter $\theta = arctan2\,(\overrightarrow{AB})$ l'angle formé entre le vecteur AB et l'axe horizontal. Ainsi, on veut tourner notre sinusoïde de cet angle θ et la faire partir au point A, on peut alors écrire $\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_A \\ y_A \end{pmatrix} + \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} . \begin{pmatrix} t \\ A*\sin(\omega*t+\varphi) \end{pmatrix}$ et obtenir une courbe de la forme :

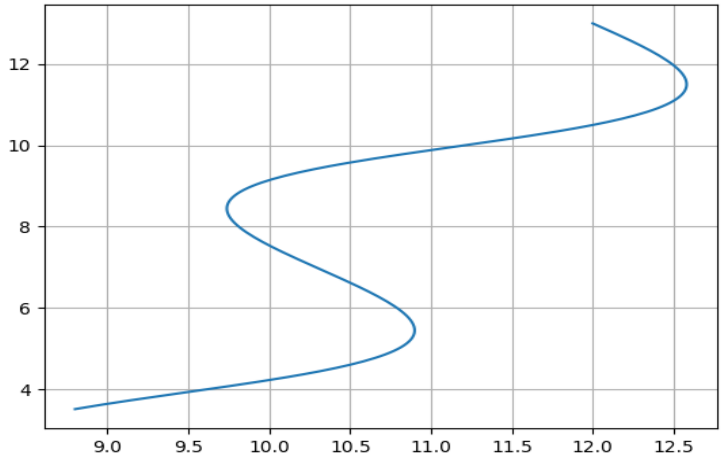

Figure 5 - Fonction sinus tournée d'un angle arbitraire

Pour l'implémenter, connaissant les équations, j'ai d'abord imaginé une méthode par bouclage linéarisant [2] permettant de calculer le cap désiré. Il n'était pas très difficile à mettre en place, un des blocages majeurs était le calcul matriciel en C++ que je ne maitrisais pas bien,

j'ai dû alors développer les équations et les écrire ligne par ligne (les matrices étant de taille 2x2, c'était faisable mais cela a entrainé quelques erreurs de calcul qu'il a fallu vérifier plusieurs fois par la suite). Cependant, nous nous sommes retrouvés confrontés à des problèmes de singularité pour l'angle $\theta = \frac{\pi}{2}$, c'est-à-dire une trajectoire en sinus suivant l'axe vertical.

Nous n'avons pas réussi à trouver de solution pratique à ce problème alors nous avons décidé de suivre une autre approche, moins jolie au niveau théorique mais beaucoup plus fonctionnelle : le suivi de points. J'ai mis en place un programme qui calcule progressivement des points que le robot doit atteindre avant de passer au suivant. En effet, on connait le point de départ et le point d'arrivée, ce qui nous donne une tendance à suivre qui est un segment autour duquel on veut que le robot oscille. On calcule alors la position des points qui donnent la trajectoire désirée.

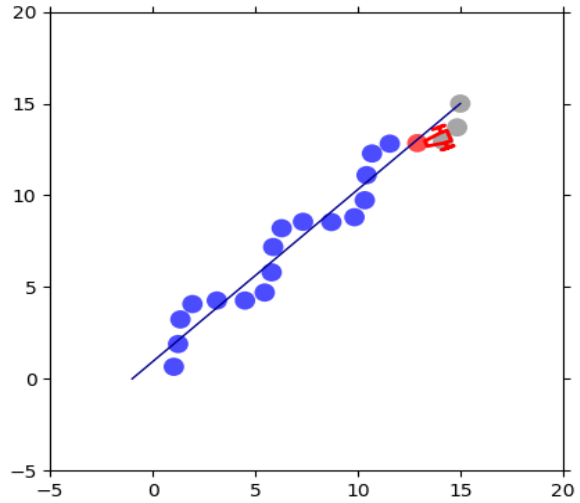

Figure 6 - Suivi de points sur une trajectoire sinusoïdale en simulation

Cette méthode s'est avérée très efficace et nous avons obtenu des résultats très satisfaisants en termes de suivi de points à l'aide du GPS seul. Nous avons pu faire une démonstration de ce programme lors de la matinée des associations de l'ENSTA Bretagne, lien vers la démonstration : https://www.youtube.com/watch?v=x7jJzU36DrU.

