Simulation hybride pour la coordination de véhicules hétérogènes au sein d'une flottille

Olivier Parodi

Université Montpellier II

SCIENCE ET TECHNIQUES DU LANGUEDOC

Thèse

En vue de l'obtention du grade de

Docteur De L'Université De Montpellier II

Discipline : **Génie Informatique, Automatique et Traitement du Signal**
Formation Doctorale : **Systèmes Automatiques et Microélectroniques**
Ecole Doctorale : **Information, Structures, Systèmes**

Présentée et soutenue publiquement par

Olivier Parodi

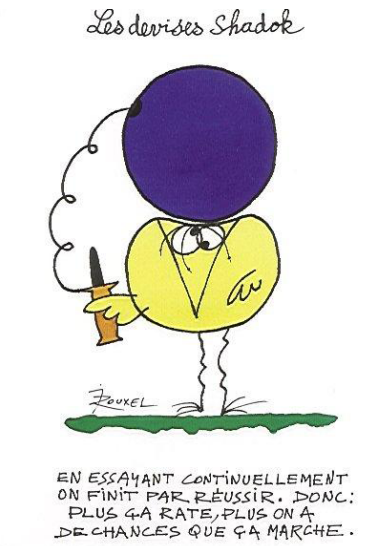
Le 18 Décembre 2008

Simulation hybride pour la coordination de véhicules hétérogènes au sein d'une flottille

Jury

Rapporteurs	Massimo	CACCIA	*Senior Researcher, CNR - ISSIA*
	Simon	LACROIX	*Directeur de Recherche, CNRS - LAAS*
Examinateurs	Frédéric	DEVIE	*Ingénieur, DGA - GESMA*
	David	ANDREU	*Maître de Conférences, Université Montpellier II*
	Lionel	LAPIERRE	*Maître de Conférences, Université Montpellier II*
Directeur de thèse	Bruno	JOUVENCEL	*Professeur, Université Montpellier II*

Thèse préparée au LIRMM Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (CNRS), 161, rue Ada 34392 Montpellier Cedex 5



A ma femme et ma fille.

Remerciements

Les travaux présentés ont été réalisés au LIRMM, Laboratoire d'Informatique de Robotique et de Micro-électronique de Montpellier, Unité mixte de recherche du CNRS et de l'Université de Montpellier 2. Je remercie Michel Robert, directeur du LIRMM ainsi que François Pierrot et Stefano Cerri, directeurs-adjoints de m'avoir accueilli au sein du laboratoire.

Je remercie les membres du jury, pour l'intérêt qu'ils ont porté à mes travaux, en particulier le président de mon jury, M. Simon Lacroix, Directeur de recherche au LAAS à Toulouse, et M. Massimo Caccia, Chercheur à L'ISSIA à Gênes pour avoir accepté d'évaluer mon travail. Je tiens à remercier M. Frédéric Devie, ingénieur au GESMA à Brest, pour la précision des remarques qu'il m'a faites et qui m'ont permis d'améliorer le présent manuscrit. J'exprime ma gratitude à Mr. David Andreu, Maître de conférence à l'Université de Montpellier 2, pour les nombreuses discussions enrichissantes dont j'ai pu tirer partie pour orienter mon travail tout au long de la thèse, ses qualités humaines et sans qui je n'aurais jamais pu accomplir cette tâche. Un grand merci à Mr. Lionel Lapierre, Maître de conférence à l'Université de Montpellier 2, pour sa patience, sa gentillesse, sa pédagogie dans ses explications et pour les nombreuses heures passées à m'aider dans l'implémentation des modèles et lois de commande. Je remercie bien sûr mon directeur de thèse, Mr. Bruno Jouvencel, Professeur à l'Université de Montpellier 2, pour l'autonomie qu'il m'a accordée durant ma thèse, ses qualités d'écoute, sa patience et son jovial optimisme (à toute épreuve), qui apporte toujours du courage.

Ce travail est le fruit d'une collaboration scientifique et humaine riche, et c'est dans ce sens que j'adresse mes sincères remerciements à Abdellah El Jalaoui, pour les nombreuses nuits de soutien et ses conseils formateurs pour tout ce qui touchait à la programmation, Vincent Creuze, pour m'avoir apporté son expertise dans le domaine de la propagation acoustique et son aide lors de la correction du manuscrit, Stéphane Georges pour sa précieuse aide dans l'installation et la configuration des systèmes hôtes du simulateur, et enfin Olivier Tempier pour m'avoir offert ses compétences en électronique.

Ces travaux au LIRMM m'ont donné l'occasion de pouvoir apprécier l'amitié de certaines personnes que je ne saurais oublier : Olivier Strauss (conseils en tout genre et à toute heure), Sébastien Krut (merci de m'avoir encouragé dans cette voie passionnante),

Sébastien Druon (Sébastien 10 / Linux 0 - victoire !), Karen Godary (World Championship... des bavardes !), Olivier Compagnie (bienvenue au club des couches/biberons !), Robin Passama (si seulement le monde était meilleur...), René Zapata (alors ces sockets... elles communiquent ?), Marc Chaumont (on a fini par l'ouvrir cette boîte - désolé pour le bruit !), Ahmed Chemori (aide personnalisée pour tous !), André Crosnier (désolé pour les réunions sauvages dans le bureau...), et Etienne Dombre (merci pour vos conseils et votre soutien).

J'en profite également pour remercier mes compagnons de thèse à commencer par Jean-Minou Spiewakounet (que je suis laid !) qui m'a chaleureusement accueilli au sein de l'équipe. Merci pour tes encouragements (allez courage !), tes séances "briefing" qui ont contribué à me faire garder les pieds sur Terre (mais si je peux !) et à éviter certains écueils (mon PC s'en souvient encore), et ton humour indéfectible (oui... un jour je t'apprendrai le nœud du Caribou !). Je garderai toujours en tête tous ces bons moments passés ensemble, que ce soit les sorties planches (oui on va partir...), les paris de l'impossible (bon OK, là tu m'as ruiné !), ou les manips dans l'eau en plein hiver pour régler les sonars (prends ton temps Bruno, on est là pour ça :), le thésard est robuste et de toute façon jetable).

Un gros merci à mon Creuzounet pour ses encouragements, ses précieux conseils (en tout genre) et son humour exceptionnel diffusé sur SMEQ en stéréo avec Olivier... Que du bonheur. Au fait, je te propose une sortie parapente au lac du Salagou (infesté par les vipères et la leptospirose n'oublie pas). Le départ s'effectuerait de Hombori en avion sur la compagnie Rwakabika Bushi Express. C'est un stagiaire qui a testé la voile, il a dit que tout était parfait donc tu n'as pas de souci à te faire. Il m'a donné une clé USB avec le compte rendu du test, que tu pourras glisser dans le port USB de ton ordi pour le consulter. Partant ? Je voulais aussi te dire que Maëlys est particulièrement intéressée par les recherches que tu mènes et qu'elle est volontaire pour venir te donner un coup de main pendant les manips :) La digne fille de son père !

Je souhaite également remercier Rémi Soulage, pour tous les bons moments passés ensemble. Je me rappellerai toujours de nos (trop)(longues) "pauses café" où l'on refaisait le monde (après tout ce temps, je n'aime toujours pas les chasseurs... et je ne crois toujours pas en Dieu !) et pour toutes les sorties chinois (on aurait du prendre un abonnement). Merci aussi pour ta patience, ton aide ultra-efficace (mais comment fais-tu pour comprendre si vite un problème si mal expliqué...) dans l'urgence au téléphone un certain soir (rahhhhh ces problèmes d'allocation dynamique) et la classe Matrice ! Finalement elle ne m'a pas servi en DEA... mais pendant la thèse ! Merci pour ta gentillesse et je te souhaite bon courage (avec Matlab) pour la dernière ligne droite (fais gaffe elle est longue quand même !).

Merci Abdellah (deuxièèèèèème tournée !) pour ta générosité, ta gentillesse et ton humour à 2 balles... qui me fait bien marrer... J'ai passé de très bons moments en ta compagnie et j'espère que notre amitié perdurera. Je me souviendrai notamment de ces séances cinés à 4H du matin (n'importe quoi...) et des longues discussions sur l'ANNUIT CŒPTIS (tu m'aurais presque fait douter dis donc, mais "The hidden treasure in the Seal is the courage and presence of mind of the people who created it and created these values for the whole country"). Ah oui... et merci aussi pour les cours de Powerpoint... Tu es le plus fort du monde à n'en point douter !

Bien sûr, je tiens à remercier Michel Benoit (OK tu n'es pas un compagnon de thèse... mais je faisais partie de ton équipage à bord du bateau !) pour tous ces moments de franche rigolade impérissables dans mon esprit ! Allez dans le désordre... Une toute petite portière d'un certain Scénic (désolé Bruno ! ...) arrachée tout à fait malencontreusement à coup de bateau... On a (presque) tous bien ri ! Une autre ? Je me souviendrai toujours de mon premier vol plané en autonome... Mais je suis sûque qu'à l'avenir, tu penseras à lever l'ancre avant de mettre plein gaz sur un bateau ; je suis d'accord, le voyage est plus monotone, mais c'est moins risqué ! Bon et sinon j'ai adoré ton bonnet "vieux loup de mer" (je veux le même), et les vidanges sauvages par dessus bord... (sisi on a des photos avec Jean Minou !) Au-delà de ces bons moments, merci pour la confiance que tu m'as accordée pour tripatouiller la torpille (il aura fallu du temps quand même !) et les conseils de rigueur (même seul, j'avais toujours l'impression que tu regardais par dessus mon épaule et je me disais : "si Michel était là...."). Ah oui et aussi...ça va me manquer de ne plus te faire sursauter...

Tant que j'en suis à remercier les ingénieurs... Une pensée spéciale pour Tempiette de mon cœur (dit "La Mauviette") (un petit sourire ? non ? OK OK ne me tape pas !), grand spécialiste (ceinture noire foncée) de Goju-Ryu KuYuKai (sisi ça existe !). J'ai particulièrement apprécié ton aide rapide et efficace accompagnée la plupart du temps de tes froncements de sourcils suspicieux (mais qui bien analysés valent plus qu'un long discours) devant mes réalisations... A ton contact j'ai au moins appris à (la liste n'est pas exhaustive, mais TOUT le reste est moins drôle !) : ranger ma table, rendre les outils, fermer les portes à triple tours, mettre sous alarme, décoder les basses fréquences vocales, et bien sûr... tout remettre dans la caisse ! ! J'ai passé de bons moments en ta compagnie et j'espère que d'autres viendront (j'attends toujours la raclette !). Ah et au fait ! T'as retrouvé la clé ?

Une pensée va naturellement à tous les (vieux) collègues thésards que je n'oublierai pas : Corbel Dallas Multi-pass (mais si tu es fort, tu fais du bon travail, et tu vas réussir à faire le bon choix !... et au fait je n'ai pas oublié le coup du tél. portable, je me vengerai :)), Gaël Pages (1Kg de carambars pour renouveler ton stock de blagues !), Walid Zarrad (merci pour les TDs !), Vincent Nabat (mon hélico s'est crashé :(), Mickaël Sauvée (le découpeur de jambon... merci pour la barbaque !), Aurélien Noce (The GEEK, the best of the world, vive linux), Michel Dominici (le futur, c'est la programmation parallèle !), Philippe Amat (Best @WOW !), Yousra Ben Zaïda (je viens de lire tes remerciements... je vois que je n'ai pas été assez désobligeant avec toi dis moi ? ! :)), Kevin Loquin (ou la force (intellectuelle) tranquille) et tous les autres que je n'ai pas mentionnés...

Je souhaite bon courage à tous les (jeunes) collègues thésards pour l'épreuve de rédaction qui les attend : Rogerio Richa (vivas las Bolivianos ! !, compagnon sanglant de mes nuits studieuses (sisi tes images de cœurs battants à 2h du mat sont répugantes !), Carla Aguiar (Bouh ! ! Ahhhhhhhh ! on a des enregistrements maintenant !), Peter Meuel (Mister Yoyo... 3/1 pour moi !), Xianbo Xiang (FT ! !, bon courage pour la suite mon gars !), Ashvin Sobet (30 millions de Dieux), Baptiste Magnier (tu ne m'as pas connu sous mon meilleur jour (désolé), et j'aurais aimé apprendre à te connaître davantage), Sébastien Lengagne (merci encore pour les articles !), Jérémy Laforêt (Mister EeePC), Vincent Bonnet (le meilleur d'entre nous), Lotfi Chickh (pour ne rien te cacher, je n'ai toujours pas compris ce que tu faisais !), Nicolas Riehl (tu bouches le passage avec tes ficelles !), Marc Bachelier,

Bastien Durand, Sébastien Cotton, Andrea Ancuta, Mourad Benoussaad, Guillaume Souquet, et tous les autres que je n'ai pas eu le temps de connaître ou que j'aurais oubliés... Courage les gars !

Je ne saurais terminer sans rendre hommage à ma famille et à mes proches pour leur soutien inconditionnel. Mes grand-parents, qui chacun à leur manière ont su m'entourer. J'aimerais particulièrement remercier mon grand-père, Raoul, pour le goût de la science qu'il a su m'insuffler dès mon plus jeune âge en essayant (peine perdue) de me protéger des dangers de la chimie (c'est pas bien de donner des fausses formules !), et pour son guidage éclairé de sa sagesse aux moments de décisions pas toujours faciles à prendre.

Mes parents Claude et Sylvie bien sûr, qui m'ont offert la possibilité de poursuivre mes études et à qui je dois beaucoup : ils ont toujours été là pour me soutenir, m'encourager, me conseiller dans les moments les plus durs. Merci maman, pour le temps passé à corriger l'anglais de mes papiers... Tu dois mieux connaître le sujet que moi maintenant !

Ma sœur Florence, qui est en dernière année de thèse (bon.. le droit c'est moins bien que la science, mais je suis admiratif des idées que tu défends. L'idéalisme fait du bien à l'Humanité parfois) et qui a toujours été là pendant ces années pour m'encourager. Je te souhaite à mon tour tout le courage nécessaire à l'accomplissement de ton travail.

Mes beau-parents (exceptionnels, il faut bien le dire - au moins 10 points, hein Bruno ?), qui se sont toujours montrés attentionnés. Un grand merci en particulier à belle-maman, qui est venue soutenir, dans les moments les plus difficiles, la petite famille lorsque celle-ci s'est agrandie !

Mes amis, anciens ou plus récents, à qui je tiens tout particulièrement : Petite Fraise Poilue, Paulo & Charlotte, Bilou du Gabon, Carolaïne, Lopette des îles, Tom, Fred & Eva (l'ordre ne compte pas !).

Galice, qui a largement contribué à cette thèse en m'offrant de grands instants de réflexion au cours de nos promenades en tête-à-tête...

Ma femme, Nolwenn, qui m'a offert le plus grand bonheur que je pouvais espérer : ma fille Maëlys. Merci pour ta patience exceptionnelle, ta gentillesse de tous les jours et ton amour sincère qui m'ont permis de mener ce travail à terme.

Table des matières

Table des figures

Introduction

"L'investigation du fond de la mer ne relève pas de notre seule curiosité ; notre survie en dépend peut-être..." Lorsqu'en 1961, le président Kennedy prononça cette phrase, la surface de la Lune était plus familière à l'homme que les grands fonds marins, un espace plongé dans le silence et les ténèbres, et trois fois plus vastes que l'ensemble des terres émergées.

Depuis les propos de Kennedy, les hommes n'ont cessé de se lancer à l'assaut des fonds marins sur une échelle sans cesse plus grande. Jacques Yves Cousteau et ses *océanautes* ont vécu en communauté sous-marine pendant des semaines dans les eaux de la mer Rouge, tout en opérant des explorations à près de 300 mètres avec la soucoupe plongeante.

Il semblerait donc que l'homme ait une propension naturelle à explorer le milieu qui l'entoure mais, à l'heure où nous envoyons des sondes spatiales toujours plus sophistiquées vers des mondes lointains, où l'on parle d'envoyer l'Homme sur Mars, force est de constater que les profondeurs abyssales demeurent inexplorées ou tout du moins très mal connues.

C'est dans les années 50, que l'exploration des océans connaît un certain engouement qui n'a cessé de croître depuis ; les besoins des hommes, en ressources pétrolières notamment, ont été un véritable moteur pour cette évolution. Les premières plongées qui ont permis de qualifier les grands fonds océaniques (aussi bien l'aspect topographique que géophysique) ont été réalisées à l'aide d'engins submersibles habités.

Depuis les objectifs d'exploration du milieu marin ont évolué et peuvent être divisés en 2 catégories principales :

- Les applications civiles :
  - Exploration/étude du plancher océanique (exploration des failles océaniques, étude de la vie des grands fonds...)
  - Mesure des caractéristiques géophysiques du milieu marin (salinité/température de l'eau, sources sous-marines d'eau douce, caractérisation des principaux courants océaniques dont dépend notre climat, cartographie des pollutions, modèles numériques etc...)
  - Détection et localisation des bancs de poissons
  - Localisation des ressources naturelles (prospection pétrolière, extraction de diamants...)
  - Établissement de cartes bathymétriques

  - Surveillance de structures industrielles (suivi/surveillance de pipelines, inspection de barrages...)
  - Déplacement de charges dans les chantiers sous-marins
  - Détection sismiques, prévention des tsunamis
- Les applications militaires :
  - Établissement de cartes de champs de mines
  - Destruction de mines
  - Suivi et attaque de cible
  - Déploiement de systèmes d'écoute
  - Communications acoustiques à longue portée
  - Guidage d'armes sous-marines
  - Délocalisation/déploiement de capteurs spécifiques

L'ensemble de ces applications a été rendu possible grâce à l'émergence de la robotique sous-marine, dont l'homme ne peut s'affranchir pour nombre de raisons parmi lesquelles :

- Endurance humaine nécessaire pour les missions en eaux très profondes.
- Dangerosité liée au milieu marin : la pression, les mines et certains phénomènes physiques tels que les expulsions de lave en fusion autour des sources géothermiques sont autant d'éléments qui rendent les missions humaines hasardeuses.
- Les coûts : le recours à un robot mobile permet généralement une économie de temps et d'argent.

L'utilisation de robots sous-marins semble donc adaptée aux missions nécessitant une intervention en grand fond, une endurance importante ou dans une zone dangereuse.

De nos jours ces applications mettent en œuvre principalement des engins qui sont soit remorqués, soit téléopérés.

En ce qui concerne les engins téléopérés ou *ROV* (Remote Operated Vehicle), le cordon ombilical dont ils sont pourvus leur permettent d'être alimentés en énergie et commandés d'une part, et de transmettre en temps-réel les informations acquises par leurs capteurs d'autre part. Les principaux inconvénients liés à l'utilisation de ce genre de matériel sont les moyens de surface lourds à mettre en œuvre et la limite de la zone d'activité de l'engin liée à la longueur de cet ombilical. Le bateau à partir duquel l'engin est téléopéré doit être stabilisé. Il s'agit en général d'un bateau spécifique avec un personnel qualifié (le bateau amiral de l'IFREMER, le *Pourquoi Pas ?*, constitue le fleuron de cette technologie en France), et le déploiement du ROV dépend fortement de l'état de la mer.

En ce qui concerne les engins remorqués, ils sont utilisés principalement pour accomplir des relevés bathymétriques ou des inspections de câbles/pipelines. Le déploiement de cette technologie nécessite encore l'utilisation d'un bateau sur zone et d'un câble de remorquage ; l'utilisation de ce type de matériel est également fortement liée à l'état de la mer. De plus la précision d'un relevé bathymétrique effectué avec ce genre de sonar dépend de l'altitude de vol de l'engin remorqué. Par ailleurs, ce matériel ne permet pas de faire de relevés en eau peu profonde : dans les ports, le long des côtes, dans les étangs...

Pour toutes ces raisons, l'utilisation d'un engin autonome semble souvent plus aisée. Les développements technologiques de ces engins sont croissants, mais l'intérêt qu'ils suscitent se heurte à plusieurs problématiques :

- Énergie embarquée à bord
- Autonomie décisionnelle

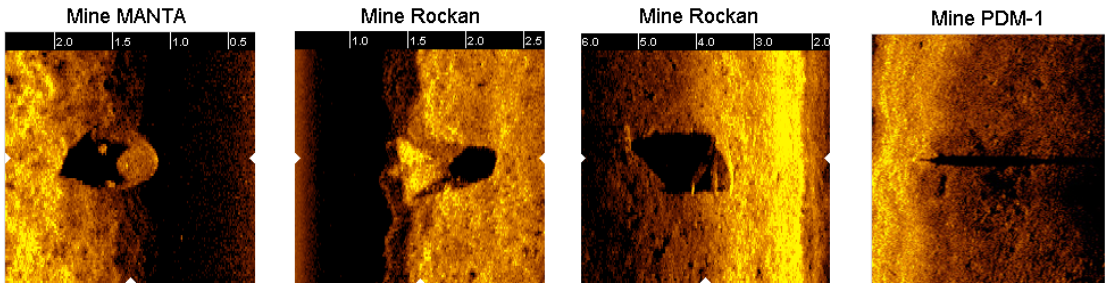


Fig. 1 – Différentes images acoustiques de mines obtenues avec un sonar à 2400kHz

- Positionnement
- Communication
- Mécanique (évolution dans un milieu extrême : haute pression, salinité etc...)
- Évolution dans un milieu peu ou pas connu (mise en œuvre de capteurs spécifiques)

Malgré ces difficultés, que le progès nous permet de surmonter d'année en année, de nombreux résultats prometteurs ont été obtenus dans un large champ d'applications.

Le déplacement des zones militaires de la haute mer vers les zones littorales a suscité l'idée de drones navals autonomes au sein de la Marine Nationale. Ces robots ont donc pour mission la lutte contre les mines ou le renseignement (écoute, cartographie, imagerie,...) avant un débarquement. Le premier intérêt à utiliser des drones pour ce type de mission est évidemment d'épargner des vies ; "Il y a quelques années, explique Pierre Leca (adjoint au chef de la division environnement à la préfecture maritime de Brest), cette technique était à classer au rayon science-fiction. Le milieu marin est en effet beaucoup plus difficilement pénétrable que les milieux aérien ou terrestre, d'ailleurs plus d'hommes sont allés dans l'espace qu'à 6 000 mètres de profondeur. On ne connaît pas le monde sous-marin. La communication, par exemple, y est très délicate ainsi que la navigation en l'absence de repères ou de GPS."

Même si en France ce type de dispositif est encore à l'étude (la Marine Nationale utilise plutôt à l'heure actuelle des engins téléopérés de type PAP (Poisson Auto-Propulsé) [1] ou SPIV (Sonar Propulsé à Immersion Variable) [2]), des engins comme le REMUS [3] existent et sont régulièrement utilisés en mission opérationnelle. Cet engin a par exemple été expérimenté à faible autonomie décisionnelle pour le déminage de port en Irak (voir la figure 1).

Parmi les missions qu'effectuent couramment les AUVs (Autonomous Underwater Vehicle), on peut également citer la bathymétrie ou le mapping. C'est le cas par exemple de l'AUV du MBARI [4] (Monterey Bay Aquarium Research Institute)(voir figure 2), qui possède un engin autonome comportant trois sonars. Chacun de ces sonars mesure un aspect distinct du plancher océanique. Les avantages d'un tel système sont nombreux : coût réduit, capacité à scanner une zone précise, capacité à naviguer près du fond (environ 30m pour cet AUV) ce qui améliore considérablement la netteté des images acquises et enfin la rapidité d'acquisition (environ trois fois plus rapide que l'utilisation de leur ROV). Cet AUV a par exemple pu acquérir en 3 dimensions une partie du canyon de la baie de Monterey (voir figure 3). Enfin, pour donner un dernier exemple de domaine dans lequel les AUVs sont couramment utilisés, on peut mentionner l'océanographie physique. Il s'agit de l'étude de l'état et des processus physiques au sein de l'océan. Ainsi il est

Fig. 2 – AUV utilisé au MBARI comportant 3 sonars.

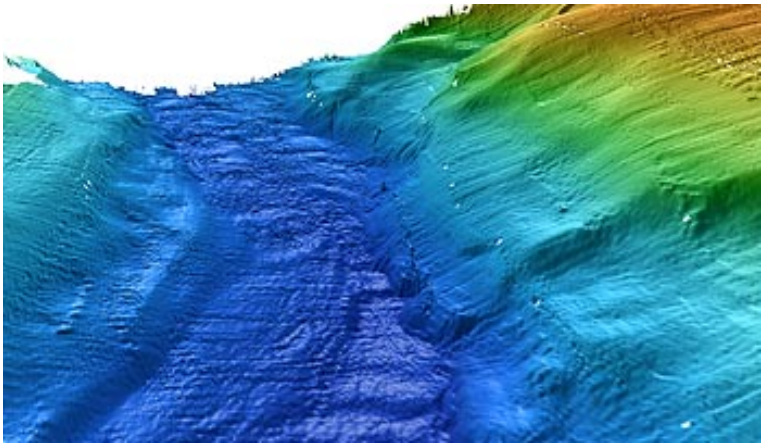


Fig. 3 – Exemple de rendu d'acquisitions faites par l'AUV et un ROV

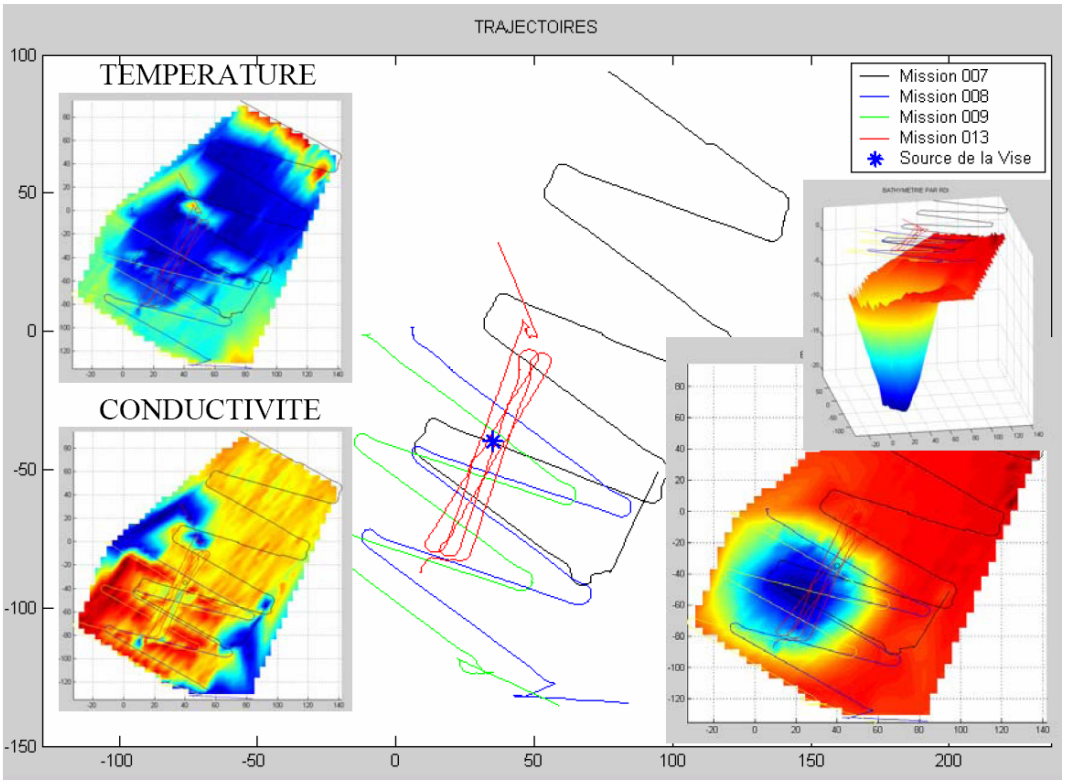


Fig. 4 – Acquisitions réalisées par Taipan2 sur la source de la Vise

plus aisé d'utiliser un AUV pour effectuer ce type de relevé pour plusieurs raisons, parmi lesquelles le coût, l'accessibilité des zones géographiques, la rapidité du relevé permettant de conserver dans une certaine mesure la cohérence spatio-temporelle : un phénomène évoluant dans le temps et dans l'espace (par exemple le panache d'une source d'eau douce sous-marine, ou bien la plume d'une rivière se jetant dans la mer), il est nécessaire d'obtenir une "capture" aussi instantanée que possible sous peine de ne pas obtenir un relevé correspondant à la réalité du phénomène observé. Taipan 2 [5], l'AUV du LIRMM (Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier) a effectué un relevé de la source de la Vise [6](voir figure 4) en Février 2006.

Contexte

L'utilisation d'un AUV pour ce type de mission apporte donc de réels bénéfices. La question qui vient naturellement concerne donc l'extension de l'utilisation d'un seul engin à une multitude d'engins potentiellement hétérogènes. En effet, un bon nombre de tâches ne peuvent être accomplies par un seul engin ou du moins sont largement perfectibles en multipliant le nombre de capteurs déployés simultanément.

Actuellement peu d'expérimentations ont mis en œuvre plusieurs engins autonomes (que ce soit des engins de surface, des drônes aériens, des véhicules terrestres, ou des engins sous-marins) simultanément. Pour autant, on imagine aisément les avantages que pourraient procurer ce genre de déploiement, parmi lesquels :

– une exploration systématique et très rapide du milieu

- lorsque la dynamique de l'environnement est importante, un seul engin peut ne pas suffire à percevoir l'aspect spatio-temporel du milieu à explorer : un seul véhicule a besoin de temps pour se déplacer d'un point de mesure à un autre, et la situation peut être instationnaire par rapport à ce temps d'acquisition.
- il est difficile de regrouper plusieurs capteurs spécialisés sur un même engin (problème de place, d'énergie, d'interférences...)
- redondance des données acquises permettant de qualifier leur niveau de crédibilité
- relevés multi-échelles permettant d'adopter la résolution spatiale requise pour capter un phénomène

Le déploiement de plate-formes multi-capteurs sous-entend donc qu'il est nécessaire de mettre en œuvre une commande et une stratégie de commande coordonnée de la formation, permettant l'utilisation de flottilles d'engins autonomes, tout en garantissant des performances et en tenant compte des contraintes du milieu (faible bande passante, difficulté de localisation ...).

Le développement et le contrôle d'engins en flottille sont loin d'être des tâches aisées. Cela est d'autant plus vrai pour les engins sous-marins, toujours difficiles à superviser durant les expérimentations. Cela tient principalement au fait qu'il n'est pas possible de suivre visuellement le déplacement d'un AUV lorsqu'il effectue une mission. Par ailleurs la localisation précise d'un véhicule sous-marin constitue une réelle problématique : que ce soit la mesure du vecteur d'état (position, vitesse, attitude...) par l'engin lui-même, ou par un opérateur extérieur, les solutions existantes sont lourdes à mettre en place (centrale inertielle de plusieurs centaines de kilo à bord des sous-marins nucléaires, ou déploiement de systèmes de bouées pour l'écho-localisation) et onéreuses. Enfin il faut mentionner la problématique de la communication aquatique dont la bande passante est limitée (quelques dizaines de bits utiles par seconde pour les distances concernées), sujette aux bruits ambiants qu'ils soient naturels (pluie, vent, animaux...), liés aux activités humaines (bateaux, chantier sous-marin, etc...), dus aux réverbérations ou aux phénomènes de multi-chemin (ondes acoustiques réfléchies par l'environnement, atteignant un recepteur avec un décalage temporel). Ce dernier point en particulier constitue un verrou technologique majeur pour la coordination de la formation des véhicules. Devant la diversité des applications et des engins ([7] et [8]) pouvant être mis en œuvre, plusieurs stratégies peuvent être envisagées pour la navigation en flottille. On peut par exemple considérer que les décisions ne sont prises uniquement que par un véhicule *Leader*, ou bien qu'au contraire les décisions sont prises en compte sous forme de votes au vu d'une situation donnée par tout ou partie des participants. Quelle que soit la stratégie retenue, celle-ci doit garantir la réussite de la mission, en partageant une quantité d'informations limitée.

Il n'est dès lors plus concevable de mettre en œuvre une flottille d'engins autonomes sans avoir testé au préalable la faisabilité de la mission, le comportement logico-temporel des engins, l'efficacité des algorithmes employés, le bon fonctionnement de tous les sous-systèmes ; et ceci à plus forte raison, lorsque les engins sont hétérogènes (subsurfaciques, surfaciques, aériens typiquement), proviennent d'organisations différentes (laboratoires, industriels...), et sont rassemblés pour l'accomplissement d'une mission commune. C'est pourquoi les outils de simulation jouent ici un rôle clé : ils nous permettent de tester et va-

lider nos lois de commande, notre architecture de contrôle et de détecter les inconsistences préliminaires des scenarii. Par ailleurs, ces technologies limitent les ressources humaines, réduisent le nombre et la difficulté des expérimentations nécessaires et par conséquent, le coût et le temps dépensé à mettre au point ces systèmes.

La problématique de la coordination de flottille est aujourd'hui une thématique de recherche complexe, étendue et très active ([9], [10], [11], [12]). Dans ce contexte, nos travaux ont pour objet l'étude et la proposition d'un outil de recherche, sous la forme d'un simulateur collaboratif, permettant de valider les nouvelles stratégies de commande de coordination de véhicules autonomes, et de préparer les expérimentations réelles. Par collaboratif, nous entendons ici qu'un tel outil ne peut être développé seul : il est nécessaire de créer une architecture pouvant accueillir des modèles créés par différentes équipes car le domaine de compétences à maîtriser est vaste. De fait, notre problématique est de concevoir un système de simulation permettant d'envisager le test et la validation de stratégies de commande de véhicules en flottille grâce notamment à la prise en compte des communications, et de préparer ces missions en testant le véritable contrôleur des véhicules impliqués. Cette thèse doit donc conduire à la création d'un outil opérationnel pour l'équipe de robotique sous-marine du LIRMM.

Objectifs

Ce manuscrit a pour objectif d'apporter des réponses aux questions suivantes :

- Pourquoi la simulation est un outil nécessaire dans le cadre de l'étude de la coordination multi-véhicule ?
- Que peut-on attendre d'un système de simulation dans ce cadre ?
- Quels sont les travaux précédents concernant la simulation de véhicules autonomes permettant au minimum la simulation d'engins sous-marins ?
- Quels phénomènes doit-on prendre en compte selon le but recherché ?

Ces questions nous amènent naturellement à déterminer :

- Quels sont les modèles pertinents à implémenter ou développer au regard des objectifs que nous nous sommes fixés ? Quel est le degré de finesse nécessaire ?
- Avec quelle granularité spatiale et temporelle doit-on mener une simulation dans un cadre donné ?
- Quel est le domaine de validité dans les quatre dimensions (espace et temps) de l'exploitation du système ?
- Quels sont les contraintes à respecter pour permettre une exploitation scientifique du système de simulation, *ie* le simulateur ?

et de fait, à s'interroger sur l'impact de l'architecture sur le simulateur :

- Comment architecturer le simulateur et rendre générique les entités participant à la simulation ?
- Pourquoi et comment distribuer les calculs sur une grille d'unités de traitement en réseau ?
- Quels sont les contraintes liées à cette distribution et quelles sont les implications sur le résultat de la simulation ?

dans le but de créer un simulateur Hybride/Matériel-dans-la boucle :

- Comment concevoir une architecture logicielle permettant de réaliser des simulations hybrides pour des engins autonomes hétérogènes ?

- Comment connecter un robot à un tel système en minimisant l'impact de cette connexion sur le comportement de son architecture de contrôle ?
- Comment faire en sorte que, vu du robot, une mission réelle ou simulée soit exécutée de la même façon ?

dans ce domaine intrinsèquement pluri-disciplinaire :

- Comment intégrer l'ensemble de ces réflexions dans l'implémentation d'un prototype de simulateur fonctionnel ?
- Quels environnement et outils offrir pour permettre un travail collaboratif ?

Organisation du manuscrit

La première partie de ce manuscrit propose une étude des simulateurs existants. Cette étude débute par un état de l'art du domaine. Nous portons une attention particulière au cours de cette étude sur un ensemble de points que nous définissons comme étant critiques dans le cadre de la préparation de missions multi-véhicules.

Par la suite, nous faisons une proposition concernant la classification de ces simulateurs : les critères classiques utilisés par la communauté pour distinguer les simulateurs présentent des limites que nous proposons de repousser grâce à notre nouvelle approche.

Enfin cette partie s'achève avec le positionnement de notre travail par rapport à l'existant, au cours duquel nous précisons point par point les aspects fondamentaux du simulateur que nous proposons.

Nous exposons dans la deuxième partie notre proposition d'un nouveau simulateur hybride permettant de préparer les expérimentations multi-véhicules. Nous commençons par évoquer dans un premier chapitre l'architecture de ce simulateur, qui a pour objectif de permettre le travail collaboratif autour d'une expérimentation commune faisant intervenir des véhicules hétérogènes appartenant à différentes entités.

Nous abordons ensuite la réalisation de ce simulateur en précisant les technologies que nous avons retenues pour le développer. L'objectif de ce chapitre est de démontrer clairement quels sont les implications de ces choix sur le respect des critères fondamentaux du système de simulation proposé.

Nous présentons ensuite la partie modélisation. En effet, les modèles mis en œuvre dans ce simulateur sont nombreux et variés (modèles dynamiques des véhicules, modèles de capteurs, d'actionneurs, de moyens de communication, d'environnement (salinité, courant, collisions, ...)), et nous étudions pour chacun de ces modèles, les implications sur le domaine de validité des hypothèses de départ. Nous proposons dans ce chapitre un ensemble de modèles permettant de prendre en charge les communications inter-véhicules au sein de la flottille, en respectant les contraintes physiques liées au milieu de propagation, et aux moyens de communication utilisés. En revanche, nombre de modèles que nous utilisons sont issus de la littérature classique, et sont donnés en annexe.

Enfin la troisième partie conclut notre travail. Nous présentons d'abord la méthodologie que nous avons utilisée pour valider notre simulateur. L'objectif n'est pas ici de valider les modèles, travail qui sort du cadre de cette thèse, mais plutôt de valider l'architecture de simulation, au sein de laquelle ils sont mis en œuvre.

Puis nous présentons une série de simulations typiques, mettant en avant l'exploitation du système de simulation. Nous proposons des exemples de missions faisant intervenir un ou plusieurs véhicules, éventuellement hétérogènes, au cours desquelles les engins doivent mettre en œuvre un capteur extéroceptif et maintenir un lien de communication leur permettant de se coordonner afin de mener à bien la mission.

Première partie

Etude des simulateurs de véhicules autonomes

Introduction

Cette première partie est consacrée à l'étude des simulateurs de véhicules autonomes. Nous nous focalisons particulièrement sur la simulation des engins sous-marins, car il s'agit du domaine de recherche de notre équipe. Nous débutons cette première partie par l'évocation des critères classiques d'évaluation des simulateurs. Nous montrons que ces critères se révèlent insuffisants pour notre contexte, et nous proposons une nouvelle grille d'évaluation nous permettant de distinguer les différents simulateurs existants.

Dans un deuxième chapitre, nous nous attachons à classifier les différents simulateurs existants, afin d'être en mesure de préciser le contexte dans lequel nous nous plaçons pour faire notre proposition. Nous montrons en particulier en quoi les solutions existantes ne satisfont pas les principes fondamentaux que nous nous sommes fixés.

Enfin, nous menons une réflexion sur la construction de l'architecture d'un système de simulation. La conception de cette architecture va être guidée par la nécessité de disposer d'une plate-forme expérimentale, permettant de tester des lois de commande multi-véhicules (notamment sous-marins) et de préparer des expérimentations réelles.

Chapitre 1

État de l'art

1.1 Introduction

Le développement d'un véhicule autonome et, à plus forte raison d'un véhicule autonome sous-marin est loin d'être une tâche aisée. En effet, au-delà des défis techniques que représentent bien souvent la mécanique et l'électronique de tels engins, se cache une architecture logicielle dite "intelligente" ayant en charge la conduite du véhicule durant la mission. Cette architecture doit être en mesure de faire évoluer le véhicule dans un environnement non structuré et la plupart du temps inconnu. Elle doit pouvoir faire face à un ensemble d'événements en adoptant les réactions adéquates. De plus, une contrainte forte pour cette partie logicielle est sa capacité à fonctionner en temps-réel. Si, il y a quelques années encore, les engins autonomes étaient gérés par quelques microcontrôleurs, la puissance, la miniaturisation et la baisse de consommation des processeurs d'aujourd'hui nous permettent d'envisager des architectures et donc des scenarii toujours plus complexes. Ce genre d'architecture demande donc un nombre de tests de mise en situation extrêmement important afin de s'assurer de leur bon fonctionnement. En effet, plus les scénarii auxquels doit faire face l'engin sont complexes, plus le nombre de tests permettant d'envisager toutes ces situations sont nombreux. Lors du développement de tels engins, une des contraintes est bien souvent le nombre de personnes (chercheurs, ingénieurs, techniciens) y travaillant simultanément.

Cela rend souvent l'engin indisponible pour effectuer des tests durant la phase de développement. De plus des éventuelles erreurs dans le code du logiciel de bord sont difficiles à éviter et peuvent mener, dans bien des cas, à la perte ou à la détérioration de l'engin ; et lorsque l'engin est récupéré, le code n'est pas toujours modifiable sur place et conduit à l'annulation de l'expérimentation. Tout ceci ajouté à la difficulté de mise en œuvre des manipulations, du coût engendré, des ressources humaines et matérielles impliquées et du temps passé, diminue l'efficacité de ces tests.
Si ces remarques sont valables pour un seul engin, on imagine aisément ce qu'il advient lorsqu'on envisage de recourir simultanément à plusieurs engins, potentiellement hétérogènes, pour effectuer une même mission. En effet, même si l'on est aujourd'hui capable de réaliser un AUV fiable d'une part, et de concevoir des scenarii complexes avec des algo-

rithmes performants d'autre part, les équipes rencontrent plusieurs difficultés pratiques ; c'est sans doute ce qui explique en partie le peu d'expérimentations dans ce domaine. Parmi ces difficultés, on retrouve principalement :

- L'utilisation de matériel hétérogène (des engins de classes et/ou types différents, provenant de différentes organisations). En effet, un véhicule autonome est souvent onéreux et les équipes de recherche sont généralement spécialisées dans un type d'engin précis (AUV, UAV (Unmanned Aerial Vehicle), USV (Unmanned Surface Vehicle), UGV (Unmanned Ground Vehicle)). Les véhicules de chaque acteur sont donc conçus différemment (par exemple les moyens de communication peuvent ne pas être compatibles voire même interférants) et possèdent une architecture de contrôle propre. Cela signifie concrètement que pour effectuer une mission commune, chaque engin sera programmé de façon idoine : il n'est pas évident par exemple que chaque architecture logicielle puisse prendre en compte toutes les contraintes de la mission (par exemple interdiction d'utiliser certains capteurs interférents). Des solutions pour pallier le problème peuvent être mises en place, mais doivent être validées durant une phase de tests dans lesquels l'aspect temporel doit être pris en compte.
- Ensuite il faut mentionner le problème de la logistique, lourde à mettre en place : on peut imaginer un scenario dans lequel interviendrait classiquement un USV, un AUV et un UAV. Cela sous-entend donc que l'on trouve un lieu adéquat, que toutes les équipes puissent se déplacer avec leur matériel un même jour, que l'on transporte et mette en place tout l'équipement (radars de poursuite, GIPS (GPS Intelligent Buoys, ateliers, ...), et bien sûr que tous les véhicules soient fonctionnels le jour J (respect des contraintes de temps). Ici encore, une plate-forme de test se révèle nécessaire pour diminuer les échecs d'expérimentations et donc éviter au maximum la lourdeur de la logistique.
- Superviser une expérimentation réelle dans laquelle interviennent de nombreux engins constitue sans aucun doute une difficulté supplémentaire. En effet, si pour le milieu aérien (UGV, USV, UAV) des solutions performantes existent, il n'en va pas de même pour le milieu sous-marin. Les systèmes de localisation (au moins pour des grands volumes de mission) ne sont pas très précis et souvent lourds à mettre en place. Bien que la simulation ne puisse pas résoudre ce type de problème, elle permet néanmoins d'entraîner les équipes à la supervision des opérations.
- Par ailleurs, chaque équipe conçoit des algorithmes potentiellement différents pour accomplir une même tâche. Cela signifie par exemple que les stratégies employées pour rechercher un objet peuvent ne pas être compatibles : un AUV peut faire un quadrillage d'une zone alors qu'un autre sera programmé pour effectuer une recherche spiralaire. Un équilibre doit être trouvé entre la liberté de développer des algorithmes pour son propre robot et l'obligation de rendre ces algorithmes compatibles pour effectuer une mission commune. Là encore une plate-forme de test s'avère indispensable pour vérifier la cohérence du comportement de l'ensemble.
- En outre il est nécessaire de considérer les problèmes dus à l'utilisation de capteurs et/ou moyens de communication interférents. On peut par exemple imaginer que si deux AUVs suffisamment proches, cherchent à utiliser leur sonar opérant sur la même bande de fréquence, aucun des deux ne sera en mesure de faire des acquisitions correctes. Différentes stratégies peuvent être mises en place (Time Sharing, synchronisation, événementiel...), mais là encore il faudra tester les solutions retenues en

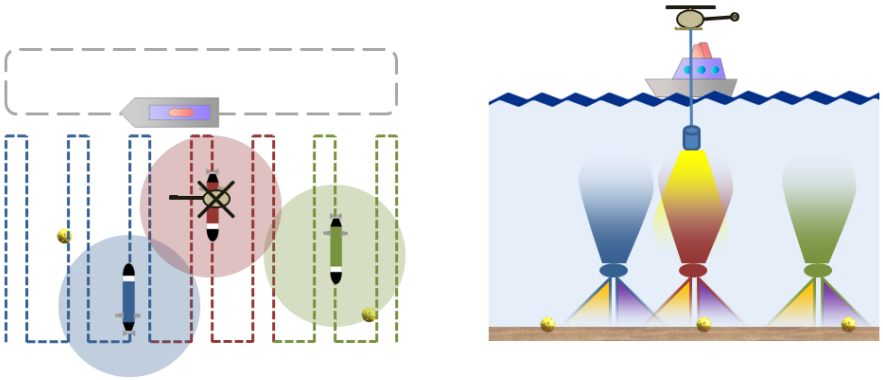


Fig. 1.1 – Trois AUVs cherchent d'éventuelles mines sous-marines. Lorsqu'une menace potentielle est détectée et qualifiée, on recommence toute la mission avec un critère de performance plus important.

veillant à l'aspect temporel.

- Qui dit flottille, dit généralement communications. Il semble évident que l'adoption d'un modèle de communication commun est obligatoire pour les missions coordonnées. Les protocoles de chacune des couches de ce modèle seront à choisir (ou inventer ?) en fonction des moyens de communication qui seront mis en œuvre (radio, WIFI, modem acoustique, liaison satellite, ...), des media de propagation (eau, air), de la criticité des données (télécommande, envoi mission, récupération de données, ...). Un simulateur pourra tester l'implémentation dans les véhicules des différentes couches de ce modèle de communication commun.
- Pour finir, la complexité des scenarii et le nombre d'engins hétérogènes mis en jeu rendent la cohérence de la mission hasardeuse. En effet, il devient rapidement difficile de mesurer toutes les implications des contraintes/ordres donnés aux véhicules. Par exemple, la figure 1.1 (d'après une idée originale de M. Devie) présente une mission de déminage envisagée par le GESMA (Groupe d'Etudes Sous-marines de l'Atlantique) effectuée par trois AUVs et un UAV. La mission des AUVs consiste à détecter l'ensemble des contacts (typiquement des mines) afin de les classer et de les identifier. Afin de minimiser les interférences entre les sonars et les modems acoustiques les véhicules travaillent suffisamment éloignés pour ne pas être perturbés, mais suffisamment proches pour pouvoir être à portée de communication par intermittence (en milieu de rail). Un drône aérien vient régulièrement mouiller un modem afin de récupérer les informations à transmettre vers le bateau (la communication verticale permet des débits plus élevés qu'à l'horizontal). Un simulateur permet clairement de tester les implications des prérogatives individuelles des véhicules sur la réussite globale de la mission.

Le développement d'un engin complexe et à plus forte raison la conception d'un scenario mettant en œuvre une multitude d'engins, nécessite donc une plate-forme de tests commune à travers laquelle chaque entité pourra réaliser des tests préalables pour accélérer et fiabiliser son développement. Un simulateur permettant de tester avec un maximum de sécurité, un gain de temps et d'efficacité apporte une réelle amélioration de la phase de

développement de l'engin et de la préparation des missions. L'augmentation de la capacité de traitement des systèmes informatiques a rendu tout cela possible, au moins dans une certaine mesure. De nombreux simulateurs existent déjà et sont couramment utilisés. Comme nous le verrons à partir de la section 1.3, il y a plusieurs types de simulateurs, chacun d'eux permettant de se focaliser sur des aspects différents ; conception de commandes, validation du comportement d'une architecture, test de stratégie d'acquisition, sont des problématiques pouvant être abordées en simulation. Bien évidemment cette simulation ne peut s'envisager que si nous sommes en mesure de connaître les lois physiques régissant le système étudié, étant entendu que ces lois ne sont qu'une vue de l'esprit permettant de décrire plus ou moins finement la réalité. Il est important de déterminer d'emblée *l'objectif* de la simulation, son *contexte d'exploitation*, son *domaine de validité* en tenant compte des *approximations* faites lors de la *modélisation*. Cela nous permettra d'évaluer plus tard dans quelle mesure les résultats obtenus sont représentatifs de la réalité et de fait, le *degré de confiance* que l'on peut avoir dans les conclusions tirées de son *exploitation*.

Une fois ce travail effectué, il est nécessaire de trouver un compromis entre précision, fonctionnalités, ergonomie et temps de développement.
Cet état de l'art est donc destiné à produire une vue d'ensemble sur les simulateurs existants, particulièrement focalisé sur les engins sous-marins.

Nous avons choisi d'utiliser les classes de simulateur telles qu'elles sont définies dans [13] pour cataloguer les différents simulateurs existants. Quatre classes sont ainsi définies : les simulateurs hors-lignes, en-lignes, matériel-dans-la-boucle (HIL), et hybrides.

1.2 Remarques préliminaires et angle de vue

Beaucoup de chercheurs utilisent la simulation pour leurs travaux ; il faut néanmoins définir correctement ce qu'est la simulation. Modéliser un système réel constitue la première étape de la simulation. Les *modèles* de simulation peuvent être basés sur des représentations physiques, mathématiques, comportementales... Ces modèles sont parfois alimentés par des données empiriques. Un système de calcul permet alors d'animer le modèle : il s'agit du processus de *simulation*. Cet outil est alors utilisé pour mieux comprendre, optimiser, améliorer la fiabilité ou tout simplement étudier les systèmes. Par ailleurs, la simulation est largement utilisée pour vérifier la cohérence de la conception (forme du véhicule, lois de commande, modèle de communication...).

Il faut donc définir clairement le cadre dans lequel le système modélisé est exploité, en d'autres termes le *cadre expérimental*, en s'assurant que cette *exploitation* soit dans le *domaine de validité* du simulateur. *Système étudié* (donc contrôle et/ou observé), *modèles* (du système étudié et de l'environnement), domaine de validité et cadre expérimental, sont les aspects fondamentaux de la simulation [14], et par conséquent du (ou des) simulateurs attenants(s).

1.3 Les simulateurs hors-ligne

Comme nous l'avons évoqué, la simulation joue un rôle primordial dans les différentes phases de développement d'un AUV. Son atout majeur est la possibilité de prévoir et de corriger les défauts éventuels qui pourraient mener à une mise en service longue et

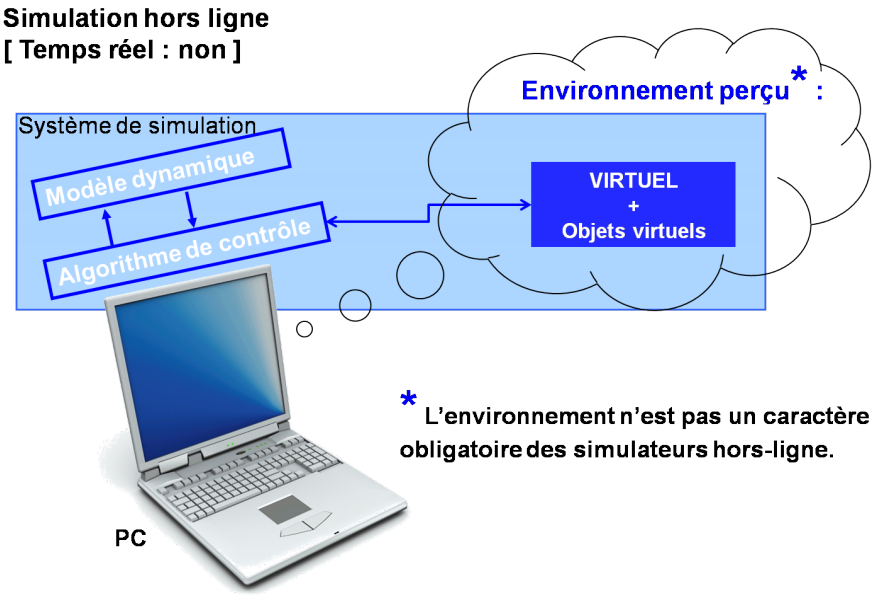


Fig. 1.2 – Un simulateur hors-ligne n'utilise ni le logiciel ni l'ordinateur du robot réel. Un ou des algorithmes de contrôle interagissent avec un modèle dynamique sur un ordinateur. Éventuellement il est possible d'ajouter un environnement virtuel pour tester certains algorithmes faisant intervenir des capteurs extéroceptifs.

coûteuse des véhicules. Les simulateurs *hors-ligne*, encore appelés simulateurs *offline*, correspondent à une classe de simulateur permettant de mettre au point les lois de commande et les algorithmes de traitement de l'information ; ils permettent notamment d'en déterminer les performances théoriques (voir figure 1.2). Durant la phase de conception par exemple, les simulateurs hors ligne sont très utiles. L'outil MATLAB/Simulink se révèle ainsi être un excellent outil pour réaliser ce type de simulation. En effet, il existe de nombreuses toolbox disponibles pour les robots sous-marins, incluant bien sûr les AUVs [15]. Cette toolbox par exemple, met à disposition un environnement qui propose les ressources nécessaires pour une implémentation rapide des modèles mathématiques des systèmes marins et permet ainsi à l'utilisateur de se focaliser sur la conception du contrôle.
Bien que cette toolbox soit utile pour la simulation du modèle des robots et pour la conception de la commande, elle ne reproduit pas tous les aspects de la réalité (notamment celle du monde extérieur). Les capteurs extéroceptifs tels que sonars et autres caméras, ne sont donc pas modélisés. Cela représente une limitation majeure, lorsque l'on souhaite tester des architectures de contrôle perfectionnées qui interagissent avec l'environnement ou avec d'autres engins leur permettant de prendre des décisions ; de telles architectures ne peuvent être testées avec ce genre de simulateur. Il est donc nécessaire de compléter ces toolboxes avec du code permettant l'utilisation de capteurs extéroceptifs. C'est par exemple le cas dans [16] qui utilise un environnement Matlab/Simulink incluant la simulation d'un monde virtuel pour tester en simulation différents contrôles d'un AUV. Bien que l'environnement de simulation permette l'utilisation d'un sonar, d'autres capteurs extéroceptifs, comme un capteur de salinité ou une caméra ne peuvent être utilisés.

Dans [17], un simulateur 3D a été développé en C++ dans lequel a été implémenté la

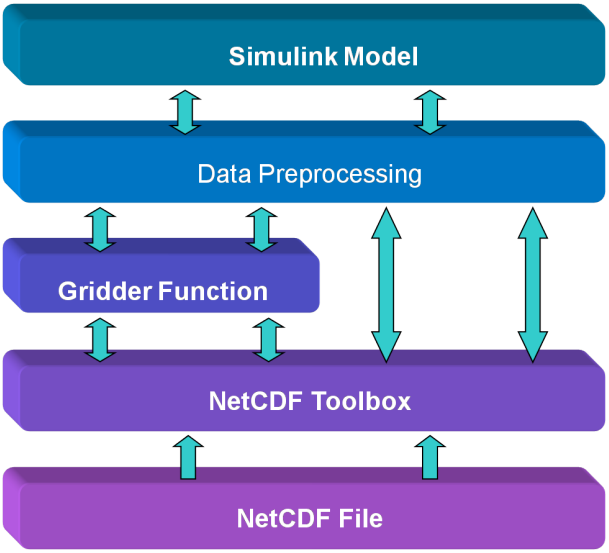


Fig. 1.3 – Incorporation d'un fichier netCDF dans le modèle Simulink

vision par sonar et caméra. En utilisant les capacités d'OpenGL, les auteurs parviennent à une simulation hors ligne réaliste du système de vision.

