IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

ENSTA Bretagne

Rapport de stage M1

Commande prédictive (MPC) pour l'inspection de la qualité de l'eau

Hugo Reubrecht

Robotique mobile et autonome - CERI Systèmes numériques

Mai 2022 - Août 2022

Tuteur école :
Luc Jaulin
luc.jaulin@ensta-bretagne.fr

Tuteur centre de recherche :
Eric Duviella
eric.duviella@imt-nord-europe.fr

Résumé

Le stage s'inscrit dans le cadre d'un projet régional des Hauts de France sur la surveillance de la qualité de l'eau par drones de surface (USV). La problématique s'est recentrée sur la réalisation d'une architecture de simulation des USV couplant les outils Matlab/ROS au logiciel de simulation VRX. Cette architecture offre de nombreuses possibilités de simulation de stratégies de contrôle, telle que la commande prédictive MPC (Model Predictive Control), en y injectant des perturbations liées au vent et au courant, mais également de scénarios réalistes de collecte d'échantillons basés sur des données réelles, et de scénarios intégrant des multi-drones. Cette architecture de simulation a été utilisée pour tester des stratégies d'exploration en utilisant des drones de surface avec des contraintes de durée maximale d'exploration nécessitant le recours à plusieurs drones afin de prendre en compte l'évolution rapide des données mesurées au cours du temps. L'architecture de simulation pourra être utilisée pour étudier d'autres scénarios liées aux missions d'exploration, ronde de surveillance ou détection rapide de sources de pollution.

Résumé

The problematic of this internship is to carry out a simulation of a USV (Unmanned Surface Vehicle) used to take samples and analyse water in rivers. This vessel control is realized through a particular mathematical model : the MPC ("Model Predictive Control"), which makes it possible to predict the state of the robot according to the motor inputs thanks to its dynamics. Through a Matlab/ROS architecture, the first objective is to carry out a tracking of previously provided GPS points. The next objective is the exploration of an area through the simulation using a real database obtained during previous expeditions. Then other strategies will be studied with multi-drone exploration simulations. However, the data on water varies over time, so the results obtained will be compared in this report to determine the most robust strategy for effective mapping according to certain criterias (speed, data accuracy, etc.).

Table des matières

1 Introduction

La filière robotique de l'ENSTA a une affinité particulière avec les robots qui évoluent dans des milieux aquatiques, cet environnement ajoutant des problématiques différentes par rapport à des robots terrestres. Inscrit dans la voie de l'approfondissement en Robotique Mobile et Autonome, la réalisation de ce stage au sein d'un laboratoire de recherche, le CERI de l'IMT Nord Europe, sur un sujet en lien direct avec la robotique en milieu aquatique m'a permis de faire valoir les compétences acquises à l'école dans ce domaine durant cette seconde année. Ce stage a été l'occasion de découvrir aussi le fonctionnement d'un laboratoire et d'entrevoir le monde de la recherche.

L'objectif principal de mon stage consistait en la réalisation d'une architecture de simulation de drones aquatiques, de type aéroglisseurs, en couplant les outils Matlab, ROS et le logiciel de simulation VRX. Ces drones de surface sont utilisés pour prendre de mesures caractérisant la qualité de l'eau. Plusieurs campagnes de mesure sur des sites de la région des Hauts de France avaient été réalisés avant le début de mon stage. Une stratégie de contrôle prédictif de type MPC (Model PRedictive Control) avait également été conçue et testée sous l'environnement Matlab.

En tant que stagiaire, j'ai dû i) m'approprier la problématique de relevés de mesure par drone aquatique, ii) concevoir l'architecture de simulation couplant les différents outils, iii) bien comprendre les concepts de la commande MPC afin de l'utiliser pour le drone proposé dans le simulateur VRX, ce drone étant différent de celui utilisé dans le cadre du projet, iv) déployer des commandes MPC pour des simulations multi-drones, v) réaliser des scénarios de simulation réalistes à partir de mesures réelles. L'architecture ainsi créée permet maintenant d'imaginer plusieurs scénarios, afin de réaliser et tester des stratégies de rondes par multi-drone, de la détection et localisation de source de pollution, etc.

Le rapport a pour objectifs de présenter toutes les tâches réalisées lors de mon stage en développant davantage les algorithmes de contrôle et des simulations mises en place pour le drone aquatique utilisé par la start-up "Bathy Drone Solutions". Il est articulé autour de deux axes principaux : l'automatisation du robot et la stratégie d'exploration d'une zone avec le mise en place de simulation multi-drones pour en tester l'intérêt.

2 Contexte

2.1 Présentation des organismes

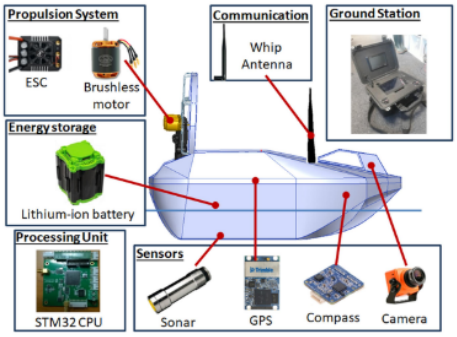
2.1.1 Présentation de l'IMT Nord Europe - CERI SN

Le centre de recherche de l'IMT de Douai dispose de différents pôles de recherche et celui au sein duquel je vais réaliser ce stage est le "Centre d'Enseignement Recherche et Innovation Systèmes Numériques" (CERI SN). Il est composé de 34 enseignants chercheurs et 6 ingénieurs qui mènent des études notamment sur divers systèmes complexes alliant Intelligence artificielle et Automatique. Je serai pour ma part plus inscrit dans un registre automatique mais avec un sujet malgré tout tourné vers la robotique avec l'automatisation d'un drone aquatique.