Figure 7 - Extrait de la vidéo de suivi de points en "zig-zag"

L'algorithme de contrôle

Le robot suit une liste successive de points, lorsqu'il est suffisamment proche du point actuel (dans un cercle autour de ce point dont le rayon est choisi en fonction de la précision que l'on souhaite obtenir) il passe au point suivant. Pour se diriger vers un des points, le robot calcule le cap qu'il doit suivre, en faisant la différence entre son cap actuel, estimé par la centrale inertielle, et le cap donné par sa distance au point. Lorsque le robot est suffisamment proche du point il calcule le nouveau cap à suivre en prenant en compte le point suivant. Il faut bien choisir le rayon autour du point GPS que l'on veut atteindre, s'il est trop grand, le robot sera très peu précis et la trajectoire globale ne sera pas celle attendue. En revanche, s'il est trop petit, le robot tournera plusieurs fois autour du point sans vraiment l'atteindre et sa trajectoire sera interrompue également. Dans notre cas, nous avons choisi une précision de 20cm autour du point objectif. Le programme qui régit ces étapes commence alors par calculer tous les points en discrétisant la sinusoïde vue précédemment, puis ils sont ajoutés dans une liste et le robot reçoit comme objectif le premier point. S'il est à plus de 20cm de ce point, le robot essaiera de le rejoindre, sinon il marque cet objectif comme atteint et passe au point suivant.

On peut voir sur ce graphique, en bleu la trajectoire prévue et en rouge le trajet réel du robot, un résultat qui était tout à fait à la hauteur de nos attentes, pour un trajet d'une dizaine de mètres entre les bâtiments E et M de l'ENSTA Bretagne.

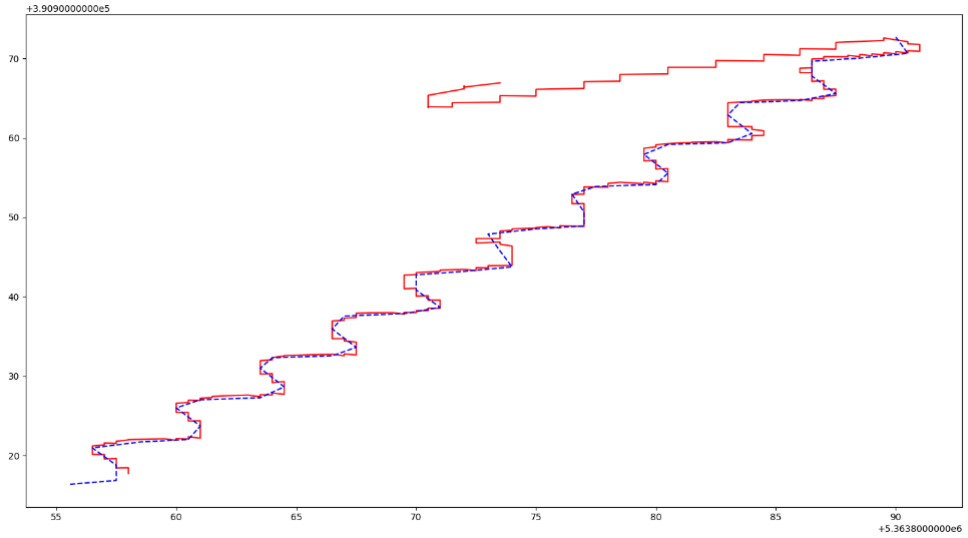

Figure 8 - Mission de prise de photo (parcours théorique en bleu, trajet réel en rouge)