Dans [18], les auteurs ont développé un simulateur hors-ligne qui permet de tester des algorithmes adaptatifs d'échantillonnage de l'océan (mesures caractéristiques physiques permettant de produire un maillage). Ce type d'algorithme permet d'optimiser la trajectoire des plate-formes de mesures en temps-réel et sans intervention de l'homme. Cette optimisation basée sur le partage des données acquises fait l'objet de nombreuses recherches. Pour pouvoir tester ces algorithmes les plate-formes doivent donc pouvoir communiquer. Le simulateur supporte cette fonctionnalité en permettant aux entités de partager les données acquises en utilisant un espace de mémoire partagée (workspace de Matlab). Un aspect intéressant de ce simulateur réside dans le fait que les données environnementales (typiquement la répartition des températures et de la salinité dans l'océan) sont incorporées au modèle utilisé dans la simulation à partir d'un format de données très utilisé dans la communauté des océanographes (Network Common Data Form appelé netCDF). Une toolbox de Matlab est utilisée pour extraire et manipuler les données contenues dans le fichier netCDF. Une fois ces données extraites, elles sont réparties sur une grille (Arakawa C-Grids) avant d'être incorporées au modèle Simulink (voir figure 1.3).

Dans [19], les auteurs présentent un simulateur capable de reproduire les comportements d'AUVs. Toutes les entités appartenant à la simulation sont décrites sous forme de classes, puis on décrit les interactions entre les objets instanciés (voir figure 1.4). Cette programmation est réalisée en langage SHIFT (langage destiné à décrire les réseaux dynamiques d'automates hybrides) qui permet de développer rapidement ce genre d'applications (4000 lignes de code en SHIFT équivalent à 14000 lignes de code en C d'après les auteurs). Ce simulateur a été créé dans le cadre de l'océanographie côtière et permet de simuler plusieurs AUVs simultanément. Le but est de tester la logique des architectures de contrôle pour le multi-AUV. Le simulateur met en œuvre un modèle d'AUV à six ddl (degrés de liberté), différents phénomènes océanographiques, les interactions entre les véhicules et l'environnement, des modèles de capteurs et d'actionneurs et enfin les communications. Ce simulateur a été utilisé pour tester une mission mettant en œuvre

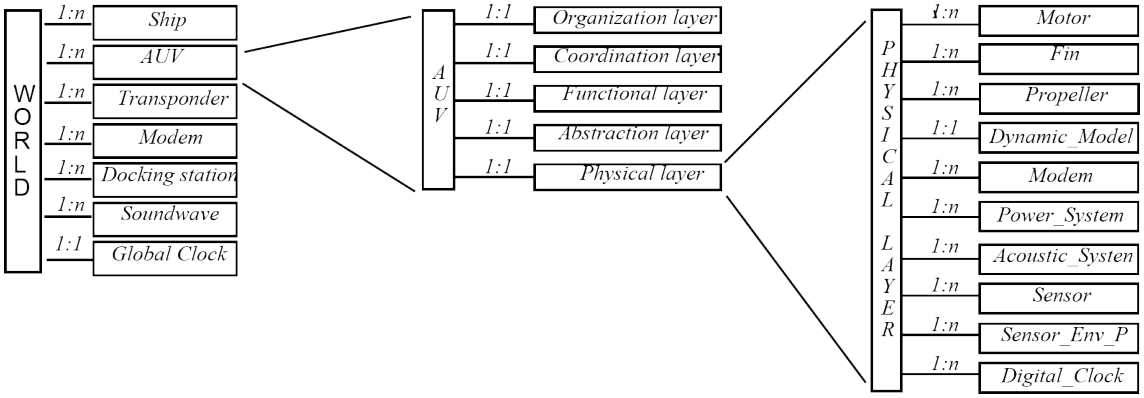


Fig. 1.4 – Représentation graphique des entités et de leur hiérarchie décrite en SHIFT

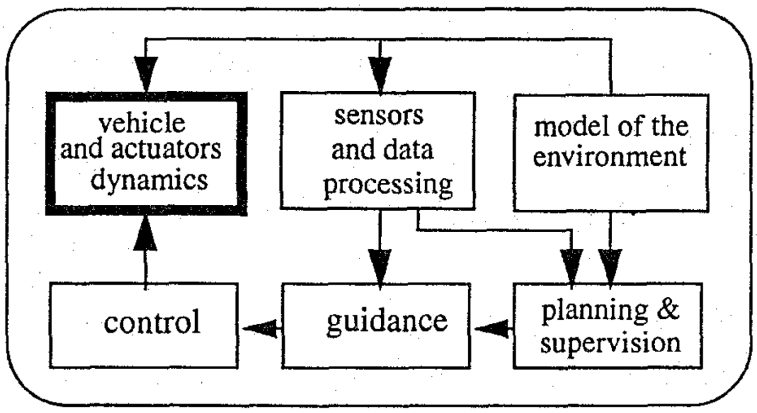


Fig. 1.5 – Architecture du simulateur

six AUVs *Phoenix*.

Dans [20], les auteurs exposent le fonctionnement d'un simulateur pour un ROV. Il a été pensé pour permettre de réaliser des tests comparatifs et des évaluations de performance pour plusieurs contrôles et guidages, dans des situations proches de celles de la réalité (courant, absence de GPS, présence d'un câble ombilical, ...). Ce simulateur a été développé dans l'environnement Matlab/Simulink et est divisé en plusieurs librairies facilement interchangeables pour réaliser différents tests. L'architecture de simulateur est présentée figure 1.5.

Enfin dans [21], les auteurs ont développé un simulateur appelé *RobSym* pour leur AUV *Roby*. Ce simulateur comporte un modèle dynamique simple permettant de reproduire le comportement de Roby ; il est utilisé pour tester les algorithmes de contrôle, de planification et de filtrage des données capteurs. Ces algorithmes peuvent ensuite être implémentés sans presque aucune modification sur le contrôleur du robot.

Il est à noter que :

- La granularité dans le temps et l'espace d'un modèle a des conséquences sur les résultats de l'exploitation du simulateur. Il est donc nécessaire de déterminer la granularité requise en fonction de ce que l'on souhaite observer : il ne sera pas forcément intéressant de modéliser les marées à la microseconde pour tester l'efficacité d'une loi de commande d'un AUV...

– Par ailleurs, sur un plan temporel, une seconde de simulation ne vaut pas une seconde dans le monde réel. Par exemple dans [17], le temps simulé s'écoule plus rapidement que dans la réalité. Cela s'avère particulièrement pratique lorsqu'on a recours à des algorithmes génétiques par exemple puisqu'il permet au système de trouver rapidement une solution. Dans d'autres cas, la charge de calculs à effectuer à chaque pas de la simulation est tellement importante, qu'une seconde de temps simulé met bien plus d'une seconde à être calculée. C'est par exemple le cas dans [19] où la simulation prend trois fois plus de temps que la mission réelle. Quoi qu'il en soit, dans un cas comme dans l'autre, l'aspect temporel de la simulation n'est pas pris en compte, et rend l'algorithme de commande inopérant lorsqu'il est transféré sur un robot réel.

Ce type de simulateur n'est donc pas totalement adapté à notre contexte ; en effet, s'il demeure possible de tester les propriétés de convergence de lois de commande permettant à plusieurs véhicules de naviguer en formation, il est en revanche impossible de tester le comportement temporel de l'architecture de contrôle.

Une autre classe de simulateurs permet de prendre en compte dans une certaine mesure l'aspect temps-réel des simulations. Ces sont les simulateurs dits *en-ligne*.

1.4 Les simulateurs en-ligne

Les simulateurs en ligne (*online*), constituent une classe de simulateurs capables de maintenir la cohérence temporelle, tout au moins dans une certaine mesure. Ces simulateurs prennent en compte les propriétés temporelles des algorithmes testés. Typiquement, ce genre de simulateur va prendre en compte le temps d'exécution d'un algorithme, les délais de traitement de l'information... Le simulateur exécutera les algorithmes de contrôle à la même fréquence que celle utilisée sur le robot réel, tout en leur fournissant les données capteurs au même taux d'échantillonnage que les capteurs réels. Le logiciel de contrôle réel du robot est généralement utilisé dans ce type de simulation (voir figure 1.6).

La simulation en ligne joue donc un rôle important dans les phases de développement et de test, car il est rare d'avoir à sa disposition plusieurs engins. Cela permet donc à plusieurs personnes de travailler simultanément sur divers aspects de l'engin (contrôle, architecture logicielle, etc...) sans avoir besoin pour autant d'en disposer. Nous pouvons tester les propriétés de convergence dans le temps des algorithmes de contrôle. Les algorithmes de filtrage et de fusion des données issues des capteurs se prêtent également à ce genre de simulations, car il est facile de perturber les réponses capteurs (que ce soit en jouant sur la fréquence d'échantillonnage ou sur le rapport signal sur bruit), dans le simulateur. De même, il est possible d'expérimenter de nouvelles lois de commande en toute sécurité pour l'engin et d'en affiner les réglages avant les essais en mer. Cette classe de simulateur permet donc, dans une certaine mesure, d'évaluer les performances *a priori* des algorithmes.

Dans [22], les auteurs présentent leur simulateur appelé *DeepC Simulator*, destiné à leur AUV *DeepC*. Il s'agit d'un simulateur temps-réel, modélisant la dynamique de leur engin et un environnement. Il permet de planifier et de surveiller la mission durant une simulation. Différents modèles sont utilisés : un modèle CAD (Computer Aided Design) pour décrire l'engin, un modèle cinématique du véhicule, un modèle du fond de la mer, de

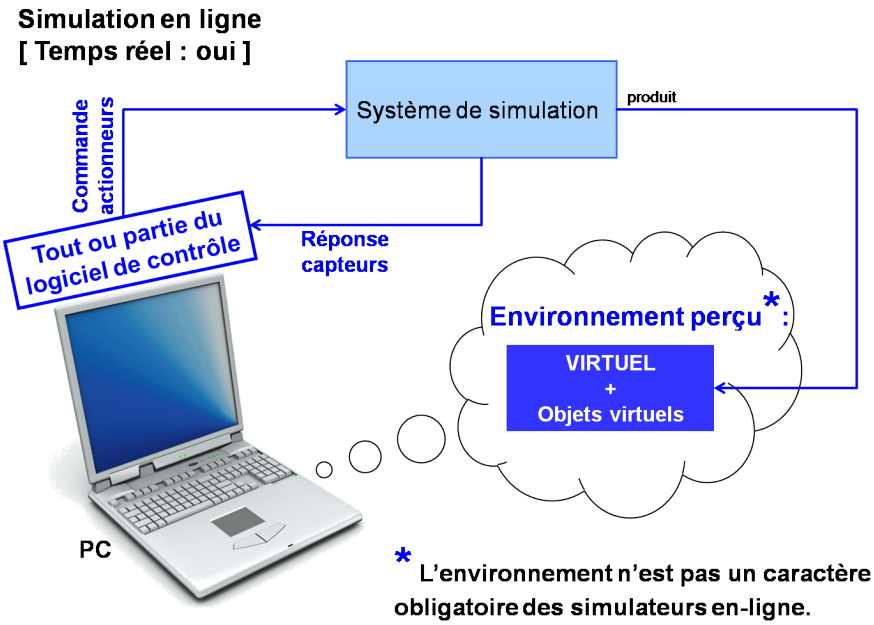


Fig. 1.6 – Tout ou partie du logiciel de contrôle du robot est déployé sur un ordinateur différent de celui du robot. Il interagit alors avec le système de simulation qui peut éventuellement fournir un environnement virtuel. Le système de simulation a donc en charge de faire évoluer le modèle dynamique du robot en fonction des commandes ; il fournit la réponse des capteurs au logiciel de contrôle.

la faune et de la flore... Un processus appelé "Virtual reality" se charge de la visualisation 3D de la scène et de la simulation de capteurs virtuels. L'ensemble de l'architecture de ce simulateur est présenté sur la figure 1.7.

Dans [23], les auteurs exposent un simulateur en-ligne utilisant pratiquement le même contrôle, les mêmes scenarii, et le même programme de contrôle que pour l'AUV réel. Les scenarii et les procédures d'urgence telles que la remontée en surface et l'évitement d'obstacles, tout comme les modes de navigation standard (navigation rectiligne à vitesse constante, suivi de trajectoire, plongée, vol stationnaire...) peuvent être vérifiés grâce à des exécutions répétées de la simulation. Ce simulateur a été utilisé avec le robot *Urashima*.

Dans [24], les auteurs ont développé un simulateur pour tester les algorithmes de contrôle de leur contrôleur appelée *Autopilot*. Le simulateur comprend un modèle d'AUV et calcule le déplacement en fonction des commandes des actionneurs, des courants, de fluctuations aléatoires et des instruments de navigation. L'utilisateur peut configurer un véhicule virtuel pour qu'il corresponde aux caractéristiques physiques d'un véhicule réel. Pour cela, il peut changer la position des gouvernes, des propulseurs, indiquer les dimensions, définir la répartition des masses, la flottabilité, les coefficients de traînée...

Dans [25], l'auteur présente un simulateur en-ligne appelé *SubSim* et destiné au robot *MAKO*. Comme présenté sur la figure 1.8, le processus de simulation se trouve au centre du système. Il exécute un programme contrôleur et calcule la physique de la simulation à partir des différentes données issues des fichiers de configuration (modèle physique et graphique de l'environnement, robots et autres objets). L'auteur a apporté un soin particulier à la structuration et la définition de la hiérarchie des classes.

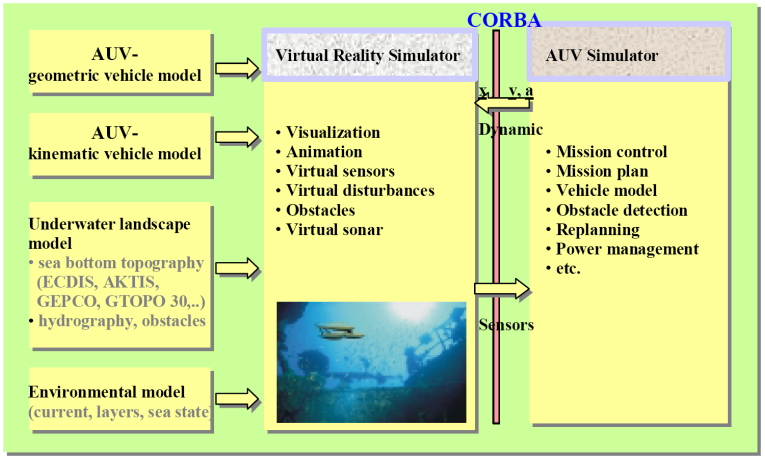


Fig. 1.7 – Architecture du simulateur DeepC

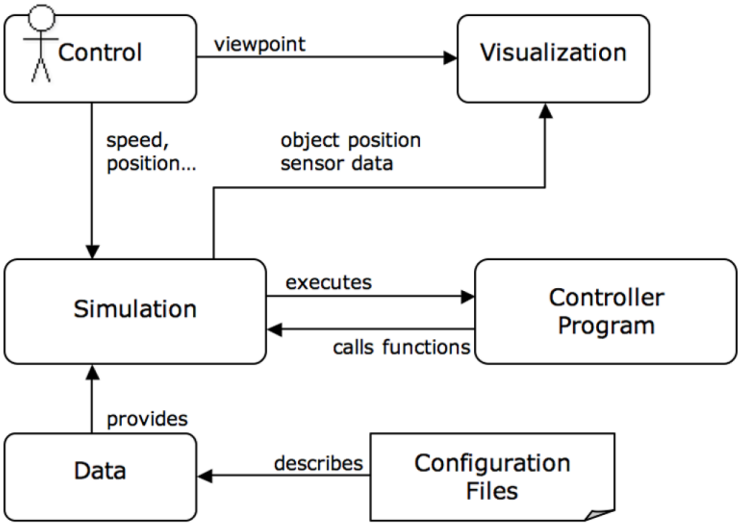


Fig. 1.8 – Concept du système de simulation SubSim

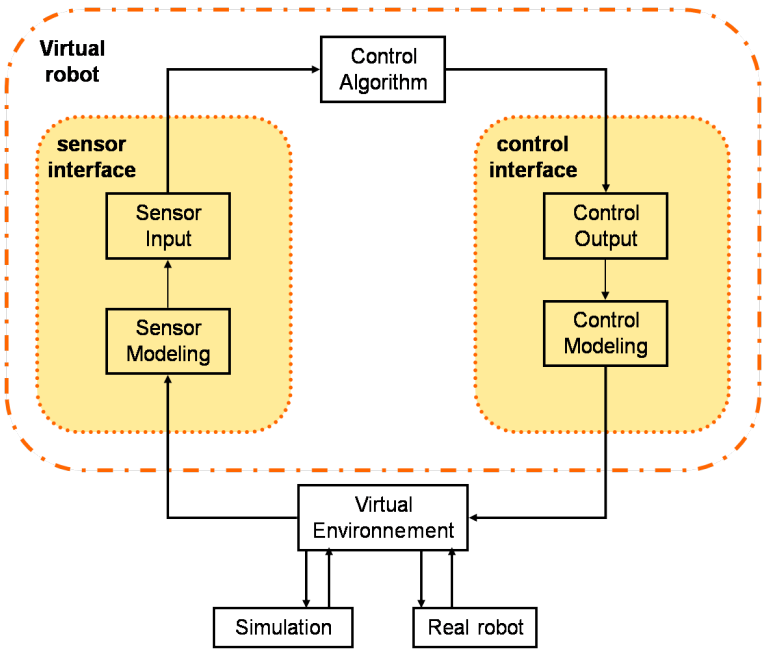


Fig. 1.9 – Simulateur utilisant le VRML (Virtual Reality Modeling Language) et le Java

Dans [26] et [27], les auteurs ont développé un environnement virtuel en VRML (Virtual Reality Markup Language) qui peut être partagé sur le web. Ce langage est utilisé pour décrire toutes les entités d'une simulation interactive : les participants, les capteurs et le son. Par ailleurs, il est possible d'ajouter des scripts pour créer un scenario. Un effort particulier a été fait sur la modélisation visuelle de l'environnement. Des robots réels ou virtuels peuvent être utilisés dans cet environnement (figure 1.9). Pour y connecter un robot réel, un client est exécuté sur l'ordinateur de bord.

Dans [28], les auteurs ont développé un simulateur appelé *SAMON*, destiné à tester des algorithmes d'échantillonnage de l'océan par des AUVs. Pour cela, le modèle comportemental des véhicules est utilisé. Son implémentation est laissée à la charge de l'institution souhaitant mettre en œuvre son véhicule au sein de l'environnement virtuel.

Les simulateurs online ne permettent pas de tester l'architecture de contrôle sur le robot réel : un ordinateur différent de celui du robot réel, est utilisé pour exécuter tout ou partie de ce logiciel. Il n'est dès lors pas possible de garantir que le comportement temporel de cet ordinateur sera identique à celui utilisé sur le robot réel :

- En effet, le matériel utilisé pour la simulation diffère de celui employé sur le robot. Par conséquent, rien ne garantit qu'un algorithme aura le même comportement temporel sur les deux plate-formes : un processeur possédant quatre cœurs ne possède à l'évidence pas les mêmes performances qu'un microcontrôleur. Cela peut avoir des conséquences importantes sur le comportement du robot. Par exemple, si on développe un algorithme pour faire de l'évitement d'obstacles à partir d'un flux vidéo, l'algorithme peut être exécuté en quelques milisecondes sur du matériel performant, mais n'aura pas un comportement forcément satisfaisant sur l'ordinateur de bord.
- Ensuite dans ce type de simulation, le simulateur peut être exécuté sur le même matériel que la partie contrôle. Par conséquent, là non plus, il n'est pas possible

de prédire le comportement temporel global des algorithmes de contrôle, et au-delà de l'architecture de contrôle. En effet, calculer l'évolution d'un véhicule, et à plus forte raison de plusieurs, consomme des ressources qui ne seront pas disponibles pour le contrôleur. Comment distinguer alors un échec de simulation du à un temps d'exécution trop important d'un algorithme utilisé par le contrôleur, d'un échec lié à la multitude de véhicules impliqués dans la simulation.

– Pour finir, dans ce type de simulateur, il n'est pas toujours possible de déployer la même architecture de contrôle que sur le véritable véhicule (matériel différent). Il en résulte souvent que seules certaines parties sont déployées à la place ou que des solutions dites "équivalentes" soient retenues. Encore une fois, cela ne permet pas de garantir un comportement identique sur le robot réel. Par exemple, il est possible de déployer la partie contrôle du logiciel du véhicule, en faisant abstraction de la partie acquisition et traitement des données capteurs, qui sont, elles, fournies par le simulateur. Les différences à l'exécution du logiciel du véhicule seront donc notables, et peuvent induire des divergences de comportement.

Les simulateurs en-ligne ne permettent donc pas de tester le fonctionnement complet d'un contrôleur déployé sur un véhicule autonome. De même, il n'est guère possible d'envisager à ce stade de simuler le comportement d'une flottille d'engins autonomes. Une autre classe de simulateur permet d'utiliser l'architecture de contrôle réelle, sur le matériel réel d'un véhicule : il s'agit des simulateurs dits *matériel-dans-la-boucle*.

1.5 Les simulateurs matériel-dans-la-boucle

Lorsque l'objectif est de vérifier la fiabilité d'un système en prêtant une attention particulière au couplage des tâches (guidage, navigation, procédure d'urgence...), le comportement du système simulé doit être le plus proche possible du système réel. Pour cela, il est nécessaire d'utiliser le véritable matériel et le véritable logiciel de contrôle dans la simulation (figure 1.10). Une simulation matériel-dans-la-boucle se réfère à une simulation qui inclut ou qui est capable d'inclure n'importe quel matériel (généralement le contrôleur du véhicule mais il peut également s'agir d'un moyen de communication, d'un capteur, ...) du système simulé. Une simulation de cette classe sous-entend qu'une partie des calculs mathématiques faisant évoluer les modèles soient effectués sous contrainte de temps. En effet, l'ensemble des calculs doit être effectué par le simulateur avec un temps bien inférieur (en général on prendra une dizaine de fois) au temps de cycle du contrôleur. Nous reviendrons sur ce point capital dans les sections suivantes. Encore appelés simulateurs *Hardware In the Loop (HIL)*, ce type de simulateur a la particularité d'exécuter le contrôleur du robot sur le véritable robot et non sur un matériel tiers. Cela a pour conséquence directe que, sous réserve de découplage temporel entre le contrôleur et le simulateur (c'est-à-dire que le contrôleur ne doit pas "attendre" une donnée en provenance du simulateur (attente bloquante)), le comportement temporel de l'architecture de contrôle pendant une simulation sera identique à celui d'une véritable mission. Les consignes aux actionneurs issues du contrôleur, sont quant à elles routées vers le simulateur. Le simulateur fournit alors les informations des capteurs au robot, en lieu et place des véritables capteurs. Le simulateur est donc généralement exécuté sur un ordinateur différent de celui du robot (bien que l'on trouve quelques exemples ou le simulateur est exécuté avec le contrôleur sur l'ordinateur de bord comme nous le verrons dans les paragraphes suivants), et a pour

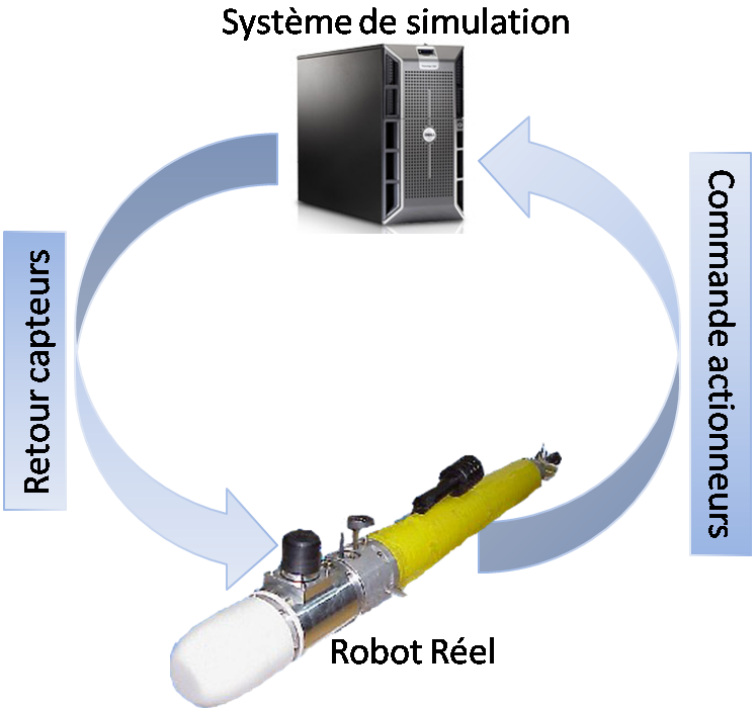


Fig. 1.10 – La simulation HIL ou hybride implique l'utilisation du matériel du robot dans la boucle de simulation

rôle de calculer l'évolution dynamique du véhicule à partir des commandes envoyées par le contrôleur du robot et de générer les réponses des capteurs à partir de l'évolution de l'engin, éventuellement plongé dans un environnement virtuel(figure 1.11).

Ce type de simulation n'est pas nouveau, et a été largement utilisé par l'industrie spatiale, dès lors que le logiciel de contrôle de vol est devenu un élément critique des engins. En plus des possibilités des simulateurs en-ligne, les simulateurs matériel-dans-la-boucle permettent de vérifier le comportement temporel de l'architecture. En effet, l'architecture de contrôle n'étant plus perturbée par l'exécution du simulateur d'une part, et s'exécutant sur le véritable ordinateur de bord du véhicule d'autre part, on a l'assurance (toujours sous réserve de découplage temporel) que son comportement sera identique durant une simulation et une véritable mission. On peut donc contrôler, plus facilement qu'en mission, le temps que prennent les différentes tâches et affiner les slots de temps processeur qui leurs sont accordés. En revanche, il n'est pas possible de tester les capteurs physiques car cela reviendrait à intervenir physiquement sur le capteur (par exemple mettre un capteur de pression sous pression) pour obtenir une stimulation. Cette classe de simulateurs permet donc d'évaluer les performances *a priori* du système.

Les premiers travaux en simulation HIL datent d'une vingtaine d'années. Le DARPA Naval Technology Office a sans doute été le premier à développer un système temps-réel pour les véhicules sous-marins autonomes (capter, décider, agir). Le but de ce projet étant de faire collaborer plusieurs engins, un simulateur simulant capteurs et environnement a été développé [29]. Le NPS (Naval Postgraduate School), a également développé un monde virtuel de l'environnement sous-marin. Il était alors possible de connecter des AUVs à

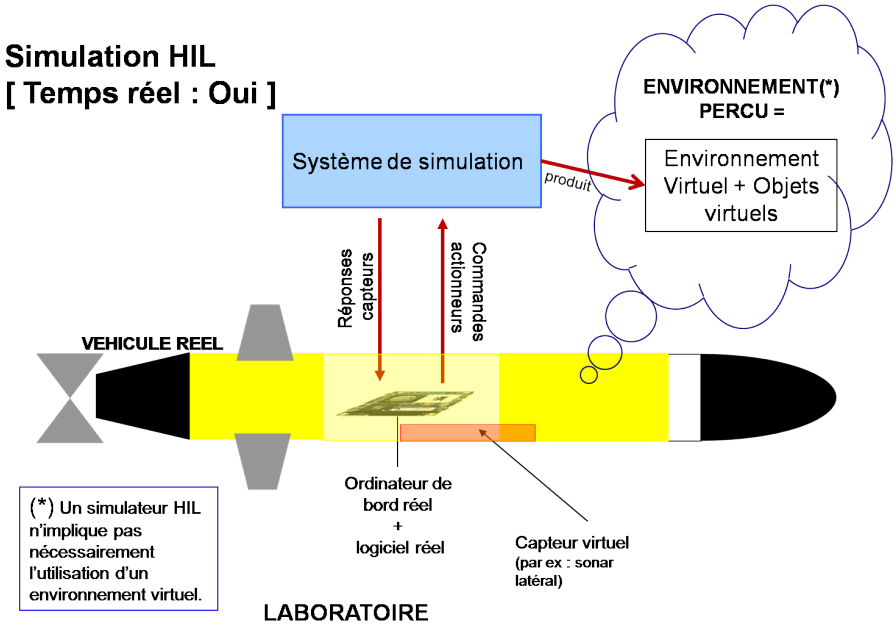


Fig. 1.11 – Dans un simulation HIL, l'ordinateur de bord et le logiciel réels sont utilisés et connectés au système de simulation qui peut éventuellement simuler un environnement virtuel. Il prend alors en charge un ou des capteurs virtuels capables de percevoir les objets virtuels faisant partie du monde simulé. Le véhicule n'est pas dans son milieu opérationnel.

cet environnement et de réaliser des simulations HIL [30] dans le but de développer et d'évaluer des contrôles non linéaires de type sliding mode. Au Naval Research Laboratory un prototype de simulateur a été créé. Ce simulateur avait la particularité de pouvoir simuler des véhicules de classes différentes (sous-marins, véhicules de surface notamment). Plusieurs modèles étaient mis en œuvre pour réaliser ces simulations (véhicules, capteurs et environnement) et une interface graphique simple avait également été développée [31]. Dans [32], l'auteur positionne le problème de la mise en œuvre d'un AUV, et propose un environnement virtuel décrit en VRML (Virtual Reality Markup Language).

Plus récemment, dans [33] et [34], les auteurs ont développé un simulateur appelé *DEVRE* dont l'objectif est d'assister les ingénieurs durant le développement du logiciel du contrôleur et de permettre de tester leur robot *GARBI* dans leur laboratoire avant les expérimentations réelles. La simulation nécessite l'utilisation de trois ordinateurs : l'ordinateur de bord du véhicule, un autre pour l'interface homme-machine, et enfin un dernier utilisé pour la simulation à proprement parler et sur lequel est calculée l'évolution de la dynamique de l'engin, l'évolution d'un modèle d'environnement, ainsi que les images virtuelles d'un sonar virtuel. L'interface homme-machine, et le simulateur ont été développés sous LabWindows et sont exécutés sur des plate-formes win32. Les capteurs virtuels correspondants aux capteurs physiques accèdent à un serveur de données de simulation (hébergé par l'ordinateur de bord) pour calculer leur réponse, à laquelle est ajouté un signal d'erreur (voir figure 1.12). Ce simulateur fonctionne spécifiquement avec le logiciel de l'AUV, qui a été conçu de façon à contrôler de la même manière l'AUV réel et l'AUV simulé. Il est à noter que le code correspondant à la simulation de l'ensemble des capteurs virtuels, est exécuté sur l'ordinateur de bord de GARBI. Par conséquent, afin de garantir

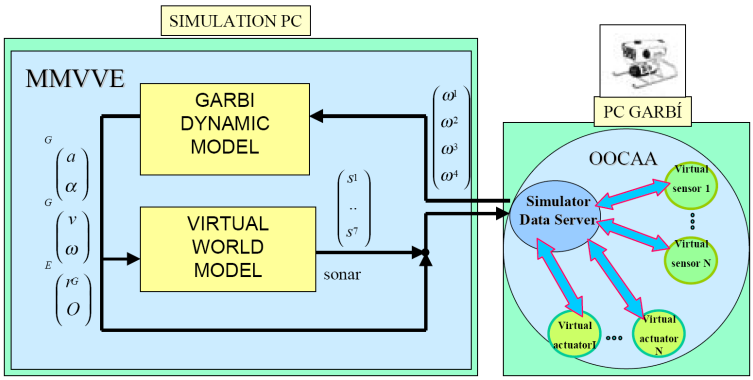


Fig. 1.12 – Architecture du simulateur DEVRE

le même comportement temporel entre les modes réel et virtuel, il serait nécessaire de veiller à ne pas empiéter sur les ressources du logiciel de contrôle en exécutant la simulation des capteurs virtuels uniquement sur le temps processeur laissé libre par le contrôleur. Dans le cas contraire, il n'est pas possible de garantir le même comportement temporel, voire logique, entre modes réel et virtuel.

Dans [35], les auteurs exposent le projet CADCON (Cooperative AUV Development Concept). Ce simulateur permet de réaliser des simulations matériel-dans-la-boucle grâce à un serveur d'environnement fournissant un environnement virtuel et supportant les communications haut-niveaux, et des "AUVSim Client" en charge de faire évoluer le modèle dynamique des robots et de calculer la réponse des capteurs après s'être abonné à l'environnement pour recevoir les données. Cet environnement virtuel permet de manipuler les interactions entre plusieurs véhicules, contrôlées par les clients du simulateur (voir figure 1.13) ; il s'agit d'un volume d'eau limité par ou contenant un ou plusieurs des éléments suivants :

- la topographie du fond sous-marin
- la topographie d'une couche de glace de surface
- un modèle de thermocline simple
- un modèle de salinité simple
- un modèle de courant simple
- un nombre variables d'agents (par agent, les auteurs désignent ici des entités autonomes différentes des robots participants)
- un nombre variable d'objets inanimés
- une source de dipôle magnétique

Toutes ces fonctionnalités, sont décrites dans un ensemble de fichiers de configuration chargés au démarrage de la simulation. Les participants sont modélisés avec des formes géométriques simples et se déplacent en utilisant un modèle newtonien. Ces formes peuvent entrer en collision avec les différents éléments constituant la simulation. Le but affiché de cet outil est de ne pas se focaliser sur la précision de l'évolution des modèles dynamiques mais plutôt sur la capacité offerte aux participants de communiquer entre eux à haut niveau d'abstraction. L'utilisation du protocole TCP/IP pose problème pour ce type de simulation : l'utilisation d'un mode connecté pour échanger des informations, sous-entend que si l'une des entités communicantes bloque, l'autre interlocuteur suspendra son exécution jusqu'à réception de la donnée. Cela est lourd de conséquences, et certains phéno-

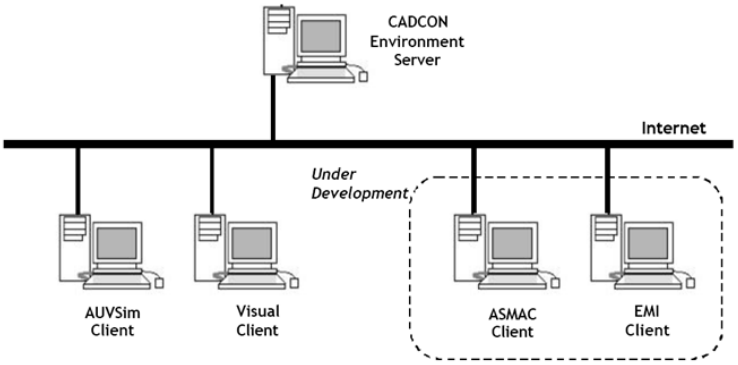


Fig. 1.13 – Architecture du système de simulation CADCON

mènes ne pourront pas être observés avec ce type de connexion. L'utilisation d'Internet ne semble pas non plus compatible avec la qualité des résultats escomptés. En effet, temps de latence, ordre d'arrivée des paquets, perte de paquets et durée d'acheminent variables sont des caractéristiques intrinsèques du réseau Internet. Ces particularités rendent impossible l'interconnexion de deux systèmes temps-réel (le simulateur d'environnement et le client) par Internet sans perdre la notion de temps-réel. Par ailleurs, les auteurs justifient le recours à des modèles dynamiques dégradés, reproduisant davantage le comportement que la dynamique des engins, en indiquant que le simulateur CADCON se focalise sur la capacité offerte aux véhicules de pouvoir communiquer entre eux. Pourtant, même si l'aspect communication est effectivement primordial (comme nous l'abordons dans la section suivante), il ne faut pas pour autant négliger la dynamique des engins. On imagine aisément les conséquences si la simulation n'est pas suffisamment (et tout l'enjeu est de déterminer le "suffisamment" au regard de l'objectif visé) proche de la réalité. Il n'est pas non plus possible de simuler un sonar ou un système de vision dans CADCON se privant de ce fait d'une partie des scenarii possibles. Enfin le client "AUVSim" fonctionne sous win32, qui n'est pas un système temps-réel. Par conséquent, il faudra prendre soin de vérifier la cohérence des résultats au regard de l'activité du système d'exploitation.

Dans [36], les auteurs décrivent leur simulateur *sauvSim*, destiné à leur AUV, appelé *SAUV* (Solar-Powered AUV). Ce simulateur est exécuté sur l'ordinateur de bord de l'AUV, avec les autres composants du système de contrôle ; il présente deux modes d'utilisation :

- le premier correspond à un mode "single" : le simulateur se charge de l'ensemble des calculs de la simulation (évolution du modèle dynamique, réponses de capteurs environnement...). Ce mode permet de simuler l'ensemble des sous-systèmes de leur AUV.
- un mode connecté : ce simulateur peut également se connecter à l'infrastructure de simulation (en fait il s'agit d'un simulateur d'environnement) CADCON [35]. Dans ce mode, les liaisons séries avec les capteurs et autres moyens de communication sont redirigées vers le simulateur qui se charge de dialoguer avec l'environnement CADCON (voir figure 1.14) via un client appelé AUVSim Client (voir la figure 1.15). Ce mode permet d'obtenir une modélisation plus fine de l'environnement, et de réaliser des missions de coopération avec d'autres acteurs.

On constate ici qu'une partie importante de la simulation est exécutée par l'ordinateur

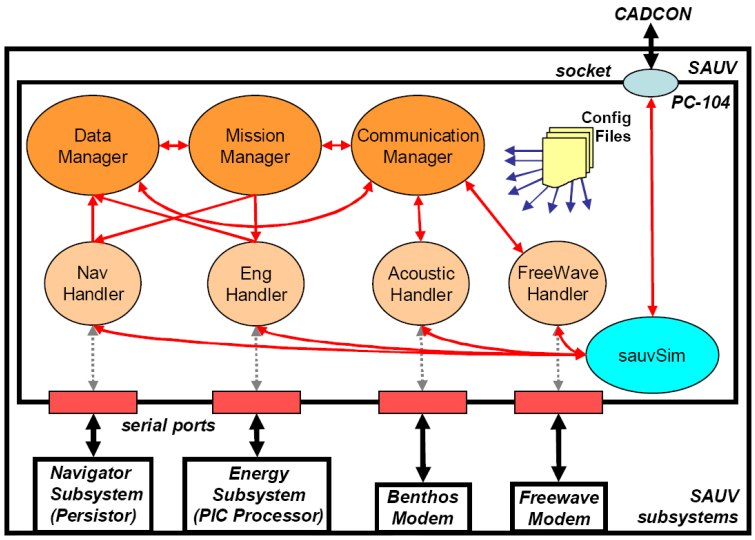


Fig. 1.14 – Vue globale du système simulé : sauvSim représente le simulateur, SAUV est le nom donné à leur véhicule

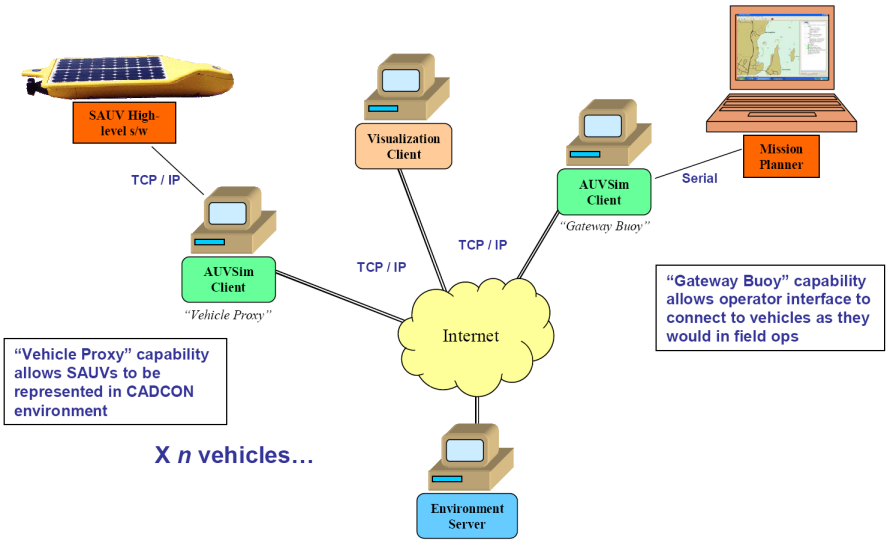


Fig. 1.15 – Liaison de sauvSim avec CADCON)

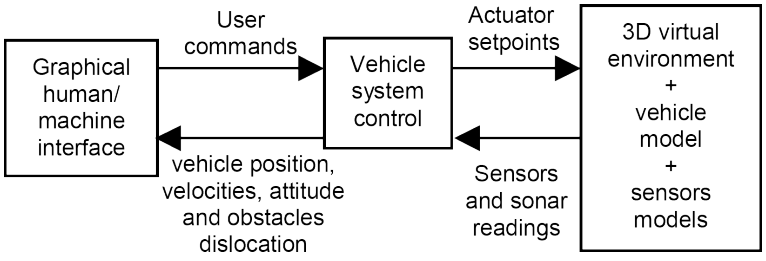


Fig. 1.16 – Distribution des éléments lors d'une simulation du robot SARA

de bord lui-même ; la reproduction du comportement temporel du contrôleur entre mode réel et mode simulé peut ici aussi être mise en doute. De plus lorsque le simulateur est utilisé en mode connecté, le protocole TCP/IP est ici aussi utilisé. Par ailleurs passer par Internet pour faire des simulations temps-réel ne semble guère possible comme nous l'avons précédemment évoqué.

Dans [37] et [38], les auteurs ont développé un simulateur permettant de simuler plusieurs véhicules simultanément. Le simulateur est en charge de calculer l'évolution du modèle de véhicule en fonction des commandes et renvoie les informations capteurs au contrôleur. La modélisation des capteurs consistent ici à produire l'ensemble des sorties numériques brutes des capteurs à partir du vecteur d'état du véhicule. La simulation des données raw (brutes), permet de tester les algorithmes destinés à "parser" (analyse syntaxique) les données capteurs pendant les missions réelles. Le modèle des capteurs intègre leurs limitations "naturelles" (échantillonnage, biais, variance...). Par ailleurs, ce système de simulation est également composé d'un simulateur d'environnement. Le simulateur d'environnement est relié à tous les simulateurs de véhicules par un lien ethernet, via le protocole TCP/IP. L'utilisation du protocole ne semble pas approprié comme nous l'avons précédemment évoqué. Enfin la simulation d'une mission, nécessite la synchronisation préalable des AUVs et de l'environnement. Cette synchronisation n'existant pas dans la réalité (sans l'ajout d'une horloge ultra-stable du moins), il ne semble pas raisonnable de se servir de ce genre d'artifice pour effectuer une simulation multi-véhicule. Enfin le simulateur d'environnement envoie un ensemble d'informations (vagues, courants, bathymétrie...) indifféremment à tous les simulateurs de véhicules (broadcast). Cela ne correspond pas non plus à la réalité : un AUV à une position donnée ne peut connaître l'état de l'environnement plus loin que ne lui permettent ses capteurs.

Dans [39], les auteurs présentent un simulateur pour le développement du logiciel de contrôle pour des AUVs. Le simulateur, en charge de calculer l'environnement virtuel en 3D, l'évolution du modèle de l'engin et la réponse de ses capteurs, est connecté à deux autres ordinateurs : une interface homme machine et le contrôleur du véhicule. Ces trois ordinateurs sont reliés par un réseau ethernet et utilisent le protocole TCP/IP pour communiquer. Ce simulateur est utilisé pour le robot *SARA*. L'architecture du système incluant le simulateur est présentée figure 1.16. Le simulateur et l'interface HMI fonctionnent sous win32, alors que le logiciel de contrôle du véhicule est exécuté sur un Motorola PowerPC sous LynxOS (système temps-réel) ; il s'agit donc d'une architecture globale couplant univers temps-réel et non temps-réel.

Dans [40] et [41], les auteurs décrivent un simulateur appelé DVECS (Distributed Virtual Environment Collaborative Simulator) permettant de tester des ROVs et des AUVs réels ou simulés, dans un monde virtuel synthétique contenant des objets réels ou virtuels,

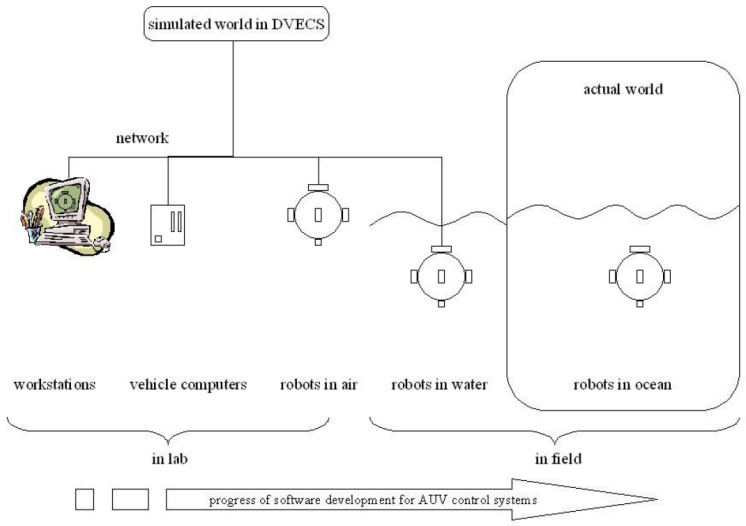


Fig. 1.17 – Utilisation de DVECS avec l'AUV Odin

des obstacles et des perturbations. Ce simulateur permet de déployer plusieurs véhicules simultanément et d'utiliser des capteurs détectant des images texturées générées par l'environnement. Les articles ne présentent aucun détail sur l'architecture, le fonctionnement du simulateur ou des résultats de simulation mais font la liste des fonctionnalités du système. Bien qu'attrayant, il n'est pas possible de juger scientifiquement ces travaux à partir de ces informations. La figure 1.17 reproduit la façon dont est connecté le système à leur AUV Odin.

Dans [42], les auteurs ont développé un simulateur pour leur AUV *Redermor*. Le but affiché du simulateur n'est pas de devoir nécessairement placer le matériel dans la boucle de simulation. Le but premier est de fournir un outil commun de développement et de test pour l'équipe qui comprend des hydrodynamiciens, des mécaniciens, et des automaticiens. Pour cela, un soin particulier a été apporté à la flexibilité de leur outil pour en faciliter l'utilisation et l'évolution. Cette flexibilité se retrouve à trois niveaux : au niveau matériel, le standard VME (Versa Module Eurocard) a été retenu pour l'enfichage des cartes électroniques ; au niveau logiciel, tous les modules sont définis comme dans Matlab/Simulink (les modèles des véhicules tournent sous SIMULINK RTW) ; enfin, au niveau de l'interface homme-machine des fichiers de description sont générés à partir de la base de données de l'environnement et du véhicule. La figure 1.18 représente l'utilisation du simulateur en mode matériel-dans-la-boucle.

Dans [43], les auteurs ont développé un simulateur appelé *CSE* (Core Simulation Engine) utilisé avec leur ROV *ANGUS*. Ce simulateur permet de réaliser des tests avec du matériel réel et a la particularité de pouvoir utiliser des composants logiques ou temps-réel au sein de son architecture. Une interface permettant l'entraînement des opérateurs, la vérification de faisabilité et le rejeu des missions a été développée. Ce simulateur est divisé en de nombreux modules répartis sur un réseau. La structure logique du simulateur est présentée figure 1.19. Le simulateur CSE se veut être un projet collaboratif impliquant industriels et organismes de recherche. CSE permet également de simuler des systèmes complexes, tel qu'un bras robotisé par exemple.

Les simulateurs HIL sont très utiles pour la préparation des missions puisqu'ils per-

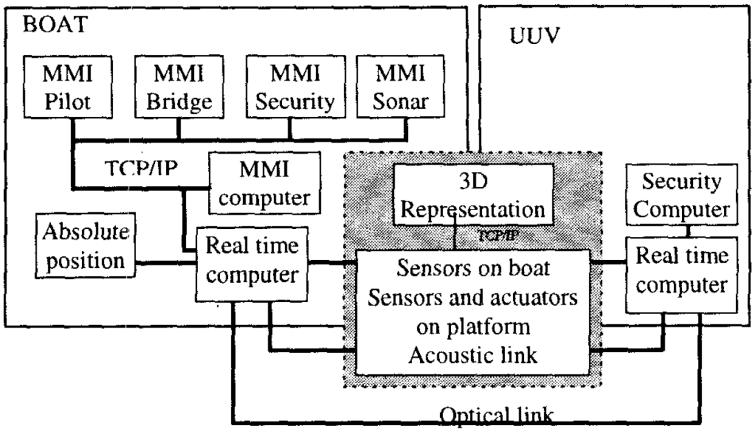


Fig. 1.18 – Architecture d'une simulation HIL pour Redermor. Le carré gris désigne le simulateur. Le lien optique n'est utilisé que durant les phases de test.

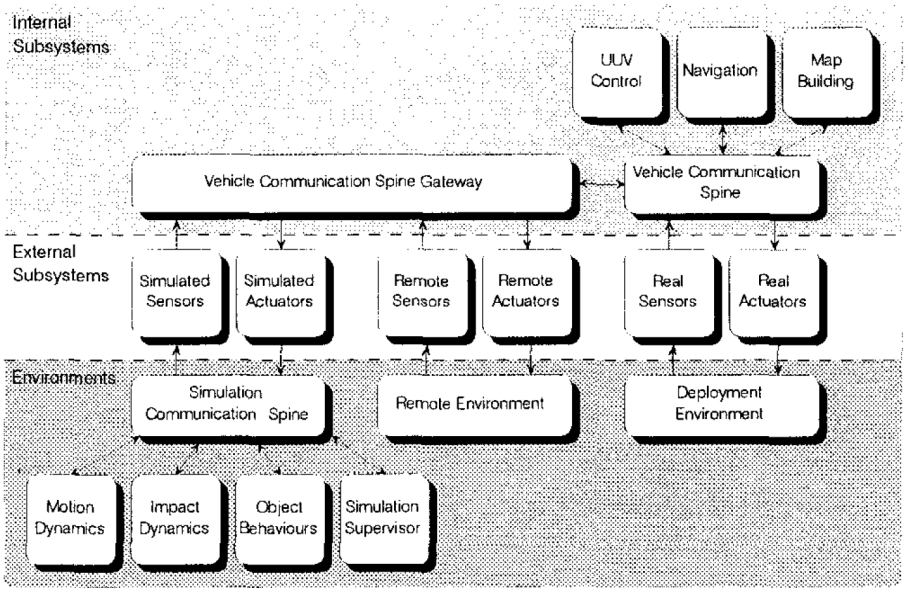


Fig. 1.19 – Structure logique du simulateur CSE

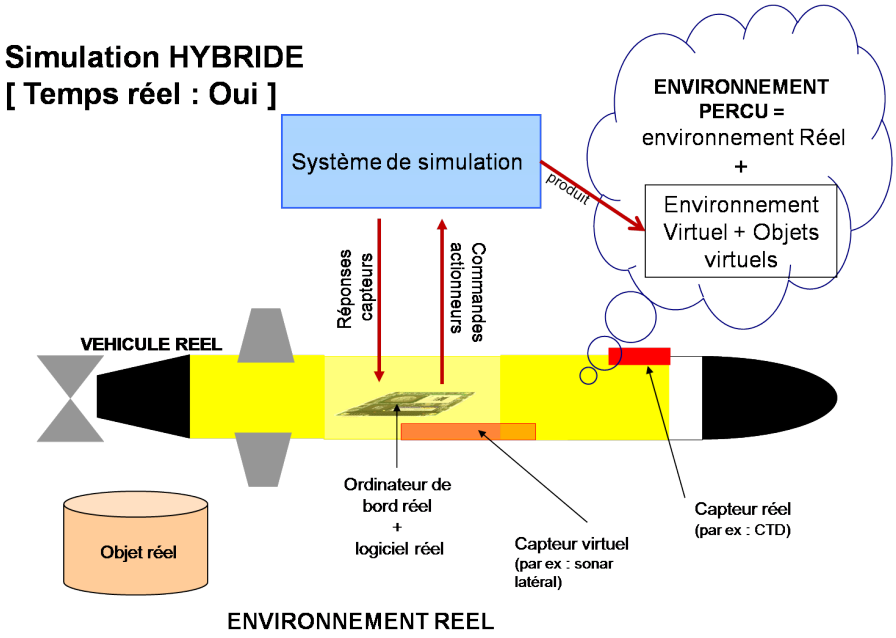


Fig. 1.20 – Dans un simulateur hybride, le véhicule réel se trouve dans son milieu opérationnel et effectue normalement sa mission. Le simulateur supporte un monde virtuel (normalement calqué sur le monde réel dans lequel le robot évolue), des objets virtuels et bien-sûr des capteurs virtuels. On qualifie alors le simulateur d'hybride car le système réel réagit durant sa mission réelle à des stimuli réels et virtuels (évitement d'obstacles virtuels par exemple).

mettent de valider presque entièrement et précisément aussi bien le déroulement de la mission, que le comportement global de l'engin. Les simulateurs HIL ne possèdent pas d'environnement dans lequel objets réels et virtuels peuvent cohabiter. Ce cadre, communément appelé *réalité augmentée*, désigne un environnement produit par une machine à l'intérieur duquel des entités réelles ou virtuelles peuvent se mouvoir et interagir.

Lorsqu'un tel environnement est simulé, on parlera plutôt de simulateur *hybride*, rendant par là possible, l'utilisation de capteurs extéroceptifs virtuels dans l'environnement opérationnel réel.

1.6 Les simulateurs hybrides

Un simulateur hybride est en fait un simulateur HIL où les systèmes réels et virtuels agissent ensemble dans une réalité augmentée (voir figure 1.20). Il est alors possible de réaliser une mission dans laquelle le robot utilise par exemple un capteur CTD réel (Conductivity, Temperature, Depth) pour constituer une carte de salinité tout en utilisant un sonar virtuel pour faire de l'évitement d'obstacles virtuels. Par ailleurs, cette classe de simulateur permet de tester davantage de matériel : il est possible de vérifier que les actionneurs fonctionnent correctement (action conforme au contrôle et à la réalité (problème de convention de signe)) et de s'affranchir du modèle dynamique de l'engin, souvent mal connu.

Ce type de simulation permet de préparer entièrement les missions. Grâce à l'ad-

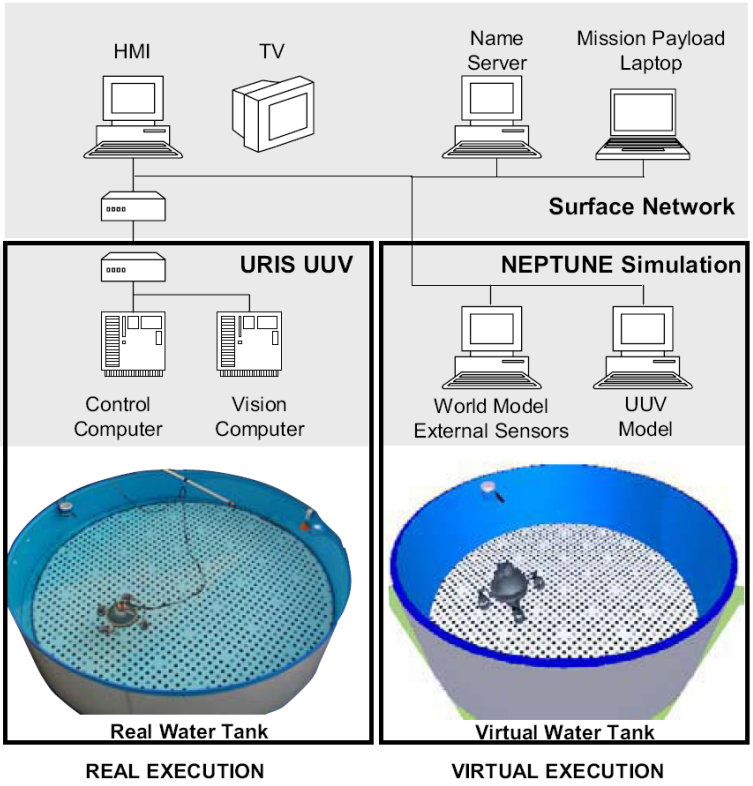


Fig. 1.21 – Vue globale d'URIS connecté au simulateur NEPTUNE

jonction d'un simulateur d'environnement dans lequel évoluent toutes les entités (qu'elles soient réelles ou virtuelles), il est désormais possible de préparer n'importe quel type de mission, incluant des scenarii faisant intervenir plusieurs véhicules dans le cadre d'une navigation en flottille. Typiquement, ce genre de simulateur permet d'utiliser les capteurs extéroceptifs réels et virtuels (notamment les sonars, caméras, et capteur CTD), et parfois les moyens de communications usuels (radio, modem acoustique, wifi etc...).

Dans [33], les auteurs ont développé un environnement distribué autour de trois ordinateurs permettant la simulation hybride de leur engin URIS. Le premier ordinateur est l'ordinateur de bord de leur AUV, un second est utilisé pour l'interface homme-machine, et le troisième enfin, est le simulateur à proprement parler. Ce dernier calcule donc l'évolution de la dynamique de l'engin et est chargé de la représentation du monde virtuel. L'interface homme-machine, peut-être connectée aux capteurs et actionneurs réels de l'engin (on parlera alors ici de *monitoring*), ou bien à des capteurs/actionneurs virtuels (voir figure 1.21).

Dans [44], les auteurs ont développé le concept du "*monde synthétique*" (voir figure 1.22). Un monde synthétique est un monde réel auquel on a superposé un monde virtuel créé d'après des acquisitions capteurs réelles. Les auteurs exposent leur méthodologie pour créer ce monde synthétique dans lequel plusieurs robots peuvent se mouvoir. Dans cet article, le robot *Twin-Burger* est mis en oeuvre au travers d'une simulation dans une piscine réelle, dans laquelle ont été disposés quatre obstacles virtuels. Le robot a correctement navigué dans le monde réel, en évitant les murs réels du bassin et les obstacles

Fig. 1.22 – Le monde synthétique auquel font allusion les auteurs constitue ce qu'on appelle une réalité augmentée

virtuels. Le système est baptisé *MVS*, pour Mulit-Vehicle Simulator.