2.1.2 Présentation de BDS

Bathy Drone Solution (BDS) est une start-up qui réalise des prélèvements bathymétriques dans la région. L'entreprise emploie un drone qui va pouvoir réaliser des prélèvements en profondeur dans différents milieux aquatiques. L'objectif de l'entreprise est de pouvoir proposer des prestations d'analyse bathymétrique qui vont permettre de se passer d'opérations potentiellement coûteuses voir dangereuses si l'on employait un plongeur pour les réaliser. Ici le drone conçu par l'entreprise permet justement à l'opérateur de rester sur la berge et de pouvoir atteindre des zones parfois inaccessibles à la nage. BDS et le CERI SN de l'IMT Nord Europe travaille donc en collaboration dans l'optique de pouvoir à terme rendre ce robot autonome.

(a) Photo du drone de BDS



(b) Architecture hardware du robot

FIGURE 2 – Drone de BDS

2.2 Objectif du stage

2.2.1 Enjeux des recherches

L'exploration de milieux aquatiques est devenu un véritable enjeu ces dernières années, c'est en effet une exploration qui apporte de nombreuses problématiques puisqu'il s'agit d'un environnement qui est la plupart du temps difficilement accessible par l'Homme notamment si l'on souhaite réaliser de l'exploration sous-marine. Cependant cette exploration est primordiale dans divers domaines comme la cartographie, la surveillance de certaines zones sensibles, l'écologie... Dans le cadre de ce stage on s'intéressera principalement à l'aspect cartographique et écologique de cette exploration avec un robot qui surveille la qualité de l'eau permettant ainsi de détecter les principales zones sources de pollution qui pourraient mettre en péril la faune et la flore aquatique. L' enjeux des recherches ici est de pouvoir proposer à BDS différents scénarios d'automatisation possibles du robot. Pour l'instant le robot nécessite un opérateur pour le télécommander à distance, cependant cette façon de faire n'est pas la plus optimale dans le cadre d'exploration de larges zones comme des lacs par exemple. L'opérateur a du mal à estimer les trajectoires et visiter chacun des recoins des différents cours d'eau. Des problématiques se rajoutent avec l'évolution des données de l'eau au cours d'une journée. Il est donc important d'être le plus optimal pour la visite d'une zone. Il faut s'assurer que tous les endroits sensibles à la pollution auront été explorés avec suffisamment de valeurs pour avoir des données moyennées plus précises et pertinentes, et tout cela dans une fourchette de temps la plus réduite possible pour éviter des changements trop brusques au cours de la journée.

2.2.2 Mes missions

Je vais détailler dans cette section, les différentes missions que j'ai reçues au cours de ce stage. Je détaillerai plus spécifiquement ces travaux dans les sections suivantes. Les missions que j'ai réalisé au cours de ce stage tournent autour de la mise en place d'une simulation pour un USV sur un plan d'eau. Des algorithmes de contrôle ont déjà été codés sous Matlab. Mon premier objectif est donc de comprendre le fonctionnement de cet algorithme pour pouvoir ensuite réaliser une passerelle entre ces codes sous Matlab et le software ROS. Une fois cette passerelle mise en place, il sera possible de faire tourner les algorithmes sur Matlab est de communiquer les résultats des calculs de contrôle au robot recevant des informations via ROS. J'ai du ensuite m'approprier le simulateur VRX et adapter l'architecture logicielle de cet environnement pour la problématique sur laquelle je travaille. La suite consiste à réaliser des simulations et tests de performances selon le vent, et la réalisation de certains parcours pour cartographier le Lac du Héron. Je réaliserai donc une évolution dynamique (selon l'heure de la journée) des différente données de l'eau au sein de cette simulation. Et pour finir je mettrai en place une simulation multi-robots et étudierai différentes stratégies de cartographie.

3 Architecture la commande de drones aquatiques

3.1 Architecture Matlab/ROS

3.1.1 Rostoolbox sous Matlab

ROS (*Robot Operating System*) est un environnement open source de développement logiciel pour la robotique. Cet environnement est très pratique pour la mise en place d'une architecture logicielle dans un projet de robotique. L'une de ses principales qualités est sa capacité à établir facilement et de façon intuitive des connexions et communications entre les différentes interfaces réseaux du projet. La première étape est de relier l'algorithme MPC codé sur Matlab avec une architecture ROS externe. Pour cela des plugins sous Matlab existe. L'un d'entre eux est le *Rostoolbox*. Cette toolbox permet de créer un réseau de noeud ROS. Cela va être utile dans un premier temps pour conserver et réutiliser ainsi les codes d'algorithme MPC déjà déployés sous Matlab. Grâce à cette fonctionnalité il suffit de connaitre l'adresse IP du noeud master qui correspond au noeud dont le rôle est de permettre aux différents nodes ROS de se localiser les uns les autres. Une fois que ces nœuds se sont localisés, ils peuvent communiquer entre eux en *peer to peer* (c'est à dire que le noeud est à la fois client et serveur) (il publie ou reçoit un topic contenant un type de message).