La prise des photographies

Pour faire une reconstitution photogrammétrique géoréférencée des bâtiments, il a fallu trouver un système pour connaitre la position exacte et l'orientation des caméras au moment où les photos ont été prises. Pour résoudre ce problème, nous avons configuré une Raspberry Pi pour nous permettre d'automatiser cette tâche. Ce microcontrôleur communique avec le robot Warthog en se connectant à son réseau Wifi, puis en utilisant un seul ROS commun aux deux systèmes. Le ROS Core est lancé sur le robot Warthog et la Raspberry Pi y a accès, il m'a suffi de configurer son adresse IP correctement. De cette façon, le robot peut envoyer comme consigne à la Raspberry si elle doit initialiser une prise de photo ou non : en effet, lors d'une mission de photogrammétrie, le robot avait des phases de déplacement entre plusieurs bâtiments, pendant lesquelles il n'était pas nécessaire de prendre des photos, et des phases de prise de photo, pendant lesquelles il effectuait ces trajectoires en « zig-zag ».

On a alors utilisé une API python pour prendre des photos avec les caméras GoPro, qui nous a donné du fil à retordre en raison de nombreux bugs et problèmes de compatibilité, afin de prendre des photos avec le microcontrôleur. Il est important pour la synchronisation que toutes les photos soient prises exactement au même moment, d'où la nécessité de le faire grâce à un programme. Marie Ponchart avait opté pour une solution différente : elle filmait une mission de photogrammétrie dans son intégralité pour ensuite extraire quelques photos permettant de reconstituer la scène. Elle pouvait les synchroniser en filmant un chronomètre précis au dixième de seconde sur chaque caméra simultanément, pour synchroniser le début du film, et elle obtenait une précision dépendante du nombre d'images par seconde de la caméra, avec des caméras GoPro elle est de 30 images par seconde, soit une précision de 33 millisecondes. Avec l'utilisation d'un programme pour automatiser la prise de photo, on

estime que la précision est de cet ordre de grandeur, l'avantage considérable de notre solution étant qu'on ne prend pas une vidéo mais bien quelques photos, offrant un gain d'espace mémoire et d'autonomie des caméras importants. On peut alors se permettre de faire de plus longues missions, allant jusqu'à quelques centaines de photos.

Enfin, la configuration du GPS fut une étape clé de cette installation. Nous avons utilisé deux GPS classiques, reliés à un module GPS Ublox. Le premier GPS nous a permis d'obtenir la position absolue des caméras et le deuxième servait à avoir l'orientation du robot, en faisant la différence entre sa propre position et celle du premier GPS. L'acquisition de ces données de position et de cap se faisait également sur le microcontrôleur (Raspberry Pi), grâce à un code permettant de déchiffrer en partie le binaire. Les données étaient ensuite renvoyées sur un nœud ROS, et écrite dans un fichier texte enregistré dans la mémoire de la Raspberry Pi avec les photos de la mission.

Pistes d'améliorations envisagées

L'utilisation du Lidar

Nous avions commencé à utiliser le Lidar (laser imaging detection and ranging) pour la détection d'obstacle d'une part, mais surtout pour pouvoir compléter les données des photographies avec les données de ce capteur très précis. Pour ce faire, nous nous sommes penchés sur des algorithmes de SLAM (simultaneous localization and mapping) [3]. Après quelques recherches, nous nous sommes aperçus que ces deux méthodes d'acquisition et de reconstitution de l'espace (Lidar d'un côté et caméras de l'autre) sont tout à fait différentes et présentent chacune des avantages et des inconvénients. Dans le cas de la photogrammétrie, elle est moins couteuse (des appareils photo d'entrée de gamme sont déjà efficace alors qu'un Lidar est un capteur onéreux) et permet une représentation en couleur d'une scène, qui peut s'avérer indispensable dans certains cas d'application. En revanche, cette méthode demande des temps de calculs en général plus longs, et s'avère moins efficace dans la détection de petits objets ou de détails. Nous avons observé que le temps de calcul pour générer un bâtiment en photogrammétrie demande parfois dix fois plus de temps que le temps de la mission de prise de photos, alors que la reconstitution au Lidar ne prend que quelques minutes. Le Lidar se rend très efficace pour faire des modèles numériques de terrain ou repérer des objets fins comme des câbles. Nous nous sommes alors demandé s'il existait un moyen d'utiliser ces deux méthodes, qui semblent complémentaires, pour améliorer la reconstitution. Après quelques recherches [4] cela semble tout à fait possible et il existe même des codes qui le font déjà, cependant nous avons assez vite renoncé à cette méthode face à la complexité des algorithmes à mettre en place et le temps qu'il nous faudrait pour pouvoir aboutir avec cette méthode. C'est une piste à explorer qui peut grandement améliorer la qualité et la précision des images obtenues par photogrammétrie.