1.7 Conclusion

Dans ce chapitre nous avons abordé en profondeur la problématique des simulateurs. Nous avons commencé par présenter des arguments pour expliquer en quoi un simulateur est utile pour le domaine de la robotique sous-marine en particulier, et constitue une étape obligatoire dans le contexte du multi-véhicule. Nous avons ensuite défini ce qu'était un simulateur en distinguant le modèle, le système, le cadre expérimental et la simulation. Il est à noter que suivant le niveau de finesse retenu, le flux entre les modèles peut varier énormément. L'architecture de simulation doit donc minimiser le temps de communication entre les modèles, sous peine de ne pas pouvoir simuler des capteurs complexes finement (sonar à ouverture synthétique typiquement). Nous avons évoqué dans ce chapitre les différentes technologies de simulateur et mentionné les travaux qui s'y rapportent. Nous proposons dans le chapitre suivant une synthèse de cette classification, puis la définition de critères supplémentaires nous permettant de mieux cerner les solutions existantes et donc de prendre position.

Chapitre 2

Synthèse et approfondissement

2.1 Synthèse

Comme nous l'avons évoqué précédemment, il existe de nombreux simulateurs que nous avons présentés selon la classification de P. Ridao [13]. Dans un exercice de synthèse, nous proposons une représentation schématique de cette classification (figure 2.1). Cette représentation se fonde sur l'identification des composants fondamentaux (indifféremment appelés blocs dans ce chapitre), dont les interconnexions (au travers de la configuration d'interrupteurs), permettent d'exprimer les différents types de simulateur. Plusieurs couleurs ont été utilisées pour représenter la nature des composants. Ainsi on retrouvera :

- la couleur verte pour les blocs *virtuels* : ce sont des composants logiciels faisant évoluer un modèle mathématique correspondant à un véritable bloc
- la couleur jaune pour les blocs *réels* : ce sont les composants matériels qui existent réellement
- la couleur grise a été utilisée, quant à elle, pour représenter les blocs fonctionnant aussi bien avec les blocs réels, qu'avec les blocs virtuels.

Nous allons maintenant présenter les fonctions occupées par les différents blocs :

- Les blocs gris représentent le contrôleur du robot qu'il soit réel, ou *émulé* (on entend par là qu'il adopte au minimum le même comportement logique pour un ensemble de fonctionnalités, non nécessairement identique, à celles du contrôleur réel). Il s'agit donc du logiciel et du matériel réel ou pas, utilisé pour contrôler le robot. On retrouve quatre blocs correspondant chacun à un type de contrôleur. Ce type est défini selon trois critères :
  - le matériel (l'ordinateur de bord) utilisé : est-il identique à celui déployé sur le robot ? ce critère est noté HW (pour HardWare).
  - le logiciel du contrôleur : est-ce celui du robot réel ? ce critère est noté SW (pour SoftWare).
  - l'aspect temps-réel du contrôleur, noté TR.

  La combinaison de ces critères mène à huit architectures différentes, dont quatre seulement sont cohérentes et représentent un intérêt réel. Ces quatre arrangements sont donc représentés par les quatre blocs gris. Les quatre autres arrangements ne

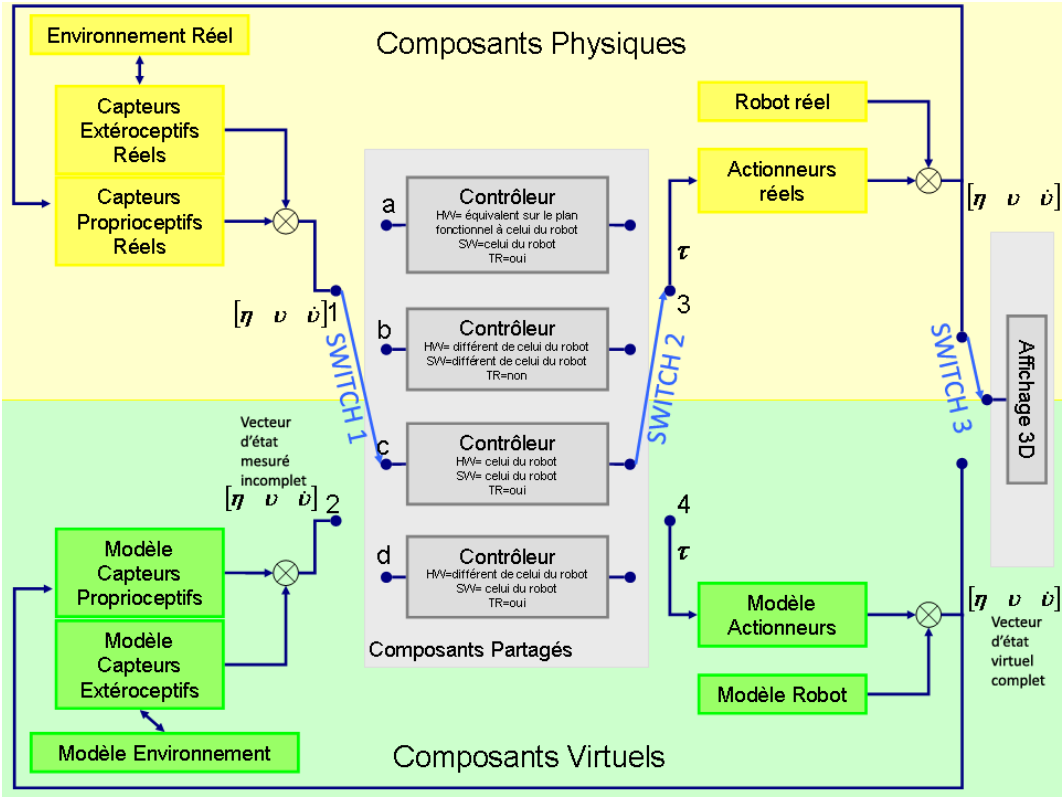


Fig. 2.1 – Architectures des Simulateurs. Un robot est considéré comme un ensemble capteurs - contrôleur - actionneurs - mécanique

présentent pas d'intérêt : par exemple la combinaison matériel du robot + logiciel du robot + pas temps réel n'est pas possible). L'entrée de ces blocs est constituée du vecteur d'état estimé (c'est-à-dire le vecteur d'état tel qu'il a été mesuré par les capteurs) et éventuellement de données provenant des capteurs extéroceptifs (vidéo, images sonar etc...). La sortie de ces blocs est le vecteur de commande transmis aux actionneurs, réels ou pas.

- le bloc environnement représente le milieu dans lequel évolue le robot. Lorsqu'il s'agit d'un milieu virtuel, ce bloc comporte les modèles mathématiques de phénomènes physiques réels : vague, vent, courants, ... C'est également dans ce bloc qu'est modélisé le fond marin ainsi que le différents objets composants le monde virtuel (structures, robots, bateaux...). L'environnement peut être stationnaire (c'est-à-dire que les différents composants présents dans l'environnement n'évoluent pas avec le temps) ou dynamique.
- Les blocs capteurs qui sont de natures différentes :
  - extéroceptifs : ce sont les capteurs qui permettent de percevoir le monde extérieur : température, images sonar, ...
  - proprioceptifs : ce sont les capteurs qui permettent de percevoir les variables internes du robot...

  Il est à noter qu'un capteur virtuel ne peut sonder qu'un environnement virtuel et qu'un capteur réel ne peut percevoir qu'un environnement réel. Il existe néanmoins une exception à cette règle. En effet, il est possible de développer une plate-forme de stimulation du robot, pour certains de ses capteurs. Cette plate-forme est alors reliée à un simulateur d'environnement et elle peut alors reproduire en réel, les stimuli que percevrait le robot plongé dans cet environnement virtuel. Ce genre de plate-forme est plutôt utilisé dans le domaine du spatial et sort du cadre de notre étude.
- Les blocs actionneurs : il s'agit des composants (mécaniques, électriques,...) (ou de leur modèle mathématique si on se place dans le cadre d'une simulation) constituant la chaîne de l'effecteur permettant au robot de se mouvoir. Dans la réalité, la sortie du contrôleur n'est évidemment pas directement connectée aux actionneurs ; elle passe généralement par un étage de puissance délivrant le bon signal à l'actionneur. Cet étage n'est pas représenté sur la figure.
- Le robot : il s'agit du robot lui-même si l'on se place dans un cadre réel, ou bien de la représentation mathématique de sa dynamique si l'on se place dans le cadre de la simulation. La sortie de ce bloc représente le vecteur d'état complet virtuel de l'engin (à ne pas confondre avec le vecteur d'état mesuré, qui peut être par ailleurs incomplet, de la sortie du bloc modèle capteurs). Dans le cas d'une simulation prenant en charge l'environnement, il existe des interactions entre le modèle du robot et les différents phénomènes physiques modélisés dans l'environnement. Il peut s'agir de vagues, de courants, de vents..., ou bien de manipulations par le robot, d'objets appartenant à l'environnement. Dans ce cas, il faut relier le bloc "modèle environnement" à celui de "modèle robot" pour être en mesure de calculer l'évolution de la dynamique du robot en tenant compte de ses interactions avec l'environnement. Dans le cas réel, ces interactions existent toujours.
- L'affichage 3D enfin permet de représenter la scène dans laquelle se trouve le robot. Cet aspect souvent négligé et peu considéré par la communauté des roboticiens, revêt pourtant une certaine importance. Il est complémentaire de l'affichage sous forme

de courbes, plus classique, et permet de mieux se représenter les évolutions du robot dans son espace. Dans le cadre du multi-véhicule notamment, ce genre d'affichage se révèle précieux pour être en mesure d'analyser la situation. Enfin, la simulation des capteurs de vision (type caméra et appareil photo) nécessite obligatoirement ce genre de rendu. Il est alors possible de tester un ensemble d'algorithmes de type SLAM (Simultaneous Localisation And Mapping) [45] ou suivi de pipeline par exemple.

Un type de simulateur est défini par la position des interrupteurs : il s'agit des trois "interrupteurs" sur la figure 2.1 appelés *SWITCH1*, *SWITCH2* et *SWITCH3*. Chacun des interrupteurs sert à relier respectivement, les capteurs au contrôleur, le contrôleur aux actionneurs, et le robot à l'affichage 3D. Suivant les interconnexions des différents blocs, nous retrouvons les différentes classes de simulateurs évoquées précédemment :

- les simulateurs hors-ligne : configuration des switchs = 2-b-4
- les simulateurs en-ligne : configuration des switchs = 2-a-4
- les simulateurs matériel-dans-la-boucle : configuration des switchs = 2-c-4 (environnement non obligatoire)
- les simulateurs hybrides : configuration des switchs = (1-c & 2-c)-4, avec capteurs extéroceptifs et environnement
- le mode réel : configuration des switchs = 1-c-3 sans affichage

Par ailleurs, ce schéma représente également des modes que nous n'avons pas évoqués, mais qui se rapprochent du contexte de la simulation :

- la surveillance en ligne [1-c-3] avec affichage : ce mode permet de suivre et de représenter l'évolution du robot réel, dans un environnement réel, mais n'est pas adapté à tous les types de robot. En effet, si ce genre de surveillance peut s'envisager pour les robots aériens, il est en revanche peu concevable de pouvoir suivre un AUV de la même façon, en raison de la faible bande passante des communications sous-marines.
- entraînement de l'opérateur [2-d-4] : ce mode permet à un opérateur de disposer d'un robot virtuel, lui permettant de s'entraîner à sa conduite dans le cadre d'un engin télé-opéré, ou de s'entraîner à la prise de décision et au-delà au commandement dans le cadre d'une mission multi-véhicule.

Nous constatons à travers cette figure qu'il existe un ensemble de simulateurs différents et il ne faudrait surtout pas en déduire, que certains sont meilleurs que d'autres. En revanche, cette diversité permet aux différents acteurs (les mécaniciens, les automaticiens, les acousticiens, ...) d'utiliser un simulateur adapté à leurs besoins : un automaticien, par exemple, aura sans doute besoin d'un modèle dynamique du robot relativement précis dans un simulateur hors-ligne. En revanche, le concepteur d'une architecture de contrôle aura davantage besoin d'un simulateur matériel-dans-la-boucle. Chacun de ces simulateurs permet donc de *valider* une approche dans un certain *domaine* et dans un *cadre* défini. Par ailleurs, l'utilisation d'un simulateur en particulier est en relation directe avec la phase de développement de l'engin ; durant les premières phases, des simulateurs hors-ligne seront privilégiés, puisqu'à ce stade le contrôleur du véhicule n'est normalement pas crée. Par la suite, les simulateurs en-ligne pourront être utilisés sur tout ou partie de l'architecture de commande. A ce stade, les lois de commande et la dynamique de l'engin commencent à être correctement définies. Les dernières étapes de conception nécessitent souvent l'emploi d'un simulateur matériel-dans-la-boucle, grâce auquel il est possible de valider le comportement temporel du contrôleur et des lois de commande. Enfin la dernière étape qui précède l'essai réel du robot, peut nécessiter l'utilisation d'un simulateur hybride,

qui permettra de tester dans un espace opérationnel (piscine par exemple), et en sécurité, l'ensemble des comportements du robot face à diverses situations pouvant être liées à l'environnement.

La question qui se pose naturellement maintenant, concerne la place des missions multi-véhicules dans le cadre de la simulation. En effet, à ce stade nous n'avons pas encore abordé la relation qui lie les expérimentations faisant intervenir plusieurs robots mobiles autonomes, potentiellement hétérogènes, au monde de la simulation. Dans le paragraphe suivant, nous introduisons les raisons qui nous ont naturellement poussées à retenir une technologie en particulier : les simulateurs hybrides ou les simulateurs Hardware-In-the-Loop.

2.2 Intérêt de la simulation hybride et HIL

Après avoir montré l'intérêt du recours à la simulation pour la commande de flottille de véhicules potentiellement hétérogènes dans l'introduction, nous avons distingué quatre classes distinctes de simulateurs et nous avons répertorié les principaux travaux dans le domaine de la robotique sous-marine pour chacune de ces catégories.

Plusieurs raisons nous ont guidé vers le choix de l'hybride :

- De nombreuses équipes ont conçu des algorithmes innovants permettant d'envisager la coordination de véhicules sous-marins. Pour autant il existe très peu d'expérimentations dans ce domaine. En effet, des étapes de tests sous forme de simulations s'avèrent nécessaires pour avancer vers l'expérimentation réelle. Si la plupart de ces algorithmes ont été validés dans des simulateurs hors-ligne, peu d'entre eux en revanche ont été réellement implémentés. Ceci est sans doute dû, entre autre, à l'absence d'outils de simulation permettant de prendre en compte plusieurs véhicules. Quoi qu'il en soit, il est aujourd'hui nécessaire de franchir une nouvelle étape en validant le comportement et les performances de ces algorithmes dans un milieu sécurisé, mais se rapprochant le plus possible des conditions d'exploitations réelles ; nous avons suffisamment avancé sur la conception de ces algorithmes, et nous abordons maintenant l'étape d'implémentation durant laquelle ils devront être validés.
- La simulation hybride est à mi-chemin entre l'expérimentation et la simulation : on utilise le robot réel, avec sa dynamique réelle, ses actionneurs et certains de ses capteurs.
- Il est possible de tester le robot réel en toute sécurité dans une piscine (et donc de s'affranchir de la simulation de son modèle dynamique qui est souvent mal connu), tout en le plongeant dans une réalité augmentée (c'est-à-dire la réalité qu'il perçoit avec ses propres capteurs, à laquelle on ajoute des objets virtuels qu'il peut percevoir avec ses capteurs virtuels) pour tester des algorithmes évolués (suivi de pipeline, détection de mine) sans avoir à chercher un emplacement réel réunissant les conditions d'expérimentations (champ de mines).
- L'équipe de robotique sous-marine du LIRMM, développe deux engins sous-marins autonomes. Ces deux véhicules font l'objet d'un remaniement complet afin notamment de leur ajouter de nouveaux capteurs. Le nombre restreint de personnes de l'équipe, ajouté à la charge de travail conséquente des chercheurs, a rendu ces robots indisponibles pour un certain temps. L'avantage que peut procurer le recours à la simulation, est ici certain : il est possible d'utiliser l'ordinateur de bord du véhicule,

ainsi que ses différents éléments (actionneurs, capteurs, et moyens de communication), pour implémenter les algorithmes de contrôle de flottille dans l'architecture logicielle du contrôleur. De nombreux tests de validation pourront donc être menés pendant la réfection des engins.
- Enfin nous collaborons avec l'équipe de Massimo Caccia, chercheur au CNR (Consiglio Nazionale delle Ricerche), en Italie. Leur équipe développe un catamaran autonome de surface baptisé Charlie et des expérimentations en vue de faire collaborer leur robot et les nôtres sont prévues prochainement. Ce genre de collaboration, dont la logistique est importante (on imagine aisément que déplacer un robot de plusieurs centaines de Kg depuis l'Italie n'est pas des plus aisé), nécessite une plate-forme de test commune afin de maximiser les chances de succès le jour de l'expérimentation.

Tous les points que nous avons évoqués nécessitent d'implémenter les algorithmes de coordination au sein de l'architecture logicielle de contrôle des robots réels. Par conséquent cela sous-entend que le simulateur devra nécessairement avoir un comportement temps-réel pour dialoguer avec le contrôleur du robot. Ces caractéristiques correspondent clairement à un simulateur matériel-dans-la-boucle ou hybride si l'on veut pouvoir tester des algorithmes de type SLAM [46], faisant partie de la problématique de la coordination de véhicules.

Après avoir exprimé notre intérêt pour les simulateurs hybrides, nous abordons dans le prochain paragraphe les caractéristiques supplémentaires requises pour traiter la problématique du multi-véhicule. Nous y présentons les simulateurs hybrides existants et leurs limitations, puis nous évoquons les motivations qui nous ont amenées à construire un nouveau simulateur.

2.3 Approfondissement des critères de classification

Maintenant que nous avons clairement mentionné les classes de simulateurs existantes en exposant les caractéristiques propres à chacune d'elles, et choisi la technologie qui nous semble la plus appropriée dans le cadre de notre étude, nous proposons dans cette section des attributs supplémentaires, qui ne caractérisent pas nécessairement les simulateurs hybrides (ou HIL), mais qui nous semblent essentiels pour envisager la simulation multi-véhicule.

2.3.1 Multi-véhicules

Si concevoir, tester et évaluer le déroulement d'une mission pour un seul véhicule peut déjà se révéler compliqué, la difficulté augmente encore lorsque l'on considère deux véhicules ou plus. Cela est dû en partie au nombre élevé d'interactions possibles entre les participants d'une mission de coopération. La stratégie de contrôle de chaque véhicule doit être en effet assez robuste pour faire face à des communications dégradées, une localisation des autres participants peu précise, des défaillances incontournables... Tester la logique de la mission dans laquelle se trouvent plusieurs engins se révèle être une chose difficile, car cela nécessite la présence de tous les véhicules, pour évaluer le comportement de chaque robot face aux actions et communications des autres. Un simulateur multi-véhicule est donc nécessaire, afin de tester l'ensemble du système avant d'effectuer les essais réels. Cela signifie que le simulateur est capable de prendre en charge simultanément plusieurs

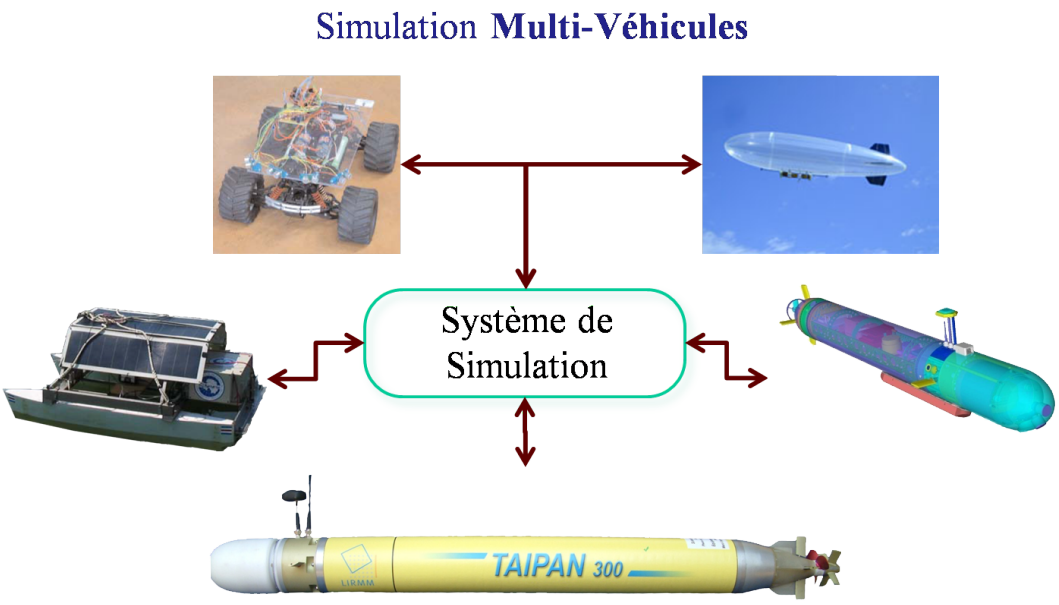


Fig. 2.2 – Simulation multi-véhicules

véhicules au sein d'une même simulation (figure 2.2). On retrouve généralement dans ce genre de simulation, la gestion des collisions inter-véhicules qui permet d'éviter aux engins de se "traverser" les uns les autres. Dans ce type de simulation, les véhicules sont synchronisés entre eux seulement si un moyen matériel permet de le faire dans la réalité. En effet, cette synchronisation n'existe pas dans la réalité et une simulation "honnête" ne doit pas s'affranchir de cette difficulté.

Dans [37] et [38], les auteurs ont développé un simulateur avec lequel il est possible de simuler un ou plusieurs AUVs. Ce système de simulation est composé de plusieurs simulateurs de véhicule, en charge de calculer l'évolution du modèle dynamique de l'AUV auquel il est raccordé et de fournir les données capteurs correspondantes (figure 2.3). Aucune démonstration d'une simulation faisant intervenir plusieurs véhicules n'est faite dans ces articles.

[41] et [40] font également mention des capacités de leur simulateur à pouvoir simuler plusieurs engins mais à l'heure où ces lignes sont écrites, nous n'avons pas trouvé d'informations ou de démonstration sur cette fonctionnalité.

Dans [13], les auteurs indiquent que leur simulateur Neptune est capable d'opérer des simulations multi-véhicules, mais peu d'information à ce sujet sont disponibles.

Dans [44], les auteurs proposent un simulateur auquel il est possible de connecter plusieurs agents (figure 2.4). Chaque agent est connecté à un (ou plusieurs processus) qui lui fournit les données du monde synthétique. Les auteurs ne donnent aucune information sur la façon dont sont reliés ces agents aux processus, ni sur l'influence de cette connexion sur le comportement du contrôleur de l'agent. Une simulation multi-véhicule utilisant ce simulateur peut être trouvée dans [47].

Dans [35], [48] et [36], les auteurs démontrent les capacités de leur simulateur à réaliser une mission multi-véhicule dans laquelle interviennent plusieurs SAUV. Ces véhicules sont connectés via un client au serveur d'environnement CADCON pour réaliser leur mission.

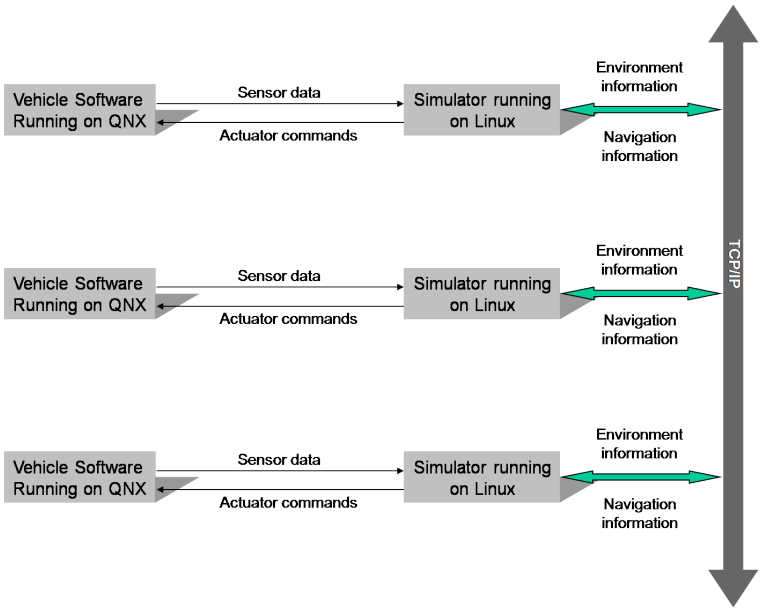


Fig. 2.3 – Implémentation du simulateur

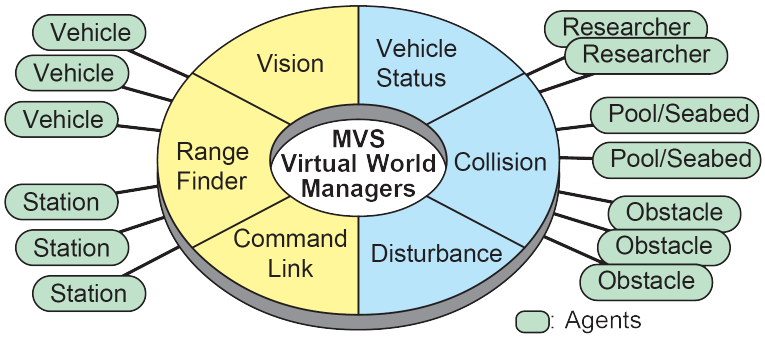


Fig. 2.4 – Liaison du simulateur MVS et de différents agents

Ces quatre simulateurs sont les seuls à prétendre aux scenarii multi-véhicules tout en étant HIL ou hybrides, ce qui semble peu au regard du nombre de simulateurs étudiés jusqu'à maintenant. Parmi ces quatre simulateurs, deux présentent des résultats concrets d'une simulation multi-robot. La possibilité offerte par un simulateur multi-véhicule de prendre en charge plusieurs robots n'est toutefois pas suffisante. En effet, on peut considérer qu'il existe deux façons de communiquer :

- une première façon, que l'on pourrait qualifier d'*implicite*, consiste à observer le comportement des autres véhicules et de prendre les décisions en conséquence. Ce genre de coopération ne permet pas de partager des informations acquises.
- La seconde manière, que l'on peut qualifier d'*explicite* consiste à échanger des messages de façon active. Ce type de communication permet donc d'échanger des données acquises par différents capteurs sur différents véhicules et permet d'envisager des algorithmes de fusion de données.

La plupart des algorithmes de coopération impliquant des communications explicites, il est nécessaire de disposer d'un simulateur de communications (simulation du moyen de communication), et d'environnement (simulation de la propagation des ondes).

2.3.2 Communications inter-véhicules

Envisager un simulateur permettant de tester des scenarii multi-véhicules, dans lequel il n'est pas possible aux engins de communiquer entre eux semble absurde. En effet, cette fonctionnalité est absolument essentielle et constitue un passage obligé pour travailler dans le cadre que l'on s'est fixé. Il existe de nombreux types de moyens de communication, s'appuyant sur des media différents (eau et air), que les robots utilisent suivant le milieu dans lequel ils évoluent. Les communications inter-véhicules se font principalement par radio UHF, WIFI et ondes acoustiques. Exceptionnellement, des moyens opto-électroniques ont pu également être mis en œuvre pour communiquer en milieu aquatique.

Ces liens de communication permettent donc aux robots d'échanger des informations telles que leur vecteur d'état, mais également de transmettre des données acquises par leurs capteurs. Ces communications sont donc nécessaires pour maintenir la cohérence du groupe et pour mener la mission à son terme. Un certain nombre de simulateurs vus dans les sections précédentes permettent de prendre en charge les communications entre les véhicules, mais aucun d'entre eux n'est utilisable pour simuler les communications aquatiques de façon réaliste dans le cadre de la flottille d'AUVs.

Dans [35] et [48], le simulateur développé permet aux véhicules de communiquer entre eux en utilisant un canal au choix (un canal parfait, un canal acoustique et un lien vers un satellite ARGOS). Très peu d'informations sont disponibles concernant la modélisation des communications, mais il semblerait que les véhicules puissent s'échanger des chaînes de caractères de façon instantanée. Tous les véhicules sont atteignables à tout moment et les communications se font en mode point à point ou en broadcast (point à tous). Il n'est fait aucune mention de la bande passante, de la qualité du lien ou bien de la portée des communications. Bien que sûrement très pratique à utiliser pour permettre à des véhicules de communiquer durant une simulation, cette implémentation des communications ne correspond absolument pas à la réalité. Cela pourrait convenir dans une certaine me-

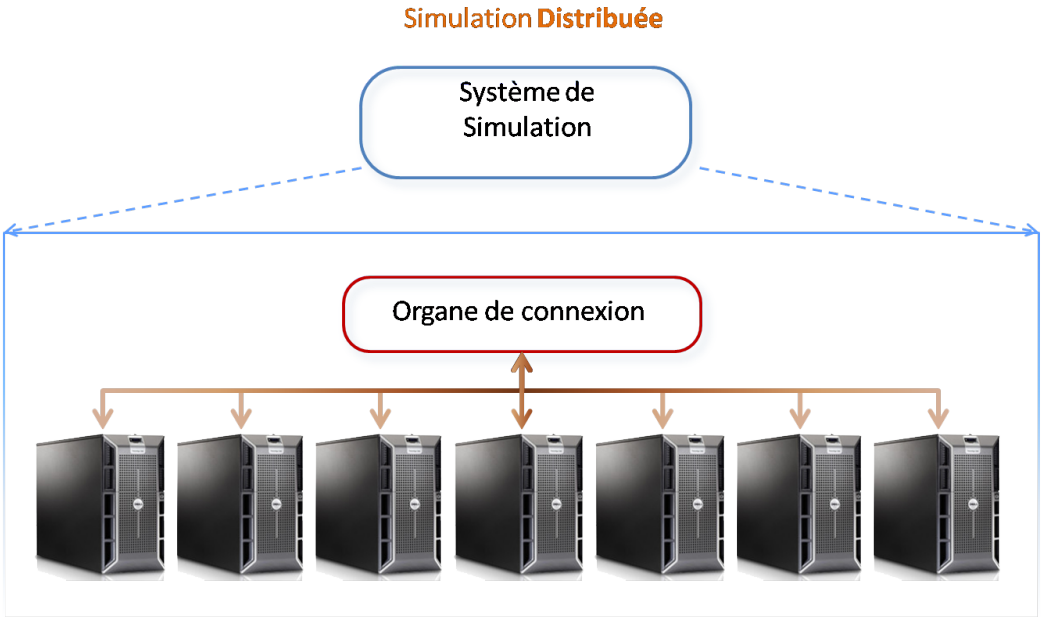


Fig. 2.5 – Simulation distribuée : le simulateur est réparti sur plusieurs ordinateurs mis en réseaux

sure pour les communications aériennes (wifi, radio, ...), mais il en va autrement pour les communications acoustiques qui sont fortement contraintes, comme nous le verrons à la section 6.2.2. Dans le simulateur DVECS [41] [40], et MVS [44] [47], les communications sont modélisées sous forme de mémoire partagée. Elles sont donc instantanées, sans durée (bande passante infinie), sans interférences, ... Cette implémentation ne convient donc pas à notre problématique.

Aucun des auteurs des simulateurs étudiés (toute nature confondue) ne s'est attaché à la problématique des communications. Il est pourtant nécessaire de simuler le comportement d'un moyen de communication (temps de réveil, demande de réémission, interférences,...) et de la propagation des ondes (temps de latence, limitation de la bande passante des canaux, multi-chemins, bruits,...) si l'on veut tester de façon réaliste des algorithmes de coordination de flottille. Permettre aux véhicules de communiquer au sein du simulateur n'est toutefois pas suffisant.

Tout au long de notre état de l'art, nous avons pu voir que la simulation permettant de considérer des scenarii multi-véhicules, implique une masse importante de données à traiter (élaboration de la réponse des capteurs) et de calculs à faire (propagation des ondes radio/acoustiques, évolution de la dynamique des engins,...). Le point suivant traite de cette difficulté.

2.3.3 Distributivité

Simuler plusieurs engins, possédant des modèles dynamiques différents, évoluant dans un milieu complexe, et interagissant grâce à leurs capteurs plus ou moins sophistiqués et leurs moyens de communication, suggère une charge de calcul élevée, n'évoluant pas linéairement avec le nombre d'engins présents dans la simulation (figure 2.5). Un seul et

unique ordinateur ne semble donc pas suffisant pour implémenter un tel simulateur, tout du moins pour la simulation temps-réel. Plusieurs ordinateurs sont, de toute évidence, nécessaires et le simulateur se doit d'être distribué sur ces moyens de calculs : on parlera alors de système de simulation distribué. Ces ordinateurs appartiennent généralement à un réseau ethernet dédié, ou sont reliés entre eux via Internet. Un simulateur distribué est donc un simulateur dont la charge de calcul de la simulation est répartie sur plusieurs machines interconnectées. Il faut préciser ici que le contrôleur du robot ne fait pas partie de ce que l'on appelle simulateur ; ainsi par exemple le simulateur DEVRE, développé dans [33] n'est pas considéré comme étant distribué au sens de notre définition.

Si le gain en terme de temps de calcul est évident, ce genre d'architecture n'est pas sans poser de problème. Les problématiques scientifiques sont nombreuses et plusieurs communautés sont actives sur le sujet. Parmi ces problématiques, on peut citer :

- le mode de connexion entre les nœuds du réseau (mode connecté avec acquittement sans perte de paquet, ou mode non connecté avec possibilité de perte)
- les aspects de synchronisation entre les machines (c'est-à-dire, le partage d'une horloge commune)
- les temps de réponse (comment garantir que la réponse d'un nœud puisse parvenir à son destinataire sans retard)
- la causalité (garantir que le calcul d'un événement à un instant donné aura les même implications sur l'évolution du système que dans la réalité)

Le chapitre 3 est consacré à notre positionnement par rapport à ces aspects importants de la simulation.

Beaucoup d'auteurs font le choix de dégrader la finesse de leur modélisation, de limiter la précision de leur capteur virtuel (sonar), ou d'augmenter la période d'intégration pour s'affranchir du problème de la surcharge de calculs [33], [35], [38], [19] entre autres. D'autres ont fait le choix d'essayer de répartir la charge de calcul sur plusieurs ordinateurs connectés en réseau, en usant de différentes stratégies et configurations.

Dans [43], le simulateur CSE est un simulateur réparti sur un ensemble de machines hétérogènes (stations de travail, PC, ...), couplées entre elles en utilisant le réseau Internet. Cet ensemble est considéré comme une seule et unique machine virtuelle distribuée. La cohérence spatio-temporelle est maintenue de la façon suivante : tous les ordinateurs calculent l'évolution de la simulation à leur rythme. Si une incohérence (collision de véhicules par exemple) intervient, les ordinateurs reviennent dans le temps, pour recalculer un nouveau futur avec les nouveaux états. Cela permet d'éviter de cadencer l'ensemble sur la fréquence du simulateur le plus lent. Pour aider à maintenir la cohérence des données et pour garder un mécanisme propre et facile pour les programmes distribués, les auteurs utilisent une approche par abonnement pour le flux de données et la synchronisation. Pour ce faire, une base de données intelligente maintient en permanence à jour l'état du monde entier simulé. La base de données utilisée, appelée dVise, est intelligente dans le sens où c'est elle qui détecte les collisions spatiales et calcule les vues virtuelles des capteurs.

Dans [28], les différents émulateurs d'AUVs sont connectés au serveur SAMON via un "wrapper". Ce terme désigne un programme permettant de transformer les ordres passés par un superviseur en commandes exécutables par l'AUV. Sur la figure 2.6, le bloc "Inherent Behavior" est en charge de reproduire le comportement de l'AUV face à différents stimuli (ordre d'un superviseur, environnement, ...). Le bloc "Ocean Environment" est

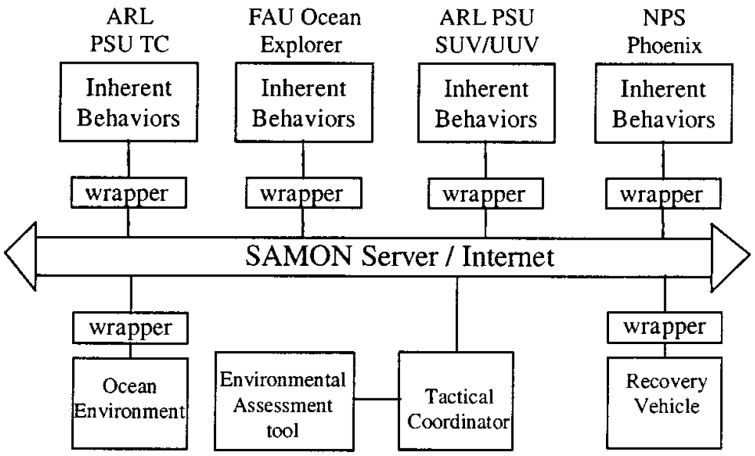


Fig. 2.6 – Simulateur utilisant le VRML (Virtual Reality Modeling Language) et le Java

chargé de fournir la bathymétrie et la valeur des paramètres océanographiques en réponse aux interrogations des capteurs.

Dans [49], les auteurs ont développé un simulateur HIL pour leur engin *Romeo*. Le système est divisé en trois réseaux locaux : un réseau dédié aux instruments de bord et au contrôleur appelé *Onboard Ethernet LAN*, un réseau dédié au simulateur appelé *Lab Ethernet LAN* et enfin un réseau dédié à la supervision appelé *Surface Ethernet LAN*. Le système de simulation est réparti sur trois serveurs qui prennent chacun en charge un aspect de la simulation : rendu graphique, environnement et capteurs, et calcul de l'évolution dynamique de l'engin (voir 2.7).

Dans [23], le système de simulation est distribué sur quatre ordinateurs reliés par un réseau ethernet (voir figure 2.8). Un premier ordinateur ("control unit") est dédié au contrôleur, qui génère les commandes à partir de la simulation des capteurs et du panneau de surveillance du bateau mère. Un second ordinateur ("Simulator of AUV") est utilisé pour simuler l'évolution de la dynamique de l'engin (pour cela un modèle reproduisant les manœuvres du véhiculé a été mis en place) et la réponse des capteurs. Un troisième ordinateur ("Monitoring panel") est utilisé pour la surveillance du statut du contrôleur et pour lui passer d'éventuelles commandes. Enfin, un quatrième ordinateur est utilisé pour la génération et la soumission des informations nécessaires à l'exécution de la simulation ("Simulator on the mother Ship").

Dans [38], chaque véhicule est connecté à un simulateur de véhicule qui sont eux-même connectés au simulateur d'environnement en TCP/IP. Dédier un simulateur par AUV constitue sans aucun doute un premier pas vers la répartition de la charge du calcul sur le réseau. Cela peut s'avérer néanmoins insuffisant si le véhicule possède un sonar et un système de vision par exemple. Le simulateur devant fonctionner en temps-réel, ses limites seront très vite atteintes avec ce type de capteurs et un choix devra être fait entre précision des modèles et vitesse de calcul. Cette architecture est donc statique et ne permet pas de répartir la charge de calcul de façon extensive.

Dans [44], les auteurs montrent comment leur système de simulation MVS répartit la charge de calcul sur plusieurs ordinateurs. MVS produit un environnement de simulation auquel plusieurs acteurs peuvent se rattacher via un ou plusieurs processus "d'abonnement". Lorsque la charge de calcul devient grande, il est possible de créer d'autres envi-

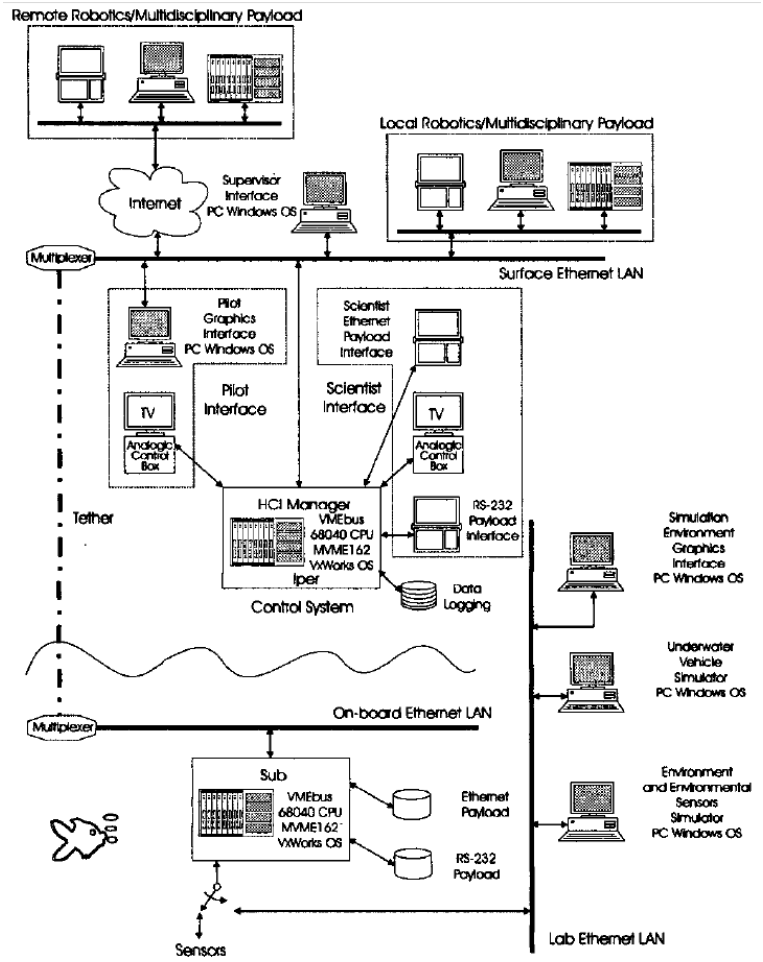


Fig. 2.7 – Vue globale du ROV romeo connecté au système de simulation

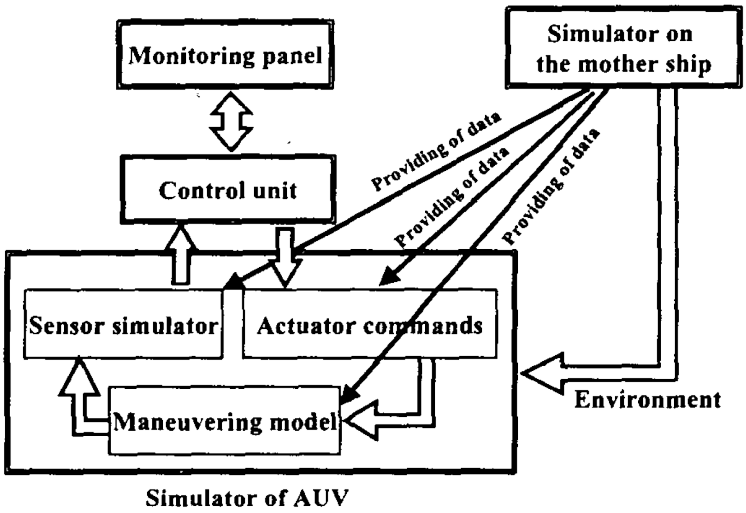


Fig. 2.8 – Fonctionnement des différents ordinateurs composant le simulateur d'Urashima ; chaque cadre en trait épais, correspond à une unité de calcul

ronnement de simulation. De nouveaux mondes sont alors créés et d'autres agents peuvent s'y rattacher. Malheureusement les nouveaux mondes ne partagent pas de données communes, et l'intérêt de ce système s'en trouve donc fortement limité.

Toutes ces solutions pour relier les moyens de calculs offrent plus ou moins d'avantages ; néanmoins, aucun des auteurs n'a soulevé la problématique du découplage temporel entre les différentes entités constituants ces systèmes. Afin d'illustrer clairement ce point, prenons l'exemple du découplage temporel de la commande d'un véhicule et du simulateur de la dynamique de l'engin. Ces systèmes de simulation sont utilisés entre autre, pour vérifier le bon fonctionnement de l'architecture de contrôle du véhicule. Admettons que celle-ci présente une défaillance quelconque, et que "pour une fois", la commande mette une seconde à être générée au lieu des 0,1 seconde habituelle. En mode TCP/IP, le simulateur de la dynamique de l'engin, va rester bloqué en attendant que la commande du véhicule lui parvienne. Une fois cette commande arrivée (en retard), il va calculer l'état du véhicule à t+1, et renvoyer ces informations au contrôleur du véhicule. Cela signifie que pendant cette seconde, l'engin est resté virtuellement immobile. Et pourtant dans la réalité, cette seconde aurait peut-être suffit à lui faire percuter un obstacle. On voit clairement, avec cet exemple, qu'il est nécessaire de s'assurer du découplage temporel de toutes les entités faisant partie du réseau de simulation, si l'on veut pouvoir mettre en évidence des comportements potentiellement divergents ; cela signifie qu'il faut abandonner le protocole TCP, au profit de l'UDP, qui est un mode de transmission non connecté. Néanmoins, cela ne résout pas entièrement le problème. Il faut également veiller à ne pas créer de blocage (du type "attendre", ou "lire tant que") au niveau de l'implémentation. Globalement, ce découplage temporel peut-être solutionné par un stockage intermédiaire.

Dans ce paragraphe, nous avons évoqué la répartition de la charge de calculs sur un réseau. Parmi ces calculs nous n'avons pas évoqué la visualisation 3D, qui constitue un dernier point important pour considérer la coordination de flottille.

2.3.4 Visualisation 3D

Ce dernier point, souvent considéré par la communauté comme accessoire, se révèle pourtant être un élément clé de la simulation multi-véhicule. En effet, il est très important de pouvoir visualiser la scène 3D dans une mission de collaboration. De nombreux contrôles de véhicule ont pour objectif de faire converger l'engin vers une zone particulière à un moment donné. Quand plusieurs véhicules sont impliqués, l'opérateur doit pouvoir vérifier que de l'implémentation du contrôle dans chaque véhicule émerge la "chorégraphie" désirée. La visualisation 3D de la scène est donc recommandée pour tester le bon déroulement de la mission avant d'effectuer les tests réels. Un affichage 3D de la scène est un programme client auquel parviennent l'ensemble des vecteurs d'état des engins simulés et qui affiche en temps-réel leur évolution dans un environnement défini. Ce programme permet de placer une caméra virtuelle à un endroit jugé utile par l'utilisateur, et lui autorise donc de suivre la scène depuis un point de vue privilégié. Cette caméra virtuelle peut également se déplacer : suivi de véhicules, trajectoires prédéfinies, ... font partie des services offerts par ce type de programme.
Il permet donc à l'utilisateur de prendre une décision adéquate grâce à une bonne appréciation de la scène (replanification, interruption, réassignation de mission...). En outre,

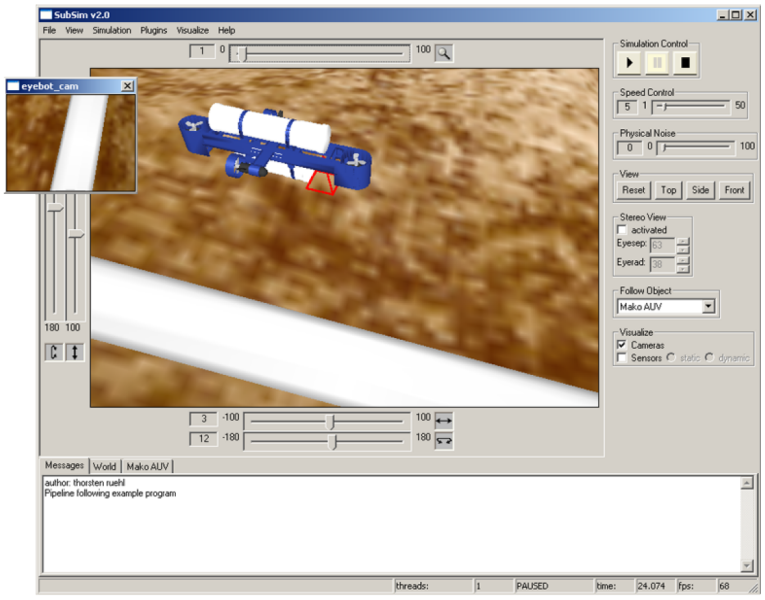


Fig. 2.9 – L'AUV MAKO placé dans un environnement virtuel utilise une caméra virtuelle pour tester un algorithme de suivi de pipeline

c'est le seul moyen de simuler une caméra virtuelle. Les débouchés scientifiques de ce genre d'application sont nombreux : on pourra par exemple mettre au point et évaluer les performances d'algorithmes destinés à calculer le vecteur vitesse d'un AUV à partir d'images vidéo texturées du fond de l'océan. Bénéficier d'un rendu 3D de la scène observée permet donc de mieux appréhender les difficultés inhérentes aux missions multi-véhicules :

- les véhicules peuvent être hétérogènes
- ils peuvent évoluer dans un milieu et dans un contexte différents
- plusieurs acteurs peuvent intervenir et ne connaissent pas forcément tous les engins mis en œuvre
- il est difficile de se représenter dans l'espace 3D, la scène qui évolue par ailleurs rapidement.

Dans [43], la simulation de l'environnement consiste en un modèle 3D de la scène fabriqué en utilisant CAD (Computer Aided Design). Des modèles de structures sous-marines et diverses informations peuvent être facilement intégrés au simulateur CSE grâce à la fonction d'import de données de dVise. Ces informations *a priori* peuvent être utilisées pour la visualisation et constituent un aspect important pour la préparation des missions. Grâce aux informations de localisation de l'engin, il est possible de calculer la scène vue par différents capteurs (sonar ou vision) du véhicule.

Dans [25], l'auteur démontre les capacités d'affichage 3D de son simulateur en utilisant une caméra virtuelle pour suivre un pipeline virtuel. Ce genre de système permet de tester efficacement des algorithmes liés à la vision (voir figure 2.9). Ce simulateur permet donc de faire un rendu 3D de la scène de simulation et a la particularité de disposer d'un mode "stéréovision".

Dans [13], les auteurs ont également développé un programme permettant de visualiser la scène de simulation. Ils illustrent les capacités d'affichage 3D de leur simulateur en reproduisant une scène réelle mettant en œuvre leur robot URIS dans une piscine. Ce rendu 3D de la scène permet de réaliser des simulations hybrides sophistiquées impliquant

des capteurs extéroceptifs évolués.

D'autres simulateurs proposent également un affichage 3D ; ce dernier point est sans aucun doute la fonctionnalité la plus implémentée dans tous les simulateurs étudiés ici. Nous avons balayé dans les sections précédentes de nombreuses technologies et l'objet du prochain paragraphe est de proposer une nouvelle classification permettant de prendre en compte les quatre derniers points que nous venons d'évoquer, à savoir les notions relatives au multi-véhicule, aux communications, à la distributivité et à la visualisation 3D.

2.4 Vers une nouvelle classification

2.4.1 Définition

Nous avons pu constater qu'il existe de nombreux critères pour classifier les simulateurs. P. Ridao a proposé de distinguer quatre catégories de simulateurs différents, mais on a constaté que cette classification n'était pas adaptée à la problématique de la simulation multi-véhicule. Par ailleurs, devant le nombre élevé de critères permettant de qualifier un simulateur, nous proposons une approche systématique permettant de clairement identifier les fonctionnalités d'un simulateur. Pour cela, nous avons choisi 20 critères séparés en 7 groupes :

- MVC : Multi-VéhiCule
  - IDT : Le simulateur permet-il de simuler plusieurs véhicules IDenTiques ?
  - MCL : Le simulateur permet-il de simuler des véhicules différents, mais de Même Classe (AUV ou USV ou UGV, ...) ?
  - CLD : Le simulateur permet-il de simuler des véhicules de CLasses Différentes ?
- COM : COMmunications inter-véhicules
  - SPP : Le simulateur permet-il à différents véhicules de communiquer Sans prendre en compte les Phénomènes de Propagation ?
  - ECP : Le simulateur permet-il d'Emuler la Couche Physique pour la propagation des ondes ?
  - MPO : Le simulateur intègre-t-il un Modèle de Propagation des Ondes ?
- SIM : SIMulateur
  - HYB : Une simulation peut-elle être exécutée en mode HYBride ?
  - HIL : Une simulation peut-elle être exécutée en Hardware-In-Loop ?
  - ONL : Une simulation peut-elle être exécutée en ONLine ?
  - OFF : Une simulation peut-elle être exécutée en OFFline ?
- WRD : WoRlD
  - OTG : Est-ce que le fond de l'Océan, un Terrain, ou une couche de Glace sont modélisés ?
  - OJS : Est-il possible d'ajouter des ObJets Statiques (mines, pipeline,...) ?
  - OJD : Est-il possible d'ajouter des ObJets Dynamiques (poisson, sous-marin,...) possédant ou non un comportement (script, loi physique, ...) ?
- ENV : ENVironnement
  - STA : Les phénomènes environnementaux (courants, vents, ...) sont-ils stationnaires ?

  - DYN : Les phénomènes environnementaux (courants, vents, ...) sont-ils dynamiques ?
- EXT : EXTéroceptif
  - AUR : Le simulateur supporte-t-il des capteurs extéroceptifs AUtRe que vision et sonar (par exemple CTD) ?
  - SNR : Le simulateur supporte-t-il des capteurs extéroceptifs de type SoNaR ?
  - VSN : Le simulateur supporte-t-il des capteurs extéroceptifs de type ViSioN ?
- DVS : DiVerS
  - DST : Le simulateur peut-il être DiSTribué sur plusieurs unités de calculs (réseau et multi-processeur) ?
  - A3D : Le simulateur propose-t-il un Affichage 3D de la scène simulée ?

Ces groupes sont ensuite répartis de la façon suivante (figure 2.10) :

MVC			COM			SIM				DVS		ENV		EXT			WRD		
CLD	MCL	IDT	MPO	ECP	SPP	HYB	HIL	ONL	OFF	DST	A3D	OJD	OJS	VSN	SNR	AUR	DYN	STA	OTG
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	1	0	0	1	0	0	1	0	1	0	0	1	0	0	1
0	1	1	0	1	0	0	1	0	0	1	0	1	1	0	1	0	0	1	0
1	1	1	1	0	0	1	1	0	0	1	1			0	1	1	0	1	1
														1	0	0	1	1	0
														1	0	1	1	1	1
														1	1	0			
														1	1	1			

Fig. 2.10 – Toutes les combinaisons n'ont pas obligatoirement un sens. Pour cette raison, on a fait apparaître dans ce tableau les possibilités combinatoires par groupe ; par exemple un simulateur permettant de simuler des véhicules de classes différentes, sera obligatoirement aussi un simulateur capable de simuler des véhicules identiques.

Grâce à cette échelle, il est possible de classifier précisément un simulateur en répondant aux différentes questions : on pourra par exemple indiquer qu'un simulateur est MVC7 COM2 SIM12 WRD7 ENV1 EXT0 DVS3 (les chiffres correspondant au code binaire des réponses aux questions précédentes). Ceci n'est toutefois pas encore suffisant : il est nécessaire de pouvoir quantifier l'efficacité (c'est-à-dire la capacité à répondre à une problématique), d'un simulateur dans un contexte donné. Pour cela nous proposons une approche par *score adaptatif*. Le mot score veut dire ici, que notre méthode va être en mesure de produire une note pour un simulateur donné. Le terme adaptatif quant à lui signifie que notre approche permet de donner une note, mais une note dans un contexte précis et pour une application clairement définie : on ne peut pas utiliser un même système de notation pour deux objectifs différents. Ainsi si l'on souhaite étudier les capacités d'un simulateur à tester la commande au sein des missions multi-véhicules, le système de notation doit être différent d'une étude de simulateur pour l'échantillonnage de l'océan... Nous présentons maintenant les étapes permettant de calculer ce score adaptatif (figure 2.11) :

- Nous choisissons les groupes intervenant dans notre contexte. Par exemple, si nous souhaitons simplement tester un algorithme, nous choisirons le groupe SIM. En revanche, si notre but est de jouer une mission mono-véhicule juste avant la mission

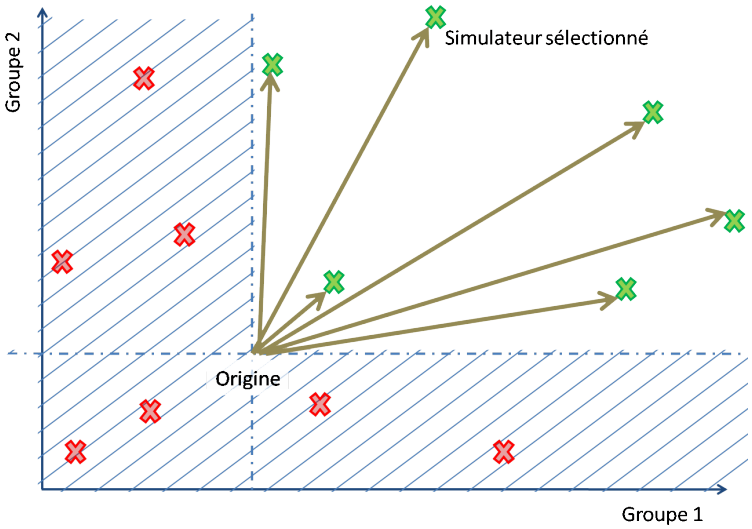


Fig. 2.11 – Calcul du score des simulateurs

réelle, nous choisirons le groupe SIM, DVS, ENV, EXT et WRD.
- Dans chaque groupe, nous classons les catégories de la plus importante, à la moins importante. Par exemple, si nous voulons tester un algorithme faisant intervenir un sonar, nous placerons SNR en premier puis VSO et AUR.
- Nous en déduisons les combinaisons possibles, puis nous en tirons une note, sur 10 par exemple, pour chaque combinaison.
- Nous définissons la classe (ou note) minimum pour chaque groupe.
- Nous éliminons les simulateurs qui n'obtiennent pas la note minimum dans chaque groupe.
- En considérant un référentiel, dont chaque axe représente un groupe sélectionné, nous plaçons les simulateurs retenus dans cet espace.
- Enfin, nous calculons la norme de chaque vecteur [origine ;emplacement simulateur], où origine est le minimum défini précédemment.