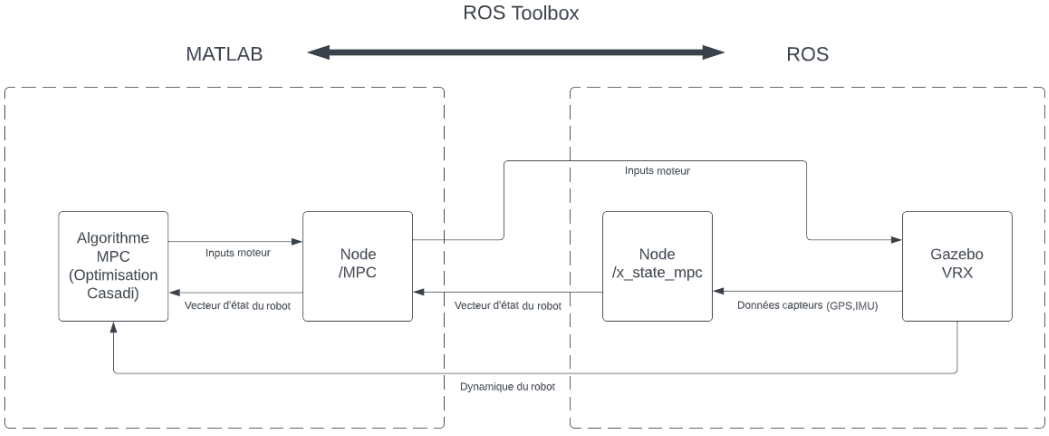


FIGURE 3 – Architecture simplifiée du Rostoolbox

3.1.2 Architecture des noeuds

Après avoir réussi la liaison Matlab-ROS, il a fallu ensuite réaliser l'architecture des noeuds ROS. Pour résumer, les différents capteurs publient leur données sur divers topics, grâce auxquels on va pouvoir publier le vecteur d'état du robot (x,y,yaw,xdot,ydot,yawdot) nécessaire pour l'algorithme MPC. Ensuite le noeud Matlab va envoyer sur le topic contrôlant les moteurs les valeurs des inputs moteurs pour atteindre l'objectif. Pour la position à partir des données en latitude et longitude dU GPS, une projection sur le plan 2D du lac est réalisée pour avoir les coordonnées du robot selon l'axe x et y du plan avec un point d'origine choisi aléatoirement sur le lac il s'agira du point GPS de référence.

Le GPS fourni également la vitesse relative du robot mais ce topic ne fourni pas les vitesses selon l'axe X et Y dans le repère du robot, or il s'agit de celui ci qui doit être utilisé dans le cadre de l'algorithme MPC. Pour les obtenir j'ai intégré les valeurs des accélérations de l'IMU selon un certain pas de temps. Cette technique est très déconseillée car très imprécise puisque les erreurs vont s'accumuler au cours du temps ! J'utilise donc en parallèle la vitesse fournie par le GPS pour corriger ces erreurs.

3.1.3 Passerelle Matlab/C++

L'avantage certain de Matlab, c'est que cela permet de réaliser du code à haut niveau, ce qui facilite la compréhension du code et son écriture, avec une interface utilisateur confortable. C'est idéal pour de la simulation, il est facile de changer les paramètres, les réadapter, ajouter des bibliothèques... Toutefois cela demande pas mal de ressources de la part de la machine pour faire tourner Matlab et lors d'expérimentations réelles ce n'est pas l'idéal, le codage Matlab n'est pas adapté pour de l'embarqué. Matlab possède des fonctionnalités qui permettent de passer des noeuds ROS sur Matlab en noeuds ROS en C++. J'ai eu l'occasion de le tester sur des noeuds très simples, mais cela n'est pas si évident puisque cette retranscription en C++ demande un format de code très restrictif sur le code Matlab d'origine. L'idéal serait de réécrire le code Matlab en C++ ou en Python et de réadapter les bibliothèques utilisées en fonction.

3.2 Simulateur VRX

3.2.1 Présentation de VRX

Le simulateur utilisé est *Virtual RobotX* (VRX) tourne sous Gazebo. Il a été initialement été développé pour des challenges et compétitions autour du contrôle de USV. L'avantage de ce simulateur c'est qu'il permet de créer un environnement plutôt réaliste pour le robot. On peut y modifier quelques paramètres tels que le vent ou la dynamique du robot. L'un des désavantages c'est qu'il n'est pas évident à prendre en main, des tutoriels existent pour le contrôle du robot mais modifier l'environnement demande d'étudier en détail les nombreux fichiers qui composent le simulateur. J'ai donc du réadapter ce simulateur pour ce que l'on souhaitait faire, en y ajoutant notamment des packages pour la liaison avec les noeuds Matlab.