La mise en place d'un bras mécanisé portant les caméras

Le calcul de trajectoire que nous avons mis en place semble fonctionner correctement, mais on pourrait envisager une autre méthode pour faire varier les angles de vues, en mettant les caméras sur un axe orientable. Le profilé que nous avions fixé au robot Warthog pour disposer les GoPros pourrait tout à fait être orientable si on installait un système mécanique avec un servomoteur contrôlant l'orientation de cet axe. Il faudrait alors connaitre parfaitement son orientation pour pouvoir faire le calcul de l'angle entre ce dernier et le repère du robot. La connaissance du cap du robot grâce à l'utilisation des deux GPS et de la centrale inertielle et la connaissance de l'angle entre le robot et les caméras nous permettrait de calculer l'angle de prise de vue. Le Warthog pourrait alors se déplacer en ligne droite, et c'est l'orientation des caméras qui permettrait d'avoir plusieurs prises de vues différentes. Nous n'avons pas retenu cette solution puisqu'il nous semblait beaucoup plus simple de commander la trajectoire du robot, que nous avons appris à faire en cours à l'ENSTA Bretagne, plutôt que de concevoir un système mécanique. Cependant, cette solution peut présenter des avantages dans des espaces étroits par exemple dans lesquels la manœuvre avec le robot Warthog est très difficile.

Conclusion

L'acquisition photogrammétrique autonome est un défi particulièrement difficile, puisque l'intervention humaine est souvent nécessaire pour apprécier si chaque photographie est exploitable ou non. Dans notre cas, on peut parler d'acquisition semi-autonome puisque nous sommes toujours restés derrière le robot pour vérifier que la mission allait aboutir à un résultat satisfaisant. Cependant, c'est un sujet qui laisse beaucoup de place à l'innovation et à la recherche surtout sur la façon dont on pourrait éviter la présence d'un opérateur humain dans le traitement des photographies.

D'un point de vue plus personnel, ce stage m'a permis de consolider une grande quantité d'acquis tout au long de la deuxième année en robotique autonome, notamment d'obtenir une bien meilleure maitrise du ROS et une meilleure compréhension du contrôle des robots. Ce stage m'a aussi fait découvrir un aperçu du monde de la recherche au sein du Lab-STICC et m'a permis de me rendre compte que je ne voudrai probablement pas continuer dans cette voie, ni faire de doctorat, une idée qui m'avait traversé l'esprit en classe préparatoire mais qui n'est plus d'actualité aujourd'hui. Enfin, travailler avec un robot terrestre comme le Warthog fut une belle opportunité et c'est un outil pédagogique très performant.

Bibliographie

[1] «Wikipédia : La photogrammétrie,» [En ligne]. Available: https://fr.wikipedia.org/wiki/Photogramm%C3%A9trie. [Accès le Juillet 2021].

[2] L. Jaulin, Mobile Robotics, ENSTA-Bretagne, France: ISTE Press - Elsevier, 2015.

[3] W. Hess, D. Kohler, H. Rapp et D. Andor, Real-time loop closure in 2D LIDAR SLAM, Stockholm, Sweden: Institute of Electrical and Electronics Engineers, 2016.

[4] F. Chiabrando, F. Nex, D. Piatti et F. Rinaudo, Integration between calibrated time-of-flight camera data and multi-image matching approach for architectural survey, Tolosa: Politecnico di Torino, 2010.

Table des figures