Un exemple concret de ce calcul est présenté dans la section suivante. Cette méthode permet de concevoir un référentiel personnalisé pour une application donnée, dans lequel vont pouvoir être évalués les simulateurs. Dans le paragraphe suivant, nous proposons de classer dans un tableau l'ensemble des simulateurs étudiés à partir des scores calculés en utilisant la méthode que nous venons de décrire. Cela nous permettra d'avoir une vue d'ensemble des technologies existantes cadrée sur nos objectifs.

2.4.2 Classification des simulateurs étudiés

Pour calculer le score, nous nous plaçons dans le contexte du multi-véhicule "pur". Cela sous-entend que seul l'aspect multi-véhicule nous interesse, et que nous ne considérons pas toutes les applications rattachées à ce contexte (suivi de gradient, suivi de pipeline, évitement d'obstacles, ...). Ce choix permet de cerner le noyau des simulateurs étudiés (2.12) :

La figure 2.13 présente un tableau regroupant l'ensemble des simulateurs étudiés.

On constate donc que, dans notre contexte, il n'existe que cinq simulateurs pouvant

MVC				COM				SIM					DVS			Note
CLD	MCL	IDT	CLASSE	PHS	ECP	SPP	CLASSE	HYB	HIL	ONL	OFF	CLASSE	DST	A3D	CLASSE	
0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
0	0	1	1	0	0	1	1	0	0	1	0	2	0	1	1	6.6
0	1	1	3	0	1	0	2	0	1	0	0	4	1	0	2	13.3
1	1	1	7	1	0	0	4	1	1	0	0	12	1	1	3	20
Note mini: 6.6				Note mini: 6.6				Note mini: 0					Note mini: 0			

Fig. 2.12 – Choix des groupes et classement des critères.

Référence	Auteur principal	Nom du simulateur	MVC	COM	SIM	DVS	Robot concerné	Classe	Score
	Conte		Non	Non	OFF	Non		MVC0 COM0 SIM1 DVS0	X
	Bono	RobySim	Non	Non	OFF	Non	Roby	MVC0 COM0 SIM1 DVS0	X
	Hornfeld	DeepC Simulator	Non	Non	ONL	A3D	DeepC	MVC0 COM0 SIM2 DVS1	X
	Robinson		Non	Non	ONL	A3D		MVC0 COM0 SIM2 DVS1	X
	Bielohlawek	SubSim	Non	Non	ONL	A3D	Mako	MVC0 COM0 SIM2 DVS1	X
	Phoha	SAMON	Non	Non	ONL	DST	SUV	MVC0 COM0 SIM2 DVS2	X
	Suriano		Non	Non	HIL	A3D	Sara	MVC0 COM0 SIM4 DVS1	X
	Brutzman		Non	Non	HIL	A3D	Phoenix	MVC0 COM0 SIM4 DVS1	X
	Devie		Non	Non	HIL	A3D	Redermor	MVC0 COM0 SIM4 DVS1	X
	Lane	CSE	Non	Non	HIL	A3D & DST	Angus	MVC0 COM0 SIM4 DVS3	X
	Bruzzone	SE	Non	Non	HIL	A3D & DST	Romeo	MVC0 COM0 SIM4 DVS3	X
	Ridao	Neptune	MCL	Non	HYB	A3D & DST	Uris, Garbi	MVC3 COM0 SIM12 DV3	X
	Gracanin		MCL	Non	ONL	A3D	Phoenix, Phantom	MVC3 COM0 SIM2 DVS1	X
	Song		MCL	Non	HIL	DST	Ocean EXplorer	MVC3 COM0 SIM4 DVS2	X
	Canell		IDT	SPP	OFF	Non		MVC1 COM1 SIM1 DVS0	0
	Komerska	SauvSim	IDT	SPP	HIL	Non	Solar-powered AUV	MVC1 COM1 SIM4 DVS0	13.3
	Choi	DVECS	MCL	SPP	HIL	A3D & DST	Odin	MVC3 COM1 SIM4 DVS3	24.9
	Chappell	CADCON	MCL	SPP	HIL	A3D & DST	Solar-powered AUV	MVC3 COM1 SIM4 DVS3	24.9
	Kuroda	MVS	CLD	SPP	HYB	A3D & DST	Twin Burger	MVC7 COM1 SIM12 DVS3	31.2
	Parodi	Thetis	CLD	ECP	HYB	A3D & DST	Taipan, Charlie	MVC7 COM2 SIM12 DVS3	32

Fig. 2.13 – Principales propriétés des simulateurs étudiés. Les simulateurs ne possédant pas de score, sortent du cadre minimal que nous avons fixé.

potentiellement répondre à notre problématique. Il est à noter que ce nombre est d'autant plus petit, que la définition de notre cadre minimal est large, car aucun critère autre que le multi-véhicule n'a été sélectionné. En effet, un très grand nombre de missions différentes (océanographie physique, évitement d'obstacles sous contraintes de flottille, cartographie coordonnée, etc...) s'offrent à nous, et les différentes communautés rattachées à ces applications pourraient sans doute mieux définir les groupes afin d'affiner la classification.
Cependant ces cinq simulateurs souffrent de défauts que nous avons évoqués précédemment, qui, dans notre contexte, remettent en question la validité des résultats obtenus par leur exploitation. Nous ne pouvons pas réduire le nombre de véhicules participant à une simulation, pour le motif d'une puissance de calcul trop faible pour faire face aux contraintes temps réel. De même, la granularité et la finesse des modèles employés sont pour nous des notions clés, qui ne doivent pas être dictées par les capacités de calcul du simulateur. Tous ces simulateurs ne permettent pas de préparer une mission opérationnelle, ce qui reste quand même un des objectifs de cette thèse. Enfin, aucun ne mentionne la problématique du découplage temporel.

2.5 Conclusion

Dans ce chapitre, nous avons proposé une synthèse des différentes classes de simulateurs. Nous avons ensuite examiné quatre autres critères qui semblent particulièrement importants dans le contexte de la simulation multi-véhicule : le concept de multi-véhicule lui même, la communication inter-véhicule, la distributivité du simulateur et la visualisation 3D. Nous avons alors constaté qu'il était nécessaire de proposer une nouvelle classification, permettant de qualifier plus finement les simulateurs dans un contexte donné, et pour une application précise. Nous avons ainsi défini un ensemble de groupes de critères précis. Il nous a alors été possible grâce à une méthode que nous avons proposée, de définir précisément les critères minimaux requis pour considérer un simulateur comme étant multi-véhicule. Nous avons conclu que, parmi les 19 simulateurs étudiés, seuls cinq pouvaient être qualifiés de la sorte, mais qu'ils souffraient de défauts de conception ou de limitations intrinsèques à leur architecture. Cette absence d'outil réellement adapté est sans doute une des raisons du faible nombre de manipulations réelles. Nous avons constaté à travers ces exemples, que beaucoup de concepts ont été développés, mais on peut affirmer, au vu de ces résultats qu'il manque réellement une architecture ouverte, permettant de rassembler les concepts et les acteurs.

L'objet de notre prochain chapitre est donc de présenter notre positionnement par rapport aux simulateurs existants. Nous présenterons point par point les concepts cruciaux qui seront retenus pour notre proposition pour une architecture de simulateur ouverte que nous exposerons par la suite.

Chapitre 3

Positionnement des travaux

Dans le chapitre précédent, nous avons démontré que le recours à la simulation était nécessaire pour aborder le contexte de la coordination multi-véhicule. Nous avons alors étudié les simulateurs existants et avons déduit grâce à notre classification que seuls cinq d'entre eux permettaient d'aborder une partie de la problématique. Dans ce chapitre, nous commençons par nous positionner par rapport aux travaux antérieurs et nous montrons qu'aucun d'entre eux ne semble réellement satisfaisant pour notre application. Nous évoquons notamment nos propres besoins actuels puis le devenir de ces besoins pour l'ensemble de la communauté. Enfin nous abordons les hypothèses et spécifications du système de simulation que nous proposons et qui sont nécessaires si l'on considère la validité de la simulation.

3.1 Un simulateur mais une architecture avant tout

Les chercheurs travaillant dans le domaine de la robotique sous-marine autonome, sans doute encore davantage que ceux travaillant dans les autres domaines de la robotique, ont massivement recours à la simulation. Du plus simple (simulateur hors-ligne), au plus perfectionné (simulateur hybride), en passant par toutes les solutions intermédiaires, on peut estimer le nombre de combinaisons de ces simulateurs à environ 36000 (estimation faite à partir des combinaisons possibles des critères exposés au paragraphe 2.4). Parmi l'ensemble de ces simulateurs seule une petite partie offre les fonctionnalités nécessaires à la préparation "real-like" d'une véritable mission (qu'elle soit multi-véhicule ou non). Pour autant un ensemble de fonctionnalités ne constitue pas un simulateur en soi. Au-delà même de l'aspect modélisation et implémentation, se trouvent les fondements de la simulation : une architecture.
Un travail est donc nécessaire avant d'aborder la problématique de la modélisation, même si modéliser et implémenter sont souvent les premiers réflexes. Par ailleurs, la simulation est souvent négligée et constitue une discipline assez peu considérée par la communauté des roboticiens : peu de chercheurs souhaitent s'investir dans ce domaine. Si cela ne porte pas à conséquences dans la plupart des situations (premières étapes de la conception des lois de commande, certains tests de convergence, ...), il en existe d'autres pour lesquelles la

conception d'une architecture est un passage obligé. Elles sont caractérisées par la quantité de calculs à effectuer, qu'ils découlent de la finesse des modèles employés ou du nombre de véhicules à simuler, mais également du temps autorisé pour que ces calculs soient accomplis : c'est la notion de temps-réel et au-delà de déterminisme et de respect de causalité. L'absence d'architecture ou une architecture mal conçue peut se révéler être un facteur limitant dans le meilleur des cas (c'est-à-dire qu'il n'est pas possible d'étendre la charge de calcul sans devoir dégrader la finesse des modèles) ou un motif d'invalidation des résultats (surcharge processeur, protocole inadéquat, ...) dans le pire des cas. Par ailleurs une architecture mal pensée ne permettra pas de faire évoluer *a posteriori* les fonctionnalités offertes. Même si ce dernier point n'est pas crucial et ne remet pas en question une simulation faite dans le cadre expérimental pour lequel le simulateur a été créé, il est évident que pouvoir faire évoluer les fonctionnalités et changer les modèles constitue un but à rechercher. Enfin, concevoir un simulateur monolithique (c'est-à-dire un simulateur créé sur mesure pour une application particulière) va à l'encontre du concept de préparation de mission de coordination. En effet, ce type d'expérimentation nécessite généralement l'intervention de plusieurs acteurs et dans ce contexte un simulateur monolithique ne peut pas être une solution satisfaisante. **Une architecture est donc nécessaire pour permettre à tous de *collaborer* à la conception de la simulation**.
Il sera donc nécessaire de créer une architecture répondant à un ensemble de critères que nous allons exposer et qui font défaut (au moins partiellement) dans les simulateurs que nous avons étudiés. Il est à noter que nous considérons cette architecture comme un *outil de recherche opérationnel* qui satisfait, au moins partiellement, nos besoins et qui sera amené à évoluer au fil du temps.

3.2 Une architecture ouverte

Nous avons évoqué dans le paragraphe précédent, la notion de collaboration des acteurs autour de l'architecture, afin de créer des simulations. Cet aspect revêt une importance capitale dans le contexte du multi-véhicule. Le terme "ouvert" est assez vague et il est nécessaire de préciser les points qui permettent de qualifier de telle architecture :

- Le code de l'architecture dans laquelle seront implémentés les modèles doit être source-ouverte, au moins pour certaines de ces parties. Les membres d'une petite équipe ne suffisent pas pour développer un projet à la frontière d'autant de domaines différents, et créer une architecture qui ne peut pas évoluer, la rendra à coup sûr incompatible avec certains modèles. Chacun doit donc être en mesure de proposer des ajouts, des extensions, ou des modifications permettant à l'architecture d'évoluer pour en améliorer le fonctionnement ou ajouter de nouvelles fonctionnalités.
- Qui dit source-ouverte, dit documentation des sources. En effet, proposer le code source ne suffit pas, il faut produire une documentation, afin de permettre aux contributeurs de participer et d'intégrer correctement leur travail. Cette documentation ne doit pas se résumer à commenter du code source. Il existe des systèmes de production de documentation automatiques (tel que Doxygen par exemple), favorisant la diffusion des informations au sein de la communauté. Cette diffusion, peut éventuellement être accompagnée d'un système d'échange et de thésaurisation des savoirs tel qu'un forum par exemple.
- Une approche modulaire permet également d'envisager un travail collectif. En effet,

cette approche permet à une personne ou un groupe de personnes de travailler spécifiquement sur une partie de l'architecture, sans nécessairement en connaître tout le fonctionnement. Une description précise des fonctionnalités du module facilite, là aussi, le travail collaboratif.
Pour cela l'architecture doit être correctement segmentée. Cela signifie que les différents domaines (dynamique des véhicules, simulation des capteurs, ...) doivent transparaître au travers de l'architecture. L'architecture doit offrir, à un spécialiste en acoustique par exemple, la possibilité de travailler uniquement sur la partie du code qui l'intéresse, sans nécessairement avoir besoin d'une vue détaillée de l'ensemble du système.

- Ouvrir la majorité du code à plusieurs contributeurs ne va pas sans poser certains problèmes inhérents au développement collaboratif. Pour assurer la cohérence du développement et le suivi des versions, de nombreux outils existent (on citera CVS par exemple) et il est nécessaire de recourir à ce type d'outils pour ce projet. Par ailleurs, une architecture bien conçue permettra à tous de personnaliser une partie du code sans compromettre la structure de l'ensemble. Cela se traduit concrètement par une approche modulaire (c'est-à-dire que l'on peut utiliser, remplacer, ajouter des portions de code dans l'architecture) et par la distributivité du système (un élément requérant une grosse puissance de calcul doit pouvoir facilement être déporté sur une autre machine pour éviter de perturber l'ensemble de l'architecture).
- La possibilité de changer les paramètres de la simulation sans avoir à re-compiler à chaque modification est un aspect pratique important. Pour cela il est possible d'utiliser des fichiers de configuration qui sont lus dynamiquement pendant la phase d'initialisation d'une simulation.
- La description des caractéristiques des entités participantes, telles que les AUVs, les modèles, ... repose généralement sur des fichiers au format propriétaire, le plus souvent sous forme de fichiers de données non formatées. L'usage de format propriétaire, le plus souvent sous forme de fichier texte, est à prohiber. Cela rend peu compréhensible le contenu de ces fichiers et il est difficile de les faire évoluer. Un exemple de ce genre de fichier est donné figure 3.1 ; de même dans [38], les auteurs font mention de fichiers texte de configuration pour leur simulateur. Il sera donc nécessaire de développer un formalisme permettant de décrire les éléments du simulateur de façon plus universelle. Ce formalisme devra pouvoir évoluer avec la pluralité des systèmes connectés au simulateur tout en restant compatible avec les différents modules existants. Il faudra donc favoriser une approche incrémentale.

Ce type de simulateur doit donc être ouvert et envisagé dans l'optique d'un travail *collaboratif*. Cela permet de pouvoir exploiter les avancées des différentes communautés en intégrant par exemple leurs modèles au sein du simulateur. En effet, le nombre de domaines différents est important, et il est peu probable qu'une seule équipe regroupe des connaissances dans l'ensemble de ces domaines (hydrodynamique, propagation des ondes acoustiques ou électromagnétiques, traitement du signal,...). Force est de constater que l'ensemble des simulateurs étudiés ne présente pas l'ouverture nécessaire à un tel projet. Seul le simulateur CADCON propose de télécharger les exécutables des clients pour leur simulateur (pas de code source disponible), qui par ailleurs n'est que très rarement "online", ce qui ne permet pas de lancer des simulations quand on le souhaite. Un gros effort doit donc être fait si l'on souhaite que "la compétence de chacun au service

```
dynamic <uris.dyn>
model <uris.wrl>
scale <0.0030>
radius <12.6>
position <0.0 0.0 0.0 0.0 0.0 0.0>
sensor <downward><
 x_max <10.0>
 beta <0.3>
 inc_x <0.3>
 inc_z <0.2>
 position <0.5 0.25 0.5 0.0 -90.0 0.0>
>
sensor <forward><
 x_max <10.0>
 beta <0.3>
 inc_x <0.3>
 inc_z <0.2>
 position <0.0 0.0 1.25 0.0 -30.0 0.0>
```

Fig. 3.1 – Les fichiers de configuration du simulateur Neptune sont décrits grâce à un langage propriétaire.

de tous" soit une phrase qui ait du sens pour ce simulateur.
Une fois l'architecture construite, l'étape suivante consiste à modéliser les éléments de la simulation, avant leur implémentation au sein de l'architecture.

3.3 Une modélisation adaptée

La première étape de ce travail consiste à définir avec précision le cadre expérimental dans lequel on se place et déterminer quelle est la précision requise, pour spécifier la finesse des modèles. Un simulateur ne devrait donc pas intégrer le "meilleur" modèle dans tel ou tel domaine, mais permettre de construire un modèle au minimum adapté au contexte dans lequel on se place. Ce travail préliminaire est donc une nécessité pour éviter de déboucher sur des incohérences. Ce n'est pas la puissance de calcul qui doit guider nos choix dans la modélisation. Nombreux sont les auteurs qui choisissent de dégrader leurs modèles (sonar et dynamique des robots notamment) car ils ne disposent pas d'assez de puissance de calcul. De même implémenter le modèle le plus exact d'un élément (capteur, actionneur, ...) ne servira à rien, si les autres modules lui fournissant des données sont grossièrement modélisés. Dans le cadre du multi-véhicule, il faut donc éviter de créer un simulateur précis pour un aspect particulier (dynamique d'un AUV en forme de torpille, ou bien propagation acoustique, ou bien modélisation exacte d'un capteur de pression,...), mais privilégier l'utilisation de *modèles suffisants* pour notre problématique. Cette étude préalable rejoint la notion d'ouverture en ce que le défaut principal des simulateurs actuels est de se focaliser plus particulièrement sur la précision d'un modèle ou deux (soit le sonar, soit la dynamique des engins, soit l'environnement, ...). Ainsi des chercheurs appartenant à des communautés différentes (élaboration de protocoles de communication aquatique, développement de modem acoustique, automatique, acoustique sonar, ...) commencent également à se trouver limités dans le développement de leur recherche et se retrouvent "confinés". Par exemple dans [50], les auteurs sont spécialisés dans le domaine de la communication aquatique et ont créé un nouveau protocole de communication. Pour autant, ils regrettent le peu d'interactions entre la communauté des automaticiens et la leur car

ils n'ont pas de véritable modèle pour le déplacement des nœuds (les AUVs, bouées, ...) de leur réseau ; ils expliquent également en quoi les simulateurs actuels ne supportent pas le protocole proposé.
Ces remarques guideront donc nos choix durant les étapes de modélisation : **nous chercherons à analyser le *degré de finesse requis* et à développer des modèles adaptés**. Nos connaissances et notre temps étant nécessairement limités, certains modèles pourraient se révéler très simples. **Il faut donc se garder la possibilité de pouvoir faire évoluer ces modèles par la suite**, en faisant éventuellement appel à des compétences externes à l'équipe ; l'architecture produite devra donc faciliter ce type d'intervention.
Une fois l'architecture créée, documentée et la réflexion sur les modèles menée, on peut songer à intégrer les robots à l'environnement virtuel.

3.4 Multi-véhicule

La plupart des simulateurs étudiés sont des simulateurs mono-véhicules. Par ailleurs parmi ceux qui gèrent le multi-véhicule, un seul ([44]) permet potentiellement de simuler autre chose que des AUVs. Pourtant plusieurs équipes possédant des engins de classes différentes commencent à travailler ensemble afin d'élaborer des scenarii multi-véhicules et de mener des expérimentations dans ce domaine. On citera notamment le Programme d'Etudes Amont Action (PEA Action), dont le but est d'étudier les moyens disponibles et de préparer les technologies futures en vue de renforcer les performances de la fonction localisation dans un réseau d'entités hétérogènes constituées de vecteurs autonomes. Plusieurs thèmes seront abordés dans le cadre de ce programme parmi lesquels la collaboration, le partage d'informations, la coordination et la gestion de flottille. Plusieurs scenarii sont envisagés, faisant intervenir des véhicules de classes différentes (voir figure 3.2).

On peut également citer le projet CONNECT [51], dans lequel l'IFREMER (Institut Français de Recherche pour l'Exploitation de la MER) est impliqué, et qui traite du problème du contrôle des systèmes multi-agents connectés par un réseau de moyens de communication hétérogènes. Le but du projet est de contrôler des groupes d'agents composés de véhicules sous-marins et surfaciques sous contrainte de communication. Il est à noter qu'un des objectifs de ce projet est de créer un simulateur permettant d'intégrer les modèles de ces agents pour confirmer les études théoriques faites en amont.

Les missions multi-véhicules sont souvent sophistiquées et le multi-véhicule ne s'arrête donc pas à mettre en œuvre plusieurs fois le même engin : les capteurs embarqués peuvent être différents, et les véhicules hétérogènes ; c'est sans doute là que réside la force de ce type de scénario : être capable d'employer plusieurs véhicules complémentaires, transportant des capteurs différents permettant des perceptions différentes.
L'architecture devra donc être mesure de simuler des *robots très différents*, évoluant potentiellement dans des *milieux différents*.

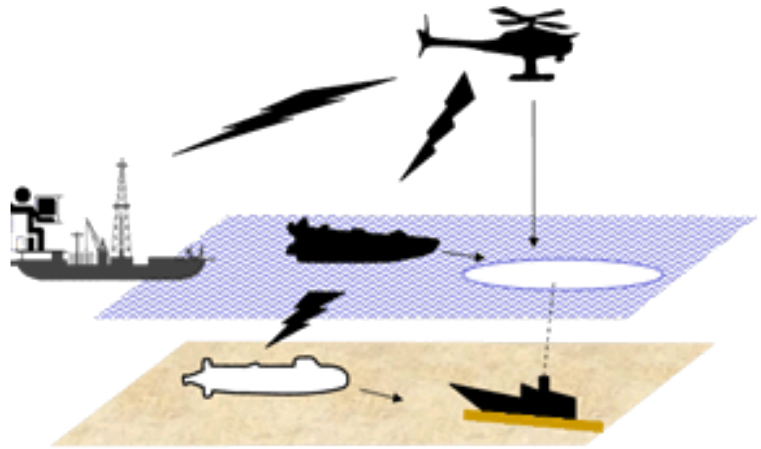


Fig. 3.2 – Partage d'information entre 1 UAV, 1 AUV et 1 USV dans le cadre d'une mission de lutte antipollution

3.5 Connectabilité

L'utilisation de robots différents, provenant d'organismes différents, induit souvent des logiciels de contrôle (de ces robots) réellement différents. En effet, les architectures de contrôle sont généralement conçues spécifiquement pour un robot et il n'est pas envisageable de demander aux différents intervenants d'opter pour une architecture générique, qui serait compatible avec le simulateur. Il faut laisser à tous la liberté d'utiliser le matériel et les logiciels habituels afin de pas induire de biais qui retarderaient à coup sûr les expérimentations. **Il faudra donc faire en sorte que l'architecture puisse se connecter à n'importe quel contrôleur en gardant comme objectif de *minimiser les modifications* sur le logiciel contrôleur et donc de minimiser les impacts sur le *comportement temporel* de ce contrôleur** [52]. Une attention particulière doit être apportée au niveau des formats de données échangées et aux protocoles des échanges entre contrôleurs et simulateurs. Cela peut prendre la forme d'une spécification de ces échanges ou par l'écriture d'un client à exécuter sur le contrôleur hôte gérant les émissions/réceptions et le codage/décodage des trames vers le simulateur. Dans ce cas il sera nécessaire de prendre soin de ne pas perturber le comportement temporel du logiciel de contrôle en préservant les ressources dont il a besoin. La norme HLA (High Level Architecture), développé par le DoD (Department of Defense) pour les systèmes de simulation distribués, pourrait représenter une solution intéressante. Cette norme permet entre autre aux ordinateurs impliqués dans la simulation de communiquer avec les autres ordinateurs sans avoir à se préoccuper des plate-formes d'exécutions.

Modéliser et décrire les entités (véhicules, capteurs, éléments de l'environnement,...) peuplant la simulation est une opération longue et fastidieuse. Un autre aspect revêt une importance particulière dans ce type de simulateur : la modularité.

3.6 Modularité

La modularité est une approche qui permet de structurer un logiciel en petites unités qui, une fois rassemblées, composent l'ensemble du logiciel. Ces petites unités, appelées modules peuvent alors être modifiées, retirées, ou ajoutées sans interférer avec le fonctionnement des autres éléments. Cette approche autorise à chacun, indépendamment des autres, de réaliser des tests unitaires. L'autre aspect de la modularité concerne la réutilisabilité du travail effectué qui permet d'éviter de redévelopper l'existant. Au final, cela permet d'obtenir un projet structuré, fruit de l'assemblage de différents modules. Cet aspect n'a pas été clairement évoqué dans les différents articles, mais il s'agit d'un fondement de la simulation pluri-acteur : il est indispensable de pouvoir connecter des modules entre eux (par exemple utiliser le module capteur pression sur l'AUV Taipan300) et de pouvoir réutiliser les modules existants (par exemple, reprendre le module capteur de pression de l'AUV Taipan300 et l'utiliser cette fois-ci sur un Remus sans aucune modification). **Notre simulateur devra donc offrir un haut niveau de *modularité*** afin de minimiser le temps de développement lorsqu'un nouvel acteur prend part à une simulation. Cette modularité s'entend à plusieurs niveaux ; le bas niveau constitué des composants "primaires" de systèmes plus complexes. Il s'agit typiquement des modèles de capteurs, d'actionneurs, et de moyens de communication. L'assemblage de ces composants donne naissance à un ensemble plus sophistiqué. Il s'agit principalement des modèles de robots (ou de certaines bouées équipées) et cela constitue le niveau intermédiaire. Enfin, l'assemblage de tous les systèmes sophistiqués forme le haut niveau. Il s'agit de la composition des modèles de scènes physiques, auxquelles sont intégrées les phénomènes environnementaux. **Le simulateur devra donc permettre de changer et réutiliser facilement ces composants *à la volée*** (c'est-à-dire sans passer par une compilation). Les composants devront donc être décrits de façon la plus générique possible, afin de respecter cette contrainte.
Si la modularité des systèmes de simulation permet de manipuler facilement les composants, cela ne résout pas pour autant le problème de la charge de calcul.

3.7 Distributivité

Ce terme désigne la capacité du simulateur à être réparti sur une ou plusieurs unités de calcul distinctes. Ces unités de calcul (qui peuvent être soit plusieurs processeurs à l'intérieur d'une même machine, soit plusieurs ordinateurs) sont connectées grâce à un réseau de communication. Chacune des unités de calcul a donc en charge d'exécuter une séquence d'instructions puis de retourner éventuellement le résultat vers une autre unité. Chacune de ces unités est autonome, et il n'existe pas de composant maître. Cette propriété permet :

- *d'étendre* les capacités de calcul du système en multipliant le nombre d'unités de calcul
- *d'ouvrir* le système car les composantes distribuées ont nécessairement des interfaces clairement définies, ce qui garantit une certaine capacité du simulateur à s'interconnecter avec d'autres entités
- d'obtenir un système potentiellement *hétérogène* : les composantes peuvent être écrites dans des langages différents, exécutées sur des architectures système diffé-

rentes (sun, x86, ...) et donc fonctionner sous des OS (Operating System) différents. A ce titre, le recours à une architecture logicielle de type CORBA (Common Object Request Broker Architecture) peut présenter certains avantages.

– *limiter* la quantité de données échangées sur un seul lien de communication

Dans le cadre de la simulation, une distinction est à faire entre simulateur distribué et simulation distribuée. En effet, dans le premier cas, le simulateur (c'est-à-dire l'ensemble des composants chargés de faire évoluer la dynamique des systèmes, de calculer les réponses des capteurs, de calculer les interactions avec l'environnement,...) est réparti sur plusieurs unités de calcul. Dans le second cas, une simulation distribuée désigne un contrôleur connecté à un simulateur, ce dernier étant exécuté sur un seul et unique ordinateur. Ce type de simulateur ne possède donc pas les propriétés précédemment énumérées. Parmi les simulateurs étudiés seuls quatre (DVECS [41][40] ; CADCON [35][48] ;MVS [44] [47] ; Neptune [13]) peuvent réellement être qualifiés de distribués. Pour autant distribuer un système n'est pas suffisant : les choix faits à la conception de ces architectures ne permettent pas d'augmenter les capacités de traitement de leur système. Tous les scenarii ne pourront donc être envisagés, pas plus que l'utilisation de modèles raffinés consommateurs de ressources. Nous introduisons alors le terme de *distributivité statique* pour qualifier ce type de simulateur. Typiquement ces simulateurs se distinguent par le fait que les concepteurs choisissent à un instant donné de dégrader volontairement la précision des modèles (par exemple dans [38]), ou de ne plus travailler en temps-réel (par exemple dans [33]). Quelle que soit la finesse des modèles choisis, un simulateur non distribué sera de toute façon limité par le nombre de véhicules simulés.

Pour toutes ces raisons, notre objectif est de développer un simulateur *distribué*, qui nous permettra donc d'aborder le contexte du multi-véhicule, en temps-réel, et au travers d'un travail potentiellement collaboratif. Dans un deuxième temps, **notre but sera de pouvoir étendre *dynamiquement* (c'est-à-dire adapter à volonté) notre puissance de calcul** ; on pourra alors parler de *distributivité dynamique*. L'architecture imaginée devra donc être capable de répondre à ces exigences.

Il est à noter que peut potentiellement se poser ici le problème de la synchronisation des simulateurs. En effet, lorsqu'ils sont tous exécutés en local sur une même machine comportant plusieurs processeurs, le problème ne se pose pas puisque tous les processus ont la même base de temps, et dans ces conditions il ne peut y avoir de dérive de temps d'un processus à l'autre. En revanche, lorsque ces processus sont distribués sur un réseau, cet aspect est à considérer. Nous avons pour l'instant fait le choix de mettre ce problème de côté au vu des faibles durées (quelques heures tout au plus) de nos simulations. Différentes réponses technologiques existent à ce problème (protocole NTP (Network Time Protocol, ajout de récepteur GPS sur les unités de traitement, ...) et pourront être étudiées ultérieurement.

Simuler plusieurs véhicules ne suffit pas. Il est nécessaire que ces véhicules puissent communiquer au sein de leur univers pour accomplir leur mission.

3.8 Communication inter-véhicules

La problématique de la communication, que ce soit en milieu aquatique ou en milieu aérien, fait l'objet de nombreuses recherches. Ces recherches sont menées dans des communautés assez "déconnectées" de la nôtre comme nous l'avons évoqué précédemment.

C'est sans doute pour cette raison que très peu de simulateurs offrent la possibilité aux véhicules de communiquer entre eux (voir le tableau 2.13) et que parmi ces cinq simulateurs, aucun ne prend en compte la vraie dynamique des communications. Elle sont toutes considérées comme étant instantanées, exemptes d'erreurs, et systématiquement distribuées à l'ensemble des véhicules. Ceci est extrêmement éloigné de la réalité, et à plus forte raison pour les communications aquatiques. Dans ce contexte, les algorithmes utilisés pour le contrôle de la flottille, ne peuvent être correctement évalués et validés au moins dans le contexte des communications aquatiques. Cette problématique sera abordée dans les paragraphes suivants, et permettra de pleinement justifier notre position sur ce point critique. On peut considérer qu'il existe au moins trois façons différentes d'aborder cette problématique. Elles correspondent au modèle de propagation sonore :

- La première méthode est celle qui est largement utilisée dans les articles étudiés, et qui consiste à considérer des communications sans erreur, non sujettes au phénomènes de multi-chemins, sans atténuation (tous les participants reçoivent le message), sans considérer le bruit ambiant, où l'ensemble du message parvient instantanément dans son intégralité (bande passante et vitesse de propagation infinie). En somme, l'implémentation consiste à partager des variables dans une mémoire partagée accessible à l'ensemble des acteurs.
- La seconde consiste à reproduire les conséquences dues aux phénomènes physiques sur les communications, sans en modéliser les causes. On introduit ici le terme d'*émulation de la couche physique*. Cela signifie que cette façon d'opérer permet d'imiter le comportement de la propagation des ondes avec une partie des conséquences de cette propagation sur l'information transportée. Pour donner un exemple concret, nous pouvons évoquer le débit de communication d'un modem acoustique. Si ce dernier essaie d'émettre à un débit supérieur à la capacité offerte par le canal acoustique, les pertes seront nombreuses et la communication ne pourra s'établir. Émuler ce phénomène physique consiste donc à détériorer artificiellement la communication pour reproduire cette réalité.
- La dernière, qui est la plus complexe, met en jeu les réelles équations de propagation des ondes. Dans ce dernier cas, l'onde se propage dans le temps et dans l'espace pendant la simulation. Ces modèles souvent complexes font l'objet d'études menées par des spécialistes du domaine (simulation de la zone de couverture pour les téléphones portables, simulation de propagation acoustique avec modèle de diffusion des matériaux impactés pour la simulation des sonar, ...). Cette façon de procéder offre souvent (cela dépend du modèle utilisé) une représentation fidèle des phénomènes réels.

Si la première méthode est insatisfaisante pour préparer des expérimentations réelles, la dernière sort largement de nos possibilités en terme de temps de travail et de connaissances (on constate bien ici, que la collaboration entre différents spécialistes est nécessaire pour ce type de travail). **Nous avons donc choisi de mettre au point une méthode *émulant* les conséquences des perturbations environnementales sur les communications**. Ceci est à notre sens, le minimum pour aborder la problématique du multi-véhicule. Une étude des phénomènes liés à la propagation d'une onde devra donc être menée pour identifier les phénomènes clés et reproduire ainsi une "certaine" réalité que nous définirons en fonction de nos besoins. En revanche, émuler cette réalité ne doit pas nous empêcher de pouvoir faire évoluer les modèles vers la troisième solution *a pos-*

teriori. **Notre architecture devra donc favoriser l'implémentation future d'un autre type de modèle de propagation**.
Si nous avons parler jusqu'à présent de la propagation, il ne faut pas perdre de vue pour autant que la notion de communication ne se limite pas à cela. En effet, pour pouvoir communiquer, les véhicules utilisent un moyen de communication. Il peut s'agir d'un modem, d'une radio, de moyens opto-électroniques,... Il est à noter que chacun de ces engins possède en général plusieurs moyens pour communiquer, adaptés au médium dans lequel ils se trouvent, à la quantité et au type d'information à transmettre. **Nous envisageons donc de considérer la *pluralité des moyens de communication* sur un robot, et d'adapter le modèle de propagation au médium dans lequel il est censé fonctionner**. Aucun des simulateurs étudiés n'aborde cette problématique qui est pourtant un point critique de la communication. En effet, chaque moyen de communication possède des caractéristiques (puissance d'émission, de réception, sensibilité au bruit, bande passante, directivité, ...) et des comportements (mise en veille automatique, temps de réveil, accumulation de données avant émission,...) qui leur sont propres. Habituellement, seuls les scientifiques appartenant à des communautés développant ce type de matériel, possèdent des simulateurs capables de modéliser ce fonctionnement. Pour autant il s'agit d'un point critique qu'il est nécessaire d'intégrer à une simulation dont l'objectif est de préparer une expérimentation. Même dans le cadre d'évaluation et de validation de lois de commande, l'implémentation de ces comportements ne semble pas superflue. **Notre architecture devra donc supporter des modèles de ces moyens de communication et reproduire leur *comportement logique et temporel***.
Un dernier point concernant les communications doit être évoqué : les interférences. En effet, lorsque les engins ne sont pas synchronisés entre eux, ils peuvent être amenés à "parler" simultanément sur le même canal, brouillant ainsi le message de ceux se trouvant dans la zone d'interférences. Si des solutions existent dans le domaine aérien (ou filaire bien-sûr) le domaine aquatique pose problème car les communications ne se font pas de façon instantanée, et la bande passante est fortement limitée. Cette problématique fait l'objet d'études menant vers la mise au point d'un protocole spécifiquement adapté au monde aquatique [50]. Cette réalité, qui n'apparaît dans aucun des articles étudiés, contraint fortement le problème et doit nécessairement être traitée. Par ailleurs, il existe potentiellement un autre type d'interaction susceptible de brouiller les communications, au moins en milieu aquatique : les interférences liées à l'utilisation de capteurs travaillant sur la même bande de fréquences : typiquement les sonars et les écho-sondeurs rentrent dans cette catégorie. La prise en compte de ces interférences est également nécessaire. Une architecture logicielle de contrôle a d'ailleurs été développée pour les robots embarquant ce type de capteurs [53], [54]. Un des objectifs de cette architecture de contrôle est d'offrir la possibilité de gérer le recrutement des capteurs interférents de façon à éviter les interférences. Être en mesure de tester de telles fonctionnalités d'une architecture nécessite donc de prendre en compte ces perturbations. **Notre objectif sera donc de tenir compte des *interférences* dans les modèles de communication utilisés, et d'offrir aux utilisateurs de définir eux-mêmes quels sont les capteurs, et les moyens de communication susceptibles d'être interférents**.
La notion du temps est très présente dans la simulation des communications. La capacité du système à travailler en temps-réel est donc une autre nécessité.

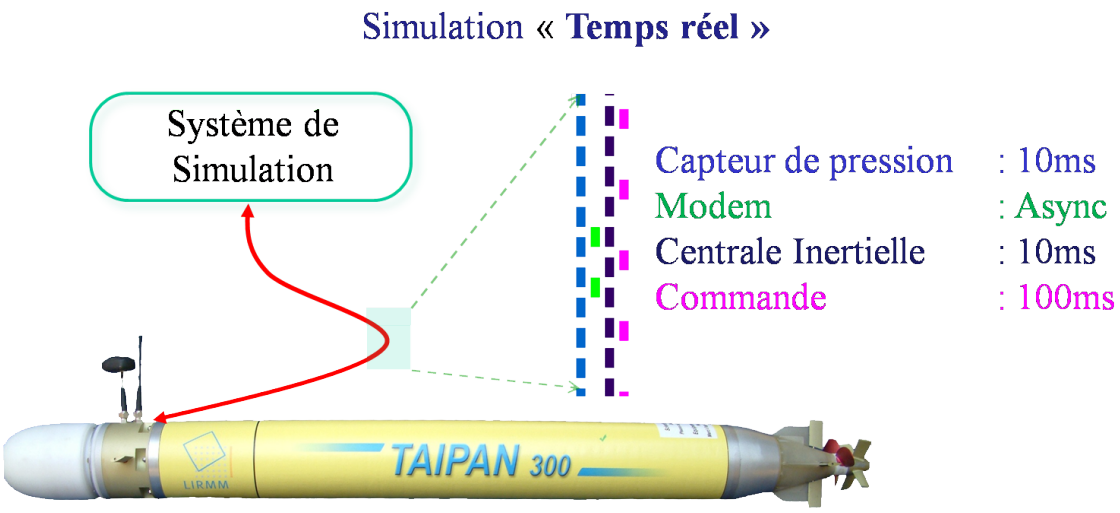


Fig. 3.3 – Le contrôleur de bord du véhicule est connecté au simulateur et échange avec lui des informations en un temps borné.

3.9 Temps-réel

Un système temps-réel a la particularité de fonctionner à une vitesse adaptée à l'évolution des entités avec lesquelles il est interfacé. Il doit donc prendre en compte les contraintes temporelles et veiller à les respecter car le respect des délais imposés est aussi important que l'exactitude du résultat. Un système ne pourra être qualifié de temps-réel, uniquement si l'ensemble de ses composants peuvent également être qualifiés de la sorte. Cela ne signifie donc pas qu'un système temps-réel est un système qui "calcule vite", mais plutôt qui "produit le résultat au bon moment". Il est à noter que cette rapidité de calcul peut ne pas être unique dans le système : différentes rapidités peuvent être employées selon la dynamique de l'entité avec laquelle est connecté le système (figure 3.3). Par ailleurs un système temps-réel est déterministe (c'est-à-dire que son comportement est connu à l'avance et reproductible), et il est possible de contrôler l'ordre dans lequel sont exécutés les tâches (ordre de précédence). **Pour opérer de la sorte, notre simulateur devra donc respecter des contraintes de *temps-réel***. Les propriétés du temps-réel sont respectées grâce au principe selon lequel chaque tâche doit s'exécuter en un temps borné. Cette contrainte a de nombreuses conséquences sur la conception des algorithmes et de l'architecture. Ainsi, un système temps-réel ne sera pas compatible avec certaines méthodes d'intégration qui ne peuvent fournir un résultat en un temps borné ; nous aurons l'occasion de revenir sur ce point important par la suite. Seuls les auteurs de [21] ont exposé leur méthode d'intégration. Ce point nous semble pourtant primordial puisque le choix de la méthode d'intégration et ses caractéristiques sont liés au problème étudié. Un pas d'intégration à 1 seconde par exemple, ne sera pas adapté à la simulation d'un problème possédant une dynamique de l'ordre de 100ms... De même, utiliser une méthode d'intégration basée sur la borne d'erreur acceptable à chaque pas, sera incompatible avec une commande de type sliding-mode.
Par ailleurs, une architecture temps-réel n'est pas compatible avec la distribution de ses composants sur un réseau non déterministe. Comme nous l'avons évoqué précédemment,

pour être qualifié de temps-réel, une tâche doit s'exécuter en un temps borné. Cela signifie donc que les données nécessaires à la tâche doivent être acheminées en un temps borné également : il faut donc être assuré que l'information mette un temps inférieur à une certaine durée, connue, pour atteindre un point du réseau. Utiliser ce type de réseau impose également des contraintes sur sa topologie. Bien que les réseaux ethernet ne puissent pas être qualifiés de déterministes, il est possible d'envisager le déploiement d'un système temps-réel sur ce type de réseau, en prenant des précautions sur lesquelles nous reviendrons en détail. En revanche nous pouvons affirmer qu'utiliser le réseau Internet pour relier les différentes entités d'un système de simulation n'est pas compatible avec la simulation matériel-dans-la-boucle de véhicules tels que des AUVs. En effet, comme pour les réseaux non déterministes, il est impossible de connaître à l'avance la durée de cheminement d'un paquet à travers ce réseau, à la différence que ces temps sont beaucoup plus souvent supérieurs à la fréquence d'échange des données avec le robot (fréquence d'échantillonnage de certains capteurs $<$ 10ms par exemple). Cet état de fait ne semble pas préoccuper l'ensemble des auteurs proposant cette fonctionnalité. Sauf cas particulier (commande temps-réel d'un système de chauffage avec une dynamique très lente par exemple), il n'est pas envisageable d'utiliser le réseau Internet comme moyen de transport de l'information temps-réel, dès lors que la dynamique des systèmes nécessite des échanges à une fréquence inférieure à 100ms. **Des réflexions sur le *type de réseau* à utiliser pour notre architecture, devront donc être menées pour en déterminer les limites, et les contraintes associées**. Même si certains auteurs utilisent des OS temps-réel pour leur simulateur, on ne trouve pas de réflexions concernant leur choix dans les articles ; une exception toutefois dans [37], qui étudie la façon dont la norme PosiX est implémentée dans différents OS, et quelles sont les conséquences sur la simulation.
Une notion connexe au temps-réel doit également être abordée : le découplage temporel.

3.10 Découplage temporel

Le découplage temporel consiste à faire en sorte que deux tâches différentes soient libres d'évoluer dans le temps, sans contraintes liées à la réalisation de l'autre tâche. Le couplage temporel n'existe pas dans la réalité : une vague n'attend pas qu'un surfer soit prêt pour se dérouler. La vague existe, et le surfer doit agir au bon moment pour surfer à sa surface (sans boire le bouillon si possible). Il en va de même pour le système de simulation : l'environnement simulé (au sens large du terme) ne doit pas attendre que le contrôleur réagisse pour évoluer. Concrètement dans le monde réel si le contrôleur donne les commandes actionneurs avec du retard, les conséquences pour l'engin contrôlé peuvent être désastreuses (il peut percuter un obstacle pour un AUV, un USV ou un UGV, ou sortir du domaine de vol et devenir incontrôlable pour un UAV...). Si le principe du découplage temporel n'est pas respecté au sein de l'architecture, les comportements divergeants ne pourront pas être observés pendant la simulation, puisque cela revient à suspendre l'évolution des modèles dynamiques, en attendant que le contrôleur réagisse. Cet aspect de la simulation n'a jamais été évoqué dans les articles étudiés, mais revêt une importance capitale pour l'ensemble des simulateur HIL ou hybrides. **Nous veillerons donc à assurer le *découplage temporel* entre la commande et l'évolution des modèles dynamiques**.
Un système de simulation temps-réel et découplé temporellement du contrôleur, sont les

bases nécessaires pour aborder la simulation HIL.

3.11 Hardware In Loop

Nous ne reviendrons pas dans cette section sur la définition de ce qu'est un simulateur matériel-dans-la-boucle puisque nous avons largement abordé le sujet dans la section 1.5. Les publications étudiées ne sont pas très explicites quant à la manière dont est connecté le véhicule au simulateur. Les auteurs précisent souvent que cette connexion est entièrement "transparente" et qu'elle ne nécessite pas d'adaptation sur le contrôleur. Néanmoins un minimum de modifications est nécessaire (au minimum, il faut redirigier les commandes vers le simulateur).
Notre but est donc de créer un *simulateur HIL*, en minimisant l'impact sur le comportement de l'architecture. L'objectif est de ne faire aucune différence entre l'écriture d'une mission réelle et celle d'une mission simulée. L'impact de la connexion de l'architecture au système de simulation est une problématique que nous avons clairement identifiée, et sur laquelle nous reviendrons dans une future section.

3.12 Hybride

De même que la notion HIL, nous avons largement abordé la simulation hybride dans le paragraphe 1.6. Seuls deux simulateurs offrent cette fonctionnalité, mais elle peut se révéler utile dans la préparation des missions : elle permet de vérifier réellement le contrôle de l'engin, et de tester des algorithmes nécessitant des capteurs qui ne sont pas obligatoirement à bord. Dans ce cas, il est possible de créer un capteur virtuel et de valider le comportement réel de l'engin.
Nous envisageons cette fonctionnalité différemment de ce que propose la littérature. En effet, dans les systèmes étudiés, l'engin est toujours relié au simulateur par un câble permettant l'échange de données entre le système et le simulateur. **Nous envisageons plutôt d'embarquer notre simulateur à bord du robot (pour le mode hybride), sur un ordinateur dédié**. Pour rappel, cette propriété permet de se placer à mi-chemin entre l'expérimentation dans le cadre opérationnel et la simulation : on utilise le véritable robot dans un milieu sécurisé (cela permet donc de s'affranchir du calcul de l'évolution de son modèle dynamique, souvent mal connu), et en augmentant virtuellement la réalité qu'il perçoit (il perçoit donc l'environnement réel avec ses capteur réels, mais aussi un environnement virtuel grâce à des capteurs virtuels également). Cette possibilité n'est pas notre priorité, mais l'architecture que nous concevrons devra permettre d'exécuter le simulateur indifféremment sur une ou plusieurs unités de calcul.

3.13 Affichage 3D

En ce qui concerne l'affichage 3D, cette fonctionnalité est largement présente dans les publications abordées. Néanmoins un seul simulateur [25] permet de simuler des caméras. Cette possibilité permet de tester des algorithmes liés à la thématique de la vision. On peut citer la surveillance et le suivi de pipeline, le calcul du vecteur vitesse du véhicule à partir de séquences vidéo, le suivi de fond, la reconstruction 3D texturée du fond (par

projection de grille laser ou stéréoscopie par exemple)...
Nous développerons donc le *rendu 3D* des scènes de notre simulateur afin de tester et valider ce type d'algorithme. Actuellement, aucun des affichages 3D existants ne permet de représenter les communications qui se déroulent pendant la simulation entre les véhicules. **Notre objectif est donc aussi de permettre la visualisation de ces communications**, notamment leur durée, la propagation de l'onde, les interférences, les niveaux de réception et d'atténuation,...

3.14 Conclusion

Dans cette section, nous nous sommes positionnés par rapport aux travaux existants. Nous avons constaté qu'aucun des simulateurs existants n'est adapté à notre contexte. Si certains d'entre eux possèdent certaines caractéristiques essentielles, il n'en demeure pas moins qu'ils souffrent de défauts rendant impossible leur utilisation pour notre application.
Nous avons alors défini les caractéristiques principales qui guideront nos choix de conception. Nous avons ainsi déterminé que la création d'une architecture ouverte permettant le travail collaboratif constituait la première étape fondamentale du processus de création d'un simulateur. Une analyse précise de nos besoins nous permettra ensuite de déterminer la finesse des modèles employés en veillant à préserver la possibilité de les faire évoluer par la suite. Des robots différents pourront se connecter au simulateur et l'objectif est de minimiser l'impact de cette connexion sur le comportement temporel du contrôleur. Une approche modulaire permettra de minimiser les temps de développement pour la connexion de nouveaux véhicules ou l'utilisation de nouveaux capteurs, et de se garder la possibilité de changer facilement de modèles. La somme de calculs potentiellement conséquente nécessite de pouvoir distribuer notre architecture sur plusieurs unités de calcul appartenant à un réseau. Cette approche nous permettra de réaliser des simulations temps-réel multi-véhicules sans concession sur la finesse des modèles. Pour aborder cette problématique, nous avons besoin d'un simulateur dans lequel les véhicules puissent échanger des messages entre eux. Nous souhaitons donc mettre au point une méthode émulant les conséquences des perturbations sur la propagation et tenir compte du comportement logico-temporel des moyens de communication associés. Nous avons montré qu'un simulateur HIL était requis et nous envisageons de le faire évoluer pour faire de la simulation hybride embarquée. Enfin, un affichage 3D nous permettra de visualiser la scène, y compris les communications s'y déroulant. Il sera également possible de simuler des capteurs d'images (indépendamment de l'afficheur 3D). Accessoirement, nous souhaitons également développer une interface graphique, permettant de configurer et superviser le déroulement d'une simulation.
L'ensemble de ces points qui nous semblent essentiels pour aborder pleinement notre problématique, ne seront pas tous implémentés dans les premières versions de notre travail. En effet, nous avons listé ici les caractéristiques devant nécessairement être prises en compte dès la conception. Nous développerons donc notre architecture de façon incrémentale, nous garantissant ainsi de toujours pouvoir enrichir et ajouter des fonctionnalités et des modèles.

Deuxième partie

Thetis, simulateur hybride multi-véhicules

Introduction

Après avoir étudié les simulateurs existants et proposé une classification permettant de nous positionner par rapport à la problématique de la simulation multi-véhicules, nous présentons dans cette partie, l'étude et la réalisation d'un système de simulation autorisant la simulation online, HIl et hybride dans le contexte de la préparation de missions multi-véhicules.

Notre proposition concerne la conception et l'implémentation d'une architecture de simulation supportant un ensemble de modèles, permettant de simuler simultanément plusieurs engins hétérogènes. En outre, ce simulateur leur permet de communiquer entre eux en respectant les contraintes liées au matériel de communication utilisé et au medium de propagation.

Cette partie s'architecture donc en trois temps. Nous commençons par présenter notre proposition d'architecture, puis exposons sa réalisation. Nous précisons dans ce deuxième chapitre, les technologies que nous avons retenues en indiquant quelles en sont les implications sur l'exploitation du simulateur.

Enfin nous introduisons les modèles mis en œuvre à l'intérieur de cette architecture ; certains de ces modèles issus de la littérature sont donnés en annexe. En revanche, nous indiquons clairement pour chacun de ces modèles, les hypothèses de départ et les implications de ces hypothèses sur le domaine de validité du simulateur.

Chapitre 4

Proposition d'une architecture

4.1 Introduction

Nous avons apporté un soin particulier à développer une architecture avant d'y intégrer les différents modèles. Cette façon de procéder nous permet de rendre le système de simulation compatible avec les objectifs que nous nous sommes fixés dans le chapitre précédent. Nous avons donc porté une attention particulière à la capacité intrinsèque de cette architecture à pouvoir être distribuée sur plusieurs unités de calcul. Cela nous a permis de respecter les contraintes liées au temps-réel, et à la somme de calculs que peut générer le recours à des modèles fins. Cette approche nous a donc permis de créer un système de simulation dans lequel les modèles sont choisis en fonction de critères scientifiques liés à une application particulière et non en fonction de la puissance de calcul disponible. Par ailleurs nous avons également veillé à produire un code source documenté et mis en ligne pour veiller à la notion d'ouverture de notre simulateur.

Notre architecture repose sur quatre simulateurs pouvant être exécutés sur des unités de calculs différentes. Ces unités peuvent être des processeurs à l'intérieur d'une même machine physique, ou des processeurs sur des machines physiques différentes, connectées entre elles par un réseau dédié. Les raisons qui nous ont poussé à opérer ce découpage sont données au paragraphe 4.2.1. Chacun de ces simulateurs occupe un rôle particulier dans le système : le premier est le simulateur de la dynamique de l'ensemble des véhicules, le second est le simulateur de l'ensemble des capteurs de chaque véhicule, le troisième est un simulateur de moyens de communication. Enfin, le dernier est un simulateur d'environnement en charge d'effectuer les calculs des phénomènes environnementaux, de la propagation des signaux, et de collisions véhicule/véhicule ou véhicule/environnement.

Ce chapitre est donc consacré à la présentation d'une architecture répondant aux exigences que nous avons établies. Nous y présentons le fonctionnement de l'ensemble, puis détaillons chacune des entités composant le système de simulation. Enfin nous vérifions que l'architecture proposée répond bien aux exigences que nous nous sommes fixées avant de conclure.

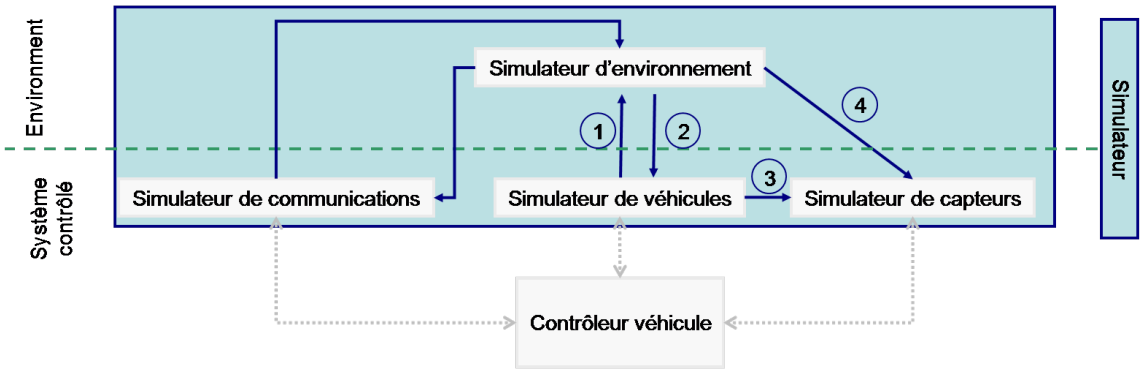


Fig. 4.1 – Thetis est articulé autour de quatre simulateurs

4.2 Présentation globale

Dans cette section, nous présentons les fondements du système de simulation "vu de haut", avant de donner les détails d'un module que nous avons qualifié d'*élémentaire* sur lequel repose l'architecture. Enfin, nous expliquons brièvement comment les véhicules sont connectés au système de simulation.

4.2.1 Une architecture basée sur quatre simulateurs

Nous avons choisi de scinder le système de simulation en quatre simulateurs spécialisés dans quatre domaines particuliers et en un ou plusieurs afficheurs 3D clients, le tout surveillé et géré (gestion uniquement pendant la phase de lancement) par un superviseur [55]. Dans un premier temps, nous allons nous focaliser sur le cœur du système à savoir les quatre simulateurs qui sont représentés sur la figure 4.1.