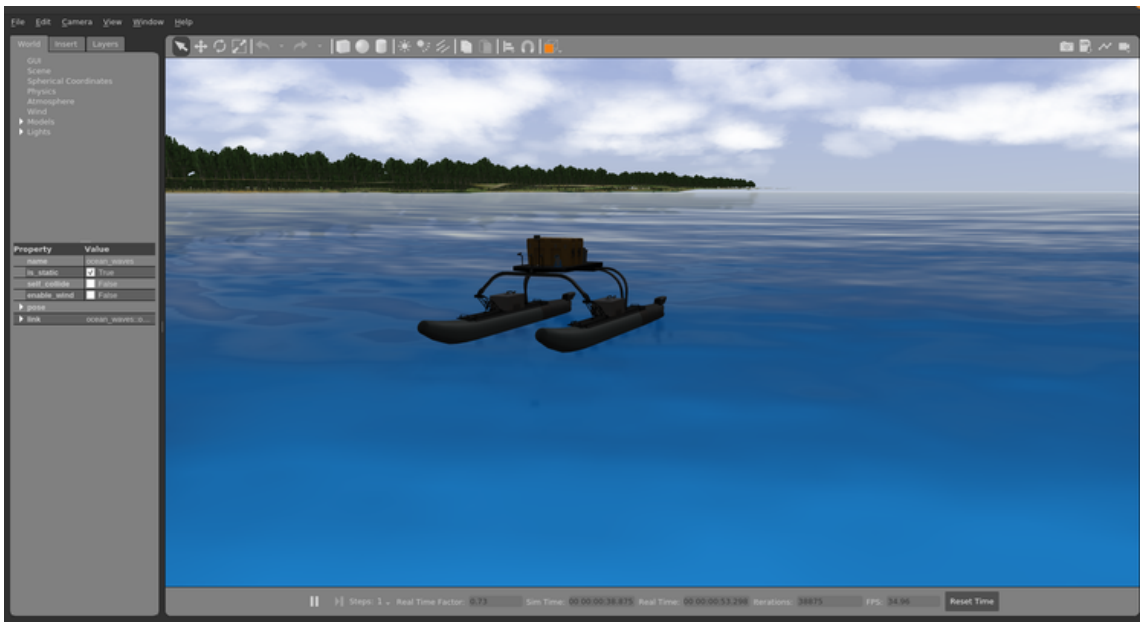


FIGURE 4 – Environnement simulé VRX avec drone

3.2.2 Modèle dynamique des robots

Le simulateur VRX possède son propre USV. N'ayant pas à disposition de modèle 3D du robot de BDS je n'ai pas pu importer le véritable robot sur VRX. La principale différence entre ces deux robots et la position des propulseurs. Chez BDS, les hélices se situent en dehors de l'eau, le robot fonctionne donc comme un petit aéroglisseur, alors que celui de VRX, les hélices se situent sous l'eau. Mais dans les deux cas les formules des équations de la dynamique de ces deux robots sont en réalité les mêmes ! La seule différence va se trouver au niveau des valeurs des paramètres dans les formules. J'ai donc du réadapter les paramètres pour que le drone sur VRX

agisse sensiblement comme le robot de BDS. Mais cela s'avère plus difficile que prévu puisque les fichiers de VRX fournissent énormément de paramètres modifiables dont certains me sont inconnus dans le cas du robot de BDS, j'ai alors approximé certains paramètres.

$$\underbrace{\boldsymbol{M}_{RB}\dot{\boldsymbol{\nu}} + \boldsymbol{C}_{RB}(\boldsymbol{\nu})\boldsymbol{\nu}}_{\text{rigid body forces}} +$$
$$\underbrace{\boldsymbol{M}_A\dot{\boldsymbol{\nu}}_r + \boldsymbol{C}_A(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r + \boldsymbol{D}(\boldsymbol{\nu}_r)\boldsymbol{\nu}_r}_{\text{hydrodynamic forces}} + \underbrace{\boldsymbol{g}(\boldsymbol{\eta})}_{\text{hydrostatic forces}}$$
$$= \boldsymbol{\tau}_{propulsion} + \boldsymbol{\tau}_{wind} + \boldsymbol{\tau}_{waves}$$

where

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

FIGURE 5 – Equations d'état du drone

Les équations d'état ne diffèrent pas selon le fait que les deux propulseurs soient immergés ou non, la différence que cela va apporter se situe au niveau des paramètres situées au sein des matrices. [1]

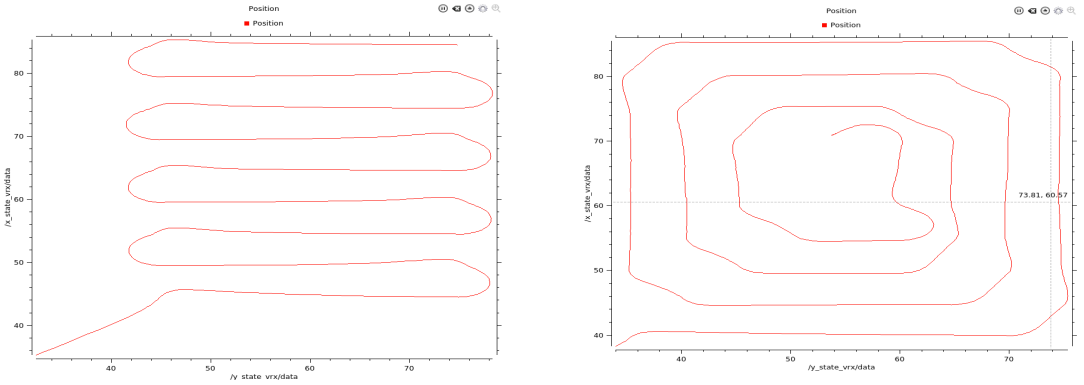
Il faut remarquer qu'ici les forces du vent et du courant sont prises en compte, ce qui ne sera pas le cas lors du calcul du modèle prédicatif puisque le drone ne possède pas des capteurs permettant de mesurer ces deux variables. Toutefois on remarquera que le MPC permet tout de même d'envoyer des commandes assez robustes pour contrer des vents et courant modérés.

3.3 Commande prédictif MPC

3.3.1 Principe et Intérêts

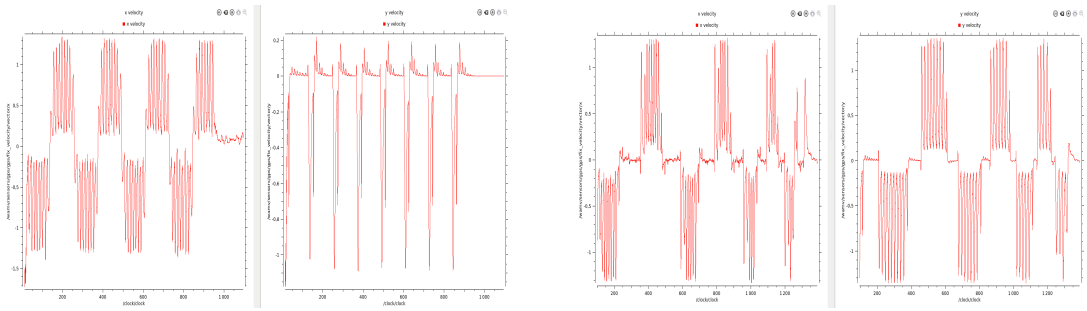
Le MPC est un modèle de contrôle qui cherche à optimiser les commandes moteurs pour rejoindre une surface cible. Cette surface cible possède 6 dimensions, la position x, y et z du drone, sa vitesse selon les axes x, y du plan sur lequel il se déplace, ainsi que sa vitesse angulaire. Pour se faire l'algorithme va chercher à réduire une fonction de coût ce qui permet notamment d'envoyer en sortie de calcul les commandes optimales à réaliser si l'on souhaite atteindre cette surface cible.

Dans ces différents cas, on cherche à recouvrir une zone rectangulaire dont les coins sont donnés en entrée, la liste des surfaces cibles est ensuite déterminée à l'avance. Ici on a implémenté de type de trajectoires possibles, celles en zigzag et celle en spirale. Dans les deux cas le MPC a su renvoyer des commandes qui ont permis d'atteindre la trajectoire prévue.



(a) Trajectoire MPC zigzag (b) Trajectoire MPC spirale

FIGURE 6 – Différents types de trajectoires réalisées par MPC



(a) Vitesse en x et y sur la trajectoire zigzag (b) Vitesse en x et y sur la trajectoire spirale

FIGURE 7 – Profils des vitesses

On peut constater que pour les vitesses, elles oscillent en permanence, cela est dû au fait que les commandes sont moins fortes à l'approche d'une zone. La vitesse est prise en compte dans la fonction de coût à optimiser, et il a été demandé ici qu'à l'arrivée sur une surface cible on souhaite avoir une certaine vitesse, d'où le ralentissement. Ensuite le robot accélère pour atteindre la zone suivante.

3.3.2 Limites et difficultés

Au cours de mes différentes expérimentations du modèle MPC, j'ai pu observer quelques problématiques et limites liées spécifiquement à cet algorithme. Le premier étant le temps de calcul assez conséquent. L'optimisation de la fonction de coût nécessite des ressources importantes pour le PC portable que j'utilisais. La fréquence d'envois des topics ROS était donc assez faible et risque d'être encore plus faible sur un PC embarqué dans le drone, souvent moins performant. Le problème lié à cette faible fréquence s'observe sur le temps de réaction du robot qui met donc plus de temps pour corriger les erreurs lorsqu'il s'éloigne de sa trajectoire.

4 Stratégies de suivi dynamique de la qualité de la ressource en eau

4.1 Conception du MPC

4.1.1 Modèle d'état du robot

Une étude sur le modèle d'état du robot de BDS a déjà été réalisée en amont lors d'un précédent stage. Ce qu'il faut retenir de cette étude est notamment l'obtention des paramètres dynamiques du robot. (extrait étude modèle dyn bds) Cependant ceux du modèle de VRX sont différents, j'ai donc eu le choix de réadapter les paramètres du modèle VRX ou alors implémenter les paramètres de VRX dans l'algorithme MPC. J'ai donc préféré implémenter les paramètres de VRX dans le modèle MPC car VRX présentait certains paramètres supplémentaires qui n'ont pas été étudiés dans le cas du modèle de BDS.

4.1.2 Présentation du MPC dédié

La première version du MPC a été codée par Alejandro Anderson avec un fonctionnement qui n'utilisait pas la simulation VRX. Mon travail a été de repartir de cette base de code pour qu'il s'implémente dans une architecture ROS. Dans un premier temps j'ai conçu différents types de trajectoires (*zigzag* et spirale) et le robot atteignait les cibles les unes après les autres. Le MPC final emploie cette structure mais cette fois ci avec une prise en compte des deux prochaines zones à atteindre pour adapter au mieux sa trajectoire, et le parcours correspond à l'exploration du Lac du Héron qui a été réalisé manuellement pour ensuite pouvoir comparer les performances.

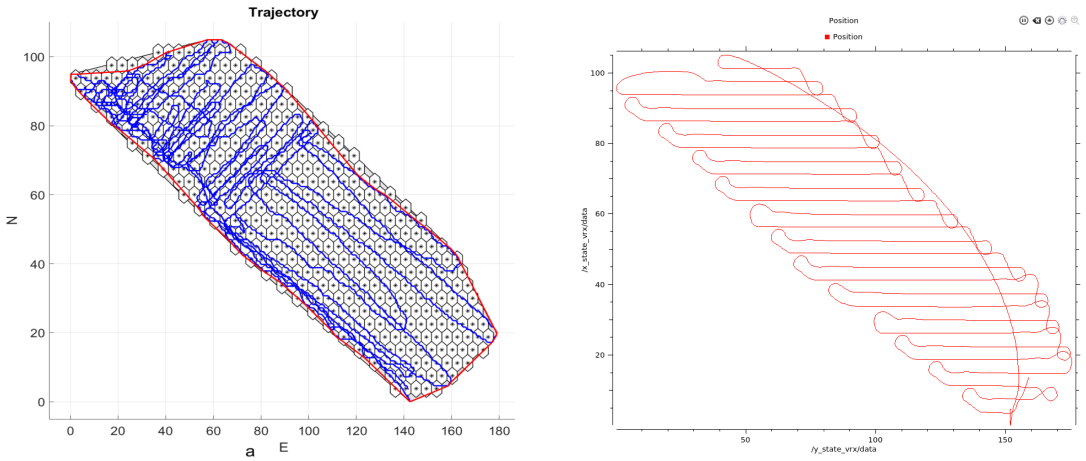
4.2 Cartographie du lac du Héron

4.2.1 Objectifs et relevés réels

La cartographie est un thème important de la robotique, ici il s'agit d'une cartographie sur les données de l'eau. Ces données permettent de souligner la présence de potentielle zones à risques pour la faune et la flore de certains lac. Le drone de BDS a réalisé des relevés réels sur le Lac du Héron avec un opérateur pour téléguider le drone. Cependant le guidage manuel du robot n'est pas une solution optimale pour de la cartographie. Toutefois cela nous permet d'obtenir un premier jeu de donnée qui permettra par la suite la réalisation de diverses simulations.