Nous y retrouvons un simulateur de véhicules en charge de calculer l'évolution de la dynamique des robots, un simulateur de capteurs calculant la valeur des sorties capteurs des engins, un simulateur de moyens de communication reproduisant le comportement logico-temporel du matériel de communication, et enfin un simulateur d'environnement calculant la propagation des ondes (éléctro-magnétiques et acoustiques), les collisions, et les phénomènes environnementaux. De façon basique, on peut dire que les véhicules sont connectés aux simulateurs et échangent des données avec ces derniers.

Un cycle de simulation se déroule de la manière suivante. Un contrôleur de véhicule envoie les commandes actionneurs au simulateur de véhicules. Celui-ci calcule l'évolution des robots à l'instant $t+1$ et envoie le vecteur d'état de chacun de ces robots au simulateur d'environnement. Celui-ci calcule alors les collisions éventuelles véhicule/véhicule et véhicule/environement, puis détermine les vecteurs forces environnementales pour chaque véhicule. Il renvoie alors les nouveaux vecteurs d'état et les vecteurs forces environnementales au simulateur de véhicules qui les utilisera pour calculer les états futurs, au cycle suivant. Il envoie ensuite ces informations au simulateur de capteurs qui est alors en mesure de calculer la réponse des capteurs proprioceptifs. La réponse des capteurs extéroceptifs, quant à elle, est déterminée grâce à ces informations et aux données envoyées par le simulateur d'environnement (nous reviendrons en détail sur ce mécanisme plus tard).

Ces réponses sont alors envoyées aux véhicules concernés à un instant précis correspondant au taux d'échantillonnage du capteur réel. Lorsque un véhicule souhaite émettre

un message, il l'envoie au simulateur des moyens de communication. Ce dernier formate alors le message comme le capteur réel et le transmet au simulateur d'environnement sous deux formes : le contenu du message en clair, et le contenu du message sous forme de signal analogique numérisé qui correspondrait au signal émis au niveau de l'antenne du périphérique. Le simulateur d'environnement calcule quand et qui reçoit le message et délivre alors le message en clair et en représentation numérique au simulateur de moyens de communication qui les renvoie à son tour aux véhicules concernés. Tous ces mécanismes seront abordés en détail dans les sections suivantes. On voit donc clairement que deux cycles coexistent. Le premier que nous avons évoqué est un cycle périodique prévu pour tourner une dizaine de fois entre chaque commande envoyée par un contrôleur de véhicule. En revanche, le second est un cycle purement événementiel.

Nous avons choisi une approche fonctionnelle pour ce simulateur, là où les autres concepteurs ont privilégié une approche par composant. Ce choix se justifie pour plusieurs raisons. Ce regroupement par fonction revient à scinder la modélisation non plus par composant (c'est-à-dire par véhicules par exemple), mais par spécialité (par exemple capteurs ou dynamique des véhicules). Cela offre plusieurs avantages, particulièrement au regard de la pluridisciplinarité des intervenants potentiels :

- Il est possible de voir un véhicule autonome comme un assemblage de capteurs, d'actionneurs, et de moyens de communication, possédant une certaine dynamique. Créer un simulateur revient donc à modéliser un ou plusieurs de ces aspects puis à animer ces modèles. Pour autant ces modèles sont différents par nature, et les spécialistes chargés de les créer appartiennent à des communautés différentes. Ainsi modéliser un capteur nécessitera plutôt des connaissances en traitement du signal alors que le modèle dynamique d'un véhicule sera plutôt établi par une hydrodynamicien ou un automaticien ; il en va de même pour la modélisation de l'environnement et des moyens de communication. Dès lors, il nous a semblé logique de regrouper les éléments (capteurs, actionneurs, ...) constituant un composant(véhicule, ...) non plus par composant (véhicule, ...), mais plutôt par la fonction qu'ils occupent en son sein. Cela a pour conséquence directe de pouvoir créer un simulateur adapté aux fonctionnalités d'une même famille d'éléments. En effet, un simulateur de capteurs, n'est pas conçu de la même façon qu'un simulateur de véhicules par exemple : les informations consommées et produites sont différentes, les dates et les événements auxquels ils le sont également, la présence ou l'absence d'étape d'intégration,... sont autant de points qui mènent à une conception différente.
  Un composant (un véhicule par exemple) n'est donc plus exécuté sur une seule unité de calcul, mais est réparti sur une grille de calcul. Cela permet aux spécialistes appartenant à une même communauté de se focaliser sur le fonctionnement et l'adaptation d'un seul simulateur sans remettre en question l'ensemble de ce système.
- Il est possible d'adapter la période de la boucle de simulation pour chaque type de simulateur : cette période est différente pour le simulateur de véhicules (environ dix fois la fréquence des commandes aux actionneurs), pour le simulateur de capteurs (à la fois événementiel pour les capteurs acoustiques commandés (type sonar), et périodique mais à des fréquences différentes eu égard au taux d'échantillonnage du dit capteur), pour le simulateur de moyens de communication (purement événementiel)... Ce n'est donc pas l'élément nécessitant la période de simulation la plus basse, qui impose sa fréquence aux autres éléments. Il n'existe pas non plus de "couplages"

artificiels (du type, "la période de simulation de tel capteur vaut dix fois celle du modèle dynamique de tel robot... Il faut donc attendre dix cycles avant de faire évoluer le robot") entre les éléments.
- Du point de vue développement, il est plus facile de maintenir à jour le code s'il n'est pas réparti sur plusieurs ordinateurs. En effet, il n'est pas rare que plusieurs véhicules utilisent le même modèle de capteur. Dans l'approche classique, le code correspondant au modèle du capteur est dupliqué sur chaque simulateur de véhicules. Notre approche nous permet d'utiliser la même section de code pour plusieurs engins. Cette section n'est présente qu'à un seul endroit et sa maintenance s'en trouve facilitée.

Ce système repose sur une approche client/serveur pour chacun des simulateurs. Pour garantir le déroulement du cycle de simulation sans interruption, nous avons développé un module élémentaire que nous présentons dans la section suivante.

4.2.2 Module élémentaire

Tous les échanges de données mentionnés dans le paragraphe précédent présentent plusieurs inconvénients pour le système de simulation :
- A tout instant un nombre important de messages sont échangés sur le réseau. En effet, la période d'intégration des simulateurs, le nombre de véhicules connectés et le nombre de capteurs de chaque véhicule sont autant d'éléments qui participent à l'augmentation du nombre de messages échangés. Nous avons ainsi estimé qu'une simulation mettant en jeu trois véhicules possédant un nombre très restreint de capteurs générait en moyenne 57600 trames/minutes (résultat obtenu à partir d'un comptage des trames durant une minute de simulation). Par ailleurs, le volume de données échangées peut-être conséquent si des capteurs d'imagerie sont utilisés.
- L'encodage et le décodage de toutes ces trames prennent du temps. Même si le traitement n'est pas complexe et ne nécessite que peu de temps processeur, il n'en demeure pas moins que le nombre de messages conséquent rend non négligeable le temps de traitement.
- L'échange de données sur un réseau s'accompagne de différents mécanismes bloquants, soit en réception (UDP), soit en émission et en réception (TCP). Cela a pour conséquence de bloquer la boucle de simulation et de ne plus assurer le découplage temporel de l'environnement, des véhicules et du contrôle.
- Enfin l'échange d'informations entre les simulateurs s'accompagnent de traitement de la donnée et de la mise en place de mécanismes d'envoi et de réception. Cet aspect n'est donc pas commode pour les personnes chargées d'implémenter les différents modèles.

Pour pallier ces inconvénients, nous avons imaginé un module élémentaire simple sur lequel reposent tous les simulateurs (voir figure 4.2).

On y distingue trois éléments principaux :
- Un premier processus appelé "processus de simulation". Il s'agit d'un des quatre simulateurs. Ce processus fait donc évoluer les modèles, puis envoie les informations calculées vers les autres simulateurs du système. Bien entendu nous utilisons des mécanismes non bloquants pour réaliser cet envoi. Pour faire évoluer les modèles, les informations requises sont lues dans une zone partagée.

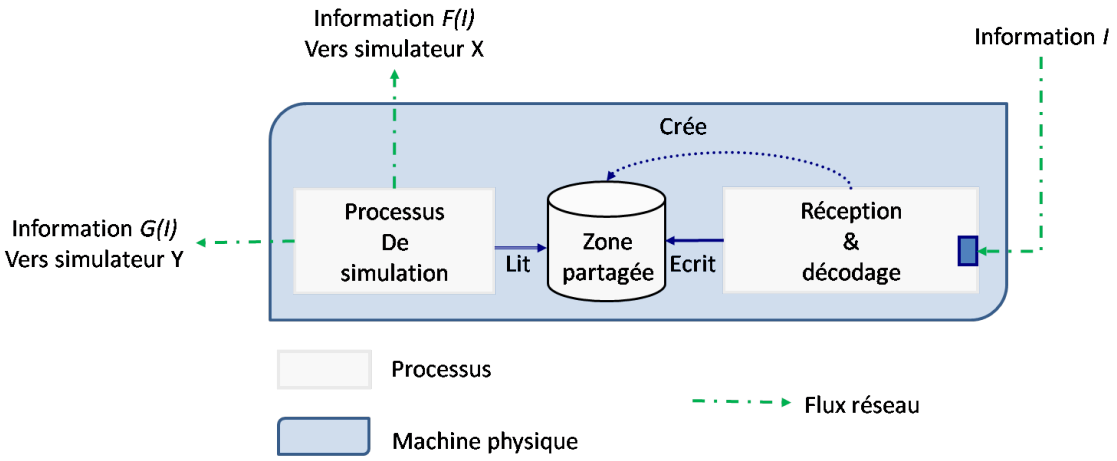


Fig. 4.2 – Module élémentaire de Thetis

- Un second processus bien distinct du premier et autonome, appelé "réception & décodage", réalise comme son nom l'indique deux fonctions. La première consiste à réceptionner les données envoyées par les autres composants du système (simulateurs ou véhicules notamment). Il est donc en attente de réception jusqu'à ce qu'il reçoive une trame. Une fois reçue, il réalise la deuxième fonction à savoir le décodage : les trames sont formatées et il est nécessaire de parser l'information contenue avant d'aller la copier dans une zone partagée.
- La zone partagée est uns mémoire dans laquelle est stockée l'information de manière à ce qu'elle puisse être directement accessible au premier processus. Ce dernier peut donc y retirer les informations mises à jour en permanence.

Il faut bien comprendre que ce sont deux processus autonomes. Ils peuvent être exécutés sur des processeurs différents (appartenant à un même nœud) après que le second processus ("réception et décodage") a créé la zone partagée et que le premier s'y est connecté. Pour éviter la décohérence des données lues dans la zone partagée (c'est-à-dire la lecture des données correspondant partiellement à l'instant t et à l'instant $t-1$), un mécanisme de synchronisation d'accès à la ressource entre les deux processus doit être mis en place.

Un simulateur est donc systématiquement connecté à une ou plusieurs zones partagées alimentées par des processus tiers chargés de la réception et du décodage des trames reçues. Cela remédie aux problèmes que nous avons évoqués précédemment : le décodage et la réception des données ne perturbent plus la simulation car un processeur leur est dédié (sur la même machine multi-cœurs).

Même si les volumes échangés sont conséquents, le fait de recourir à plusieurs processus de ce type permet de distribuer largement cette partie, davantage liée à la problématique de la répartition de l'information au sein d'un réseau qu'à la simulation elle-même. Par ailleurs, le cycle de simulation n'est jamais perturbé par une connexion bloquante : l'entité simulée continue à évoluer indépendamment du fait qu'une information tarde à lui parvenir. Ceci est capital pour la validation du comportement du contrôleur. Enfin l'échange des données sur le réseau est tout à fait transparent pour les utilisateurs du simulateur : ils peuvent développer leurs modèles et utiliser les variables calculées par les autres simulateurs, en lisant directement une zone partagée.

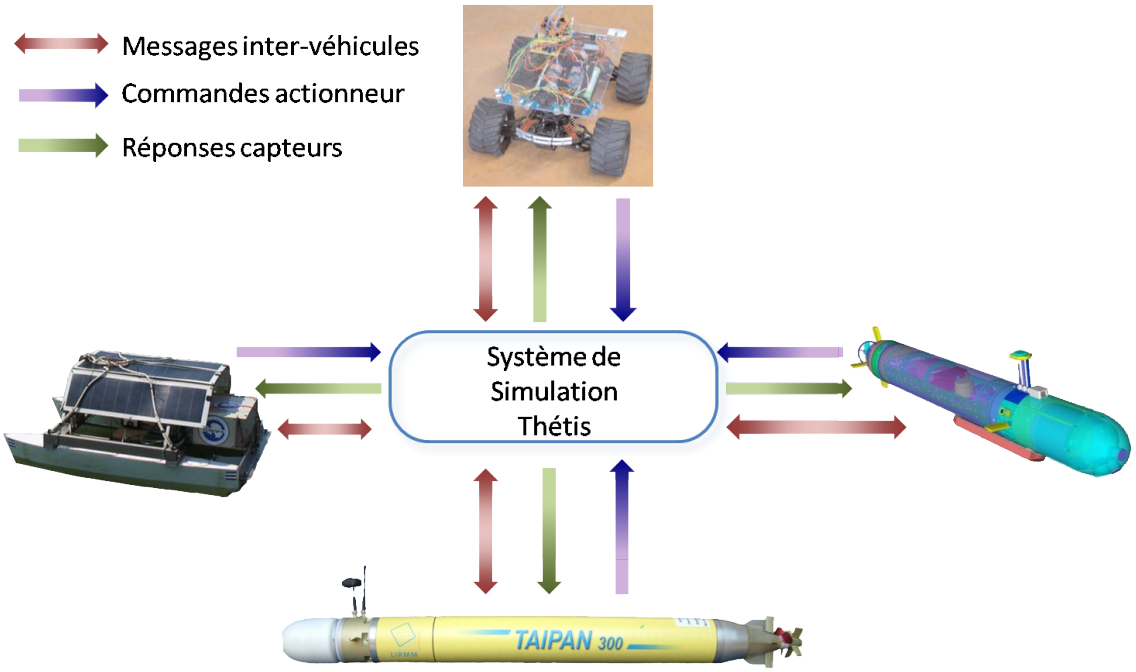


Fig. 4.3 – Connexion des véhicules au système Thetis

Il faut bien comprendre que chaque modèle n'a le droit d'exploiter que les variables qu'il est censé percevoir dans la réalité. Ainsi, il sera par exemple impossible à un modèle dynamique d'un engin d'obtenir les variables issues de son capteur de pression. Les seules données auxquelles il a accès sont les commandes envoyées par le véhicule qu'il est censé représenter. L'utilisateur n'a donc pas le choix des données à utiliser dans son modèle : elles lui sont imposées par la réalité. Ce choix empêche la création de règles virtuelles qui n'ont aucune existence réelle.

4.2.3 Connexion aux véhicules

Dans les précédents paragraphes nous avons présenté rapidement le cœur du système et le module élémentaire sur lequel il repose. Nous exposons maintenant la façon dont les véhicules sont reliés au système de simulation. Les robots échangent trois types de données avec le système de simulation : les commandes actionneurs, les réponses capteurs, et les messages envoyés/reçus vers/en provenance des autres véhicules (figure 4.3).

Il est à noter qu'il n'existe aucune connexion directe entre les robots. Tout passe nécessairement par le système de simulation. Cela représente bien la réalité car pour échanger un message, ces derniers passent obligatoirement par un moyen de communication, puis par l'environnement.

Sur la figure 4.4 on peut voir en détail les connexions avec les simulateurs et la nature des données échangées.

Pour connecter le contrôleur des robots au système, trois types de liens sont utilisés :

– une connexion purement événementielle utilisée pour la diffusion des messages entre les véhicules (et éventuellement vers les opérateurs). Lorsqu'un message doit être envoyé ce lien est utilisé. Le message est formaté de la même façon pour une simulation et pour une expérimentation réelle. La seule différence réside dans le fait qu'il

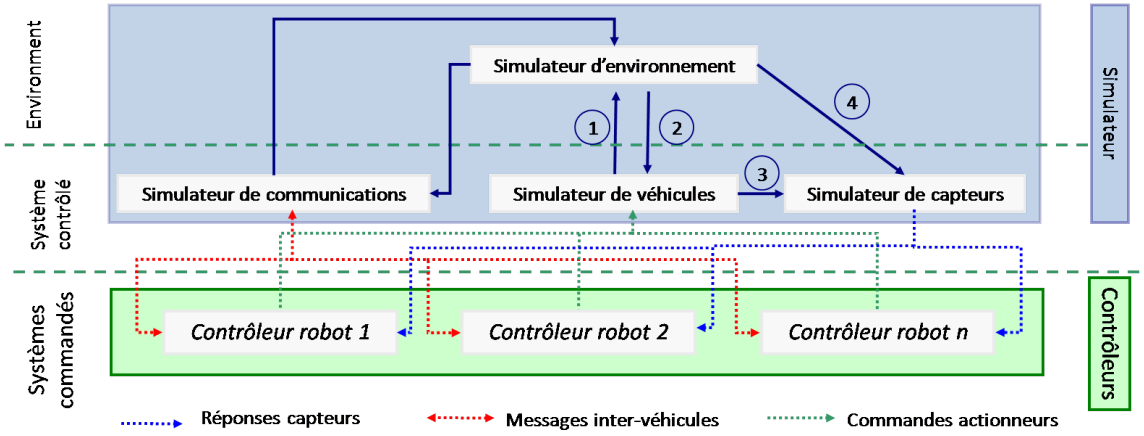


Fig. 4.4 – Détail des connexions au système Thétis

est envoyé vers le simulateur plutôt que vers le moyen de communication réel. Ce lien est bidirectionnel puisque les messages reçus par un véhicule suivent le même chemin (à l'aller et au retour).

- Une connexion sur laquelle sont envoyées des trames périodiquement : il s'agit des commandes envoyées aux actionneurs. Une fois encore l'information envoyée vers le système de simulation est identique à celle envoyée à la carte de commande de l'étage de puissance des actionneurs réels.
- Enfin la troisième connexion transporte des messages multi-périodiques ou événementiels. En effet, chaque capteur possède un taux d'échantillonnage qui lui est propre et certains capteurs, tel que le sonar, sont des capteurs commandables : un ordre peut leur être donné pour faire une capture, et un délai variable est souvent nécessaire (capteur acoustique) avant d'obtenir une réponse. Durant la simulation, le contrôleur du véhicule dispose donc de données capteurs rafraîchies à la même cadence que durant une expérimentation réelle.

Il est à noter que comme dans le cas réel, les informations échangées le sont sans attente de confirmation de réception de la part du simulateur ou du contrôleur du véhicule. Cela permet de ne pas perturber le comportement temporel de l'architecture.

Connecter les véhicules de cette façon nous permet donc de lancer les simulations comme s'il s'agissait de misions réelles. La seule différence se situe en fait au niveau des drivers des éléments constitutifs du véhicules (capteurs, actionneurs, moyens de communication) : les données sont redirigées vers le simulateur plutôt que vers l'organe concerné. Elles sont envoyées sans attente de confirmation de réception, et au moment opportun (similaire à la réalité) ; le contrôleur et le système de simulation évoluent chacun de leur côté de façon autonome. Il y a bien un découplage temporel entre le robot et le système de simulation. Un changement dans le taux d'échantillonnage d'un capteur (capteur de pression par exemple) aura donc une conséquence sur le comportement de l'engin. De même, le comportement divergent du véhicule, lorsque son contrôleur tarde à envoyer une commande aux actionneurs, peut être observé.

4.3 Les simulateurs

Dans cette section nous présentons en détail le fonctionnement du cœur du système, à savoir les quatres simulateurs, piliers de la simulation. Nous abordons en particulier le rôle de chacun d'eux, décrivons les processus qui les composent et détaillons les liens établis entre ces simulateurs et le reste des composants de la simulation.

4.3.1 Le simulateur de véhicules

Le simulateur de véhicules a en charge le calcul de l'évolution de la dynamique de l'ensemble des robots connectés au système de simulation. Cette évolution est calculée à partir des commandes envoyées par les robots et les perturbations environnementales locales, au travers de modèles pouvant être propres à chaque véhicule. En effet, deux AUVs n'auront pas nécessairement le même modèle et d'autres modèles pour d'autres types de véhicules (UGV, ASV,...) peuvent être utilisés simultanément.

La fréquence d'intégration pour ce simulateur est d'environ dix fois la fréquence d'échantillonnage des commandes actionneurs. L'évolution des modèles se calcule de façon séquentielle (dans l'ordre avec lequel ils sont décrits dans le fichier de configuration) et l'ensemble des vecteurs d'état de ces engins est envoyé en une seule fois au simulateur d'environnement qui est chargé de détecter les collisions éventuelles.

Dans cette section nous présentons figure 4.5 la structure interne du simulateur de véhicules. Ce simulateur est composé de trois processus distincts et de deux zones de données partagées. Un premier processus se charge de réceptionner, décoder et partager les commandes d'un actionneur d'un véhicule. Un second processus opère de même avec les vecteurs de forces environnementales calculées localement et exercées sur un véhicule en particulier. Les zones de données partagées sont donc constamment mises à jour pour l'ensemble des véhicules. Le troisième processus, qui est le processus de simulation à proprement parler, accède aux deux zones de données partagées afin de calculer l'évolution des modèles des véhicules connectés.

Après une étape de vérification des collisions (faites par le simulateur d'environnement), les vecteurs d'état de tous les véhicules à l'instant $t + 1$ sont envoyés sans accusé de réception au simulateur de capteurs.

4.3.2 Le simulateur de capteurs

Le simulateur de capteurs est chargé de calculer la réponse de l'ensemble des capteurs de tous les véhicules connectés au simulateur. Il s'agit aussi bien des capteurs proprioceptifs (type centrale inertielle, capteur de pression, ...) que des capteurs extéroceptifs (type sonar, CTD, appareil photographique ou caméra). Ce simulateur permet également à plusieurs modèles différents de coexister ; les comportements temporels (*eg* taux d'échantillonnage, délais propres au capteur, ...) et logiques (demande d'acquisition, ...) sont pris en compte.

La période de la boucle de simulation vaut environ le dixième du taux d'échantillonnage le plus élevé de l'ensemble des capteurs. Ce choix permet de garantir moins de 10% d'erreur sur la fréquence d'échantillonnage propre d'un capteur. La réponse d'un capteur est élaborée après chaque envoi pour les capteurs autonomes, et sur commande pour les autres.

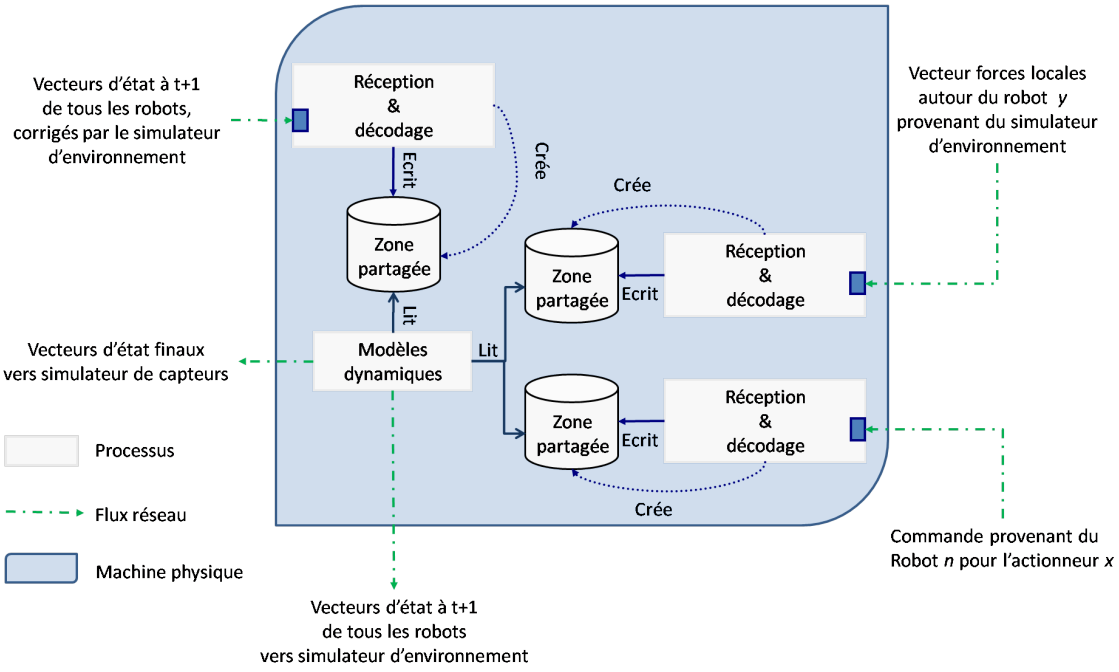


Fig. 4.5 – Structure interne du simulateur de véhicules

La figure 4.6 présente la structure interne du simulateur de capteurs. Ce simulateur est composé de quatre processus distincts et de trois zones de données partagées.

Comme précédemment, trois processus sont chargés de réceptionner, décoder et partager les données en provenance des autres entités participant à la simulation. Parmi ces entités, on retrouve les véhicules, qui ont la possibilité d'envoyer des commandes d'acquisition pour certains capteurs commandés (sonar, vidéo, ...), le simulateur de véhicules qui envoie tous les vecteurs d'état de tous les véhicules et enfin le simulateur d'environnement qui propage les données environnementales locales et adaptées pour les véhicules concernés (c'est-à-dire ceux qui disposent de capteurs extéroceptifs).

Les réponses des capteurs proprioceptifs sont élaborées à partir des vecteurs d'état des robots envoyés par le simulateur de véhicules alors que les réponses des extéroceptifs sont élaborées grâce aux données environnementales, sur requête véhicule ou de façon périodique. L'ensemble des informations nécessaires pour élaborer ces réponses se trouvent dans les zones de données partagées rattachées au processus concerné ; elles sont en permanence mises à jour, et c'est au processus de simulation d'élaborer les réponses en tenant compte du taux de rafraîchissement propre à chaque capteur. Elles sont alors envoyées au moment adéquat et nominativement (c'est-à-dire une réponse d'un capteur en particulier à un véhicule unique) sans accusé de réception.

Il existe un cas particulier que nous n'avons pas détaillé ici : la simulation d'un capteur d'images. Qu'il s'agisse d'une caméra ou d'un appareil photo, ce type de périphérique étant un fort consommateur de bande passante, nous envisageons de lui dédier du matériel (réseaux dédiés entre autre) pour garder les propriétés "real-time like" du système.

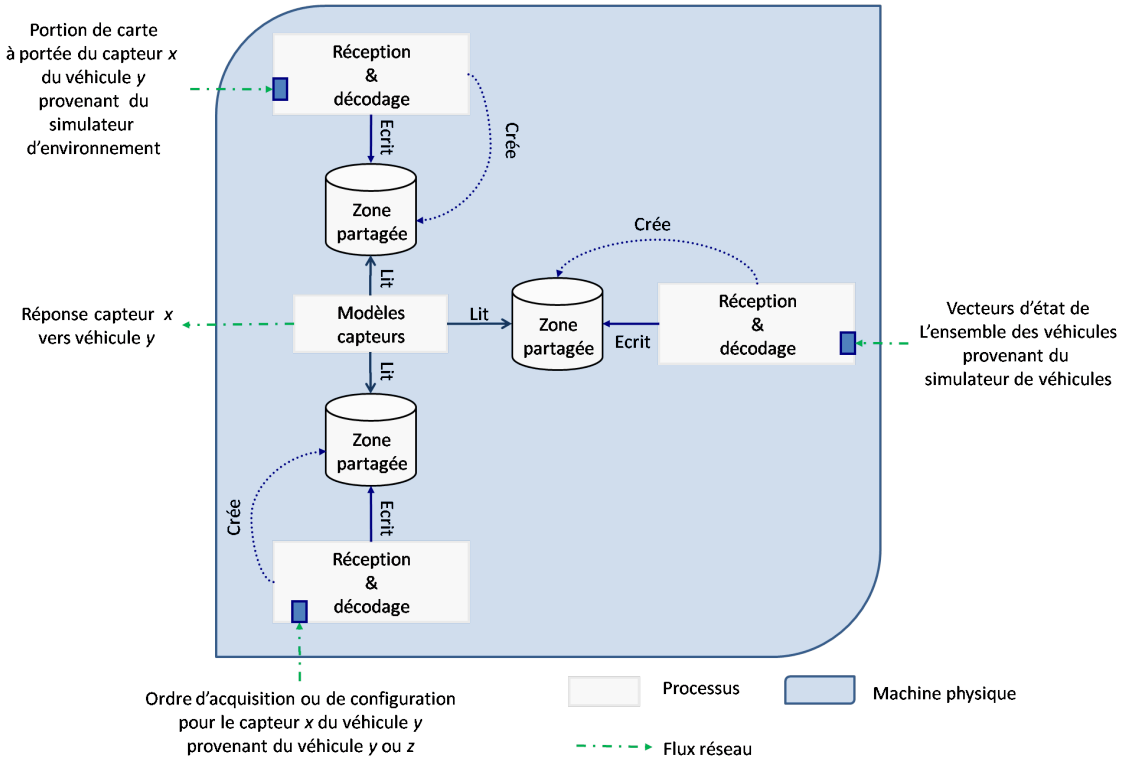


Fig. 4.6 – Structure interne du simulateur de capteurs

4.3.3 Le simulateur de moyens de communication

Le simulateur de moyens de communication est chargé de reproduire le comportement logico-temporel des différents périphériques de communication présents à bord des robots, et de calculer la forme analogique du message avant émission, au niveau de l'antenne. Il est à noter que le message est transmis au simulateur d'environnement en clair (c'est-à-dire une chaîne de caractères) et sous forme de représentation numérique du message analogique (respect du critère de Shannon). Le comportement logique du moyen de communication désigne son changement d'état (mode réception, mode émission, en sommeil...) en fonction des stimuli externes (réception d'une porteuse, buffer d'émission plein, commande utilisateur,...) et interne (délai avant mise en veille écoulé, rapport signal sur bruit dépassant un certain seuil,...). Le comportement temporel correspond aux différents délais mis par le moyen de communication pour passer d'un état à un autre.

La figure 4.7 présente la structure interne du simulateur de moyens de communication. Ce simulateur est composé de trois processus distincts et de deux zones de données partagées.

Comme précédemment, deux processus sont chargés de réceptionner, décoder et partager les données en provenance des autres simulateurs. Le simulateur de moyens de communication reçoit les messages en provenance des robots participant à la simulation. Chacun d'entre eux peut émettre un message à diffuser dans le milieu à n'importe quel instant. Le robot peut donc envoyer un message formaté comme dans la réalité : la seule différence est que ce message est dirigé vers le simulateur de moyens de communication à la place du périphérique de communication réel. Ce formatage identique permet de simuler les différentes couches d'un modèle de communication (typiquement couche physique,

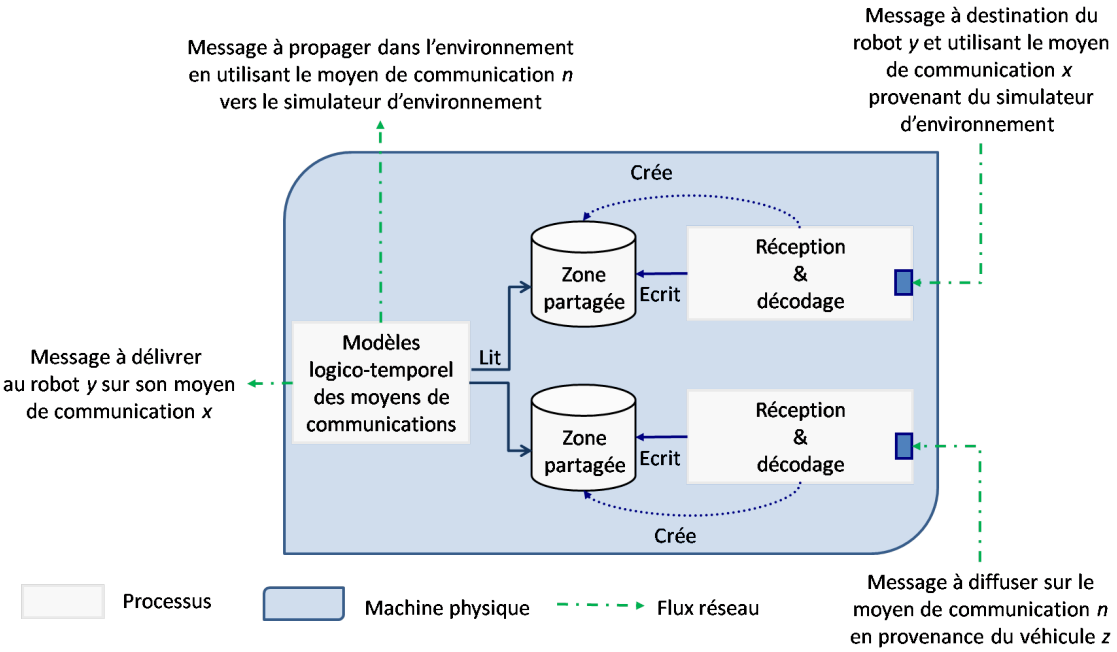


Fig. 4.7 – Structure interne du simulateur de moyens de communication

liaison de données, application).

Le message est ensuite traité par le simulateur qui, après avoir reproduit le comportement du périphérique de communication réel (mise en file d'attente, réveil, ...), l'envoie en clair et sous forme numérisée vers le simulateur d'environnement. Le message revient éventuellement, et au bout d'un certain temps, au simulateur de moyens de communication. En outre, ce message a été étiqueté afin de préciser quels sont les moyens de communication et les véhicules susceptibles de le recevoir. Un traitement inverse (décodage du message numérique, toujours avec prise en compte du comportement logico-temporel du périphérique traitant) est effectué par le modèle du moyen de communication concerné, puis envoyé au véhicule concerné.

Il est à noter que bien qu'implémentée, nous n'utilisons pas actuellement la forme numérique du message analogique. En revanche, nous utilisons le message en clair, ce qui constitute un premier pas suffisant au vu de notre problématique. La possibilité d'utiliser la représentation numérique s'adresse davantage à la communauté de scientifiques travaillant sur le traitement du signal des ondes reçues/envoyées par des moyens de communication (modem acoustique, wifi, UHF, ...).

4.3.4 Le simulateur d'environnement

Le simulateur d'environnement est chargé de plusieurs tâches :

- Calculer l'évolution des modèles environnementaux. Il s'agit principalement des vagues, des courants, du vent, mais aussi de la salinité de l'eau, de sa température, ... Le fond de l'océan, une éventuelle calotte glacière et le sol peuvent également être représentés dans ce simulateur.
- Calculer les collisions se produisant entre les véhicules ou entre un véhicule et l'environnement (fond, objet,...)

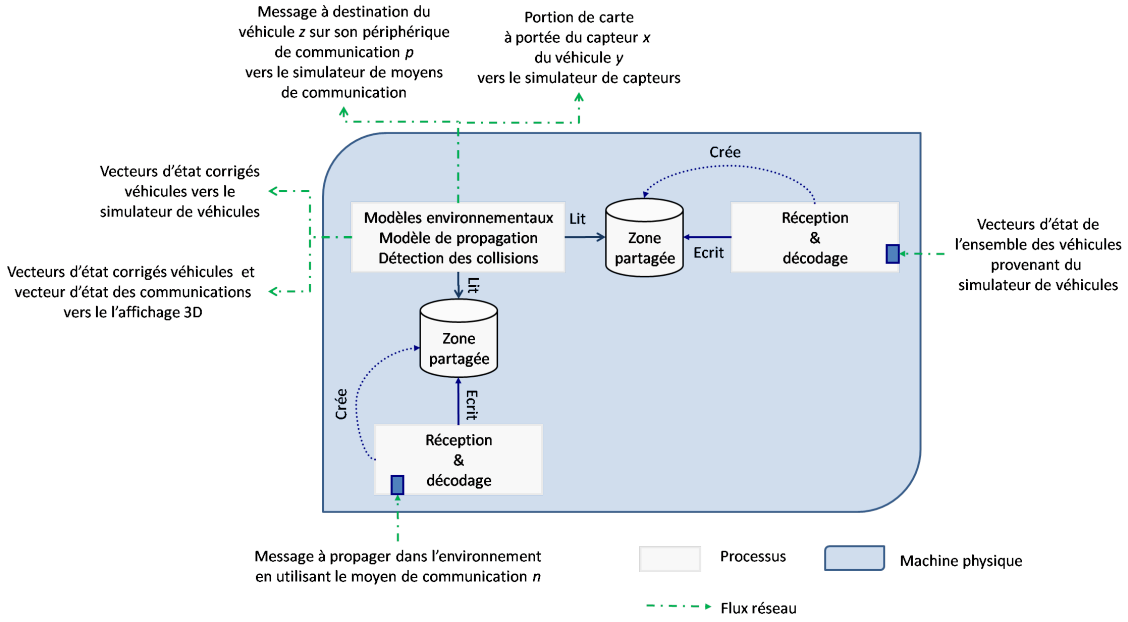


Fig. 4.8 – Structure interne du simulateur d'environnement

- Calculer la propagation des ondes électro-magnétiques ou acoustiques. Ce calcul permet de déterminer quels récepteurs perçoivent quoi et où.
- Partager les informations nécessaires avec les autres entités participant à la simulation. Il s'agit principalement du simulateur de capteurs, du simulateur de véhicules et de l'affichage 3D.

Il est à noter qu'aucun robot n'est directement connecté à ce quatrième simulateur. Seuls les trois autres simulateurs, l'affichage 3D et l'utilisateur peuvent échanger des informations avec le simulateur d'environnement.
La figure 4.8 présente la structure interne du simulateur d'environnement. Ce simulateur est également composé de trois processus distincts et de deux zones de données partagées.

Comme précédemment, deux processus sont chargés de réceptionner, décoder et partager les données en provenance des autres composants de la simulation. Le simulateur d'environnement reçoit les vecteurs d'état de l'ensemble des véhicules. Ces vecteurs d'état lui permettent de vérifier l'absence de collisions véhicule/véhicule ou véhicule/environnement. Si une telle collision se produit, il calcule alors les nouveaux vecteurs d'état des robots entrés en collision. Il est à noter ici, que nous ne considérons pas pour l'instant les interactions avec l'environnent. Les chocs n'ont donc d'effet que sur les véhicules.

Une fois cette vérification faite, il renvoie au simulateur de véhicules les nouveaux vecteurs d'état et les vecteurs forces environnementales pour chacun de ces véhicules. Cela permettra au simulateur de véhicule de calculer les effets du vent, courants et vagues sur les robots. Connaissant le vecteur d'état de chaque robot et les capteurs embarqués pour chacun d'entre eux, le simulateur d'environnement est en mesure d'envoyer une "portion" de carte environnementale de tous les capteurs extéroceptifs de tous les véhicules, au simulateur de capteurs. Ces données permettront à ce dernier d'élaborer la réponse de chacun de ces capteurs.

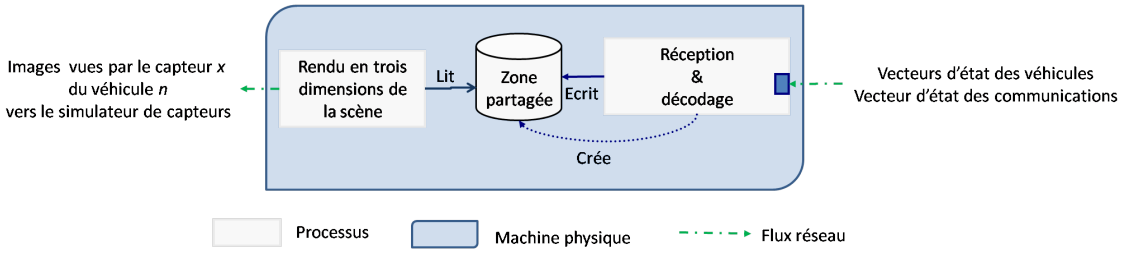


Fig. 4.9 – Structure interne de l'afficheur 3D

Enfin lorsque le simulateur d'environnement reçoit de la part du simulateur de moyens de communication un message à diffuser, celui-ci calcule sa propagation, déterminant ainsi quels sont les véhicules susceptibles de le recevoir et quand. Il est également possible de calculer durant cette étape quels sont les "déformations" du message propagé (interférences inter-symbole par exemple). Le simulateur d'environnement renvoie donc les messages à diffuser à un véhicule en particulier au simulateur de moyens de communication, qui sera alors en mesure de le délivrer au bon véhicule après avoir reproduit le comportement du périphérique de communication utilisé.

Enfin le simulateur d'environnement envoie à l'affichage 3D le vecteur d'état des véhicules et l'état des communications (quels sont les liens de communication actifs, quels sont les messages en cours de propagation, quel est l'état des moyens de communication des véhicules,...). Ces informations permettront de calculer un rendu 3D de la scène.

4.4 L'afficheur 3D

L'afficheur 3D est un programme client qui permet de visualiser la scène de simulation. Bâti sur un modèle similaire à celui des simulateurs, il est également en charge de calculer la réponse des capteurs de vision éventuellement embarqués par les robots. Il est à noter que plusieurs afficheurs peuvent être utilisés simultanément si nécessaire. En effet, il peut s'avérer utile de pouvoir suivre visuellement l'évolution de plusieurs véhicules "de près" à la fois.

La figure 4.9 présente la structure interne de l'afficheur 3D. Cette structure simple met en œuvre deux processus et une zone de données partagées. Cette structure a été adoptée pour simplifier la création de l'afficheur 3D. En effet, il est possible de développer des afficheurs différents en se focalisant sur le développement du rendu 3D ; les programmeurs n'ont ainsi pas à gérer l'échange de données avec le système de simulation puisqu'ils n'ont qu'une zone de données partagées à lire. Nous avons réellement utilisé cette possibilité puisque le développement du rendu de la scène a été produit par Sébastien Druon (Maître de Conférence à l'Université Montpellier 2) qui ne connait pas l'architecture du système.

Nous retrouvons sur le schéma un premier processus chargé de réceptionner, décoder et partager les données en provenance du simulateur d'environnement. Ce dernier envoie les vecteurs d'état des véhicules ce qui permet de calculer le rendu de la scène (position et attitude des véhicules). Par ailleurs, l'afficheur 3D reçoit également du simulateur d'environnement le vecteur d'état des communications. Cela permet d'afficher les communications qui ont lieu entre les véhicules en temps-réel.

4.5 Le superviseur de simulation

Démarrer ce système de simulation est relativement complexe. En effet, le système est largement distribué sur différentes machines et/ou différents processeurs. Par conséquent l'exécution n'est pas "classique" dans la mesure où les différentes étapes ne peuvent s'enchaîner linéairement comme pour un programme non distribué ou multi-threadé (c'est-à-dire un ensemble de processus, dits légers dans la mesure où le passage de l'un à l'autre n'implique pas de changement de contexte, se partageant un même espace mémoire).

Un programme externe doit donc veiller au déroulement de la séquence de lancement sous peine de voir le système diverger dès les premiers instants. Pour donner un exemple, on peut supposer que le simulateur d'environnement soit le premier à être exécuté. Les lectures dans la zone de données partagées en provenance du simulateur de véhicules fourniraient des indications erronées, au moins durant les premiers cycles, puisque ce dernier n'aurait pas encore eu le temps d'y envoyer des données. En revanche, les premiers vecteurs d'état des véhicules basés sur ces données erronées parviendraient au simulateur de véhicules au début de son exécution. L'état du système à l'instant initial pourrait donc être faux, et rendrait le reste de la simulation inexploitable.

Cet exemple montre clairement qu'il est nécessaire de lancer le simulateur de véhicules avant le simulateur d'environnement, quitte à ce que les premiers instants de simulation ne tiennent pas compte des perturbations environnementales. Il est donc nécessaire de respecter une séquence de lancement pour lancer une simulation et nous avons développé un programme dans ce but. Son rôle est donc de gérer cette séquence de lancement, puis de surveiller l'état du système en faisant parvenir à l'utilisateur un certain nombre d'informations produites par les simulateurs : temps de simulation restant, confirmations de franchissement d'étapes, warnings de décohérence temporelle, ...

Ce superviseur respecte des étapes pour lancer une simulation :

- La première étape consiste à charger la configuration du réseau, décrite dans un fichier par l'utilisateur. Il s'agit d'une étape permettant de déterminer l'emplacement réseau des différentes entités et leur configuration (accès, port de communication,...). Une fois cette étape franchie, le superviseur est en mesure de dialoguer avec tous les composants.
- La seconde étape consiste à distribuer l'ensemble des fichiers sur le réseau ; les exécutables et les fichiers de configuration sont ainsi répartis nominativement (c'est-à-dire que l'ordinateur chargé de la simulation des véhicules ne reçoit que les exécutables et les fichiers de configuration nécessaires à cette simulation) sur le réseau.
- Une fois les fichiers reçus par l'ensemble des ordinateurs, le superviseur lance leur exécution à distance. Tous les processus se mettent alors en attente : ils attendent une trame particulière du superviseur.
- Une fois tous les programmes en attente, le superviseur "interroge" chaque processus sur le réseau pour s'assurer que tous sont en attente du top de synchronisation (servant uniquement à disposer d'une horloge commune pour dater les logs).
- Une fois la présence des processus testée, il reste à s'assurer que l'ensemble des véhicules devant participer au scenario sont connectés ; cela se fait par une simple requête ping.
- Le top synchro-temps est émis sur le réseau, et tous les simulateurs disposent ainsi d'une base de temps commune pour enregistrer les logs.

- Les simulateurs se lancent ensuite les uns après les autres en respectant une certaine séquence (d'abord tous les processus de réception et décodage) puis les processus de simulation (Véhicules, Capteurs, Environnement, Communication) qui peuvent alors se connecter aux zones de données partagées créées par les premiers processus)).
- Le superviseur réceptionne alors les messages d'état de l'ensemble du système de simulation et les affiche pour l'utilisateur.
- Une fois la simulation terminée, le superviseur se connecte à chaque ordinateur pour télécharger les fichiers logs puis stoppe la simulation en terminant les processus sur le réseau.

Ce superviseur est donc nécessaire pour gérer l'architecture, s'assurer du bon déroulement de la simulation, et obtenir les logs en fin de simulation.

4.6 Cohérence de l'architecture proposée

Nous avons évoqué dans les sections précédentes le fonctionnement de chacune des entités composant le système de simulation et explicité les liens qui les unissaient. Après avoir présenté une vue complète du système, nous menons une réflexion sur le choix du mode de connexion permettant aux composants d'échanger des données sur le réseau. Nous évoquons dans un troisième temps les limitations de l'architecture présentée, avant de monter en quoi elle respecte les spécifications évoquées au chapitre précédent.

4.6.1 Vue globale

Nous présentons sur la figure 4.10 le synoptique du système entier. Par commodité nous n'avons représenté la connexion du système qu'avec un seul et unique véhicule et nous n'avons pas explicitement représenté les liens existants entre le superviseur et tous les processus.

Dans sa version actuelle le système de simulation totalise donc 16 processus autonomes, connectés à 10 zones de données partagées et peut être réparti sur six ordinateurs différents, chacun d'entre eux comportant plusieurs processeurs. Un dernier ordinateur fait également partie du système de simulation mais n'a pas été représenté ici car il n'y participe pas en tant que tel : il s'agit du serveur de configuration et de compilation. Cet ordinateur permet de préparer des scenarii et de configurer des véhicules. Par ailleurs, linux/RTAI y est installé afin de tester en local les différents modules des architectures (contrôle et simulateur), avant de les distribuer sur le réseau. Nous reviendrons sur le rôle et les fonctionnalités de ce serveur dans la section 5.2.1. Il est à noter que le superviseur télécharge les dernières versions des exécutables, et l'ensemble des fichiers de configuration depuis cet emplacement.

Nous avons représenté sur la figure 4.11 les ordinateurs impliqués dans une simulation faisant intervenir trois véhicules de type AUV. Nous avons précisé l'OS et le nombre de processeurs mis en jeu des ordinateurs hébergeant le système de simulation. T300_1,2,3 sont les noms donnés aux trois AUVs utilisés.

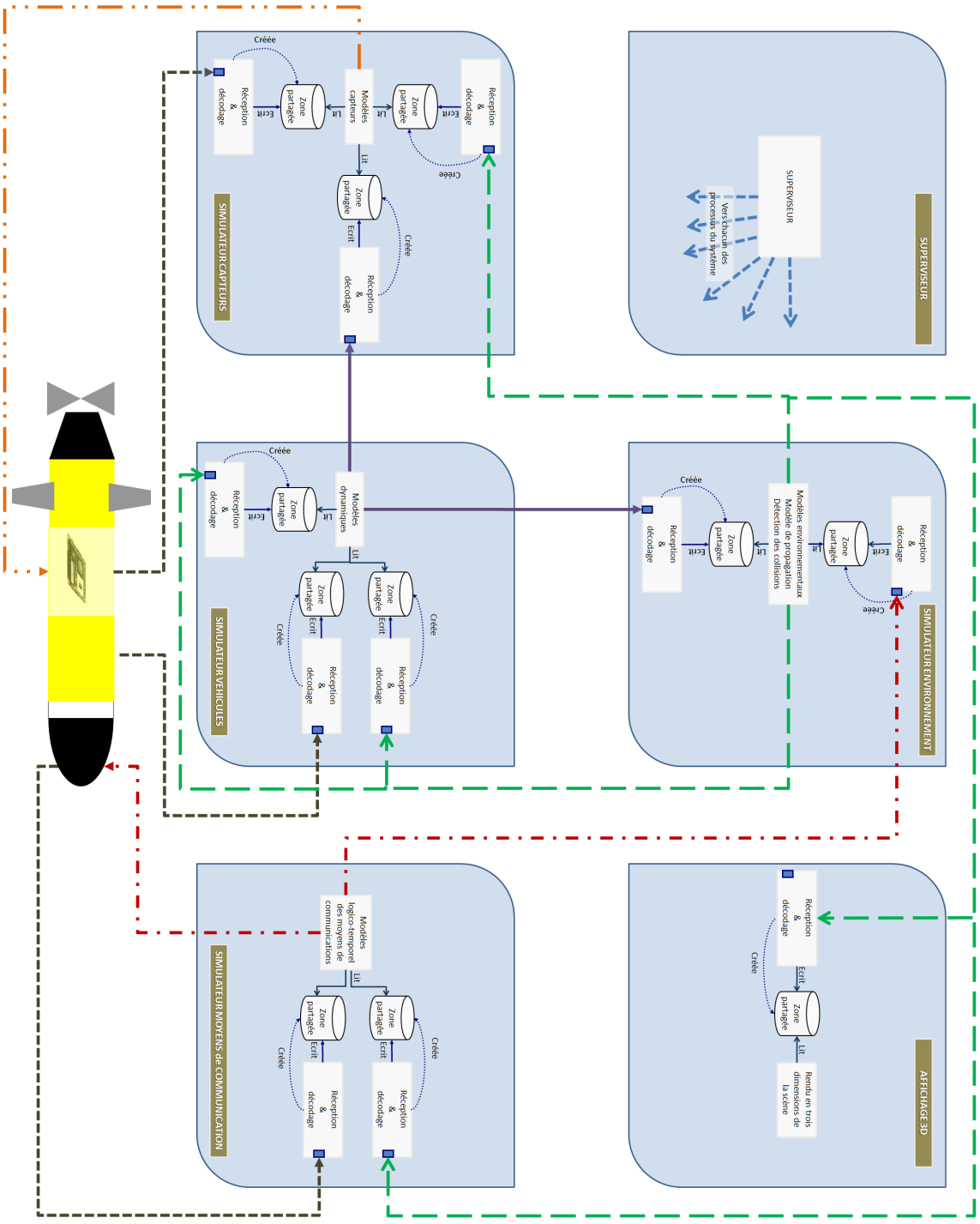


Fig. 4.10 – Synoptique de Thetis

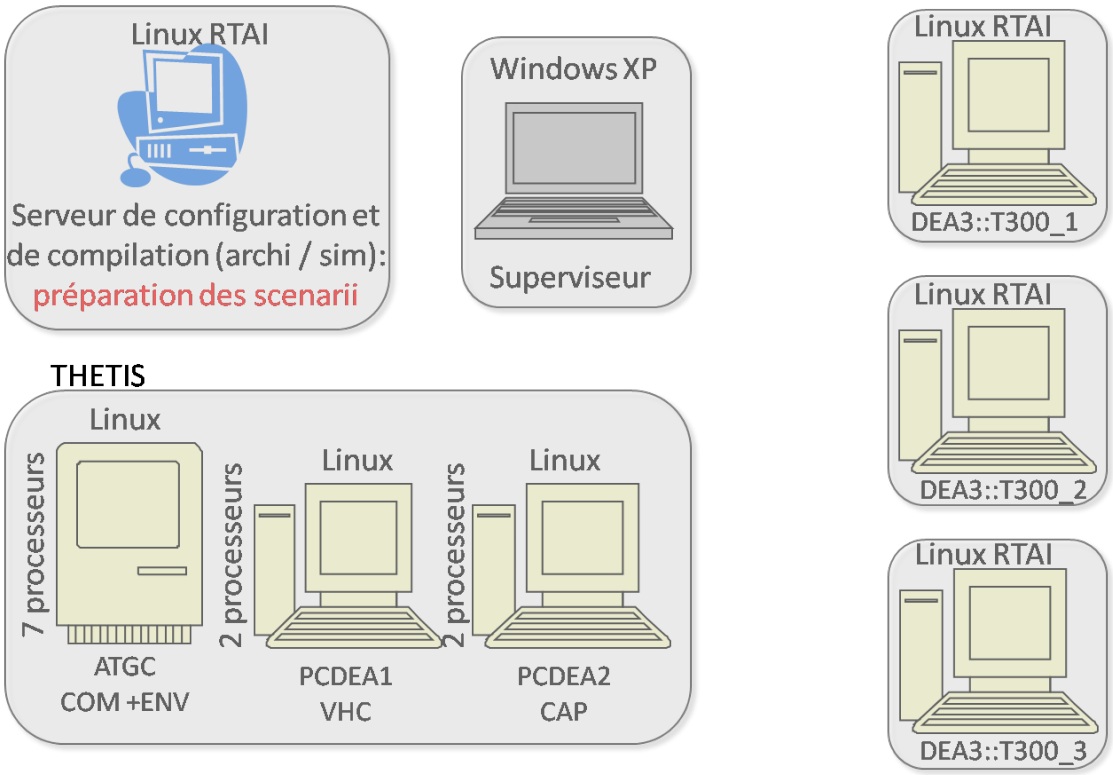


Fig. 4.11 – Un exemple de distribution du système

4.6.2 Choix du protocole

Distribuer l'ensemble de ces processus sur un réseau nécessite l'adoption d'un protocole de communication leur permettant d'échanger des données.

Sur un plan théorique, le choix d'un réseau déterministe s'impose naturellement. Cependant, le temps d'implémentation de ce genre de technologie était incompatible avec les contraintes de la thèse. Par conséquent nous avons choisi d'utiliser un réseau ethernet en minimisant l'impact de ce choix sur le déterminisme du réseau.

En effet, une première solution consiste à maîtriser les instants d'échange sur le réseau, autrement dit, faire appel à un superviseur qui ordonne à chaque entité de parler dans une fenêtre de temps déterminée. Cette approche implique qu'il est nécessaire de s'isoler des autres traffics non contrôlés, en particulier cela signifie qu'il n'est pas possible d'utiliser internet pour faire transiter des trames entre les constituants du simulateur.

La deuxième solution que nous avons retenue pour sa simplicité de mise en œuvre, offre cependant moins de garantie que la première. Nous avons utilisé un réseau ethernet micro-commuté : un seul ordinateur est connecté à une entrée d'un switch, ce qui limite fortement les risques de collisions et de latence sur le réseau.

Sur ce réseau, nous avons choisi d'utiliser le protocole UDP/IP, qui nous permet de faire transiter des paquets en mode non connecté, ce qui est bien le but recherché pour les raisons que nous avons précédemment évoquées.

4.6.3 Respect des spécifications

Dans cette section, nous vérifions que l'architecture proposée respecte bien les spécifications que nous avons précédemment établies. Un des premiers points que nous avons soulevé, était l'aspect collaboratif du simulateur. L'architecture proposée permet effectivement à plusieurs scientifiques appartenant à des domaines différents de travailler sur un aspect sans avoir à connaître le fonctionnement de l'ensemble pour autant. Les mécanismes mis en place permettent aux développeurs de s'affranchir des communications inter-processus et de la mise à jour des données ; ils peuvent se consacrer pleinement au développement et à l'implémentation de leur modèle. Un autre point important était la capacité de l'architecture à pouvoir simuler des robots différents ; l'architecture proposée ne se préoccupe pas du type de robot connecté : seuls les modèles qui seront intégrés permettront de simuler différents robots. Le système de simulation sépare donc clairement l'architecture à laquelle se connectent les robots, des modèles qui les simulent ; n'importe quel contrôleur de n'importe quel robot est donc potentiellement compatible avec cette architecture.

L'architecture proposée est clairement distribuée, ce qui correspond à un point critique que nous avions soulevé. En effet, le simulateur est composé d'une dizaine de processus autonomes, répartis et exécutés sur plusieurs machines différentes, connectées sur un réseau dédié. Cette approche nous permet donc clairement d'aborder le contexte du multi-véhicule en respectant les contraintes de temps réel en partie grâce à la répartition des traitements sur plusieurs unités de calcul.