4.2.2 Comparatif campagne de mesure "Manuelle" vs "Automatique"

Les premiers résultats sont concluants avec la simulation, toutes les zones *targets* sont observées par le robot à l'exception de certaines situées sur les bords de la surface à explorer. Ceci est notamment due à une difficulté pour le robot à réaliser certaines rotations car il possède un rayon de courbure assez important avec le modèle dynamique qu'il possède. Ce rayon de courbure risque d'être probablement moindre pour le robot de BDS. Pour ce qui correspond de la carte finale obtenue, elle se rapproche beaucoup de la carte réalisée avec la campagne manuelle. Ceci s'explique notamment par le fait que les données de l'eau employées sont celles de cette campagne, il est donc normal de retrouver une carte très similaire.



(a) Trajectoire avec un contrôle manuel (b) Trajectoire avec le MPC

FIGURE 8 – Comparaison des trajectoires

4.2.3 Génération de scénarios réalistes (génération de Cartes)

Pour la cartographie en simulation, j'ai ajouter à l'environnement de VRX, un nouveau paramètre correspondant à l'oxygène dissous dans le Lac. Pour se faire j'ai utilisé comme base de données les cartes réalisées durant les campagne de contrôle manuel du robot réalisées précedemment. Cette carte manuelle a été réalisée en plusieurs étapes. La première a été la génération d'une enveloppe convexe contenant l'ensemble du parcours réalisé par l'opérateur avec le robot. Ensuite cette zone a été décomposée en plusieurs sous-zones appelées polyhédrons (qui correspondront par la suite aux *targets* à atteindre). Et pour chacun de ses polyhédrons la valeur associée correspond à la valeur moyennée des valeur prélevées par le drone dans cette sous-zone. Toutefois cette carte présente de nombreuses zones sans données dans l'enveloppe convexe car il est évidemment très difficile de parcourir tous les polyhédrons. Une hypothèse a été réalisée et les valeurs attribuées aux zones vides

correspondent à la valeur moyenne des valeurs correspondant aux polyhédrons voisins et cette hypothèse est répétée jusqu'à obtenir une valeur pour chacune des zones. Cette hypoyhèse n'est pas très précise pour compléter les cartes, d'où l'emploi d'un contrôle autonome du drone qui pourrait parcourir toutes les *targets*. Cependant cela me permet d'avoir une carte remplie qui me servira de base de donnée pour la simulation. J'ai donc ensuite inclus dans l'architecture ROS un nouveau topic qui publie en temps réel les données de l'oxygène dissous. Cela permet d'obtenir des valeurs simulées des données de l'eau sur le Lac, est de reconstruire une carte à partir de ces données ensuite. Pour plus de réalisme dans la simulation, j'ai rajouter une dynamique par rapport au temps avec une fonction qui viendra modifier la base de données d'oxygène dissous en fonction de l'heure de la journée.

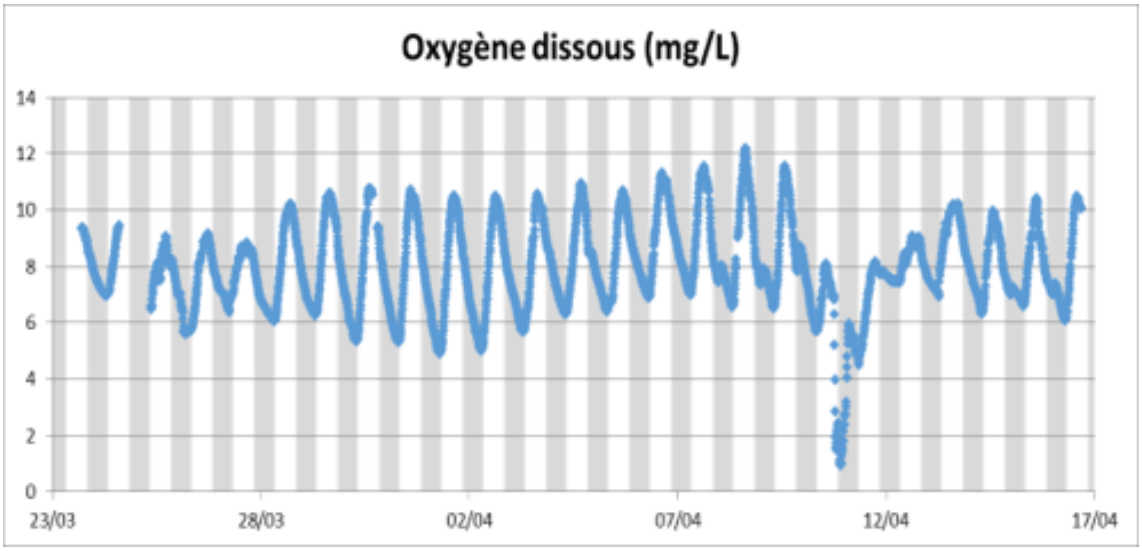
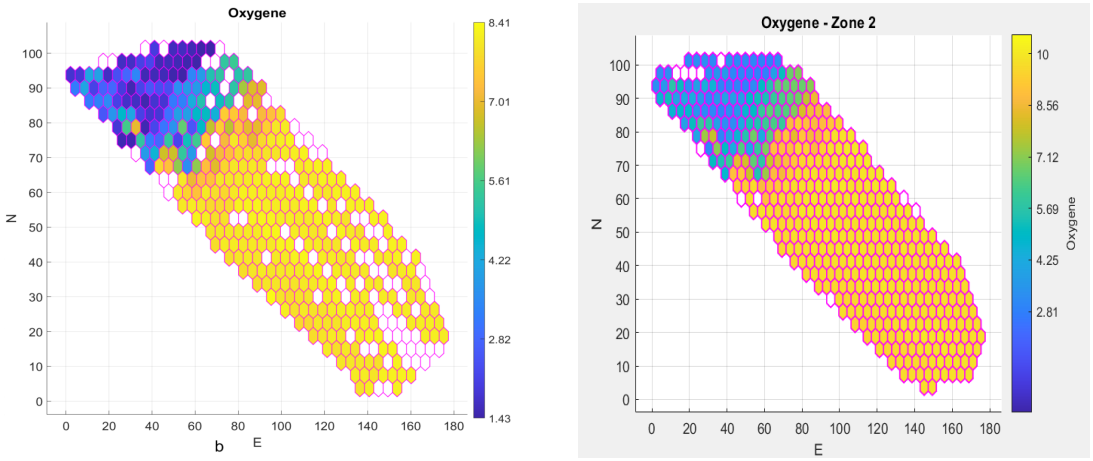