Notre architecture supporte également les communications inter-véhicules et la pluralité des moyens de communication pour chaque robot. Notre proposition permet de faire le distinguo entre le comportement logico-temporel du moyen de communication et la propagation des ondes dans le medium ; ceci était également un point clé à respecter. Notre architecture prévoit de pouvoir connecter directement le contrôleur des robots au système ; nous avons donc bien créé un simulateur Hardware-in-Loop. Par ailleurs nous avons veillé à assurer le découplage temporel entre la commande et la boucle de simulation grâce à l'adoption du protocole UDP d'une part et à la mise en place d'un module élémentaire d'autre part. Ce module permet d'affranchir le processus de simulation des contraintes de communication réseau, et donc de garantir sa libre évolution. Cela nous permet de continuer à faire évoluer les véhicules sans considération du taux de rafraîchissement de la commande.

L'architecture proposée permet également de lancer une mission simulée, exactement de la même façon qu'une mission réelle ; seul le routage des données capteurs/ actionneurs/ communications vers ou en provenance de l'architecture différencie les deux types de mission. Cela n'a pas d'incidence sur le comportement temporel de l'architecture de contrôle. Enfin, notre architecture permet d'envisager la simulation hybride. En effet, même si elle est distribuée telle que nous l'avons présentée, il est possible d'exécuter l'ensemble de ces processus sur un seul et unique ordinateur. Il suffit pour cela de spécifier différemment les adresses réseau dans les fichiers de configuration des composants. Cela permet donc d'exécuter le simulateur à bord du robot, et de faire de la simulation hybride embarquée.

Pour finir, l'affichage 3D constitue un module de cette architecture et permet effectivement d'afficher un rendu en trois dimensions de la scène simulée. Même si cette fonctionnalité n'est pas encore implémentée, l'architecture est pensée pour cela et permettra facilement son ajout.

4.7 Conclusion et perspectives

Dans ce chapitre, nous avons présenté une nouvelle architecture de simulateur. Elle a été créée en veillant à respecter un certains nombres de contraintes que nous nous étions imposées au départ, pour aborder la problématique de la simulation Hardware-In-Loop multi-véhicule. Parmi ces contraintes, certaines ont été identifiées comme étant critiques pour notre application ; il s'agit principalement de la possibilité de distribuer le simulateur pour permettre de faire de la simulation temps-réel multi-véhicule sans avoir à dégrader les modèles utilisés pour diminuer la masse de calcul. Par ailleurs, les possibilités de communication entre les véhicules offertes par cette architecture permettent de préparer les missions simulées de la même façon que les missions réelles.

Pour ce faire, l'architecture est basée sur quatre simulateurs, un superviseur et un afficheur 3D. Tous ces éléments sont reliés ensembles par un réseau local dédié. L'ensemble de ces éléments représente 16 processus indépendants accédant à 10 zones de données partagées. Cette approche se distingue des autres par sa séparation fonctionnelle. En effet, là où les autres ont préféré décomposer leur système de simulation par composant, nous avons préféré cette approche ; celle-ci a de réels avantages au regard de la pluridisciplinarité des intervenants potentiels.

Nous avons constaté que l'architecture ainsi créée représente un système complexe, nécessitant un programme tiers pour lancer une simulation. En effet, une certaine séquence doit être respectée afin de ne pas faire diverger le système lors du lancement d'une simulation ; nous avons donc développé un superviseur capable de gérer cette séquence, de surveiller le déroulement de la simulation et de télécharger les logs en fin de simulation.

Si cette approche semble pleinement convenir dans un premier temps, elle ne respecte cependant pas un dernier objectif que nous nous étions fixé : la possibilité d'étendre dynamiquement la puissance de calcul du système de simulation. En effet, bien que fortement distribuée, l'architecture proposée ne permet pas à l'heure actuelle d'être divisée en plus de 16 processus. La prochaine étape est donc de pouvoir exécuter l'évolution d'un modèle (modèle de capteur, de véhicules, d'environnement, ...) particulier sur un processeur dédié. Il ne sera pas nécessaire d'apporter de grosses modifications à l'architecture existante, puisque nous pourrons nous servir à nouveau de notre module élémentaire pour déporter ces calculs. Un simulateur enverra donc à l'ensemble de ses clients les données nécessaires pour faire un pas de simulation et attendra pendant un temps compatible avec le temps de cycle, le retour de ses clients.

Une fois cette fonctionnalité implémentée, l'architecture sera alors massivement distribuée et permettra de paralléliser l'ensemble des calculs.

Chapitre 5

Réalisation du logiciel

5.1 Introduction

Après avoir proposé une architecture de simulateur, nous consacrons ce chapitre au passage de la théorie à la pratique. En effet, nous présentons dans les sections qui vont suivre l'implémentation de notre système de simulation et plus particulièrement nos choix technologiques. L'objectif principal de ce chapitre est de montrer en quoi ces choix corroborent les objectifs que nous nous sommes fixés au chapitre 3. En effet, décider d'avoir recours à une technologie à la place d'une autre peut remettre en cause les concepts de base étudiés dans la première partie. Le but est donc de veiller à la cohérence du simulateur au regard des objectifs annoncés, notamment : ouverture, modularité, temps réel, distributivité et découplage temporel.

Dans un premier temps nous précisons les aspects liés à notre environnement de travail, à savoir les systèmes d'exploitation retenus, le matériel physique mis à notre disposition, et l'organisation de l'environnement de développement. Nous abordons ensuite certains de nos choix technologiques qui nous ont permis de modéliser (au sens informatique du terme) puis de développer notre simulateur. Nous exposons dans un troisième temps les mécanismes informatiques liés à l'implémentation de notre module de base (cf 4.2.2) avec notamment une discussion autour de l'utilisation des sockets. Enfin, une quatrième partie est consacrée à la description des composants mis en jeu dans une simulation. Nous y proposons un formalisme permettant de décrire un robot autonome et présentons l'utilisation du XML (Extensible Markup Language) comme un moyen de représenter les données au sein des fichiers de configuration et de résultats de simulation.

5.2 Environnement de travail

Dans cette section nous présentons notre environnement de travail au sens large. Il s'agit de présenter les plate-formes impliquées dans le développement, en précisant nos choix quant au système d'exploitation mis en œuvre. Nous présentons le matériel que nous utilisons d'une part pour lancer nos simulations, et, d'autre part, pour développer le système tout en précisant les solutions retenues pour permettre le travail en commun.

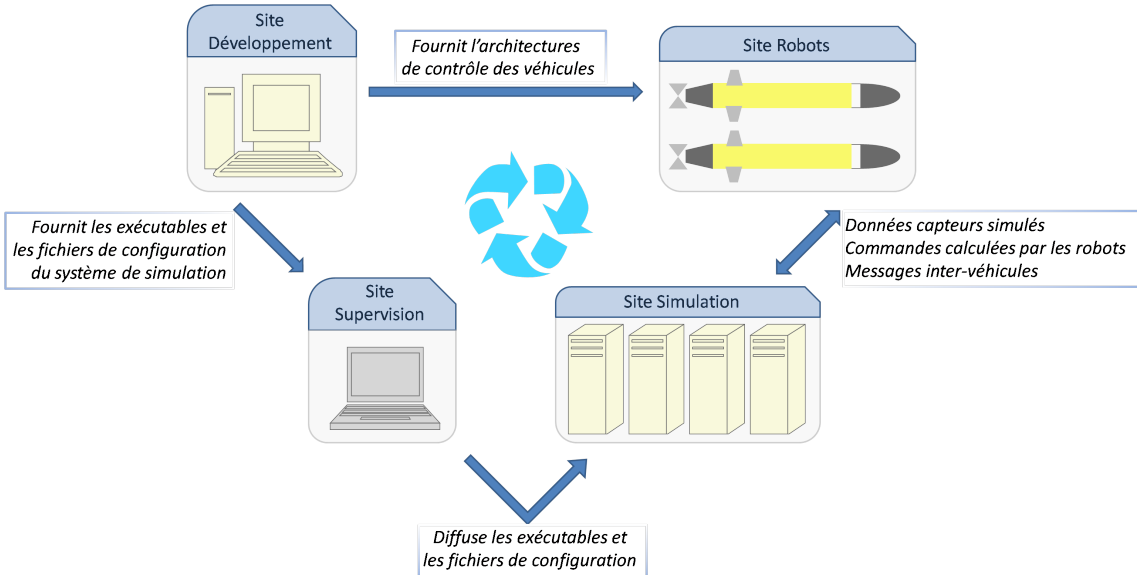


Fig. 5.1 – On présente sur cette figure les 4 sites de l'environnement de travail avec les flux de données inter-sites.

L'environnement de travail est donc réparti en quatre groupes de machines que nous présentons sur la figure 5.1.

5.2.1 Site de développement

Nous avons évoqué, dans les chapitres précédents, le caractère distribué de notre système de simulation. En effet, celui-ci est réparti en quatre simulateurs ayant chacun une fonctionnalité spécifique (simulation de capteurs ou de la dynamique ou ...). Par ailleurs les modèles à développer pour chacun d'entre eux sont nombreux et appartiennent à des domaines d'expertise clairement différents. Une solution pour permettre le travail en commun aurait été de permettre à chacun de développer en local sur sa propre machine puis d'aller tester le code sur le simulateur correspondant. Cependant, il n'est pas envisageable que chaque intervenant potentiel ait Linux/RTAI installé sur sa machine, ni qu'il ait l'expertise requise au niveau système pour mener à bien les développements, les configurations ou les mises à jour.

La solution retenue a été de centraliser les développements sur un serveur commun appelé *"ROB-RSM-SIMUDEV"* potentiellement accessible sur le réseau du LIRMM à l'ensemble des développeurs. Nous avons installé Linux/RTAI sur cette machine afin de permettre les tests unitaires avant le déploiement des architectures (simulateur et contrôleurs) sur le réseau.

Afin de garantir un accès simple et efficace (copier, modifier et transférer des fichiers d'un ordinateur à un autre) à l'ensemble des développeurs (qui ne sont pas nécessairement des spécialistes en informatique), nous avons choisi d'utiliser un système de fichiers de réseau : nous avons donc installé un serveur *samba* sur *ROB-RSM-SIMUDEV* qui rend accessible par le réseau un répertoire contenant l'ensemble de l'arborescence du système de simulation. Cette arborescence permet de séparer clairement les différentes entités du

système de simulation, les librairies à partir desquelles sont construites ces entités et enfin l'ensemble des fichiers de configuration.

Par ailleurs nous avons installé un serveur *ssh* (Secure SHell, mode de connexion sécurisé), afin d'autoriser les développeurs à compiler et exécuter les différentes entités du système de simulation. De plus un ensemble de scripts utilisables en ligne de commande (donc potentiellement *via* SSH) a été créé afin de faciliter les phases de compilation des améliorations apportées par les développeur et leur intégration au sein d'un exécutable. Typiquement, un développeur peut apporter une modification à la librairie de modèles de centrale inertielle par exemple, la compiler sans se soucier des dépendances et reconstruire automatiquement le système de simulation pour prendre en compte les modifications, à travers une simple ligne de commande.

Nous avons également adopté *Doxygen* pour produire la documentation du code source du simulateur. Doxygen [56] est un logiciel libre permettant de générer cette documentation à partir des commentaires écrits dans un format particulier du code source. L'intérêt d'utiliser ce logiciel réside entre autre dans l'insertion directe de la documentation dans le code source. Cette approche permet de maintenir la cohérence entre le code et la documentation et engage les développeurs à documenter le code qu'ils produisent. Outre les commentaires, Doxygen est capable de produire de façon automatique la documentation des structures de données, les prototypes et documentation des classes et leur hiérarchie, différents types de graphes (diagramme de classe, de collaboration, ...). Ces fichiers peuvent être directement navigables, ce qui facilite d'autant la consultation et le compréhension du code et de sa documentation associée.

Enfin, nous avons déployé un serveur *HTTP* (HyperText Transfer Protocol) afin de mettre à la disposition de tous la documentation technique relative au code source du logiciel. Cette documentation est générée automatiquement avec Doxygen et mise en ligne automatiquement grâce à un script.

Concernant la configuration matérielle de cet ordinateur, il s'agit d'un Intel/Pentium III cadencé à 600 MHz avec 192Mo de RAM. Nous avons installé Linux/RTAI sur cet ordinateur pour les raisons précédemment évoquées. Par ailleurs, il est à noter que les architectures de contrôle de nos AUVs sont également développées et stockées sur cet ordinateur. Nous ne rentrons pas dans ces détails, mais davantage d'informations à ce sujet peuvent être trouvées dans [57]. Nous signalons enfin que ces serveurs ne sont pour l'instant accessibles qu'en intranet.

En ce qui concerne la plate-forme de développement, notre approche et les technologies que nous avons mises en œuvre sont pleinement en adéquation avec la notion d'ouverture que nous avions précédemment évoquée. La figure 5.2 reprend les différents services offerts par cette plate-forme.

5.2.2 Site de simulation

Outre le site de développement, notre environnement de travail est constitué par le système de simulation lui-même. Le simulateur que nous avons développé peut être distribué sur un réseau constitué de un à quatres ordinateurs, chacun d'entre eux possédant une ou plusieurs unités de calcul. L'utilisateur peut donc utiliser au mieux les ressources de calcul disponibles afin de distribuer la simulation.

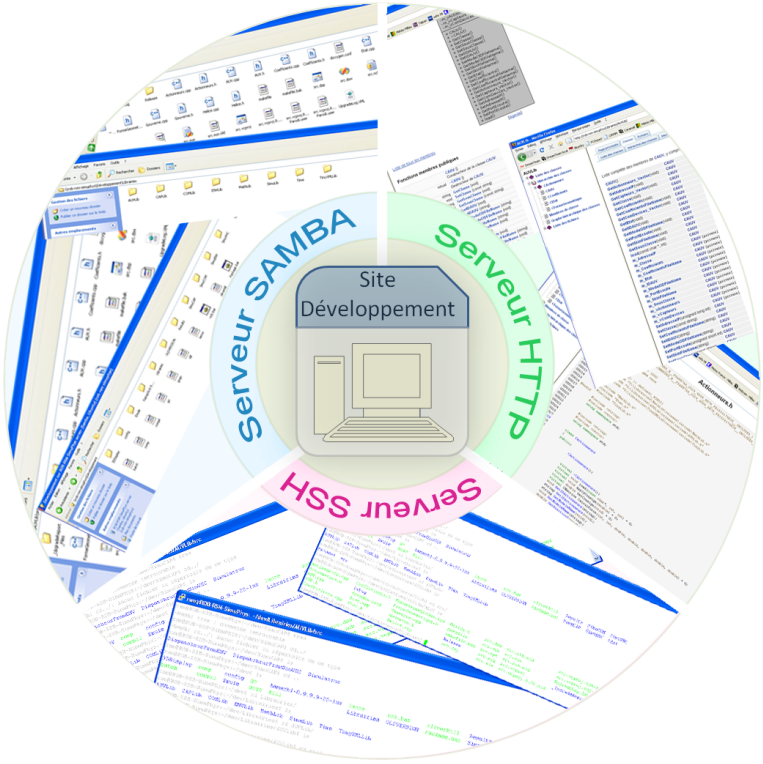


Fig. 5.2 – Le site de développement est un serveur http, un serveur ssh et un serveur samba sur lequel le code source est documenté et mis à disposition des développeurs.

L'utilisation d'ordinateurs disposant de plusieurs unités de calcul s'avère être une bonne solution pour distribuer en interne les processus d'un simulateur. Ainsi, si l'utilisateur dispose d'un quadri-coœur, il sera possible de répartir au mieux les quatre processus du simulateur de capteurs : chacun des trois processus dédiés à l'écoute, au décodage et au partage des trames en provenance des autres entités de la simulation sera exécuté sur sa propre unité de calcul, évitant ainsi de perturber le déroulement du quatrième processus de simulation s'exécutant sur la quatrième unité de calcul. Ces quatre processus doivent néanmoins être exécutés sur des unités de calcul ayant en commun une zone de mémoire ; cela est généralement synonyme d'exécution sur une même machine physique. Il est à noter qu'il est tout à fait possible de lancer tous les processus appartenant à un même simulateur sur un seul et unique processeur. Cette approche présente néanmoins moins d'intérêt si le temps processeur consommé par les changements de contexte répétés et par les processus de décodage devient non négligeable par rapport à celui du simulateur.

Par ailleurs, les serveurs possédant un grand nombre d'unités de calcul ne sont pas toujours accessibles facilement. C'est la principale raison qui nous a amené à distribuer les simulateurs sur un réseau. Il est effectivement souvent plus aisé d'avoir à sa disposition quatre ordinateurs quadri-cœurs qu'un serveur intégrant 14 unités de calcul (hormis la supervision et l'affichage 3D, notre système de simulation est actuellement composé de 14 processus).

La mise en réseau de plusieurs ordinateurs effectuant des tâches temps réel nécessite certaines précautions. En effet, même si certains protocoles tels que RTnet [58] permettent dans une certaine mesure de faire du temps réel distribué sur un réseau ethernet, nous

n'avons pas retenu pour l'instant ce type de solution pour des questions de temps d'implémentation. Nous avons privilégié l'usage d'un réseau dédié, c'est-à-dire un réseau sur lequel la bande passante est constante et dédiée à la simulation. Dans ce cadre, et en veillant à ce que la bande passante utilisée reste inférieure à 10% des capacités offertes par le réseau, les probabilités de collisions et les retards sont faibles. Enfin, nous avons recours à un switch qui permet un fonctionnement en réseau commuté : à la différence d'un hub qui fonctionne en diffusion, les paquets sont envoyés uniquement au destinataire. Cela a pour conséquence de réduire très significativement le risque de collision puisque la seule source possible serait d'avoir un récepteur qui prendrait la parole au moment où il reçoit un paquet. Globalement, le réseau que nous utilisons possède un comportement compatible avec les contraintes temps-réel sans toutefois nous en apporter les garanties.

Au LIRMM, nous utilisons notre système de simulation dans deux configurations.

- La première a recours à plusieurs ordinateurs multi-processeurs. Dans ce cadre ces ordinateurs sont déconnectés du réseau interne du LIRMM et connecté sur un switch dédié. Pour pouvoir utiliser facilement n'importe quel poste libre, nous utilisons des clés USB/CD bootables sur lesquelles est installé Linux/RTAI. Cette approche nous permet de déployer rapidement notre système de simulation en s'adaptant à la disponibilité des ressources ; elle est utilisée dans le cadre de la préparation réelle des missions. Il est à noter toutefois que les tâches ne sont pas encore ordonnancées en temps-réel car nous sommes toujours en phase de développement et de validation. Néanmoins, les mesures que nous avons effectuées, qui sont en partie présentées dans le chapitre 7, nous montrent qu'au regard des constantes de temps de notre contexte, Linux peut suffire dans un premier temps.
- La deuxième configuration de simulation consiste à utiliser un des serveurs de calcul du LIRMM mis à notre disposition ; il 'agit du serveur ATGC, composé de sept processeurs et sur lequel nous exécutons deux simulateurs. Parallèlement, nous continuons à exécuter les deux autres simulateurs sur les ordinateurs précédemment évoqués. Cette configuration est utilisée durant les phases de développement du simulateur, car elle présente l'avantage de déployer plus rapidement le système sur le réseau : tous les calculateurs restent connectés sur le réseau interne du LIRMM (réseau commuté).

Cette approche, permettant de répartir les calculs effectués par le système de simulation localement mais également sur un réseau, est bien en adéquation avec la contrainte de distributivité du système que nous nous étions fixée.

5.2.3 Site de supervision

Le superviseur possède plusieurs rôles. Il est chargé de gérer la distribution des fichiers de simulation (fichiers de configuration et exécutables) sur le réseau, d'ordonnancer la séquence de lancement d'une simulation, de superviser l'exécution de la simulation en faisant remonter les informations au niveau de l'utilisateur et enfin il doit récupérer les logs sur les différents simulateurs en fin de mission.

Ce logiciel a été développé pour windows et est exécuté sur un ordinateur portable muni d'un processeur Intel/Pentium M à 2GHz. Le programme est écrit en C++ et consiste en partie à générer des lignes de commandes shell d'après le contenu des fichiers de configuration. On utilise alors la commande *system* afin de lancer ces commandes qui

appellent le logiciel *Putty* [59]. Ce dernier est un programme permettant de se connecter à distance à des serveurs en utilisant le protocoles SSH et lance des commandes à distance sur les unités distribuées. Par ailleurs, nous utilisons les mécanismes de socket, sur lesquels nous revenons au paragraphe 5.4.2, pour communiquer avec la couche application des simulateurs.

Le superviseur est un élément clé de notre système de simulation car il gère ce qui gravite autour de la distributivité de notre simulateur.

5.2.4 Site robots

Le caractère HIL de *Thetis* en fait un simulateur sur lequel les robots participant à la simulation se connectent directement. Ainsi, chaque ordinateur de bord sur lequel est déployé l'architecture de contrôle de ces robots est relié au système de simulation par un lien ethernet.

Nous utilisons deux configurations pour réaliser nos simulations :

- La première correspond à la configuration nous permettant de préparer réellement les missions. Il s'agit d'une simulation en mode HIL dans lequel le véritable robot est connecté au système. L'ordinateur de bord de nos véhicules est un PC-104 (norme de PC embarquée qui définit une forme dont le bus permettant de connecter des cartes d'extension est formé de 104 points) intégrant un processeur Intel/Celeron cadencé à 400MHz ; le système d'exploitation installé est Linux/RTAI. De façon générale, tous les détails concernant l'architecture logicielle de nos véhicules peut-être trouvée dans [57].
  La configuration précédente présente l'intérêt d'utiliser le véritable matériel et permet de préparer la mission en garantissant un comportement temporel identique de l'architecture de contrôle pendant les phases de test et d'expérimentation. Néanmoins, les véhicules ne sont pas systématiquement disponibles pour réaliser ce type de simulation et leur nombre limité ne nous permet pas de valider des stratégies de coordination faisant intervenir plus de deux véhicules.
- Par conséquent, nous avons recours à la simulation online qui présente l'intérêt d'utiliser exactement la même architecture de contrôle que celle du robot réel, mais déployé sur du matériel différent. En effet, dans cette configuration nous déployons Linux/RTAI sur un ordinateur standard et y exécutons l'architecture de contrôle. Pour simplifier les opérations, nos utilisons un CD bootable nous permettant de travailler dans un environnement Linux/RTAI depuis n'importe quelle machine connectée au réseau. Nous avons également écrit un script permettant de se connecter directement au site de développement, pour y télécharger les architectures de contrôle des véhicules et les exécuter (figure 5.3). Il est à noter que, dans cette configuration, le temps d'exécution des algorithmes peut varier significativement. Si les tests de simulation online s'avèrent particulièrement pratiques et rapides à mettre en place, ils ne dispensent pas pour autant de refaire les tests en configuration HIL avant d'effectuer une mission réelle.

Pour compléter ces deux configurations de simulation nous travaillons actuellement sur une troisième possibilité qui nous permettrait de simplifier davantage le déploiement préalable à une simulation multi-véhicule. En effet, même s'il n'est pas nécessaire de disposer des véhicules réels pour lancer une simulation, il faut en revanche trouver autant

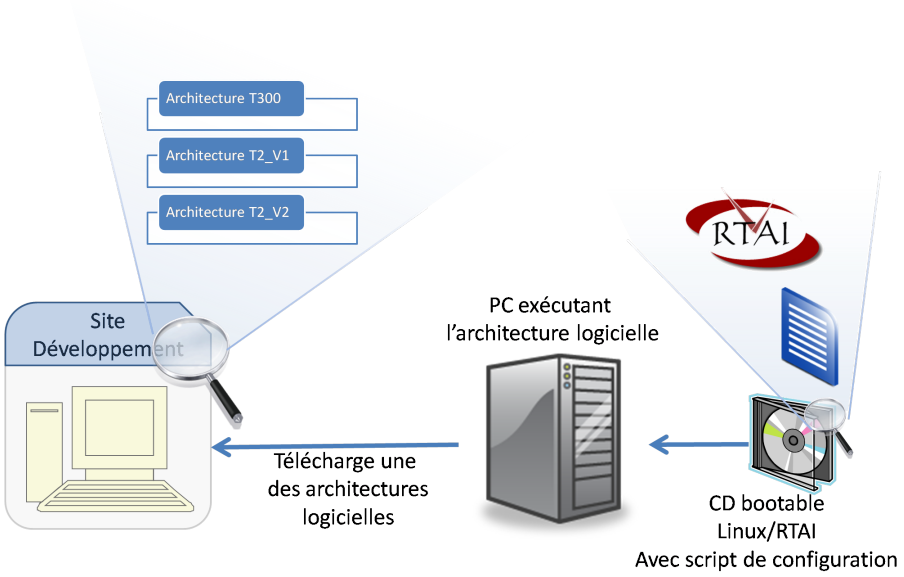


Fig. 5.3 – En configuration online, le véhicule est remplacé par un ordinateur standard qui télécharge et exécute l'architecture logicielle du véhicule sur le serveur de développement grace à un script.

d'ordinateurs que de véhicules impliqués dans la simulation et y déployer, *via* le CD bootable précédemment évoqué, Linux/RTAI et l'architecture de contrôle. Cette opération, bien que facilitée par l'utilisation de CD, peut s'avérer longue et fastidieuse si le nombre d'engins impliqués est conséquent. Pour pallier le problème nous testons actuellement un mode que nous qualifions de *virtualisation online* et qui consiste à déployer plusieurs architectures logicielles de nos AUVs sur plusieurs machines virtuelles, exécutées par une même machine physique (figure 5.4).

Les premiers tests ont été réalisés avec l'intervention de Stéphane George (technicien réseau au LIRMM) montrant qu'il était possible de faire tourner trois Linux/RTAI sur un Intel/Pentium 4HT @ 3GHz en utilisant *VMware Server*. Il est à signaler ici que, bien que les résultats des premiers tests nous aient montré que les performances étaient acceptables sur le plan du comportement temporel, il n'est plus possible ici de garantir la notion de temps-réel puisque le système hôte n'est pas temps réel. Il s'agit donc avant tout d'une configuration nous permettant de tester très rapidement et en première approche une première implémentation de stratégie de commande coordonnée qui doit être validée par des tests HIL avant l'expérimentation réelle.

Ces approches permettent de réaliser des simulations multi-véhicules de plusieurs façons différentes, choisies en fonction de la disponibilité des ressources et de la phase de test. Ces technologies nous permettent donc de remplir nos objectifs concernant la notion du multi-véhicule.

5.2.5 Conclusion

Nous avons constaté dans ce paragraphe que la configuration des machines, les services mis en place, les technologies choisies concourent à la réalisation de nos objectifs : temps

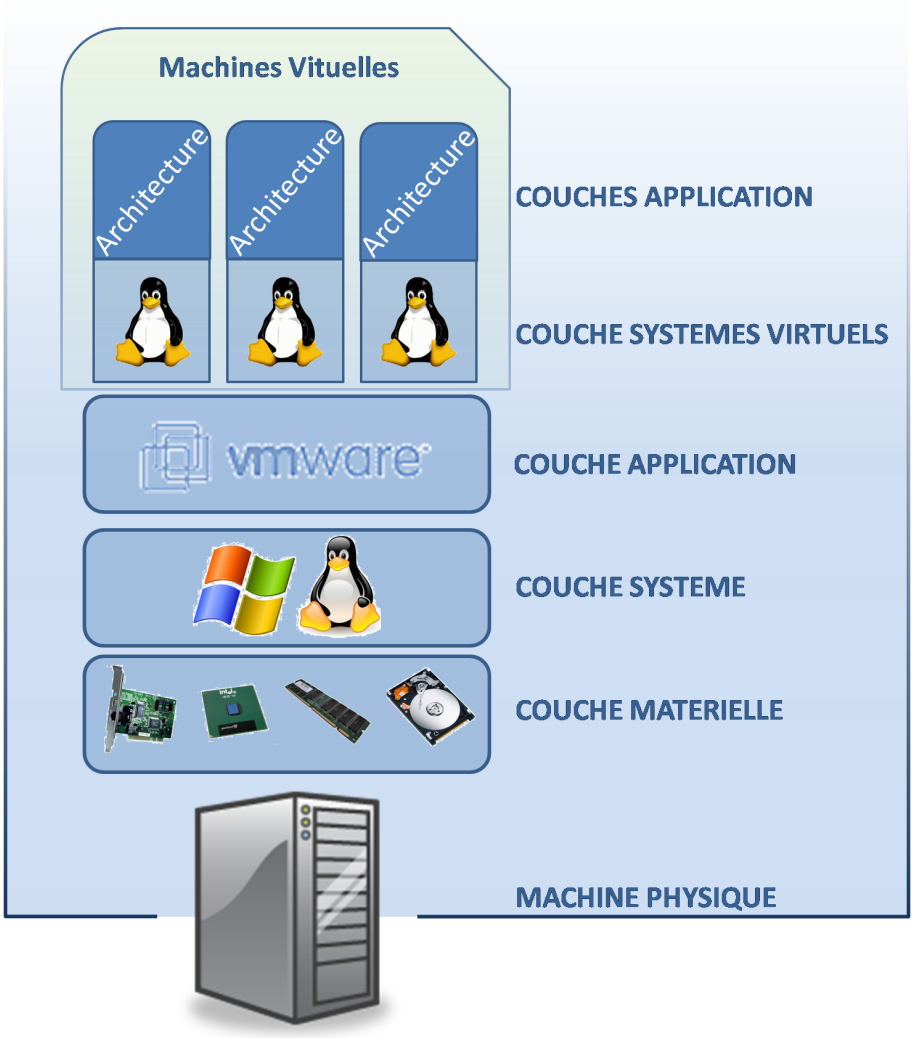


Fig. 5.4 – Créer des machines virtuelles, nous permet d'exécuter plusieurs architectures logicielles de contrôle sur une même machine physique.

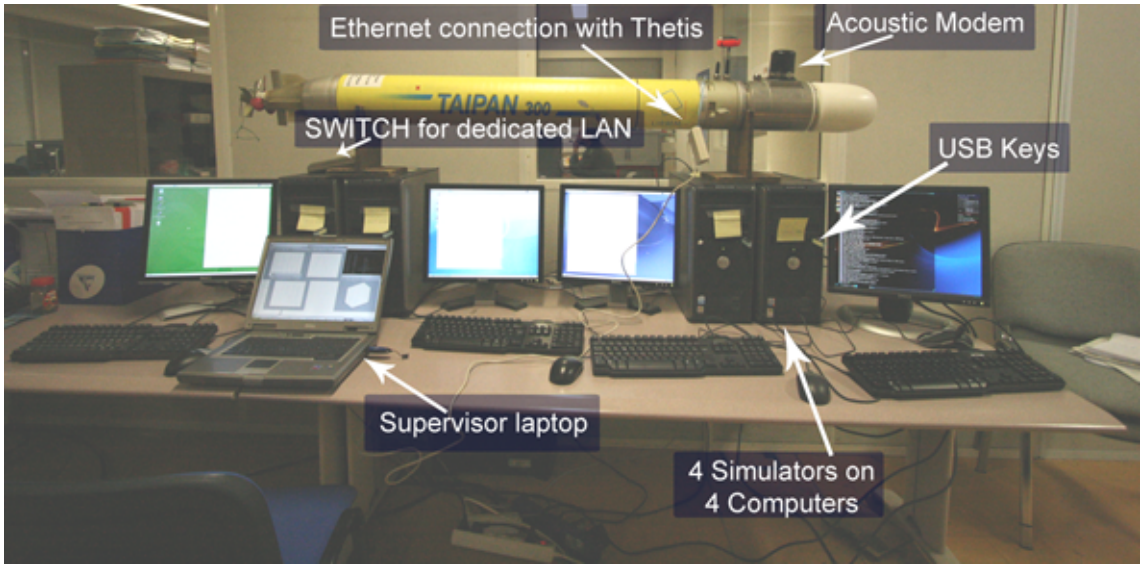


Fig. 5.5 – Le système Thetis connecté à l'ordinateur de bord de Taipan300.

réel, multi-véhicule, HIL (avec des déclinaisons online), ouverture...

Sur la figure 5.5, nous présentons une vue du simulateur en mode "préparation de mission" : on utilise le véritable véhicule connecté sur un réseau dédié, au simulateur réparti sur quatre ordinateurs multi-cœurs.

5.3 Les choix technologiques

Pour réaliser ce système de simulation, nous nous sommes appuyés sur un ensemble de technologies existantes que nous allons brièvement présenter.

Il est à noter que nous avons choisi de développer notre simulateur en C++ pour plusieurs raisons :

- sa rapidité d'exécution (comparée au java par exemple)
- RTAI accepte ce langage
- il s'agit d'un langage orienté objet, qui nous permet d'atteindre le niveau d'abstraction requis pour la réutilisabilité du code
- c'est un langage très largement utilisé dans la communauté scientifique et nous permet de préserver la notion d'ouverture
- la possibilité de compilation commune avec le langage C (utilisé pour le bas niveau) *via* la commande *extern*

Le C++ est donc compatible avec nos objectifs de temps réel, de modularité et de réutilisabilité.

Développer un projet complexe nécessite de passer par une étape de modélisation que nous allons présenter dans le paragraphe suivant. Ensuite, nous examinerons les librairies existantes que nous avons utilisées dans ce projet.

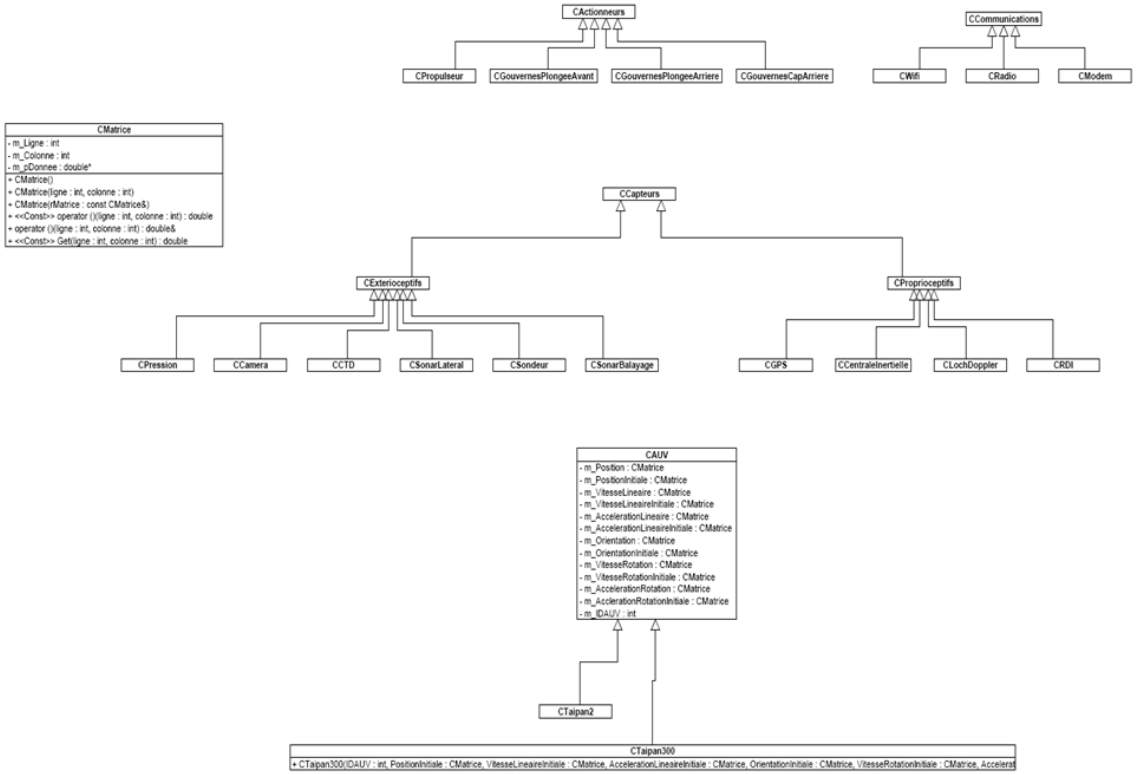


Fig. 5.6 – L'UML a été utilisée au commencement du projet pour modéliser les classes de Thetis.

5.3.1 La modélisation

Pour aborder ce projet, nous sommes passés par une étape de modélisation. Pour ce faire, notre choix s'est porté sur l'*UML* (Unified Modeling Language) et plus particulièrement sur le diagramme des classes, ce qui nous a permis de représenter les dépendances entre les classes. L'UML est un langage graphique de modélisation des données et des traitements et est composé de 13 types de diagrammes [60]. Le diagramme de classe que nous avons utilisé permet de représenter une collection d'éléments de modélisation statiques qui montre la structure de notre modèle. En revanche, ce diagramme ne permet pas de représenter les aspects dynamiques et temporels. La figure 5.6 présente une partie des diagrammes de classes modélisés.

Il est à noter que ce travail de modélisation n'a pas été maintenu durant la construction du projet ; une fois la modélisation des classes de base accomplie, nous avons continué ce travail de représentation en utilisant le programme Doxygen précédemment évoqué. La représentation sous forme de diagramme de classe concourt donc à la notion d'ouverture, facilitant la compréhension des dépendances entre les objets et leur contenu pour les développeurs.

5.3.2 Choix des librairies existantes

Pour réaliser notre simulateur nous avons utilisé quelques librairies existantes. En ce qui concerne les interfaces homme machine, permettant à l'utilisateur de créer, configurer

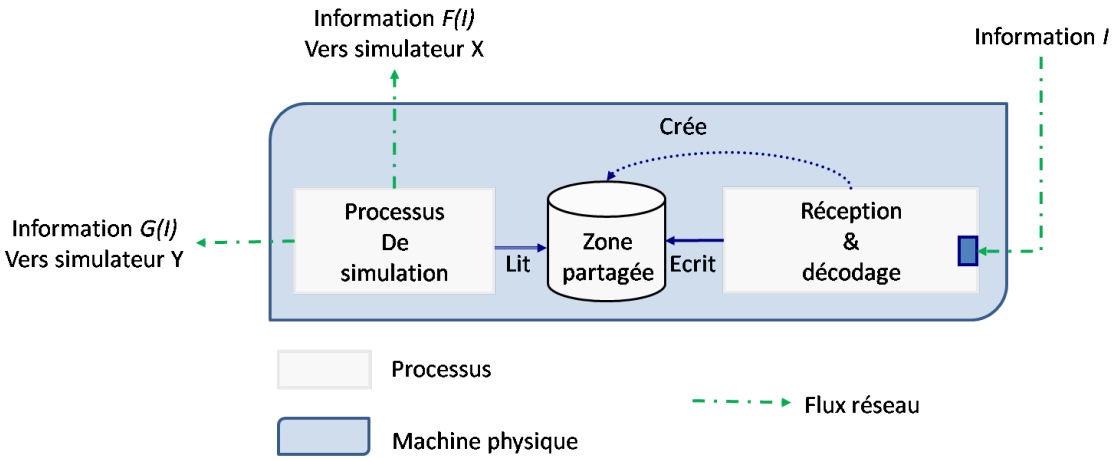


Fig. 5.7 – Module élémentaire de Thetis

et lancer les simulations, nous avons choisi la librairie *GTK+*. Cette librairie présente l'intérêt d'être portée pour les principaux systèmes d'exploitation (Windows, Linux,). Cette bibliothèque permet de réaliser des interfaces graphiques et peut être utilisée conjointement avec le logiciel *Glade* qui permet d'en faciliter l'exploitation. En effet, ce logiciel permet de dessiner l'interface, et de générer le code GTK correspondant ; à la charge de l'utilisateur de relier les *signaux* (c'est-à-dire les événements utilisateur) aux fonctions créées.

Comme nous le verrons au paragraphe 5.5, nous avons utilisé le *XML* (Extensible Markup Language) pour décrire les différents composants de notre système de simulation. Afin de pouvoir exploiter facilement cette technologie, nous avons utilisé une librairie nommée *TinyXML* [61]. Il s'agit d'un parser XML libre dédié au langage C++ et simple d'utilisation. Elle nous permet de lire et écrire les différents fichiers de configuration et de créer les fichiers logs.

Enfin, nous utilisons une toolbox Matlab créée par *Geodise* [62] qui nous permet d'exploiter les fichiers de log générés en XML. Cette librairie permet de convertir et enregistrer les variables et structures internes de Matlab dans un format lisible en XML et vice-versa.

Ces trois librairies nous permettent d'utiliser des technologies qui améliorent la notion d'ouverture et d'interopérabilité de notre système.

5.4 Implémentation du module de base

Dans cette section, nous présentons l'implémentation du module élémentaire présenté en 4.2.2 et dont nous rappelons le schéma figure 5.7.

5.4.1 Mémoire partagée

Afin de garantir la libre évolution des processus de simulation, nous avons divisé notre application en processus appartenant à deux classes différentes : la classe des processus de simulation proprement dite, et la classe des processus de communication, décodage et partage d'informations en provenance des autres simulateurs. Cette approche nous permet de dédier un processeur pour chaque processus, notamment ceux en charge de la

simulation. Pour permettre à l'application de fonctionner, ces processus sont néanmoins amenés à échanger des données. Parmi l'ensemble des mécanismes de communication inter processus (dits *IPC*) nous avons choisi les mémoires partagées, car elles nous permettent d'échanger de gros volumes de données de façon presque instantanée en évitant une copie pour chaque processus destinataire. La question de l'utilisation des mailboxs s'est posée, mais ces dernières, outre le fait que les informations à partager sont à copier pour chaque destinataire, ne permettent pas de modifier une partie des informations : l'intégralité du message doit être postée à chaque modification.

La structure interne de toutes ces mémoires partagées est différente pour chacune d'entre elles. En effet, leur structure est définie dynamiquement pendant la phase de lancement du système de simulation, au regard du nombre de véhicules, de capteurs et de moyens de communication lu dans les fichiers de configuration. Nous utilisons pour cela la classe *"vector"* qui permet de créer des vecteurs de pointeurs de mémoires partagées facilement manipulables. Les figures 5.8 et 5.9 présentent deux exemples de ces structures. Il est à noter qu'en fait ce n'est pas une mémoire qui est crée mais plusieurs : une pour chaque AUV.

5.4.2 Socket

Permettre à plusieurs processus d'être exécutés sur une même machine physique n'est toutefois pas suffisant. En effet, chaque simulateur peut être exécuté sur un ordinateur différent et il est alors nécessaire de leur permettre de communiquer à travers le réseau. Les *sockets* constituent un vecteur de communication inter processus répondant à cette nécessité. Il s'agit d'une *API* (Application Program Interface) pour la communication entre plusieurs processus répartis sur un ou plusieurs hôtes en utilisant la couche réseau de la pile IP. Les sockets fournissent un mécanisme pour échanger les données en masquant le travail nécessaire, pris en charge par le système. On distingue deux modes de communication :

- Le mode connecté qui peut être comparé à une communication téléphonique : la communication est établie de façon durable entre les deux processus après une phase d'identification et l'adresse de destination n'est pas nécessaire à chaque envoi de données. En cas d'interférence, le protocole demande la ré-émission du message : il s'agit d'un mode de communication fiable. Ce mode peut être associé au protocole TCP. Il est à noter que l'écriture peut devenir bloquante si le récepteur ne lit pas suffisamment vite de son côté.
- Le mode non connecté qui s'apparente à une communication par courrier : l'adresse de destination est nécessaire à chaque envoi et il n'y a pas d'accusé de réception. Le processus destinataire regarde régulièrement dans sa boîte et peut y découvrir un ou plusieurs messages. La relation d'ordre n'est pas conservée, et aucun mécanisme de ré-émission n'est implémenté dans ce protocole : la couche logicielle peut en revanche parfois s'en charger. Ce mode peut être associé au protocole UDP.

Les sockets utilisent un modèle client-serveur pour fonctionner. Ainsi un serveur doit être à l'écoute de messages éventuels ; cette écoute se fait différemment selon le mode de connexion du socket et est représentée sur la figure 5.10.

Sans entrer dans les détails, la différence fondamentale entre les deux modes de communication est le fait qu'en mode connecté, le client doit attendre l'acquittement de la

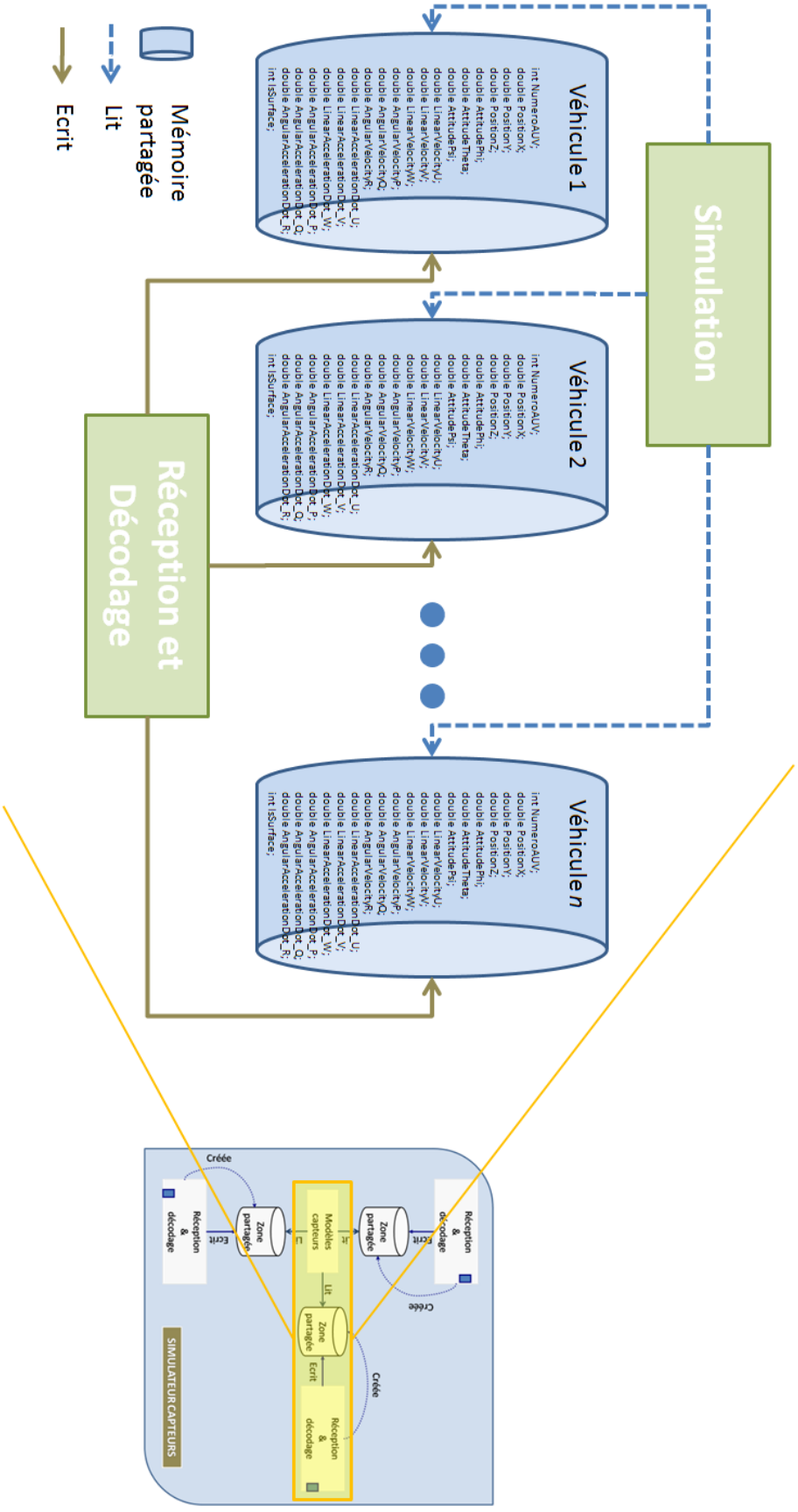


Fig. 5.8 – Mémoires partagées entre le processus de simulation des capteurs et le processus de réception et décodage des données en provenance du simulateur de véhicule. Chaque mémoire partagée contient le vecteur d'état d'un véhicule.

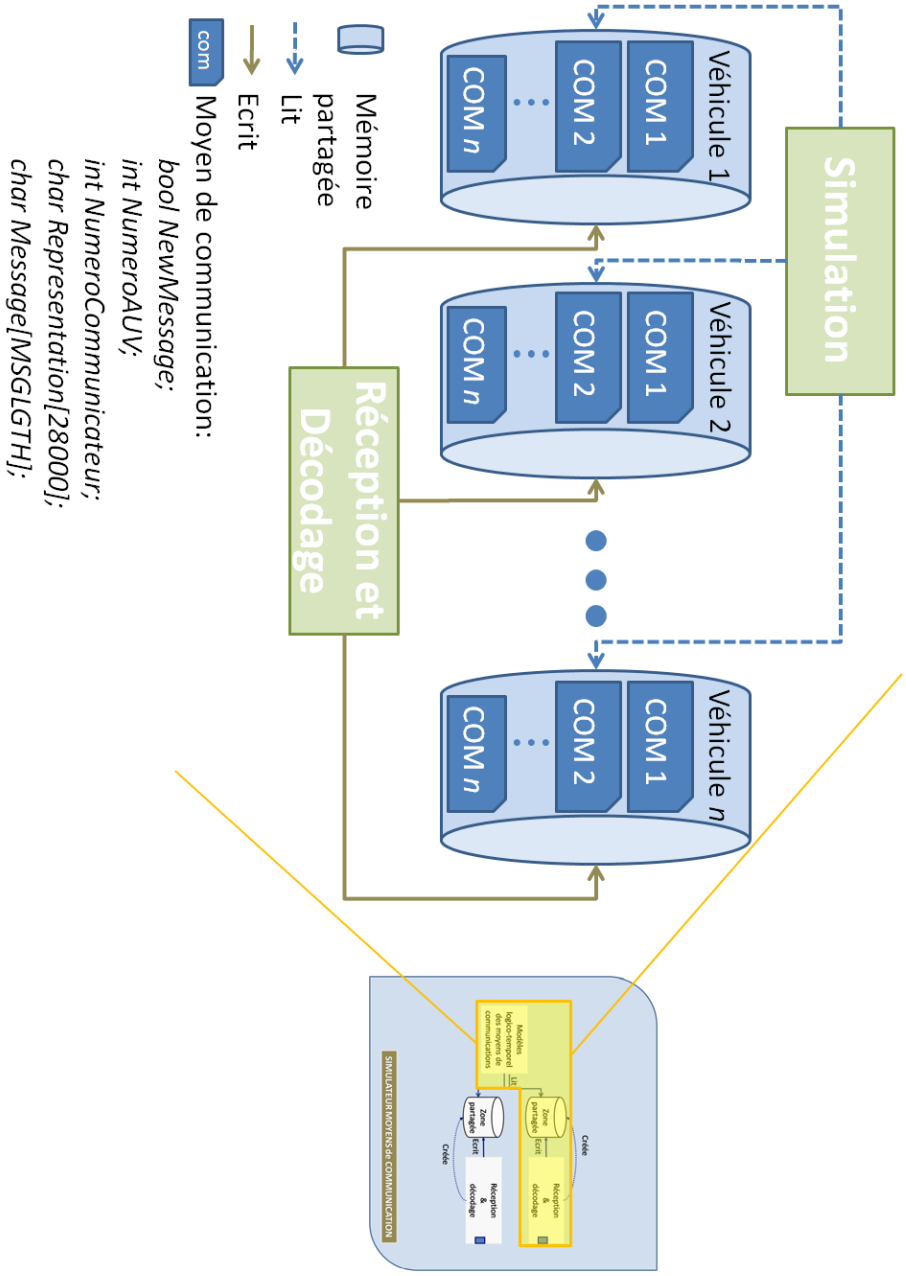


Fig. 5.9 – Mémoires partagées entre le processus de simulation des moyens de communication et le processus de réception et décodage des données en provenance du simulateur d'environnement. Chaque mémoire partagée contient un nombre de structures crées dynamiquement au lancement du système contenant entre autre le message "en clair" et un buffer de 28ko dans lequel est stockée la représentation numérique du signal analogique émis/reçu au niveau de l'antenne.

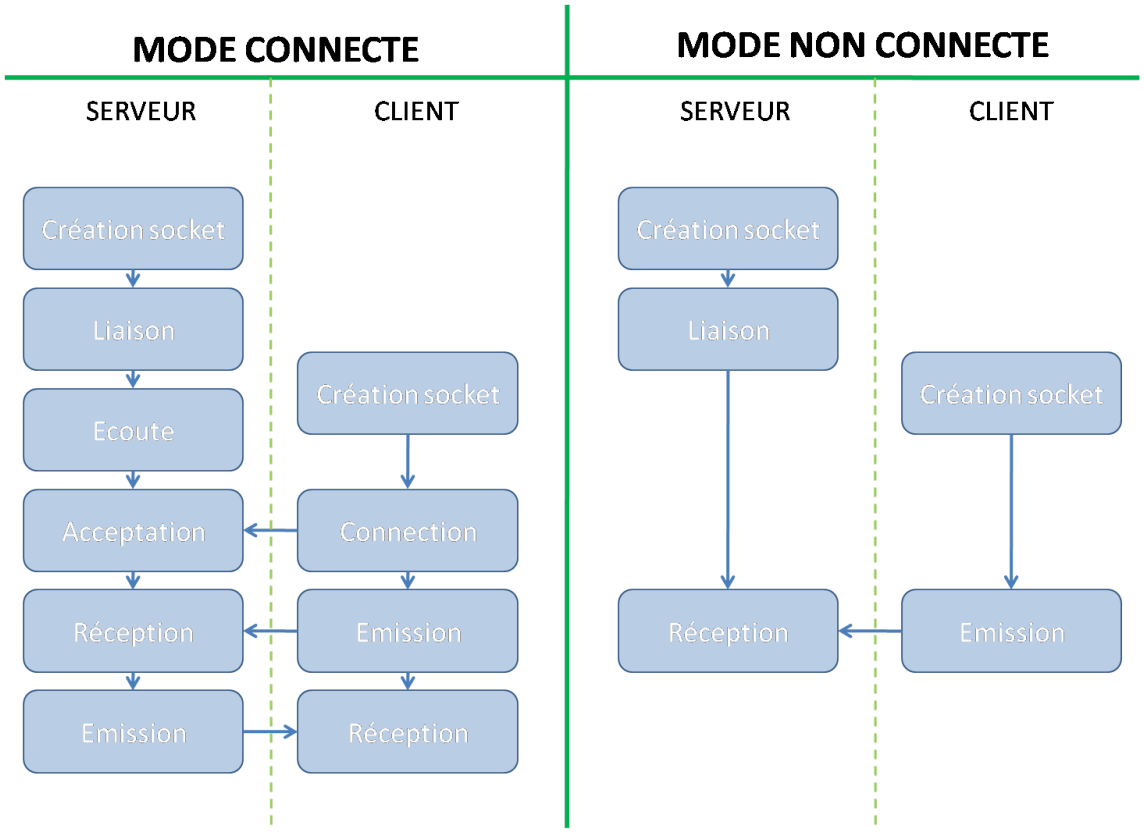


Fig. 5.10 – Différence de mécanisme entre les sockets utilisées en mode connecté ou non connecté

transmission de la part du serveur avant de pouvoir ré-émettre une donnée. Utiliser ce mode pour relier nos modules élémentaires reviendrait à faire patienter le processus de simulation (qui pour rappel est chargé de l'envoi des données qu'il génère) jusqu'à la réponse du processus serveur. Cela aurait pour conséquence de ne plus assurer le découplage temporel entre les deux processus. Dans notre architecture le mode connecté est donc à proscrire au profit du mode non connecté. En effet, dans ce mode le processus client (le simulateur donc) peut envoyer une trame à tout moment sans nécessiter l'accord préalable du processus serveur ; le découplage temporel est donc assuré.

Il est à noter que l'inconvénient du mode non connecté est l'absence d'accusé de réception des messages. Pour pallier cet inconvénient, il serait envisageable d'isoler le processus de simulation entre deux mémoires partagées : la première telle que nous l'avons décrite, et la seconde serait destinée à l'écriture des données à envoyer sur le réseau, envoi réalisé par un troisième processus de communication. Ce schéma permettrait d'utiliser le mode connecté tout en garantissant le découplage temporel des processus de simulation.

5.5 Modularité et XML

Nous avons utilisé le XML pour décrire plusieurs composants de notre système de simulation. Outre la facilite de parser un fichier écrit dans ce langage, le XML permet de ne pas utiliser de format propriétaire sous forme de fichiers de données non formatées. Son utilisation facilite la compréhension du contenu des fichiers tout en garantissant une souplesse d'évolution. Nous avons donc développé un formalisme permettant de décrire les éléments du simulateur. Quatre types de fichiers sont décrits en ayant recours au XML : les fichiers de configuration des véhicules (et des éléments qui leurs sont associés), ceux des simulateurs, ceux des résultats de simulation et enfin ceux des interférences.

5.5.1 Proposition de description formelle pour un robot autonome

Dans ce paragraphe, nous proposons une façon générique de décrire les robots mobiles. De notre point de vue, un robot mobile peut toujours être décomposé en un ensemble d'actionneurs, de capteurs, de moyens de communication. Par ailleurs, lorsque l'on place ce robot dans un contexte de simulation, celui-ci :

- a un nom
- possède un vecteur d'état initial (position, vitesse, accélération linéaires et de rotation)
- possède une représentation graphique dans le monde virtuel
- appartient à une classe de véhicule
- est représenté par un modèle auquel sont associés des coefficients
- possède une interface d'échange de données

Cette vision nous a mené à représenter les véhicules à l'aide de fichiers XML répartis au sein d'une arborescence. Sur la figure 5.11, nous montrons une représentation du code XML du fichier de configuration d'un véhicule de type AUV. On y retrouve les informations suivantes :

- le type du véhicule (AUV) qui pourrait être un ASV, UAV, UGV
- sa classe et sa sous-classe ([Taipan, Taipan300]) qui pourrait être un [Remus, Remus6000]

Fig. 5.11 – Principaux nœud du fichier XML de configuration d'un véhicule

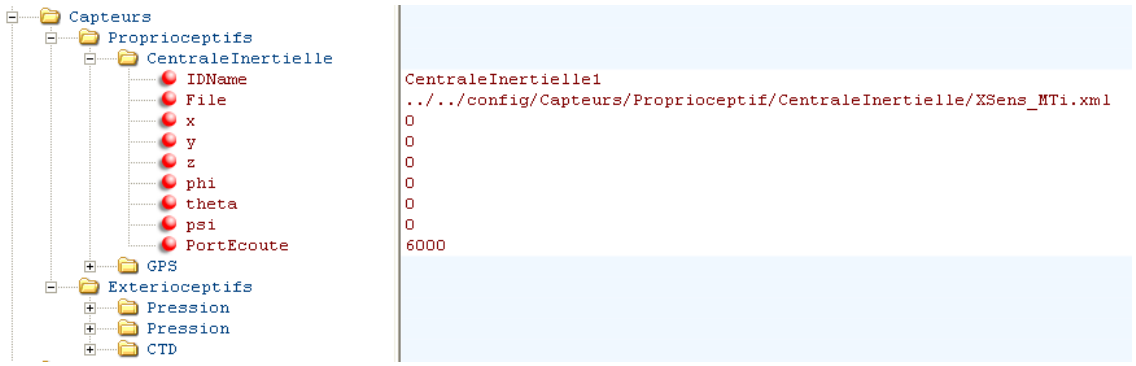


Fig. 5.12 – Nœud capteur : on distingue des capteurs proprioceptifs ou extéroceptifs. Chaque capteur possède un nom, un fichier de paramètres associé à son modèle, et un positionnement

- son nom au sein de la simulation
- son adresse IP et son port d'écoute : ces informations permettent d'envoyer les données capteurs et de délivrer les messages au véhicule durant la simulation
- un nœud "Général" dans lequel on retrouve le chemin d'accès aux fichiers décrivant son aspect géométrique, son skin (permettant la représentation graphique) et les coefficients de son modèle
- un nœud "Initial" dans lequel se trouvent la valeur des composantes de son vecteur d'état initial (figure 5.13)
- un nœud "Capteurs" dans lequel se trouvent l'ensemble des positions/orientations des capteurs dans le repère du véhicule, avec le chemin du fichier comportant les paramètres des modèles (figure 5.12)
- un nœud "Actionneurs" avec des informations similaires au nœud "Capteurs"
- un nœud "Communicateurs" avec des informations similaires au nœud "Capteurs"

Cette approche permet de configurer facilement un véhicule et permet d'interchanger les capteurs, actionneurs, moyens de communication, en réutilisant des fichiers de configuration et des modèles existants. Ainsi un UAV pourra partager le même fichier de description de la centrale inertielle qu'un autre AUV : il suffira de déclarer le chemin vers le fichier de configuration de cette centrale (dans lequel sont décrits les paramètres du

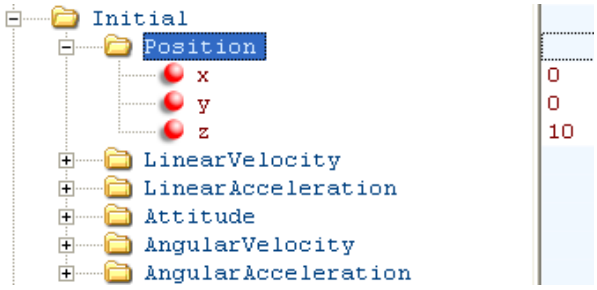


Fig. 5.13 – Nœud vecteur d'état initial : on retrouve les valeurs de chaque composante du vecteur d'état d'un véhicule

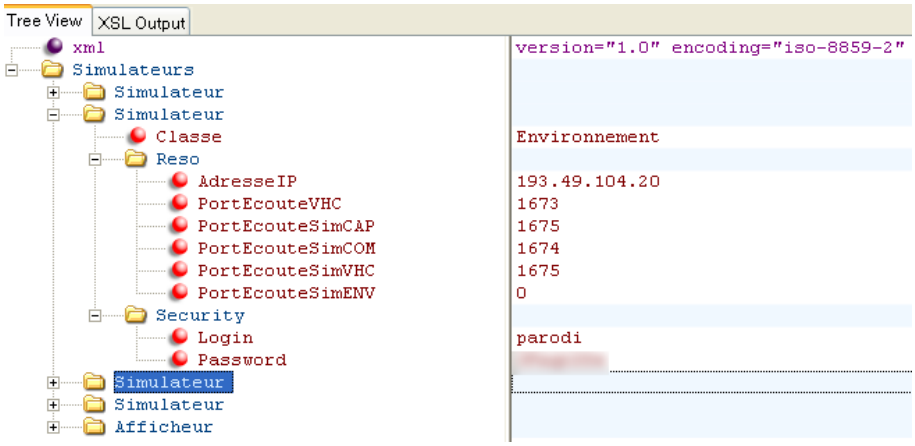


Fig. 5.14 – Fichier de configuration réseau des simulateurs

modèle de la centrale en question) dans le fichier de configuration des deux véhicules.

L'utilisation d'un modèle se fait par le nom du nœud auquel est rattaché un fichier de configuration. Ainsi, par exemple, toutes les centrales inertielles utilisées dans la simulation auront le même modèle ; seuls les paramètres associés au modèle changent. Nous envisageons de rajouter une balise modèle permettant d'utiliser des modèles différents pour un élément de même type.

Notre proposition favorise la modularité, l'ouverture, la réutilisabilté et la notion de multi-véhicule de notre simulateur.

5.5.2 Les simulateurs

Nous utilisons également le XML pour décrire la configuration réseau des différents simulateurs et de l'affichage 3D. Le fichier comporte également des informations liées à la sécurité de l'accès distant des simulateurs. Un exemple de fichier est donné figure 5.14 Cette approche nous permet de configurer très rapidement le système de simulation en nous adaptant aux ressources disponibles.

5.5.3 Les résultats

De même les résultats de la simulation sont sauvegardés au format XML. Cela permet de développer des macros d'affichage des données personnalisées : les données sont

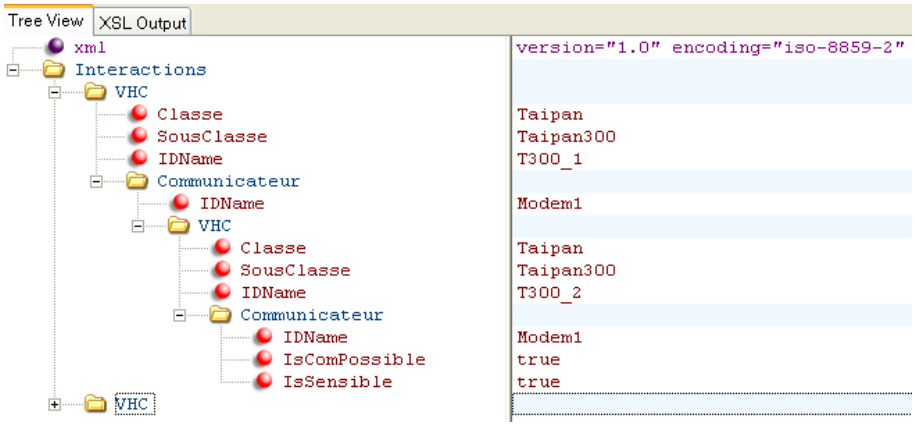


Fig. 5.15 – Fichier de configuration des interférences entre les moyens de communication des différents véhicules

toutes rendues dans le même formalisme et peuvent être exploitées après avoir parsé le fichier. Il existe plusieurs fichiers de résultats correspondant chacun à un aspect de la simulation : dynamique, commandes reçues du contrôleur du véhicule, données capteurs générées, messages échangés. Cet ensemble de fichiers est généré pour chaque véhicule ; les données sont datées et classées. Des exemples de ces fichiers peuvent être trouvés sur le site Internet de Thetis.

5.5.4 Les interférences

Un dernier fichier explicitant les interférences entre les moyens de communication et les capteurs de chaque véhicule, nous permet de configurer une simulation. Par défaut il n'existe aucune interférence, et il est à la charge de l'utilisateur de les spécifier si besoin. Sur la figure 5.15, nous présentons un fichier de configuration d'interférences entre deux véhicules. Les interférences sont orientées (par exemple le modem acoustique A du véhicule 1 est perturbé par les émissions du modem acoustique A du véhicule 2 alors que l'inverse n'est pas vrai). Nous voyons par exemple que le véhicule T300_1 possède un modem appelé Modem1 qui est perturbé par le Modem1 de T300_2. Une démarche similaire est à effectuer pour déclarer les interférences avec certains capteurs (IsSensible = true). Il est à noter que nous déclarons également quels sont les moyens de communication qui peuvent communiquer entre eux. Par défaut, aucun lien de communication n'est possible. Dans le fichier que nous présentons, le Modem1 de T300_1 peut communiquer avec le Modem1 de T300_2. Cette approche nous permet de représenter le fait que certains moyens de communication sont interférents sans pouvoir toutefois échanger des données.

5.6 Conclusion

Dans ce chapitre nous avons présenté l'implémentation de notre système de simulation. Nous avons en particulier explicité le choix des différentes technologies que nous avons retenues et leurs implications sur le système conçu. Ces choix ont été guidés par les notions importantes que nous avions fait ressortir dans la première partie. Nous avons ainsi pu clairement respecter les contraintes que nous nous étions imposées, même si

certains aspects (que nous avons évoqués au long de ce chapitre) peuvent encore être améliorés du point de vue de l'implémentation.

Dans ce chapitre, nous avons également proposé un formalisme de description des véhicules mobiles, qui permet de les configurer rapidement de façon générique. Cette approche nous permet de garantir un niveau de réutilisabilité intéressant pour la communauté.

Chapitre 6

Modélisation

Dans les chapitres précédents, nous avons proposé une architecture nous permettant d'aborder la problématique de la simulation HIL ou hybride dans le contexte du multi-véhicule. Nous avons proposé un système distribué sur un réseau dédié et précisé la nature des liens et des données échangées entre ces différents blocs.

Concevoir un simulateur ne constitue pas une finalité en soi : il s'agit avant tout de développer un outil fonctionnel nous permettant de tester la pertinence des lois de commandes et de scenarii que nous proposons. Cet outil doit être développé avec suffisamment de finesse pour supporter la comparaison avec la réalité, sans toutefois prétendre la remplacer. Nous modélisons donc l'environnement, les robots et les communications au travers d'un ensemble de phénomènes considérés comme significatifs par la littérature, dans notre contexte. Les méthodes de modélisation que nous avons choisies impliquent certaines limitations de fonctionnement. Les hypothèses faites nous conduisent à définir un domaine d'exploitation dans lequel nous considérons nos résultats de simulation comme fiables, relativement aux hypothèses de simulation.

L'objet de ce chapitre est de présenter les modèles que nous avons développés (pour certains d'entre eux) et mis en œuvre dans l'architecture proposée (les modèles analytiques issus de la littérature sont donnés en annexe). Ce chapitre est scindé en quatres sections. Nous commençons par la modélisation des véhicules, puis nous introduisons la modélisation du monde. Finalement, après avoir présenté la modélisation des différents capteurs utilisés à bord de nos véhicules, nous achevons ce chapitre avec la modélisation des moyens de communication.

6.1 Les véhicules

6.1.1 Introduction

6.1.1.1 Choix des engins simulés

Dans le chapitre précédent, nous avons proposé une architecture capable de simuler conjointement plusieurs véhicules. Ces véhicules peuvent être hétérogènes de même classe (plusieurs AUVs différents par exemple) ou hétérogènes de classes différentes (un AUV

et un ASV par exemple). Nous avons choisi d'implémenter deux modèles correspondant à deux véhicules de classes différentes. Ce choix s'explique pour différentes raisons. La première est que le temps de thèse est limité.

Par ailleurs, nous ne possédons pas nécessairement l'ensemble des connaissances requises pour développer tous les modèles des véhicules dont nous ne nous servirons pas de toute façon dans un premier temps. Enfin, nous avons préféré nous focaliser sur les modèles des véhicules que nous possédons. En effet, avant d'étendre le simulateur à de nouveaux participants, nous souhaitons valider le concept en comité restreint. Nous avons donc choisi de développer des modèles pour trois véhicules autonomes. Il s'agit de nos deux AUVs *Taipan 2* et *Taipan 300*, et du catamaran autonome développé par l'équipe de Massimo Caccia (CNR-ISSIA), *Charlie*. En effet, un projet de collaboration entre nos deux équipes a été mis en place, visant à mener des expérimentations impliquant l'ensemble de ces véhicules. Nous avons donc implémenté deux modèles de véhicule : un premier pour la classe AUV et un second pour la classe ASV.

6.1.1.2 Présentation des véhicules

L'équipe de robotique sous-marine (RSM) du LIRMM a conclu plusieurs contrats relatifs à des opérations en mer, tels que l'étude des résurgences d'eau douce, la bathymétrie, l'inspection de zones côtières (détection de mines) et l'étude de la navigation en flottille. Pour mener à bien ces différentes missions, deux véhicules sous-marins ont été construits **Taipan II ou H160** et **Taipan 300**.

6.1.1.3 Modélisation des véhicules : différentes approches

Il existe plusieurs méthodes permettant de déterminer les paramètres hydrodynamiques basés sur une géométrie donnée. Parmi ces méthodes, on peut distinguer les méthodes dites analytiques, expérimentales, calculatoires et semi-empiriques.

Les méthodes *analytiques* permettent de déterminer les valeurs des paramètres du modèle grâce à la *théorie des bandes*. Cette théorie ne peut s'appliquer qu'aux corps minces et permet de calculer les coefficients hydrodynamiques (les masses ajoutées notamment) en utilisant les propriétés des sections 2D du corps. Cette théorie permet également d'approximer les autres coefficients des équations du mouvement comme l'amortissement. Une autre hypothèse impose que les sections du corps varient graduellement sur sa longueur et que les mouvements de l'objet se limitent à de petits angles. Sur la figure figure 6.1, les paramètres hydrodynamiques des sections 2D sont intégrées sur la longueur de l'objet pour obtenir un coefficient de masse ajoutée.

Réaliser une série d'expérimentations constitue une autre approche possible pour déterminer les coefficients hydrodynamiques d'un véhicule. Ces études incluent des tests en mer et en bassin des carènes (figure 6.2). Cependant ces études sont coûteuses car il est nécessaire de construire des modèles de véhicules à différentes échelles et de les tester dans un bassin équipé, également onéreux. Par ailleurs, il est difficile d'obtenir les termes de masse ajoutée et d'inertie à partir de tests réalisés en mer. Cette étape constitue davantage un moyen de validation de ces coefficients qu'un procédé pour les obtenir.

Une troisième approche permettant de déterminer les caractéristiques d'un écoulement autour d'un objet, consiste à utiliser des méthodes de calcul numérique. Cette approche, appelée MFN (Mécanique des Fluides Numérique) permet de résoudre numériquement

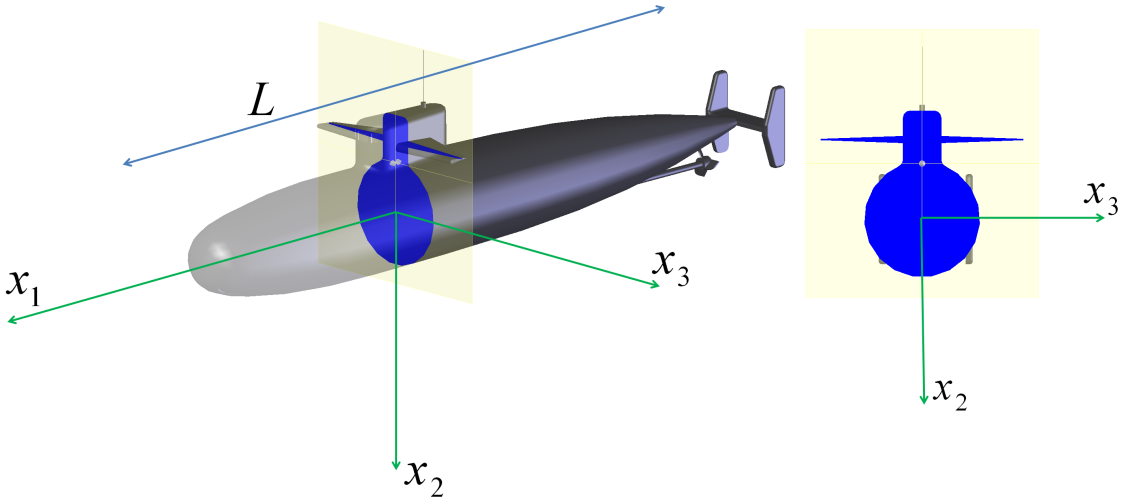


Fig. 6.1 – Théorie des bandes : pour estimer la masse ajoutée associée à la force exercée sur un corps mince en 3D dans la direction x_i due à l'accélération dans la direction x_j, on somme les contributions en 2D des sections du corps sur la longueur L.

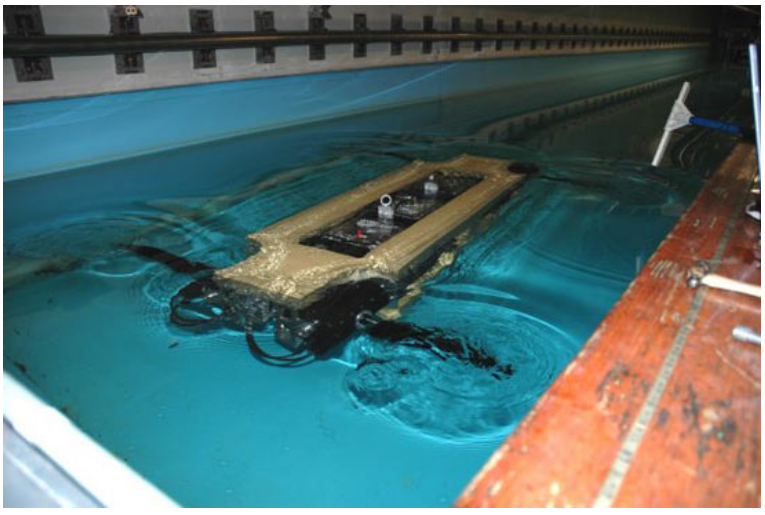


Fig. 6.2 – Test du RoboTurtle du MIT dans un bassin.

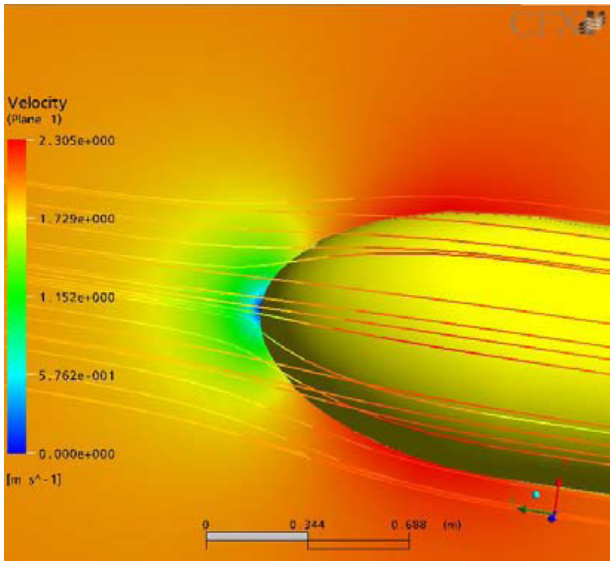


Fig. 6.3 – Profil des vitesses autour du nez de l'AUV Autosub [Alex Phillips]

les équations de Navier et Stokes en ayant recours à un ordinateur (figure 6.3). Ces programmes sont moins coûteux que des tests en mer ou en bassin. Cependant, ils nécessitent l'intervention d'un expert pour mailler le modèle et valider les résultats. Cette méthode permet de traiter n'importe quel type de trajectoire et apporte des résultats qui ne sont pas accessibles par l'expérimentation. La résolution d'un problème de MFN se fait généralement en trois étapes :

- définition de la géométrie du modèle, maillage et choix des méthodes numériques au regard des hypothèses retenues (fluide réél, fluide parfait, écoulement irrotationnel, ...)
- résolution numérique du problème en utilisant un programme informatique
- exploitation des résultats après en avoir vérifié la cohérence.

La MFN nécessite un temps de calcul très élevé et ne peut être mis en œuvre dans le contexte de la simulation temps réel.

Enfin, il est possible d'utiliser une méthode semi-empirique pour déterminer les paramètres d'un modèle. Cette méthode se base sur des données obtenues empiriquement à partir de formes géométriques génériques.

6.1.2 Modélisation des AUVs

Cette section est consacrée à la modélisation des AUVs. Les équations du modèle décrivent les lois qui régissent le comportement du véhicule dans l'espace (6 degrés de liberté). Elles modélisent ainsi deux aspects distincts : cinématique et dynamique. Les coefficients de notre modèle se basent sur une approche semi-empirique couplée à l'expérimentation. Cependant, l'obtention et l'identification des coefficients hydrodynamiques sortent du cadre de ces travaux de recherches et ne seront pas détaillées dans ce manuscrit. Le lecteur peut se référer à [63], [64] ou encore [65] pour obtenir des informations sur le calcul de ces coefficients.

6.1.2.1 Hypothèses du modèle dynamique des AUVs

Nous considérons que l'AUV simulé possède un centre de flottabilité et un centre de gravité distincts. La flottabilité est considérée comme non nulle. La vitesse axiale des AUVs étant de l'ordre de deux à dix mètres par secondes, nous admettons que l'effet du mouvement propre à la Terre sur la cinématique est négligeable. Nous considérons donc le repère terrestre comme étant galiléen. On admet que les forces hydrodynamiques ne dépendent que du module de la vitesse. La présence de tourbillons rend théoriquement cette hypothèse fausse. Néanmoins, dans les domaines usuels ces phénomènes peuvent être négligés [64]. Il est tout de même à noter que des tourbillons peuvent apparaître de manière significative à l'arrière de surfaces masquantes (par exemple des gouvernes à l'avant) induisant des effets sur les surfaces de contrôle arrière (cap, profondeur) avec un retard fonction de la distance les séparant et de la vitesse d'écoulement du fluide [66].

Par ailleurs les forces et moment dus à l'inertie et à la masse d'eau ajoutée d'une part et aux forces réactives dues aux frottements visqueux du fluide sur le corps (portance, traînée) d'autre part sont réputées de nature additive. Nous admettrons que la résultante des efforts hydrodynamiques est composée par la somme des efforts hydrodynamiques s'exerçant sur la coque et les surfaces de contrôle externes. Leur couplage est pris en compte en reprenant les travaux décrits dans [64]. Enfin, en ce qui concerne les actionneurs, nous ne considérons pas les effets dus à la masse et à l'inertie d'eau ajoutée car nous estimons qu'ils sont négligeables par rapport à ceux du corps du véhicule.

Le modèle utilisé est donné en annexe B.

6.1.2.2 Domaine de validité du modèle

Le domaine de validité du modèle présenté ici reste fortement lié à l'obtention et à la précision des nombreux coefficients hydrodynamiques le composant. Ces coefficients sont le fruit de plusieurs années d'expérimentations et de validation. Le modèle reste d'autant plus proche de la réalité que les manœuvres effectuées par le véhicule sont lentes et progressives (limitation des écoulements instationnaires turbulents à l'origine des tourbillons de décrochage) pour respecter notre hypothèse de départ.

Par ailleurs, la vitesse de déplacement par rapport au fluide est également limitée à une dizaine de mètres par seconde pour éviter les phénomènes de décrochage qui ne sont pas pris en compte. Enfin, ce modèle ne permet pas de simuler les phases de plongée du véhicule. En effet, les robots Taipan ont une flottabilité positive et il leur est nécessaire d'arriver à une certaine vitesse en surface requise pour la manœuvre de plongée (brusque inversion des gouvernes pendant un temps limité). Cette phase délicate est largement conditionnée par l'état de la mer, le cap du véhicule par rapport à la houle, le vent et comporte une part d'aléatoire qui ne peut être reproduite par ce modèle.

6.1.3 Modélisation des ASVs

Cette section est consacrée à la modélisation des ASVs. Les équations du modèle décrivent les lois qui régissent le comportement du véhicule dans l'espace (3 degrés de liberté). Nous reprenons ici le modèle exposé dans [67]. Ce modèle est celui qui est utilisé par le CNR pour modéliser le comportement de leur ASV Charlie.

6.1.3.1 Hypothèses du modèle dynamique des ASVs

Nous considérons que les mouvements du véhicules sont restreints dans le plan horizontal. Par conséquent, ne sont pas pris en compte le tangage, le roulis et le pilonement. Comme précédemment, nous considérons deux repères : le premier est attaché à la Terre mais considéré comme galiléen $< e >$ et un second lié au véhicule $< b >$. Les termes d'inertie et de traînée sont estimés à travers des données acquises par les capteurs de bord au cours de manœuvres particulières décrites dans [68]. La vitesse d'avancement du véhicule par rapport au fluide est estimée au regard de la vitesse de rotation des propulseurs et de l'angle des gouvernails.

Le modèle utilisé est donné en annexe B.

6.1.3.2 Domaine de validité du modèle

Dans B.28, le couple exercé par le gouvernail $\bar{n}^2\delta$ a été identifié comme une fonction de la vitesse de rotation du propulseur au lieu de la vitesse d'avancement du véhicule. Cela a pour conséquence que l'action du gouvernail lorsque le véhicule est encore en mouvement (mais avec le moteur stoppé), est considérée comme nulle. La limite de validité du modèle s'entend donc pour $\bar{n} > 0$. Par ailleurs cette approche nous impose une seconde limitation : la vitesse d'avance du véhicule doit suivre l'évolution temporelle de la vitesse de rotation de façon rapprochée. Cela interdit donc toutes variations brusques de la vitesse eu égard à l'inertie du véhicule. La limitation du problème dans le plan ne nous permet pas de prendre en considération les déplacements du véhicules dus à l'impact des vagues. Par conséquent l'utilisation de ce modèle s'entend sur un plan d'eau virtuel calme.

6.2 Modélisation du monde

6.2.1 Introduction

Dans la précédente section, nous avons présenté deux modèles de deux véhicules appartenant à des classes différentes. Nous avons vus que dans les équations de la dynamique de ces engins, certains termes représentaient les efforts exercés par l'environnement sur le robot. Ces efforts sont dus à un ensemble de phénomènes naturels (vagues, vent, courants, ...) et font partie du *monde*. Par ailleurs nous avons évoqué dans le chapitre 3 la nécessité pour les véhicules de pouvoir communiquer entre eux durant les phases de simulation. Un des moyens de communication passe par la diffusion d'une onde acoustique ou électromagnétique qui se propage dans le milieu. Cet aspect de la simulation fait donc également partie du monde simulé.

Dans cette section nous proposons donc dans un premier temps un modèle représentant la propagation des ondes dans le milieu, puis nous exposons les modèles relatifs aux phénomènes environementaux (vagues, vent, courants, salinité, ...). Enfin, nous abordons la modélisation des collisions véhicule/véhicule ou véhicule/environnement dans un troisième temps.

6.2.2 Propagation

6.2.2.1 Introduction

La coopération entre plusieurs véhicules présente un réel intérêt, mais il faut pour cela que les véhicules soient en mesure de communiquer entre eux. Il semble intéressant de pouvoir aborder cette problématique avec une vision "réseau" du système. En effet, la mise en réseau de ces robots augmente la capacité de propagation d'une information en permettant la communication entre plusieurs véhicules qui n'auraient pas pu communiquer en "direct".

Pour autant, créer un réseau dans un environnement sous-marin avec des véhicules mobiles et des liens acoustiques de qualité et de portée variable constitue un réel enjeu scientifique [69]. La complexité du problème augmente encore si le réseau devient hybride : il est possible d'utiliser des bouées munies de liens radio et acoustique permettant de propager des informations sur de longues portées à des débits plus élevés (qu'en pure communication acoustique).

Le problème est alors de développer un logiciel capable de déterminer comment transporter les données d'un point à un autre du réseau de façon optimale ou du moins avec certaines garanties (qualité de service). Ces logiciels, appelés protocole de routage ont déjà été largement développés (et font encore l'objet d'études) pour les réseaux aériens. Malheureusement, les propriétés du medium de propagation pour les liaisons acoustiques sont très éloignées des réseaux aériens, et il est nécessaire de développer de nouveaux protocoles pour ce type de propagation. Tester un protocole de routage nécessite un nombre d'essais très important. Les coûts et la difficulté de ces essais rendent nécessaire le développement d'un simulateur permettant de définir, tester et évaluer ces nouveaux protocoles. De nombreux simulateurs dédiés offrant cette fonctionnalité existent mais aucun ne prend en compte la dynamique des véhicules.

Dans [50], l'auteur travaille sur l'élaboration de tels protocoles mais souhaiterait pouvoir intégrer la dynamique des nœuds (véhicules, bouées, ...) du réseau dans son simulateur. A l'évidence un rapprochement des communautés doit s'opérer au travers d'une plate-forme de travail commune afin de franchir une étape dans la problématique de la coordination de véhicules.
Deux types de propagation sont à considérer : la propagation des ondes acoustiques et la propagation des ondes électromagnétiques. Les communications radio permettent dans beaucoup d'applications de remplacer le filaire par le sans fil permettant ainsi de se déplacer sans perdre la connexion en cours. Cependant, ces communications sans fil sont fortement dépendantes du milieu de propagation. Ainsi la propagation des ondes radio est d'autant plus atténuée que la densité du milieu de propagation augmente. Les ondes radio se propagent donc bien dans l'air ou dans l'espace ; en revanche, leur utilisation est fortement limitée en milieu aquatique.

En effet, ce milieu est très sélectif par rapport aux longueurs d'onde qui le traversent. Il n'est guère possible que d'utiliser des ondes VLF (Very Low Frequency) ou ELF (Extreme Low Frequency) de l'ordre de quelques Hz. Par conséquent, le débit binaire atteignable est très faible (de l'ordre de quelques bits), la taille des antennes est d'autant plus élevée que la longueur d'onde utilisée est faible, et enfin la puissance requise pour émettre est très importante (de l'ordre de quelques megawatts). Aux USA il existe des centres opérationnels, situés dans le Michigan et le Wisconsin, qui permettent de communiquer avec des

sous-marins en plongée à n'importe quelle profondeur en ayant recours aux ondes VLF.

La lumière est également une onde électromagnétique, dont la longueur d'onde est comprise entre 10^{-3} et 10^{-9}. Il est très simple d'observer que la longueur d'onde correspondant au rouge (à partir de 600nm) est fortement absorbée alors que la lumière bleue (en dessous de 500nm) se propage plus loin. De ce fait, une autre solution pour communiquer sous l'eau a été envisagée : l'utilisation de laser bleu-vert (environ 480nm). Cependant la nécessité de pointer correctement l'émetteur, et la présence de particules en suspension dans l'eau, limite son utilisation à une centaine de mètres. L'utilisation d'ondes radio ou de laser n'est donc pas adaptée à cause de l'atténuation par le milieu. A ce jour, la transmission de signaux acoustiques constitue le seul moyen de communiquer sans fil en milieu aquatique.

On utilise donc des ondes acoustiques car l'énergie se propage grâce aux vibrations du milieu. Par conséquent, les signaux acoustiques se propagent d'autant mieux et d'autant plus rapidement que la densité du milieu est grande. La gamme de fréquences utilisée varie en général de 30Hz à 1,5MHz [70]. La couche d'eau se comporte comme un guide d'onde dans laquelle l'énergie sonore est réfléchie aux niveaux des interfaces (surface et fond marin principalement). Cette propagation s'effectue sur des centaines de Km et n'est en aucune sorte affectée par les particules en suspension. Cependant la variabilité des caractéristiques physiques du milieu (température, vagues, salinité, pression etc...) affecte la qualité des canaux de communication aquatiques. Seul un débit relativement faible de quelques Kb/s est possible à obtenir. De même que les ondes électromagnétiques, l'atténuation des ondes acoustiques est principalement due à trois phénomènes : l'absorption (phénomène par lequel une partie de l'énergie est dissipée dans un milieu matériel), la diffusion (phénomène par lequel un faisceau est dévié dans de multiples directions) et la dispersion (étalement de l'onde pendant la propagation). Les ondes acoustiques sont donc adaptées aux communications aquatiques mais présentent toutefois quelques limitations :

- une faible vitesse de propagation ($1500m/s$ environ)
- la propagation par trajets multiples pouvant générer des interférences
- une faible capacité des canaux acoustiques entraînant un faible débit binaire.

6.2.2.2 Hypothèses du modèle

L'étude présentée en annexe C, montre qu'un nombre conséquent de phénomènes entrent en jeu dans la propagation d'une onde acoustique en milieu aquatique. Notre but n'est pas ici de créer un modèle parfait, au sens où il reproduirait avec précision l'ensemble de ces phénomènes. Cela est impossible pour plusieurs raisons :

- Tout d'abord l'objectif de notre simulateur est de pouvoir fonctionner en temps réel. Cela signifie qu'il n'est pas possible avec les moyens techniques "grand public" dont nous disposons d'implémenter un tel modèle en respectant ce critère.
- Des simulateurs reproduisant finement la propagation des sons dans l'eau existent déjà, et il ne s'agit pas de ré-inventer l'existant.
- Aucun des membres de notre équipe ne possède de connaissances avancées en acoustique sous-marine qui nous permettrait de créer un tel simulateur.

Par conséquent notre but est de proposer un modèle compatible avec les contraintes de temps-réel que nous nous sommes imposés mais qui soit capable d'émuler (c'est-à-dire de reproduire les conséquences des phénomènes sur la propagation sans en modéliser les causes) les principaux phénomènes de la propagation acoustique [71].

Nous considérons donc un modèle dans lequel la propagation des ondes est sphérique. Cette approximation est grossière car la propagation est fortement dépendante du diagramme de rayonnement de l'antenne acoustique utilisée. Nous abordons ce problème dans le paragraphe C.2.3. Un effort devra être fait sur cet aspect qui ne peut être négligé. Nous considérons une propagation en ligne droite : les phénomènes de multi-trajet sont ignorés. Si cette hypothèse peut sembler forte de prime abord, il faut cependant prendre en compte le fait que nous utilisons notre modem acoustique en mode *sûr*. Dans ce mode le débit est fortement limité (environ 20 bits/s) car de nombreux codes correcteurs d'erreur sont insérés. Par conséquent même si les phénomènes de multi-trajet sont présents, on peut les ignorer car ce débit est garanti. Nous tenons compte de l'atténuation liée à la distance parcourue par une onde, à la puissance d'émission et à la sensibilité en réception. Nous considérons donc qu'un message est délivré si son niveau de puissance est supérieur au seuil de sensibilité du récepteur au point de rencontre. Notre modèle tient compte du bruit lié à l'activité humaine et au vent. La vitesse de propagation est fixe et nous ne tenons pas compte de l'effet Doppler. Enfin nous considérons les interférences et supposons qu'aucun des messages reçus simultanément par un véhicule n'est compris par le véhicule récepteur.

Ce modèle permet de prendre en compte :

- tous les phénomènes liés au retard dans la réception d'un message (vitesse de propagation lente)
- le fait qu'un message n'est pas délivré instantanément au vu de la très faible bande passante
- le fait qu'un message ne soit pas distribué à l'ensemble des participants (à cause de l'atténuation, de la puissance d'émission, de la sensibilité de réception, du bruit, du diagramme de rayonnement de antennes)
- le fait qu'un véhicule ne puisse pas recevoir un message pendant qu'il émet et *vice versa* (ceci ne fait pas vraiment partie de la propagation mais plutôt du fonctionnement des moyens de communication ; nous en parlons ici par commodité mais ce point sera abordé plus tard)
- le fait qu'un véhicule qui perçoit plusieurs messages à la fois n'en reçoit aucun
- le fait qu'un véhicule doit nécessairement se trouver dans la zone de réception pendant toute la durée de la transmission pour recevoir un message
- le fait qu'un véhicule puisse émette à nouveau alors qu'une onde qu'il a précédemment émise est toujours en cours de propagation ; ce point peut paraître trivial mais nécessite de créer dynamiquement ce que l'on appelle des *contextes*
- le fait que l'on puisse définir quels sont les moyens de communications interférents (très utile pour le debuggage notamment)
- le fait que l'on puisse définir des interférences entre certains capteurs et certains moyens de communication

Notre apport porte donc sur un "moteur de distribution des messages" : il permet au travers d'un ensemble de modèles, de déterminer quels sont les véhicules à qui un message doit être délivré et le moment où cette distribution doit avoir lieu. Finalement l'aspect "simpliste" des modèles utilisés ne joue que sur les temps et éventuellement sur la possible délivrance d'un message à un véhicule. Une fois ce moteur créé, il est maintenant très simple de faire évoluer les modèles si le besoin s'en fait sentir. Mais *Thetis* est le premier simulateur à supporter la distribution des messages en tenant compte de ces

paramètres ; tester les stratégies de commande existantes avec ces contraintes constitue une première approche pour en évaluer les performances.

6.2.2.3 Modélisation

Prise en compte de la couche physique

Nous présentons ici un résumé des équations utilisées dans notre modèle que nous avons par ailleurs détaillées dans le paragraphe précédent :

- Vitesse du son : fixe $c = 1449.2m/s$
- Durée d'émission : $\delta_e = \frac{d}{c}$
- Délai de réception : $\delta_t = \frac{d}{c} + \frac{S}{D}$, où c est la vitesse du son, d la distance la plus courte séparant l'émetteur du récepteur, S la taille du message en bit, et D le débit d'émission en bit/s. Le message est intégralement délivré au récepteur au bout d'un temps représentant le temps de propagation additionné à la durée d'émission.
- Modèle de bruit : Wenz équation C.5
- Pertes de transmission : $TL = g(f) + \frac{d}{1000} + TL_d$ avec g défini à l'équation C.3, f la fréquence en KHz, TL_d le terme de dispersion défini à l'équation C.2
- Seuil de réception : $SL - TL - NB > seuil$ avec SL le niveau sonore à l'émission, TL les pertes de transmission, NB le niveau de bruit spectral isotropique rapporté à la bande passante du récepteur, *seuil* le niveau de réception au dessus du bruit ambiant minimum pour qu'une onde soit perçue par le récepteur.

Prise en compte des zones interférentes

Nous proposons un modèle dans lequel les interférences de communication sont représentées à travers un modèle logique et non mathématique. En effet, il est à la charge de l'utilisateur de définir au préalable quels sont les relations d'interférence entre les moyens de communication et les capteurs acoustiques. Le modèle consiste donc à vérifier qu'un moyen de communication en état de réception n'est pas atteint par une ou plusieurs ondes acoustiques ayant un niveau d'intensité suffisant pour être perçues par ce moyen de communication. Si tel est le cas, nous considérons qu'aucun des messages (le message qui était en cours de transmission, et le ou les suivants) n'est perçu par le moyen de communication.

Prise en compte des phénomènes de bord

Le modèle que nous proposons permet de prendre en compte les phénomènes de bord : pour qu'un message puisse être délivré à un véhicule, celui-ci doit se trouver à portée de réception pendant toute la transmission. Nous avons essayé de représenter en deux dimensions (figure 6.4)les différents cas pouvant survenir.

On considère deux AUVs dans le plan, dont un est immobile. Pour que le second soit en mesure de recevoir un message il faut que plusieurs conditions soient remplies :

- le véhicule récepteur doit se trouver dans la zone de réception au moment où l'onde acoustique émise par l'AUV émetteur lui parvient (voir cas (3) et (4) sur la figure). Il est à noter que l'AUV récepteur peut se trouver hors zone de réception lorsque l'AUV émetteur commence à émettre ; pour autant, eut égard au temps de propagation, il peut recevoir le message s'il a atteint la zone de réception au moment où il commence à percevoir l'onde (voir cas (1) et (2) sur la figure).
- Le véhicule ne doit pas quitter la zone pendant la transmission (voir cas (5) et (6) par exemple).

Bien entendu il ne s'agit là que d'une représentation. Notre modèle est capable de trouver une solution pour une scène 3D dans laquelle plusieurs véhicules sont en mouvement

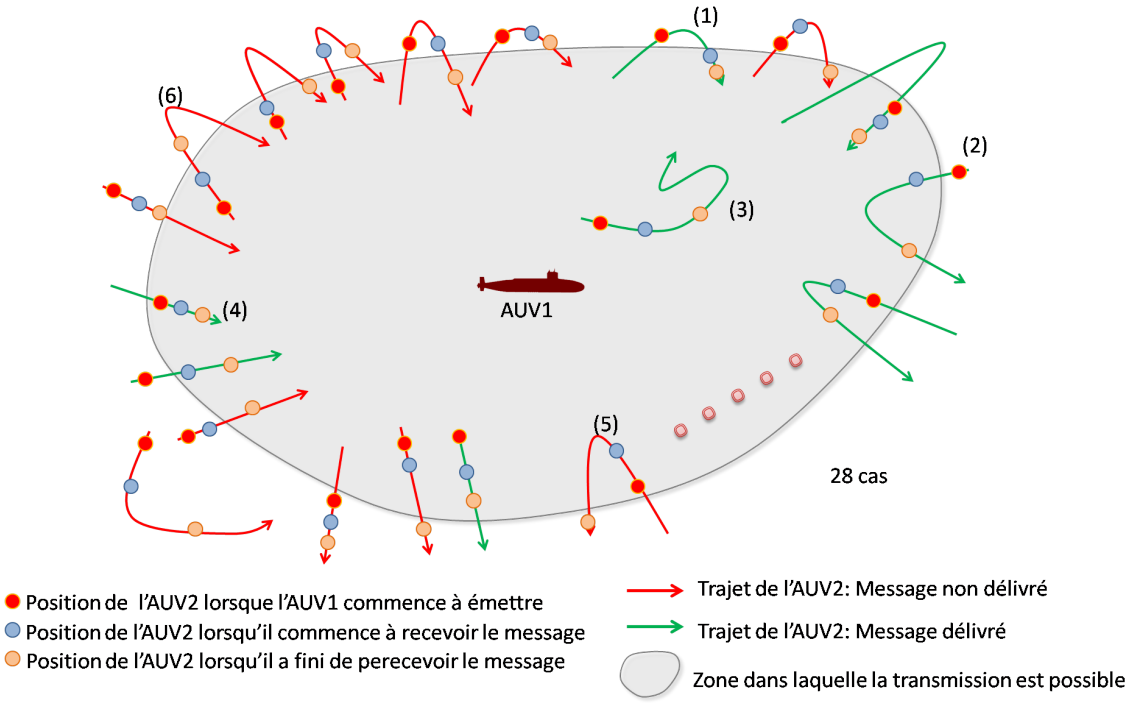


Fig. 6.4 – Phénomènes de bord : le message n'est délivré à un véhicule que s'il reste dans la zone de réception pendant toute la durée de la transmission

les uns par rapport aux autres.

6.2.2.4 Domaine de validité

Comme nous l'avons évoqué précédemment, il faut bien distinguer l'émulation de la couche physique qui consiste à reproduire les effets sur l'onde dus aux phénomènes de propagation, des modèles qui permettent de quantifier ces effets. En ce qui concerne la partie émulation, le modèle que nous proposons est capable de reproduire les principaux phénomènes réels (atténuation, interférences, latence de propagation, perception). Cette partie restera valable quand nous implémenterons la propagation des ondes électro-magnétique.

Ce modèle nous semble suffisant car il est sans doute plus pessimiste que la réalité pour nombre de phénomènes (interférences et limite de perception notamment). Pour d'autres, en revanche, le modèle implémenté est très optimiste (diagramme de rayonnement des antennes considéré comme omnidirectionnel, débit en réception constant sans prise en compte des phénomènes de multi-trajet). Il sera nécessaire de prendre en considération ces points dans un futur développement si l'on veut utiliser le mode "frequency hopping" du modem (mode dans lequel l'emission se fait à 100 bit/s sans garantie de réception). Pour ce faire on pourra implémenter une fonction de probabilité de réception en fonction du nombre et du niveau de puissance des rebonds perçus par le récepteur.

Enfin il est nécessaire que la vitesse des véhicules soit faible devant celle du son puisqu'on néglige l'effet Doppler. Cela semble être largement le cas pour les robots que nous utilisons. Enfin le fait que nous considérions une propagation rectiligne ne constitue pas une hypothèse forte dans notre contexte. En effet, si l'on regarde la figure C.13 on voit

qu'entre 0 et 100m (profondeur de travail max pour nos véhicules) il n'existe pas réellement de zones d'ombre acoustique dans lesquelles un robot ne percevrait rien. Ceci est d'autant plus vrai si l'on considère la limite de portée des modems que nous sommes susceptibles d'utiliser (10km au plus). On se situe donc dans la zone rouge en haut à gauche et notre approximation peut être considérée comme juste dans ce domaine.

Cette approche nous permet de tester et valider des protocoles de communication en distinguant les effets de chaque phénomène. En effet, ce modèle permet de déterminer qu'elle est la robustesse d'un protocole face aux différents facteurs (bruit, temps de propagation, multi-trajet, atténuation,...) et d'en élire un en fonction des conditions opérationnelles (faible fond, milieu pélagique, flottille rapprochée, pluie, ...). On peut artificiellement dégrader ou améliorer le milieu de propagation virtuel afin d'adapter le modèle à la phase de développement (conception préliminaire du protocole, limite de performance,...).

6.2.3 L'environnement

6.2.3.1 Introduction

L'environnement est composé de

- l'ensemble des phénomènes climatiques (vent, vague, courant, ...)
- l'ensemble des phénomènes physiques (salinité, température, polluants, ...)
- le fond de l'océan, le sol terrestre, une calotte glacière
- des objets statiques (pipeline, mines, ...) ou dynamiques (poisson, nageur, ...) animés par des scripts

Au-delà des stratégies de commande qui constituent notre principal intérêt, modéliser un environnement complet permet d'envisager des scenarii plus complexes. Par ailleurs, il s'agit d'un élément essentiel de la phase de test et de validation des lois de commande puisqu'il permet d'ajouter des perturbations dont il faudra tenir compte lors des expérimentations réelles. Enfin, simuler un environnement est un des points essentiels qui permettent de qualifier un simulateur d'hybride (voir paragraphe 1.6). L'approche décrite dans [18] semble intéressante puisqu'elle utilise des fichiers au standard *netCDF*, qui sont formats de données indépendant autorisant la création, l'accès et le partage de tableaux de données scientifiques (utilisés notamment en climatologie et de façon générale pour les données géo-référencées).

Nous avons ainsi développé trois modèles nous permettant de valider notre concept. Il s'agit d'un modèle de salinité, de courant et de fond océanique. Le modèle de salinité va nous permettre de tester l'emploi de capteur extéroceptif au sein de notre architecture, le fond de l'océan nous permet de tester des algorithmes de *ray tracing* impliqués dans le représentation sonar et de valider l'architecture en ce qui concerne les collisions véhicule/environnement. Enfin, la modélisation du courant nous permet de valider l'architecture en ce qui concerne l'envoi du vecteur force environnementale au simulateur de véhicules. Pour ce faire, nous avons implémenté le standard netCDF pour décrire l'environnement.

6.2.3.2 Hypothèses du modèle

Une hypothèse forte est que nous considérons que ces modèles n'évoluent pas au cours du temps. Il est à noter que cela évoluera si besoin très facilement car nous utilisons le format netCDF qui supporte la datation des données. Si cela est évident en ce qui concerne le fond océanique, il n'en va pas de même avec le courant qui peut varier au gré des marées et de la salinité qui évolue dans le temps aux alentours d'une source d'eau douce par exemple. Ces modèles font l'objet de recherches actives dans le domaine [72] et nous n'avons pas la prétention de développer de tels modèles. Une fois encore une coopération est à envisager. Le caractère stationnaire de nos modèles nous engage à davantage les qualifier de représentations de données géo-référencées que de modèles à proprement parler.

6.2.3.3 Modélisation

Le fond océanique est représenté sous la forme d'une élévation négative à partir de la surface de l'océan. Dans cette représentation, un point est d'autant plus profond que sa valeur absolue est grande :

$$f(x, y) = z$$

avec f la fonction de distribution, x et y les axes d'un repère lié à la Terre et donc l'origine se situe au niveau de la surface de l'océan, et z la distance séparant la surface de l'océan du fond de la mer aux coordonnées (x, y). Cette modélisation nous permet de tester des algorithmes de suivi de fond par diffraction acoustique [71]. En ce qui concerne la salinité et le courant, une représentation similaire a été adoptée :

$$f(x, y, z) = v$$

avec v la valeur de la salinité ou du courant aux coordonnées (x, y, z).

6.2.3.4 Domaine de validité

Le principal reproche que l'on peut faire à ces modèles est leur caractère stationnaire. Cela exclut donc toute exploitation des résultats lorsque les phénomènes à observer possèdent une dynamique. Par ailleurs, comme nous l'avons précédemment évoqué notre travail dans ce domaine n'est pas encore suffisamment abouti. Ces représentations sont actuellement en cours d'implémentation et ont été temporairement abandonnées afin de consacrer le temps à la rédaction de ce manuscrit. Un travail supplémentaire sera nécessaire pour implémenter la possibilité de représenter des objets statiques ou dynamiques.

6.2.4 Les collisions

6.2.4.1 Introduction

Il existe de nombreux modèles décrivant la physique des collisions. Un état de l'art dans le domaine constituerait un point de départ nous permettant de déterminer quel est le modèle le plus à même de satisfaire nos exigences. Néanmoins, nous avons implémenté un modèle basique nous permettant de prendre en considération les collisions véhicule/véhicule et véhicule/environnement.

6.2.4.2 Hypothèses du modèle

On considère que du point de vue du modèle de détection de collisions, tous les véhicules sont des sphères dans lesquelles sont inscrits les véhicules.

6.2.4.3 Modélisation

Connaissant la position des véhicules et leur plus grande dimension, on vérifie que deux sphères ne s'inter-pénètrent pas. Considérons deux sphères S_1 et S_2 de rayons différents : respectivement R_1 et R_2. Ces deux sphères ont pour centre les coordonnées de deux véhicules : $V_1(x_1, y_1, z_1)$ et $V_2(x_2, y_2, z_2)$. Appelons d la distance séparant les deux centres des sphères : $d = V_1V_2$. La condition pour vérifier l'absence de collision entre les deux sphères est $d > R_1 + R_2$. Dans tous les autres cas, il y a collision et le vecteur d'état à l'instant d'avant la collision des véhicules est envoyé au simulateur de véhicules. Par ailleurs une notification est envoyée au superviseur. Toutes les positions des véhicules sont testées deux à deux (par exemple si la simulation comprend quatre véhicules, on teste (1,2) ;(1,3) ;(1,4) ;(2,3) ;(2,4) ;(3,4)).
En ce qui concerne les collisions véhicule/environnement, elles s'opèrent en deux temps. Tout d'abord on sélectionne un ensemble E de points tel que :

$$E = \{(x, y)/(x < |x_1 + 2R_1|) \cap (y < |y_1 + 2R_1|)\}$$

puis nous testons l'appartenance de chacun de ces points à la sphère d'un véhicule ; une collision a lieu si

$$\left\{ (x_T - x_n)^2 + (y_T - y_n)^2 + (z_T - z_n)^2 \leq R_n^2 \quad n = 1..N, (x_T, y_T) \in E \right.$$

avec (x_T, y_T, z_T) le points à tester, N le nombre de véhicules. En cas de collision, on procède comme exposé pour les collisions inter-véhicules.

6.2.4.4 Domaine de validité

Ce modèle ne représente pas la réalité. En effet, les véhicules sont considérés comme des sphères et l'évolution du choc consiste uniquement à revenir à l'état précédent le choc. Néanmoins ce modèle à l'avantage de nécessiter peu de calcul et nous semble suffisant dans un premier temps. En effet, si les véhicules sont amenés à naviguer aussi près du sol marin ou les uns des autres, on peut considérer qu'il y ait de fortes chances pour que dans la réalité ils entrent en collision. La mission est alors considérée comme un échec. Nous n'avons donc pas besoin d'un modèle beaucoup plus fin. Par ailleurs des librairies permettant de détecter ces collisions et simuler la physique du choc existent. On pourra citer à titre d'exemple *ODE* [73] qui est une librairie haute performance permettant de simuler les corps rigides.

6.3 Les capteurs

6.3.1 Introduction

Le contrôleur d'un véhicule autonome se base sur les informations issues des capteurs pour élaborer les commandes à envoyer aux actionneurs. Le comportement et les performances de ces capteurs ont donc un impact important sur le comportement global du

véhicule. Parmi les caractéristiques influentes, on peut citer le taux d'échantillonnage, la sensibilité au bruit, le temps de réponse et la plage de mesure. Par ailleurs la réponse élaborée par certains de ces capteurs est liée à son emplacement sur le véhicule. Ainsi la distance renvoyée par un écho-sondeur différera d'un autre disposé à un autre emplacement et avec une autre orientation. Les véhicules autonomes sont en général équipés de plusieurs capteurs leur permettant de se déplacer dans un environnement plus ou moins connu. Il existe deux types de capteurs : les proprioceptifs, c'est-à-dire ceux qui renvoient une information sur l'état du système robotique (position, orientation, ...), et les extéroceptif qui servent à percevoir l'environnement (caméra, sonar, CTD, ...).

Comme nous l'avons évoqué au paragraphe 6.1.1.2, les véhicules que nous utilisons possèdent un nombre important de capteurs. Modéliser l'ensemble de ces capteurs sort du cadre de cette thèse et nous avons été amené à n'en sélectionner qu'un certain nombre. Les capteurs que nous avons choisi de modéliser sont ceux qui sont nécessaires au véhicule pour se déplacer. Ainsi nous avons retenu le capteur de pression et la centrale inertielle pour les AUVs, et un GPS pour les ASVs. Par ailleurs nous avons modélisé un capteur extéroceptif simple, en l'occurrence un CTD, afin de valider certains aspects de l'architecture que nous avons proposée et sur lesquels nous reviendrons plus tard.

Développer des modèles précis de capteurs sort également du cadre de cette thèse : il s'agit avant tout d'un travail d'ingénierie (mesurer *in situ* les caractéristiques du capteur afin de fournir des paramètres valides au modèle) et de développement (reproduire les trames envoyées par un capteur) pour la plupart d'entre eux. Ces propos sont toutefois à nuancer puisque certains capteurs, tels que les sonars, font l'objet de recherche ; la modélisation de cette classe de capteurs sort également du cadre de cette thèse. Cette remarque nous a donc amené à développer des modèles basiques de capteur.

6.3.2 Hypothèses

Plusieurs caractéristiques des capteurs sont prises en compte dans les modèles que nous utilisons. Ainsi, nous considérons le taux d'échantillonnage et les plages de mesure pour chacun d'entre eux. Nous posons l'hypothèse de non perturbation du capteur que ce soit par des phénomènes externes (champ électromagnétique entre autres), ou internes (dérive thermique, ...). On considère pour chacun de ces capteurs un bruit gaussien additif ou multiplicatif. Enfin, les modèles implémentés ne tiennent pas compte de l'emplacement et de l'orientation du capteur à bord du véhicule.

6.3.3 Modélisation

La modélisation des capteurs consiste juste à ajouter un bruit gaussien aux valeurs obtenues à la sortie du simulateur de véhicules (pour les capteurs mesurant les attitudes, position, vitesses de l'engin) ou à la sortie du simulateur d'environnement (pour les capteurs extéroceptifs). Le simulateur de capteur se charge alors de faire parvenir ces valeurs bruitées aux véhicules concernés (c'est-à-dire à ceux disposant du capteur en question) et au bout d'un temps déterminé (correspondant au taux d'échantillonnage).

6.3.4 Domaine de validité

Ces modèles ne sont pas précis et il faudra sans doute les améliorer. En effet, nous considérons un bruit blanc et constant alors que pour certains capteurs (en observant les courbes de réponse du capteur de pression notamment) cette hypothèse est fausse. Par ailleurs, ne pas considérer la position et l'orientation du capteur à l'intérieur du véhicule fournit des résultats approximatifs. En effet, si l'on considère le capteur de pression de Taipan 300 qui se trouve dans la queue, on constate à l'évidence que la mesure de pression va être fonction du tangage, de l'emplacement du centre de rotation et de la longueur de l'engin. Le modèle implémenté est donc valable pour des angles de tangage faibles.

De même la réponse de la centrale inertielle est dépendante de son emplacement et de son orientation dans le véhicule ; néanmoins, la centrale inertielle de nos robots étant alignée et orientée avec le repère du véhicule, le modèle reste pertinent. En ce qui concerne le GPS, le modèle implémenté ne tient pas compte de la visibilité et du temps d'acquisition des satellites. Ceci constitue une différence majeure avec la réalité puisqu'il faut parfois patienter plusieurs minutes afin de pouvoir acquérir une position.