FIGURE 9 – Cycle nychthéméral de l'oxygène dissous au Lac du Héron

4.2.4 Cartographie autonome (Contrôle prédictif du drone)

L'objectif pour le robot est donc de parcourir chaque *targets* constituant l'ensemble d'une zone à explorer. Pour se faire, le robot va suivre une trajectoire précise qui est établie avant la mission. La trajectoire désignée correspond à une liste des *targets* à explorer. Cette liste est générée à partir de l'enveloppe convexe de la zone à explorer, il est ensuite possible de trier cette liste pour parcourir la zone en *zigzag* dans le sens que l'on souhaite. Toutefois il peut y avoir quelques erreurs de tris. Une relecture de la liste peut être parfois nécessaire pour éviter que le robot puisse réaliser l'exploration des *targets* dans un ordre non-optimal. Une réévaluation de cette liste de *targets* n'est pas réalisée en cours de mission, d'où l'importance de s'assurer en amont qu'il n'y aucune erreur. Mais une réévaluation de la trajectoire et des commandes moteurs est continuellement réalisée pour atteindre chacune de

ces *targets*.



(a) Carte avec contrôle manuel (b) Carte obtenue en simulation avec MPC

FIGURE 10 – Comparaison des trajectoires

4.2.5 Discussion (temps d'exploration + Qualité de la stratégie + répétabilité...)

Les résultats précédents sont plutôt concluants, les simulations ont été réalisées dans un environnement assez réaliste avec des contraintes comme le vent qui ont été générées de façon aléatoire, cela reste un vent raisonnable mais capable de déplacer le drone à l'arrêt de quelques centimètres par seconde. Le MPC a su malgré tout atteindre l'ensemble des *targets* souhaitées. Le temps d'exploration est d'1h30 en moyenne, la campagne manuelle a duré approximativement 2h. La stratégie choisie ici à été de parcourir l'intégralité de la zone avec une trajectoire en *zigzag*. Mais il aurait été possible de ne viser que les zones les plus polluantes avec un premier passage plus succincts (espacement des *targets* plus large) et un second passage plus précis sur les zones faibles en oxygène dissous. L'avantage de cette simulation est sa répétabilité avec des trajectoires, des contraintes de vent et des dynamiques de drones différentes au cours des différents tests.

4.3 Cartographie multi-robots du lac du Héron

4.3.1 Architecture pour la simulation Multi-robots

L'intérêt du multi-robots est de limiter le temps nécessaire pour une expédition à cause notamment du cycle nycthéméral de l'oxygène dissous qui peut fausser la cartographie. Pour la mise en place de ce système, j'ai d'abord essayé l'utilisation d'un *namespace* où j'aurai pu dupliquer ainsi facilement le nombre de robots et de leurs topics respectifs, mais VRX ne le permet pas, j'ai donc recodé les fichiers du

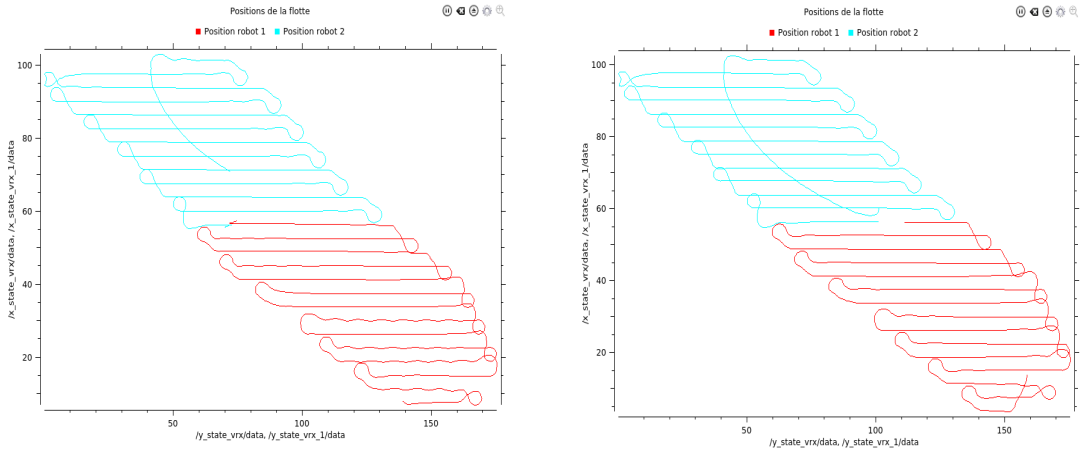
simulateur pour faire *spawn* (générer) deux robots sur la map. J'ai ensuite renommé tous les topics du second robot pour ne pas les confondre avec ceux du premier. Je n'ai pas eu le temps d'automatiser ce processus si l'on souhaite ajouter plus de deux robots, il faudra donc les recoder à la main avec leurs propres noeuds ROS pour les inclure dans cette architecture. Certes ce n'est pas pratique, mais l'étude avec deux robots est suffisante pour mettre en avant l'avantage du multi-robots.

4.3.2 Stratégies de cartographie multi-robots

Le temps serait alors divisé par deux avec deux robots si la stratégie employée est la répartition des *targets* entre les deux robots. Ici on fournit la même liste de *targets* à explorer pour les deux robots, on considère qu'ils connaissent tous les deux cette liste. Mais ils vont également connaître la liste des targets parcourus par l'un et par l'autre. Cela est possible uniquement si une communication entre les drone est établie. Cette communication n'est pas simulée ici en conditions réelles, les deux robots ont accès à cette information via des topics ROS.

4.3.3 Résultats

Les résultats obtenus sont concluants, avec une répartition des zones à explorées réussies. Au début la gestion des collisions n'était pas prise en compte, ensuite la liste des *targets* a été mise à jour en fonction de l'avancement de chacun des deux robots. Avec deux robots le temps d'exploration a bien été divisé par deux. Et il n'y a pas eu de collisions entre eux durant les simulations.



(a) Multi-robots avec collisions (b) Multi-robots sans collisions

FIGURE 11 – Trajectoire MPC avec deux robots

5 Apport du stage

5.1 Compétences développées

Première compétence essentielle qu'il faut mobiliser est l'autonomie dans son travail. J'ai pris l'habitude de rechercher avec efficacité dans les documentations, les solutions aux divers problèmes que j'ai pu rencontrer. Autre compétence que j'ai pu développer c'est d'être capable de proposer des pistes d'avancement lors des réunions. En effet la prise de parole lors de réunions d'avancement n'est pas quelque chose dont j'ai eu l'habitude lors de projets à l'ENSTA, entre camarades, je ne réalisais pas vraiment des réunions de ce type et je pense que j'en réaliserai sûrement plus souvent à l'avenir car cela permet vraiment de comprendre ce que les autres font et d'organiser la suite précise des travaux de chacun.

Pour ce qui est des compétences techniques, les attendus ont été ceux qui correspondent à ce que je sais faire grâce à la formation de l'ENSTA.(Python, C++, ROS). Mais ce stage m'a permis de beaucoup évoluer sur Gazébo et ROS avec la mise en place de multiples simulations en environnement aquatique. J'ai également pu en apprendre davantage sur le fonctionnement du MPC qui était un tout nouvel algorithme de contrôle à comprendre.

5.2 Impressions personnelles

Ce stage m'a permis de découvrir le domaine de la recherche en robotique. L'ambiance de travail a été agréable et j'en garde un très bon souvenir. Si il y avait une chose à améliorer ce serait peut être avoir une fréquence de réunion plus élevée. J'ai pu faire preuve d'autonomie même si ce n'est pas toujours évident de déterminer dans quels axes mener les recherches car les problématiques sont nombreuses (cartographie avec données dynamiques, contraintes environementales, gestion du MPC avec le choix de ces paramètres, multi-robots...). Au début les pistes de travail sont claires et précises comme construire l'environnement ROS et la simulation permettant le contrôle du drone par l'algorithme MPC. Mais après la phase de mise en place de l'environnement, il y a la phase de test, et cette fois-ci les pistes sont multiples. Tout l'enjeu des recherches est de savoir quelles pistes seront exploitables et intéressantes à étudier.

6 Conclusion et perspectives

Pour conclure, l'utilisation d'un algorithme de contrôle tel que le MPC permet d'évaluer en temps réel les commandes moteurs à fournir pour atteindre un objectif en terme de position et de vitesse. Cette approche utilisée en simulation ici permet l'exploration complète d'une zone que l'on aurait plus de mal à réaliser manuellement. Malgré tout cet algorithme peut présenter quelques défauts comme le temps de calcul ou les choix de trajectoires qui peuvent parfois paraître non-optimales.

Le MPC peut prendre en compte la force du vent dans ces paramètres mais à condition de la connaître en temps réel ce qui n'était pas le cas ici. Malgré tout l'algorithme est assez robuste pour faire face à des contraintes de vent modérée.

Le multi-robot a permis de mettre en évidence le gain de temps primordial lors d'explorations sur de plus grandes surfaces où les données de l'eau peuvent varier durant la journée. Il est donc important de construire la carte avec des données relevées sur un créneau temporel le plus court possible. Avec deux robots il est facile d'éviter les collisions et de programmer le parcours de chacun. Il pourrait être intéressant de réaliser une cartographie en simulation avec une flotte plus grande.

La suite des évènements pourrait être également de réaliser des tests du MPC en expérimentations réelles. Pour se faire il faudrait passer l'algorithme codé ici en Matlab avec un langage haut niveau sur un langage bas niveau tel que le C++. Cela permettrait d'utiliser l'algorithme en embarqué avec des temps de calcul plus faibles.

7 Annexe

7.1 Git

J'ai réalisé un dépôt GIT où se trouve l'ensemble des packages utilisés pour l'utilisation de l'environnement simulé VRX ainsi que mes codes commentés pour l'architecture ROS déployée pour le contrôle via le MPC des robots. Un tutoriel est présent pour faciliter l'implémentation. Je vous conseille vivement d'y aller faire un tour.

Lien du dépôt : https://github.com/Hugo5959/stg2022-hugo

7.2 Vidéo robot de BDS

Lien YouTube pour visualiser le robot en téléopération : https://youtu.be/Ns0s1Zaa0do

8 Bibliographie

Références

[1] Yoann HERVAGAULT (2016) *Conception et réalisation d'un système efficace de communication et de coordination au sein d'une flotille de drones aquatiques de surface*, Thèse.

Lien pdf : ici

[2] A. Anderson , J.G. Martin J. Mougin N. Bouraqadi E. Duviella L. Etienne L. Fabresse K. Langueh G. Lozenguez C. Alary G. Billon P.J. Superville J.M. Maestre (2022) *Water Quality Map Extraction from Field Measurements Targetting Robotic Simulations*, IFAC Paper

Lien pdf : ici

[3] A. Anderson , J.G. Martin N. Bouraqadi L. Etienne K. Langueh L. Rajaoarisoa G. Lozenguez L. Fabresse J.M. Maestre E. Duviella (2022) *Set-based Model Predictive Control for Exploration : Application to Environmental Missions* , Article

Lien pdf : ici