Il est donc nécessaire de faire évoluer ces modèles dans un avenir proche afin de prendre en compte ces aspects qui ne sont pas négligeables et influent fortement sur le comportement des véhicules et au delà, sur le déroulement du scenario.

6.4 Les moyens de communications

6.4.1 Introduction

Offrir la possibilité aux véhicules de communiquer au travers du simulateur ne se limite pas à modéliser les phénomènes de propagation des ondes. En effet, avant de se propager dans le milieu, l'information est traitée par un moyen de communication qui possède des caractéristiques qui lui sont propres. Ces moyens de communications possèdent notamment (au moins pour les plus sophistiqués d'entre eux) un comportement logique et un comportement temporel. Le comportement logique fait référence à tout ce qui concerne les différents états et les conditions permettant de passer d'un état à un autre. Le comportement temporel concerne les délais pour opérer ces changements d'état. Modéliser ces comportements est essentiel dans notre contexte car les conséquences sur les échanges entre les véhicules sont considérables.

La problématique de la simulation des moyens de communication sophistiqués est un sujet actuellement traité dans le domaine de la recherche. En effet, devant la complexité des traitements à effectuer généralement en temps réel par ces moyens de communication, des simulateurs dédiés aux tests de ce type de matériel sont développés [74]. Dans [75], les auteurs développent un modem reconfigurable appelé *rModem*. Ce modem est basé sur un DSP (Digital Signal Processor). L'emploi de ce type de processeur permet de développer et tester les algorithmes de traitements sous Matlab, puis de générer et envoyer le code temps-réel correspondant sur la carte du modem. Bien que cette méthode offre une souplesse indéniable, une phase de test en mer est toujours nécessaire pour valider les algorithmes implémentés.

Un simulateur HIL de modem pourrait prendre ici tout son sens. On pourrait envisager de connecter la sortie analogique du modem à une carte d'acquisition du simulateur

d'environnement. Ce dernier pourrait alors reproduire les effets des phénomènes environnementaux précédemment évoqués sur le signal en temps réel avant de le renvoyer sous forme analogique (en passant par la carte d'acquisition) au modem des véhicules concernés. Si cette approche pourrait constituer un aboutissement dans la prise en compte des comportements des moyens de communication, nous avons privilégié dans le cadre de nos recherches une approche plus ordinaire.

Un véhicule autonome possède en général plusieurs moyens de communication. Cela est principalement dû au fait qu'il doit être en mesure de propager de l'information dans différents milieux. Ainsi, un AUV utilisera son modem acoustique pour diffuser un message en milieu sous-marin mais utilisera sa radio pour communiquer en milieu aérien. Par ailleurs un véhicule peut disposer de plusieurs moyens de communication adapté à la nature des informations à échanger. Un liaison WIFI permettra par exemple d'échanger de gros volumes de données, mais la portée reste limitée. A contrario, une liaison radio UHF permettra d'échanger peu d'information (typiquement une mission, ou des signaux de télécommandes, localisation, ...) mais à grande portée.

Le contexte de nos recherches nous ont naturellement mené à nous focaliser en premier lieu sur la modélisation des moyens de communication dédiés au domaine aquatique : le modem acoustique.

6.4.2 Le modem acoustique

6.4.2.1 Introduction

Transducteurs aquatiques

Un élément indispensable à tout appareil de détection sous-marine ou de télécommunication utilisant l'acoustique est le transducteur électroacoustique. Il permet la réception et l'émission des sons dans l'eau en transformant l'énergie électrique en énergie acoustique (et vice versa). On distinguera le *projecteur* utilisé pour émettre des ondes acoustiques de *l'hydrophone* qui sert uniquement à l'écoute de ces mêmes ondes.

Une courte étude des transducteurs abordant le principe de directivité des antennes est donné en annexe C.

Le modem TAPAC

Ce matériel a été développé par la société ORCA en 1999. TAPAC signifie Télétransmission Acoustique Pour AUVs Côtiers (figure 6.5). Il s'agit d'un système de transmission acoustique bidirectionnel entre le fond et la surface permettant de transmettre des données et de mesurer la distance séparant les deux équipements.

Ce modem utilise une technique de transmission fiable basée sur l'émission de signaux modulés linéairement en fréquence (chirp) mais offrant un débit très faible (20 bits/s codage inclus). Un second mode de transmission moins robuste que le premier est également utilisable Il s'agit d'une modulation appelée "frequency hopping" associée à un codage convolutif. Ce type de modulation permet d'établir des liens à 100 bits/s. Un mode permettant de basculer d'un mode à l'autre est disponible et permet d'utiliser le mode sécurisé tant qu'une certaine quantité de données accumulées dans le buffer et paramétrée par l'utilisateur n'a pas été atteinte.

Comportement logico-temporel

Ce modem possède deux modes de fonctionnement. Le premier qui est dit "transparent" consiste à envoyer les caractères un par un (donc octet par octet). Le second appelé

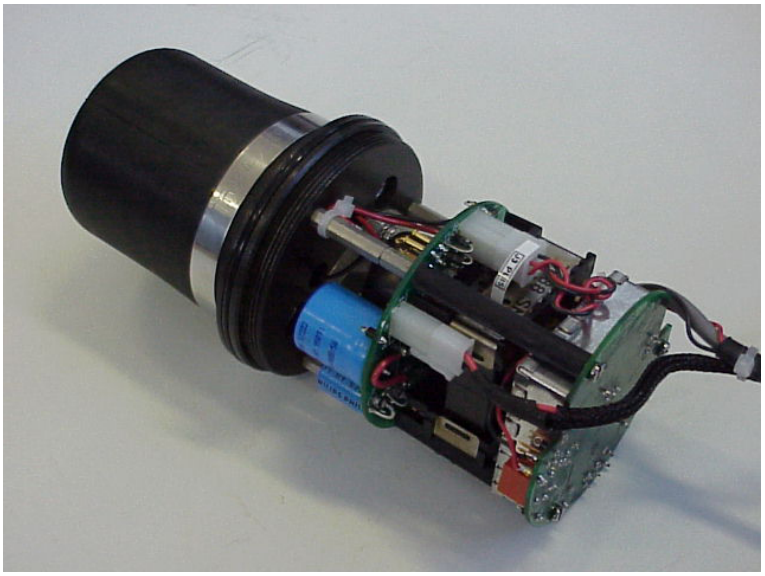


Fig. 6.5 – Système de transmission de données TAPAC

"data logger" envoie les caractères par paquet de 14 octets. Ce second mode est plus robuste que le premier car il intègre des CRC (codes correcteur d'erreurs). Quel que soit le mode utilisé, le modem commence à émettre, une fois que le seuil (fixé par l'utilisateur) de quantité de données (stockées dans sa FIFO (First In First Out)) est atteint.

Lorsque le modem n'est pas utilisé, celui-ci est en mode sommeil. Un signal de 300ms est envoyé par l'émetteur afin de le réveiller. Une fois la réception terminée et si il n'a aucun émission programmée, le récepteur reste en écoute pendant une période fixée par l'utilisateur puis retourne en mode sommeil.

Le modem peut émettre à deux niveaux de puissance (174 dB ou 177 dB). Outre la possibilité de pouvoir transmettre des données, il est possible d'utiliser ces modems en mode fréquencemètre.

6.4.2.2 Hypothèses du modèle

Le modem est capable de fonctionner de plusieurs façons différentes, en combinant les modes précédemment évoqués. Nous n'avons implémenté qu'un seul mode : transparent en modulation chirp. Cela signifie que les caractères sont envoyés un par un et que le débit d'émission est de 20bit/s. Ce mode correspond à celui que nous utiliserons lorsque nous serons à même de faire des expérimentations. Actuellement nous ne modélisons pas la file d'attente du modem. Par conséquent c'est au contrôleur du véhicule de segmenter le message à émettre en caractères avant de les faire parvenir au modem. La deuxième conséquence (qui est liée à la modulation chirp) est que nous estimons qu'à ce débit les données parviennent systématiquement au récepteur si ce dernier rempli les conditions exposées dans le paragraphe 6.2.2.3.

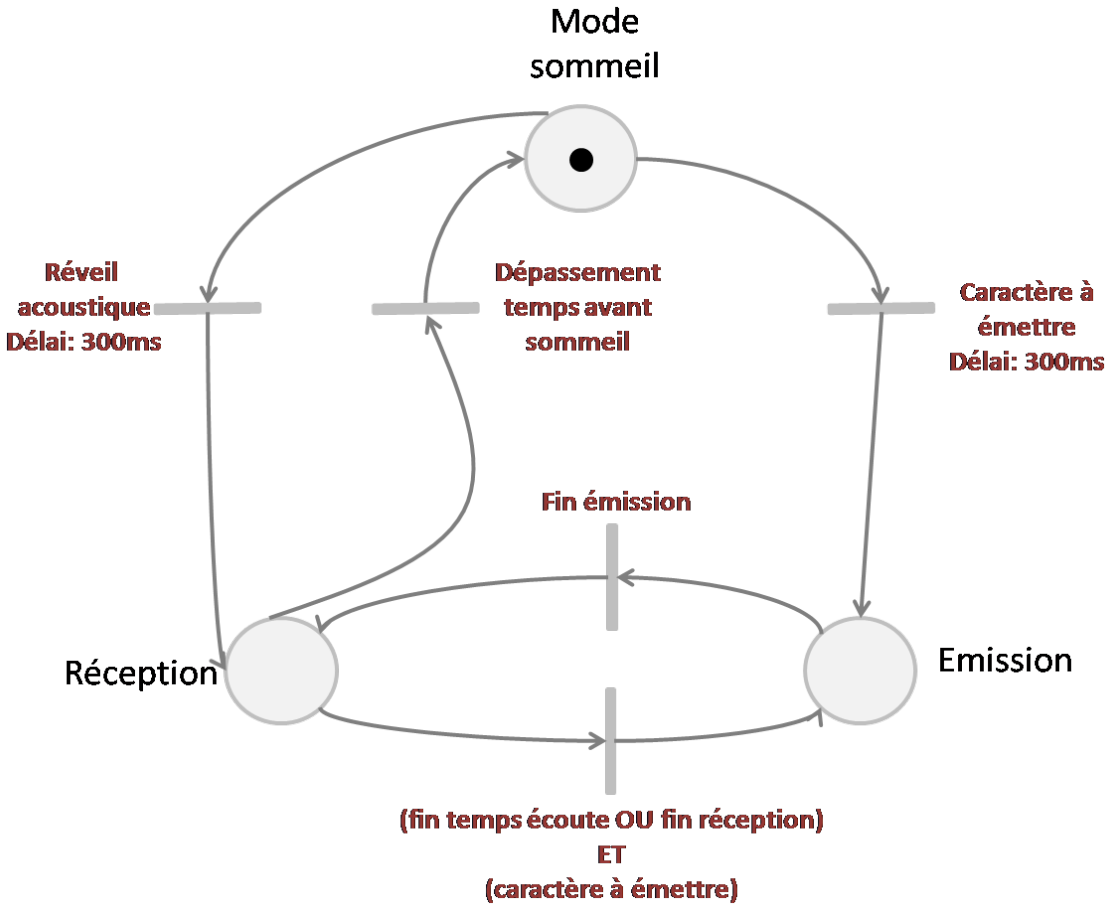


Fig. 6.6 – Diagramme d'état du modèle du modem

6.4.2.3 Modélisation

Nous avons représenté sur la figure 6.6 le diagramme d'état du modèle que nous utilisons. Conformément aux hypothèses du modèle, nous n'avons modélisé que trois états, sans file d'attente, avec un délai pour passer du mode sommeil au mode réception ou émission.

6.4.2.4 Domaine de validité

Si le comportement logico-temporel de ce moyen de communication est correctement représenté, il n'en va pas de même en ce qui concerne le traitement du signal effectué par le modem. En effet, nous ne modélisons pas cet aspect et considérons que ce traitement est instantané (émission sans autre délai que ceux mentionnés sur la figure 6.6).

Par ailleurs il existe une limitation liée au couplage de ce modèle avec notre architecture de simulation. En effet, si tout le système de simulation traite les informations en temps-réel, on constate qu'en ce qui concerne les moyens de communications les données sont échangées par bloc : un message (un caractère dans notre cas) sera transmis instantanément et intégralement au simulateur de moyen de communication juste après la date réelle de fin de réception.

Cette approche ne permet donc pas de traiter en temps-réel le décodage des messages échangés entre le véhicules : il n'est pas possible de faire réellement du "modem-in-loop". Une solution consistant à utiliser des cartes A/N avec le logiciel réel du modem pourrait être envisagée si le besoin de tester les algorithmes de décodage du modem se faisait ressentir.

6.4.3 La radio UHF, le WIFI et les moyens opto-électroniques

Bien que présents et utilisés sur nos véhicules, nous n'avons pas encore créé de modèles pour les moyens de communications radio, wifi et moyens opto-électroniques. Ce travail reprendra les mécanismes mis en place pour la diffusion des ondes acoustiques, en adaptant certains paramètres et comportements.

6.5 Conclusion

Après avoir proposé une architecture de simulateur dans le chapitre précédent, nous nous sommes focalisés dans ce chapitre sur les modèles à mettre en œuvre pour être en mesure de considérer des scenarii multi-véhicules. De nombreux modèles appartenant à des domaines différents et de nature différente sont nécessaires pour atteindre cet objectif. Les contraintes de temps liées à cette thèse d'une part et notre intérêt pour la robotique sous-marine d'autre part, nous ont mené à travailler en priorité sur certains de ces modèles.

En ce qui concerne le simulateur de véhicules, nous avons présenté un modèle permettant de simuler les véhicules sous-marins de type torpille, et un modèle pour les véhicules de surface de type catamaran. Ces modèles ont été choisis après avoir proposé un rapide aperçu des méthodes de modélisation hydrodynamique existantes. Cet aperçu nous a amené à considérer le modèle proposé par Fossen en ce qui concerne la torpille et pour lequel nous avons obtenu ces dernières années un certain nombre de coefficients, et un modèle proposé par M. Caccia pour le catamaran.

Nous avons ensuite évoqué les modèles à intégrer dans le simulateur d'environnement et dans ce cadre, nous avons proposé un modèle de propagation des ondes acoustiques en milieu aquatique après avoir réalisé une étude détaillée des phénomènes physiques reliés à cette propagation. Au-delà d'un modèle il s'agit d'un moteur de diffusion des messages entre les véhicules dont le comportement logique (à qui faut-il délivrer le message ?) et temporel (quand faut-il délivrer le message) est déterminé par un ensemble de modèles liés à la propagation (modèle d'atténuation, modèle de célérité d'une onde sonore, modèle de latence, ...). Le principal avantage de ce "moteur" est de pouvoir effectuer le traitement en temps réel et de pouvoir affiner son comportement à travers l'amélioration des modèles qui lui sont rattachés.

Par ailleurs ce même moteur pourra être utilisé pour la propagation des ondes électromagnétiques avec des modèles adaptés. Il ne modélise pas directement les phénomènes physiques, mais permet de reproduire leurs conséquences sur le message diffusé. Toujours en ce qui concerne le simulateur d'environnement, nous avons présenté un modèle de collisions véhicule/véhicule ou véhicule/environnement simple mais néanmoins efficace et suffisant. La représentation de l'environnement a également été évoquée.

En ce qui concerne le simulateur de capteur, nous avons limité nos développements aux capteurs nécessaires à la commande des véhicules et à la validation de notre architecture.

Des modèles simples pour lesquels nous avons considéré le taux d'échantillonnage et un bruit gaussien ont été présentés.

Enfin, nous avons sélectionné un moyen de communication pour lequel nous avons proposé un modèle logico-temporel. Ce modèle est à même de reproduire le comportement d'un modem acoustique que nous utilisons dans le cadre de nos recherches. Nous n'avons implémenté qu'un mode de fonctionnement qui correspond à celui que nous utiliserons vraisemblablement lors de nos futures expérimentations. Par ailleurs nous avons proposé une façon de connecter des modems réels de développement à notre simulateur, afin d'établir un pont avec une autre communauté.

Pour chacun des modèles que nous avons évoqués, nous avons clairement fait apparaître les hypothèses et le domaine de validité ou du moins les faiblesses des modèles considérés. La diversité des modèles présentés tend à prouver la capacité offerte par le système de simulation aux utilisateurs, de pouvoir développer librement un modèle adapté à leur besoin. Le pendant de cette fonctionnalité est la multitude de modèles à développer avant de pouvoir créer, tester et valider des scenarii innovants. En effet nous avons vu que certains de ces modèles sont simples mais parfois suffisants, ou inexistants (sonar, loch doppler, ...) et qu'ils nécessitent clairement l'intervention de différents spécialistes.

Troisième partie

Validation et exploitation de Thetis

Introduction

Après avoir proposé une architecture de simulateur respectant l'ensemble des critères cruciaux évoqués dans la première partie, nous avons évoqué l'implémentation du logiciel puis l'aspect modélisation. Nous présentons maintenant la validation et l'exploitation de notre simulateur Thetis.

Dans un premier chapitre, nous explicitons la méthode de validation à laquelle nous avons recouru pour valider d'une part le module élémentaire, et d'autre part l'architecture dans son ensemble. Nous présentons à la fin de cette validation, quelques missions impliquant un ou plusieurs véhicules, destinées à mettre en exergue la capacité du simulateur à prendre en compte les comportements potentiellement divergents des véhicules sous certaines conditions. Il est à noter que nous n'avons pas validé les modèles employés pour deux raisons : le travail d'identification des modèles est une thèse à part entière et sort donc du cadre de celle-ci. Par ailleurs, nous ne disposons pas à l'heure actuelle de véhicules capables de naviguer (nous leur apportons d'importantes améliorations et ils sont donc indisponibles) et nous ne pouvons donc pas réaliser de comparaisons, même qualitatives.

Le second chapitre est consacré à l'exploitation du simulateur. Nous montrons à travers une série de missions typiques (relevé CTD, stratégie de coordination, recalage des centrales inertielles, ...) les simulations qui pourront être réalisées grâce à ce simulateur.

Chapitre 7

Validation

La norme ISO 9000 définit la validation comme étant la *confirmation par des preuves tangibles que les exigences pour une utilisation spécifique ou une application prévue sont satisfaites*. Dans ce chapitre nous validons donc notre système de simulation en cherchant à :

- mesurer le degré de confiance que l'on peut avoir dans les résultats
- définir des bornes de domaine d'utilisation au regard des capacités de calcul du simulateur.
- mettre en exergue les propriétés que nous avons décrites comme étant des éléments clés de la simulation

Cette validation doit donc permettre de confirmer que les résultats sont justes en date et en valeur. Elle se déroule sur deux plans : la validation du système qui consiste à vérifier qu'il correspond bien à ses spécifications, puis la validation des modèles qui consiste à s'assurer qu'ils représentent effectivement ce qu'ils sont censés modéliser.

Dans un premier temps, nous présentons la méthodologie et les résultats de validation de l'architecture de simulation. Puis nous évoquerons brièvement la problématique de la validation des modèles. Enfin, nous montrons, à travers une série de simulations, certaines des propriétés de Thetis que nous avons mises en avant jusqu'ici.

Il est à noter que nous avons procédé à une validation incrémentale à chaque étape de conception (notamment le comportement des modèles numériques) que nous n'avons pas exposé ici.

7.1 Validation de l'architecture

Ce premier niveau de validation va nous permettre de valider l'intégrité des données transitant au sein de l'architecture, de borner les retards dus au temps de traitement de l'information et d'estimer les capacités de calcul du système.

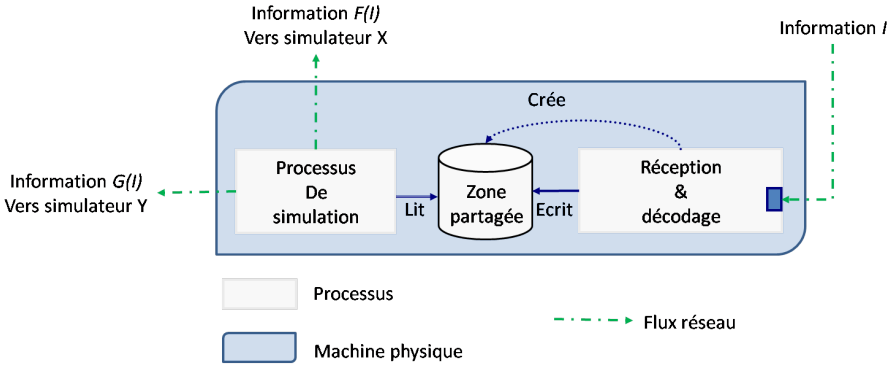


Fig. 7.1 – Module élémentaire de Thetis.

7.1.1 Module de base

7.1.1.1 Introduction

Nous rappelons sur la figure 7.1, la structure du module élémentaire de Thetis. Nous cherchons ici à mesurer deux choses. La première est le temps que nous appelons *temps de relaxation*. Il s'agit du temps au bout duquel le processus de décodage et partage des données, revient en mode "écoute" : c'est en fait le temps de traitement d'une trame entrante. La deuxième est le *retard* : c'est le temps qui s'écoule entre le moment où une trame parvient au processus de décodage et partage, et le moment où les informations contenues dans cette trame sont effectivement disponibles dans la mémoire partagée pour le processus de simulation qui y est raccordé.

Le temps de relaxation va nous permettre d'estimer la capacité de traitement d'un module élémentaire, alors que le retard nous permettra de quantifier les conséquences sur les résultats de la simulation. Nous présentons sur la figure 7.2, les différentes étapes de l'algorithme de décodage et partage. Nous y précisons les points de mesure que nous allons effectuer.

7.1.1.2 Méthodologie de la mesure

Plutôt que de tester tous les modules élémentaires du système, nous avons choisi celui qui est le plus sollicité en terme de trames reçues. Ce choix nous permet de majorer la charge de calcul et d'en tirer des conclusions pour l'ensemble des modules du système. Dans ce cadre, nous avons sélectionné le module réception et décodage du simulateur de capteurs, traitant les trames envoyées par le simulateur de véhicules. En effet, il reçoit 100 trames par secondes, multiplié par le nombre de véhicules. Par ailleurs, la longueur de la trame fait partie des plus importantes du simulateur.

Concernant la mesure du temps, nous n'utilisons pas les primitives temps réel du système pour créer nos processus ; par conséquent une approche classique et de vérifier à l'aide d'un oscilloscope l'activité du module car utiliser le temps système est trop peu précis au regard des temps à mesurer. La lourdeur de cette opération nous a amené à développer une autre approche : nous avons développé une routine en assembleur qui permet de relever un registre interne du processeur qui s'incrémente continuellement à la même fréquence que ce dernier. Grâce à ce relevé, nous pouvons connaître le nombre de cycles processeur entre deux parties du programme.

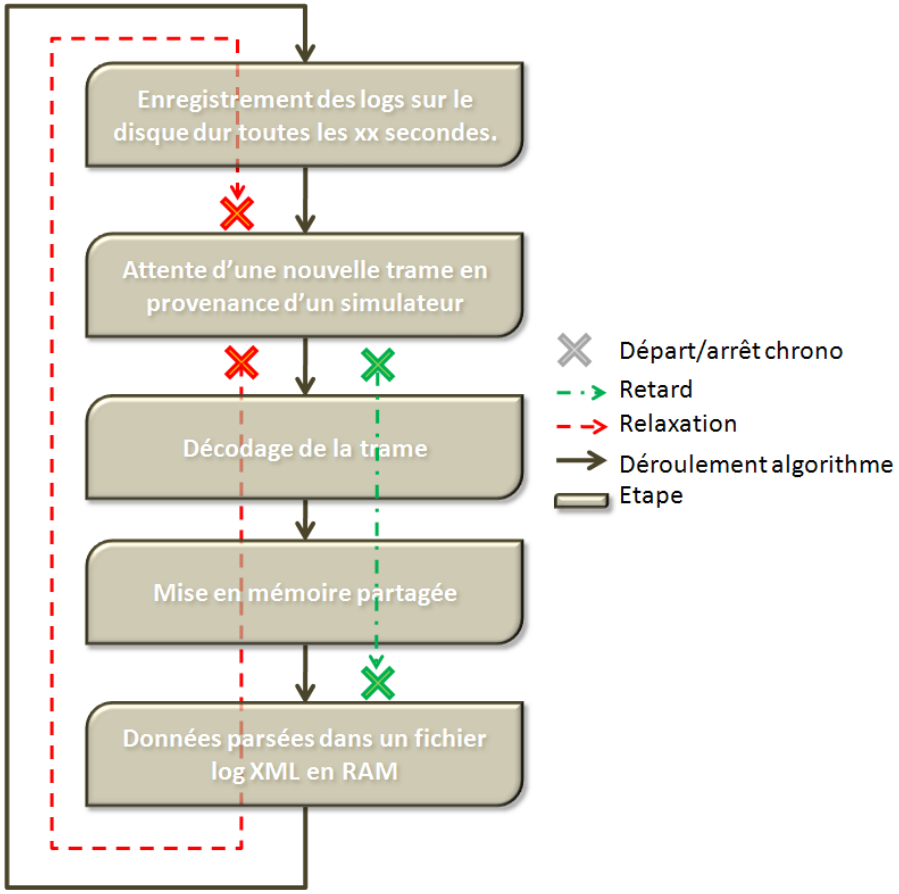


Fig. 7.2 – Points de mesure permettant de quantifier retard et temps de relaxation d'un module élémentaire.

Il est à noter que cette interrogation se fait en une seule et unique instruction assembleur. Le temps de traitement est bien sûr relatif à la fréquence du processeur. Pour quantifier une durée d'exécution d'une partie d'un programme, il est donc nécessaire de parler en cycles pour être indépendant de la fréquence du processeur. Néanmoins, pour mieux appréhender les durées que nous présentons, nous avons converti ce nombre de cycles en temps au regard de la fréquence de fonctionnement de notre processeur.

7.1.1.3 Retard

Les mesures du retard tel qu'il est défini sur la figure 7.2, sont présentées dans la colonne de gauche de la figure 7.3. Pour ces mesures, nous avons fait varier le nombre de véhicules participant à la simulation. Comme nous pouvions le prévoir, le délai au bout duquel les données sont mises à disposition en mémoire partagée n'évolue pas en fonction du nombre de véhicules : une trame met bien toujours le même temps à être traitée quel que soit son contenu. Sur une échelle de 100s, nous n'avons pas constaté de retard significatif dans le délai de décodage et partage. Par ailleurs, on constate sur le graphique que les valeurs sont très groupées : cela signifie que le temps de traitement varie très peu d'une fois sur l'autre. La moyenne de ce retard est de l'ordre de $10^{-5}s$ ce qui est négligeable par rapport aux constantes de temps du système simulé (5.10^{-3}s min).

7.1.1.4 Temps de relaxation

Les mesures du temps de relaxation sont présentées dans la colonne de droite de la figure 7.3. Nous avons à nouveau fait varier le nombre de véhicules participant à la simulation. Le temps de traitement d'un trame comprenant son décodage, sa copie en mémoire partagée, son parsing dans le fichier log en RAM ne varie pas non plus en fonction du nombre de véhicules. La légère augmentation du temps moyen de traitement apparaissant sur les graphiques ne correspond pas à une réalité physique : nous avons confirmé cela par des tests complémentaires mettant en œuvre jusqu'à six véhicules : nous avons constaté un temps de relaxation de $1.4.10^{-4}s$ ce qui est moins que pour la figure présentant un seul véhicule.

En revanche, on voit clairement apparaître deux temps de traitement différents sur les graphiques. La figure 7.4 présente les évolutions du temps de relaxation au cours de la simulation. On voit apparaître des motifs distincts sur le graphique. Nous avons d'abord pensé que ces différences étaient dues au transfert de données de la RAM vers le swap du disque dur car le fichier log devenait trop gros pour être contenu dans la RAM. Cette hypothèse n'est pas la bonne, car le temps de relaxation diminue à nouveau à partir de 20000 échantillons environ. Nous pouvons donc raisonnablement supposer que ce que nous observons ici, est corrélé à une activité périodique du système d'exploitation.

Les limites de notre système sont donc atteintes ici : le lancement de nos processus en utilisant les primitives RTAI devrait nous permettre d'améliorer les performances de ce module. Néanmoins, il faut noter que malgré ces variations, nous demeurons largement en deçà des contraintes temps-réel que nous nous sommes imposées. Ces variations sont donc sans conséquence pour le résultat de la simulation, mais traduisent bien le fait que le système d'exploitation peut reprendre la main n'importe quand. Nous avons donc intégré ces mesures dans les modules, et en cas de retard de traitement, le superviseur est averti en précisant le délai et la nature du dépassement du temps de relaxation. A la charge

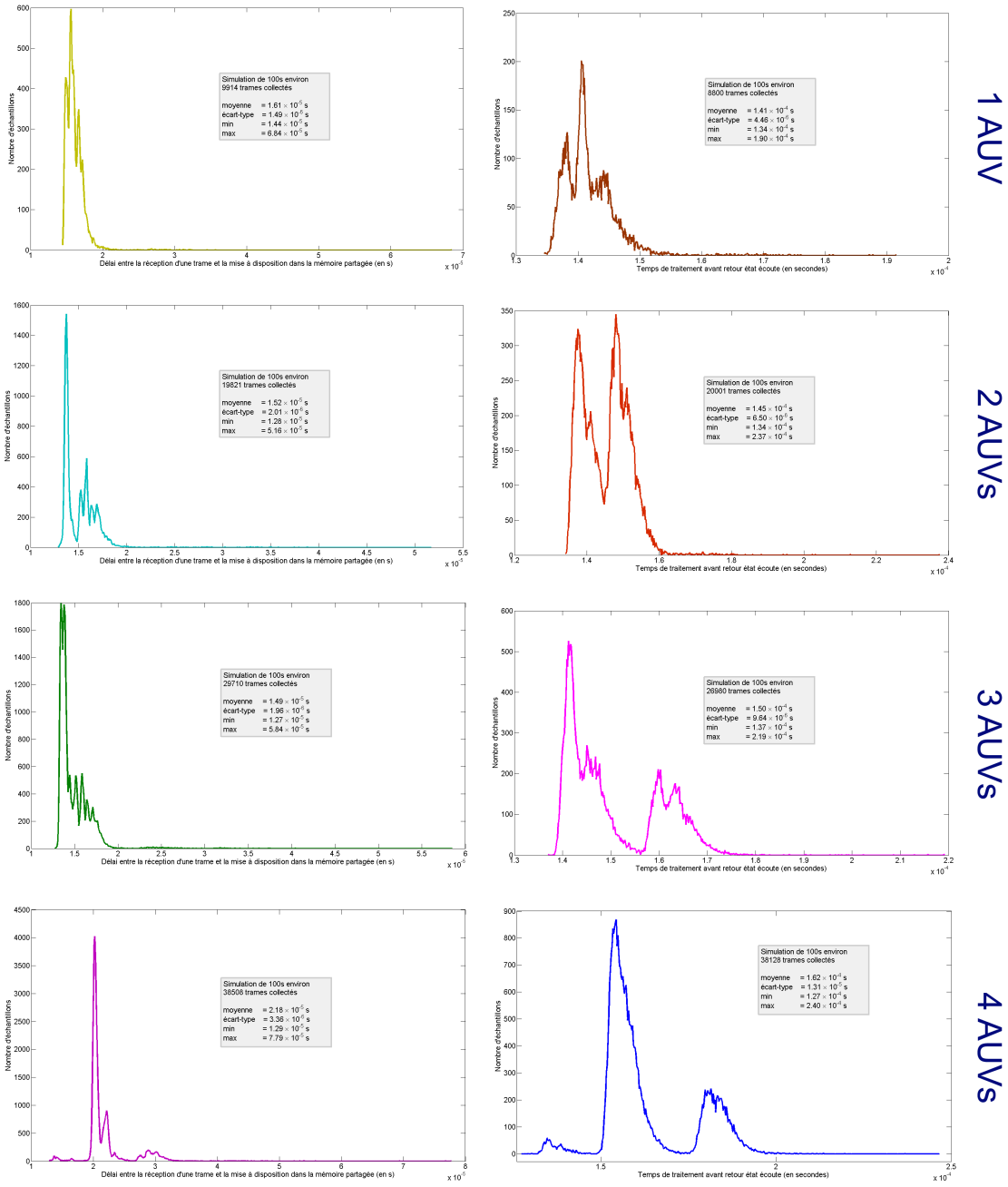


Fig. 7.3 – La colonne de gauche présente le retard et celle de droite le temps de relaxation.

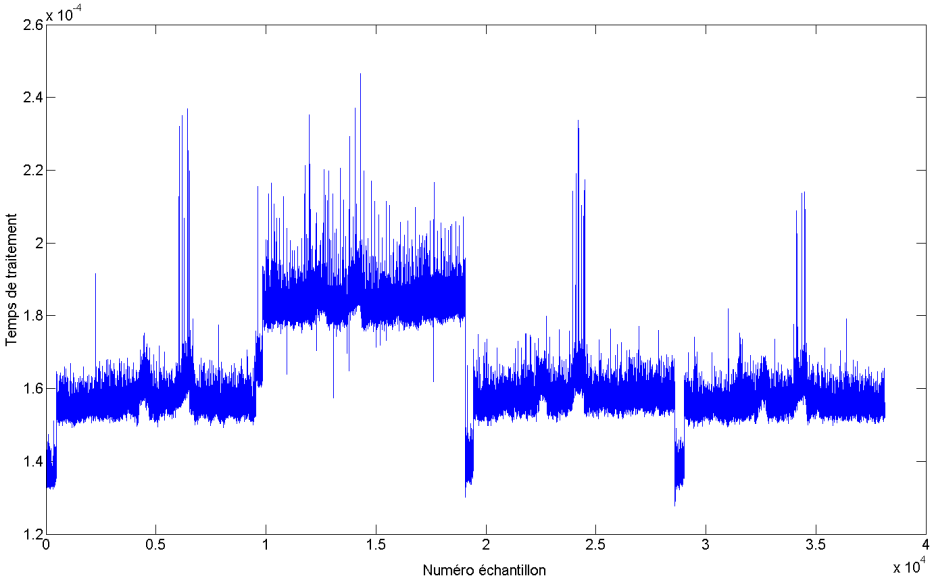


Fig. 7.4 – Evolution du temps de relaxation au cours de la simulation.

de l'utilisateur de définir si ce dépassement porte à conséquence sur les résultats de la simulation (auquel cas il faut envisager de la recommencer) ou pas. Ces problèmes seront résolus lorsque nous utiliserons les primitives RTAI.

Il est à noter que durant la simulation, nous ne sauvegardons pas les données sur le disque dur. En effet, dans les premières versions, nous sauvegardions le fichier XML de log toutes les 3s environ, mais lorsque celui-ci atteignait une taille importante (environ 5Mo), des latences (de 1 à 3s) apparaissaient, invalidant notre simulation. La sauvegarde du fichier se fait donc en fin de mission. Par ailleurs, notre expérience de l'utilisation des fichiers XML comme format de sauvegarde des logs nous a montré que ce format ne convenait pas pour sauvegarder en temps réel les données. En effet, nos mesures indiquent que le temps de relaxation était majoritairement constitué du temps d'insertion d'une donnée dans le fichier. Si cela ne pose pas de problème particulier pour l'instant (nous restons toujours en deçà des limites autorisées du temps de traitement), cela pourrait constituer un problème sur un processeur peu puissant. Par ailleurs, le temps d'extraction des données du fichier XML sous Matlab est conséquent.

Ce format ne présente une utilité que dans le cadre d'échange de fichiers de résultats de simulation. Nous envisageons donc de sauvegarder simplement les données dans un fichier texte pour diminuer le temps de traitement pendant la simulation et lors de l'exploitation, et de développer un programme permettant d'encapsuler ces données dans un fichier XML pour les échanges.

L'ensemble des mesures réalisées, nous a permis d'estimer la validité du système de simulation au regard du nombre de véhicules, de la fréquence du processeur utilisé et de la fréquence d'échantillonnage. On estime ansi que les ordinateurs que nous utilisons dans le système de simulation permettent de réaliser théoriquement des simulations allant jusqu'à 61 véhicules sans problème du côté du module élémentaire :

$$NB_{VHC} < \frac{T_x F_p}{452489}$$

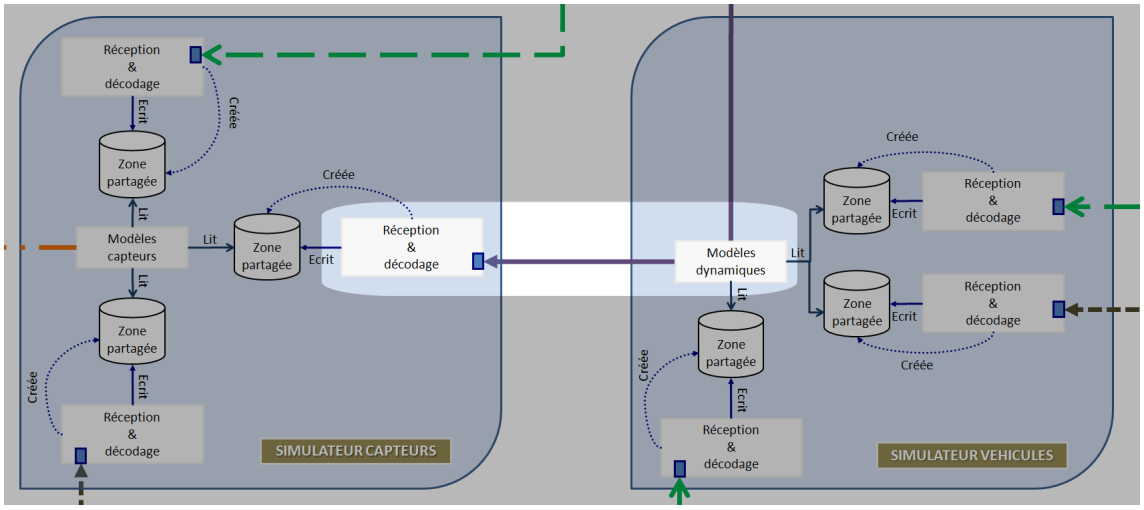


Fig. 7.5 – Validation de l'intégrité du flux réseau reliant un simulateur et un décodeur.

avec T_x le taux d'échantillonnage (période d'envoi des trames en fait, soit 0.01s dans notre cas pour ce module), F_p la fréquence du processeur en Hz (2.8GHz dans notre cas) et NB_{VHC} le nombre de véhicules pouvant participer à la simulation.

7.1.2 Lien inter-simulateur

Après avoir validé et déterminé le domaine de validité du module élémentaire le plus sollicité, nous nous intéressons maintenant au lien réseau unissant un processus simulateur et un processus décodeur. Nous contrôlons en particulier deux points : l'intégrité des données (c'est-à-dire le fait qu'un processus décodeur reçoive tout ce qu'un processus simulateur envoie) et la latence entre l'envoi et la réception d'une trame.

7.1.2.1 Intégrité des données

Nous avons à nouveau sélectionné le même bloc pour valider le lien inter-simulateur, pour les même motifs que précédemment (figure 7.5).

Pour mesurer les pertes de trames, on réalise un test basique : on enregistre de part et d'autre l'ensemble des paquet émis / reçus, puis on compare le contenu des deux fichiers. Nous avons effectué ce test dix fois pendant 100s en faisant varier le nombre de véhicules (de un à six) et n'avons pas constaté de pertes de données (mise à part dans les tous premiers instants de la simulation, lorsque le décodeur n'est pas encore à l'écoute).

7.1.2.2 Temps de latence

La mesure du temps de latence entre les deux processus nécessite une base de temps commune. Il est par exemple possible d'utiliser un récepteur GPS sur chaque ordinateur pour obtenir la synchronisation de leur horloge. Néanmoins, nous n'avons pas mis en place un tel dispositif car nous avons apprécié la latence du réseau au travers de requêtes ping. Nous avons ainsi constaté une latence de 9.410^{-5} secondes, ce qui est largement négligeable par rapport aux constantes de temps du système.

7.1.3 Temps de cycle

7.1.3.1 Période des boucles internes de simulation

Les processus n'utilisant pas les primitives RTAI, il est nécessaire de valider les temps de cycle de chaque simulateur. Chaque simulateur a un pas de simulation qui lui est propre et qui est déterminé par la fréquence d'envoi des commandes aux actionneurs. Ainsi nous choisissons de faire évoluer les modèles dynamiques des véhicules une dizaine de fois entre chaque nouvelle commande émise par le contrôleur du véhicule. La période de ces commandes est de 0.1s sur les AUVs Taipan. Par conséquent, la période de simulation du simulateur de véhicules est de 0.01s.

L'environnement doit détecter les éventuelles collisions et corriger le cas échéant le vecteur d'état des véhicules avant de le renvoyer au simulateur de véhicules. Cette opération doit se faire en moins d'un temps de cycle du simulateur de véhicules afin d'être prise en compte au cours du cycle de simulation de véhicule suivant. En utilisant cette approche, nous garantissons bien le découplage temporel entre les véhicules et l'environnement (pas d'attente bloquante du simulateur d'environnement), tout en minimisant le retard de la prise en compte d'une collision (ce retard vaut donc le temps de cycle du simulateur de véhicules). Nous avons donc choisi une période de 0.005s pour le simulateur d'environnement. Concernant le simulateur de capteurs, la période de la boucle de simulation est choisie en fonction de la plus grande fréquence d'échantillonnage des capteurs simulés. Le retard d'un échantillon capteur est borné et vaut le temps de cycle du simulateur de capteurs.

Nous avons donc choisi la fréquence d'échantillonnage de la centrale inertielle soit $8.10^{-3}s$ puisque son taux de rafraîchissement est de 120Hz. Enfin, en ce qui concerne le simulateur de moyens de communication nous avons choisi arbitrairement une fréquence de 10Hz. Ce choix implique un retard maximum de 0.1s lors de l'émission ou de la réception d'un message.

Nous présentons sur la figure 7.6 la répartition des temps de cycle de chaque simulateur pour un ou quatre véhicules.

Nous pouvons observer qu'en moyenne les temps sont respectés, avec toutefois quelques écarts apparaissant à intervalles réguliers (traduit par la présence de deux pics distincts) qui correspondent aux accès disque pour l'enregistrement des fichiers logs. Cela n'a pas d'influence sur les résultats de la simulation car ces temps restent très faibles. De même que précédemment, une information est envoyée au superviseur en cas de retard significatif ($> 100\%$ du temps de cycle).

Le fait que des valeurs de temps soient légèrement supérieures à celles spécifiées (ex : le simulateur de véhicule), vient de la méthode que nous utilisons pour endormir la tâche de simulation et laisser du temps processeur au système. Bien que cela ne soit pas utile, nous pouvons réduire ce temps (comme pour le simulateur de capteurs par exemple), afin de faire correspondre exactement valeur spécifiée et valeur réelle. Cela ne traduit pas une saturation de calcul d'un simulateur en particulier.

7.1.3.2 Période de la boucle de simulation du système

Nous avons indiqué précédemment qu'il était nécessaire de faire évoluer les modèles dynamiques des véhicules une dizaine de fois entre chaque commande envoyée. Nous avons vu

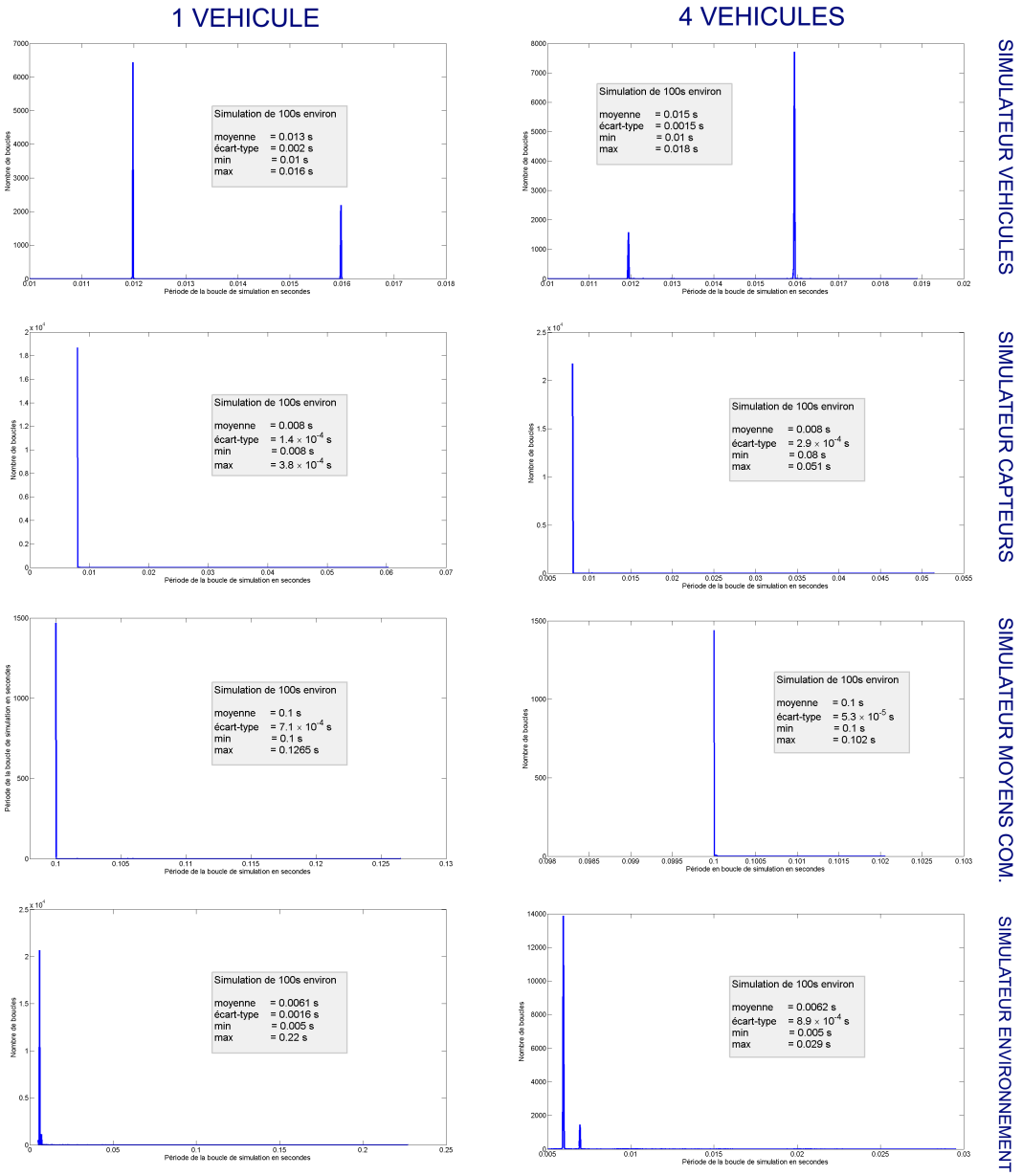


Fig. 7.6 – Répartition du temps de cycle de chaque simulateur pour un et quatre véhicules.

qu'une boucle complète de simulation comprend plusieurs échanges entre les simulateurs. Ne disposant pas d'horloge synchrone sur chaque simulateur, nous proposons d'estimer un temps de cycle complet de simulation à partir des données enregistrées précédemment.

Ainsi chaque commande arrive avec une période de $0.1s$; cela signifie que la période de la boucle de simulation doit être inférieure à $0.01s$. Autrement dit, le vecteur d'état des véhicules doit parvenir à cette fréquence au simulateur de capteurs. En ajoutant les temps de mises à disposition en mémoire partagée, les temps de traitement et d'aller-retour entre les différents simulateurs, nous obtenons un temps maximum de $0.0054s$.

7.2 Validation des modèles

La validation des modèles utilisés, sort du cadre de cette thèse ; en effet, nous avons utilisé des modèles dynamiques largement acceptés par la communauté. En ce qui concerne les modèles des capteurs, nous avons pu vérifier que leur comportement correspond bien à celui des capteurs réels. La mesure virtuelle qu'ils fournissent est sans doute légèrement différente de la réalité mais cela n'a finalement que peu d'incidence sur les résultats de la simulation. Nous avons intégré un bruit gaussien multiplicatif ou additif (selon le capteur) correspondant aux caractéristiques du capteur délivrées par le fabriquant. De façon générale, la représentativité du réel est définie par les paramètres des modèles utilisés.

7.3 Mise en exergue des comportements divergents

Nous avons mis en avant tout au long du manuscrit la nécessité de respecter certaines notions essentielles pour obtenir des résultats de simulations fiables. Cette section a donc pour but de présenter quelques cas pratiques de simulation mettant en exergue certains comportements potentiellement divergents qui ne sont pas pris en compte dans les simulateurs existants.

7.3.1 Taux d'échantillonnage des capteurs

Thetis est un simulateur temps réel, dans lequel il est possible de simuler des capteurs différents. Ces capteurs possèdent tous un modèle auquel est associé un fichier de configuration. Parmi les paramètres présents dans ce fichier, la période d'échantillonnage tient un rôle primordial, puisqu'elle permet de déterminer à quels instants est envoyée la réponse d'un capteur en particulier, à un véhicule en particulier. A l'évidence, le comportement d'un robot est fortement affecté par cette période, et il est donc nécessaire d'en tenir compte dans les simulations dont le cadre est la préparation d'une mission réelle. L'objectif est donc ici de montrer à travers deux simulations la différence de comportement d'un véhicule sur deux missions dont la seule différence est la période d'échantillonnage d'un capteur.

7.3.1.1 Présentation de la mission

Nous programmons une mission dans laquelle le véhicule, un AUV de classe Taipan300 en l'occurrence, doit atteindre une profondeur et un cap donné et s'y maintenir. Nous utilisons un contrôle proportionnel-intégral pour contrôler le véhicule dans le plan horizontal,

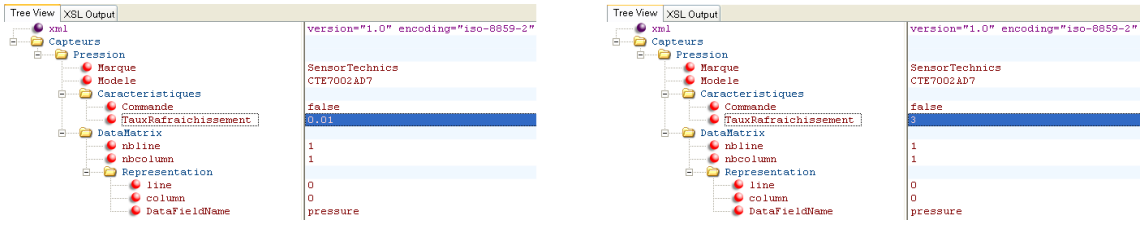


Fig. 7.7 – Configurations du capteur de pression de Taipan 300.

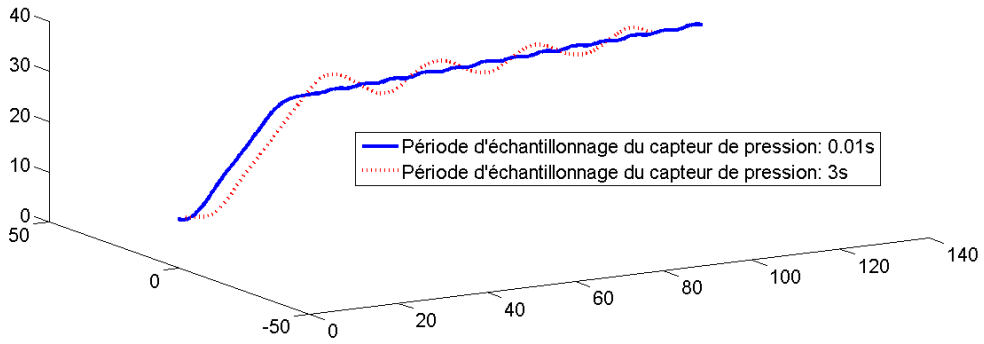


Fig. 7.8 – Trajectographie 3D du véhicule

et une commande de type sliding-mode dans le plan vertical. Nous plaçons le véhicule à $10m$ sous la surface à l'instant initial, et il doit rejoindre le cap 350° à une profondeur de $30m$ à une vitesse d'avance $u = 1.5m.s^{-1}$. Nous configurons la période d'échantillonnage du capteur de pression à sa valeur réelle (soit $0.01s$), puis à une valeur fictive de $3s$ (figure 7.7).

7.3.1.2 Résultats

Les résultats sont présentés sur les figures 7.8 / 7.9 / 7.10 .

On peut observer que le véhicule est capable de suivre la consigne de cap dans les deux simulations puisque le capteur de pression n'intervient pas dans le calcul de cette commande. En revanche, on observe figure 7.9 que les gouvernes de plongée avant sont fortement affectées par la période d'échantillonnage du capteur de pression. Dans la première simulation, l'angle des gouvernes varie immédiatement après que le véhicule a atteint la profondeur désirée, et réagissent instantanément pour corriger la profondeur de l'engin.

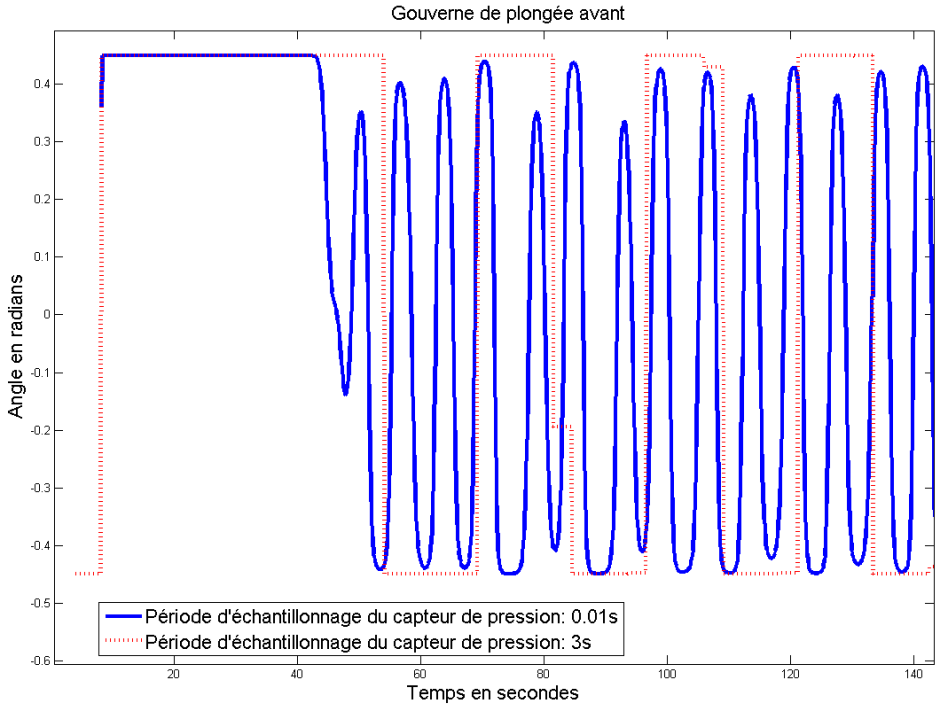


Fig. 7.9 – Angle des gouvernes de plongée

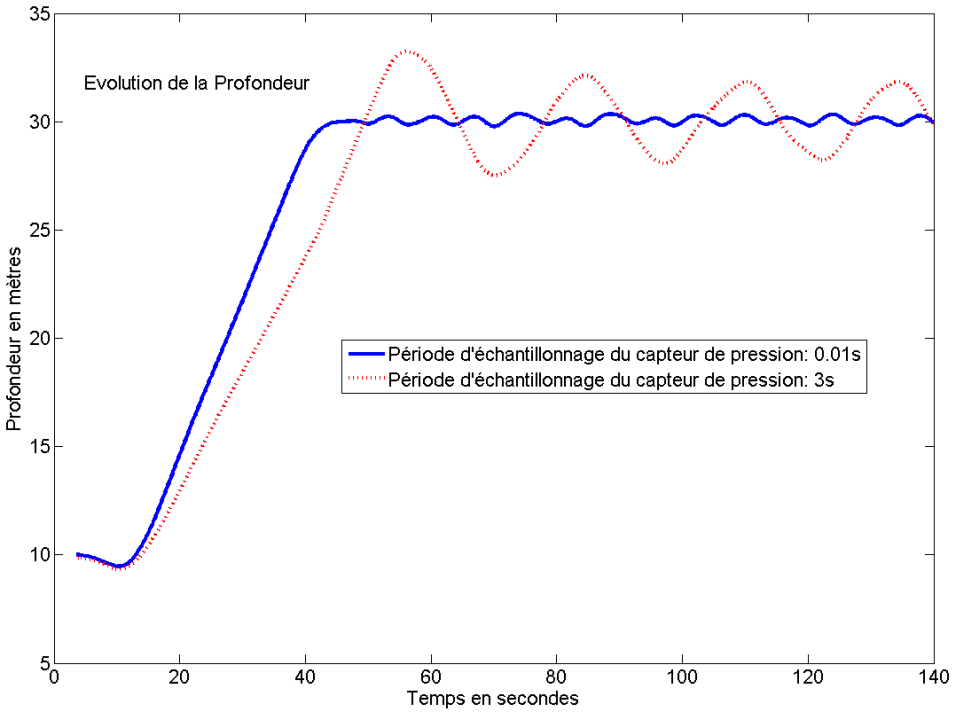


Fig. 7.10 – Evolution de la profondeur du véhicule

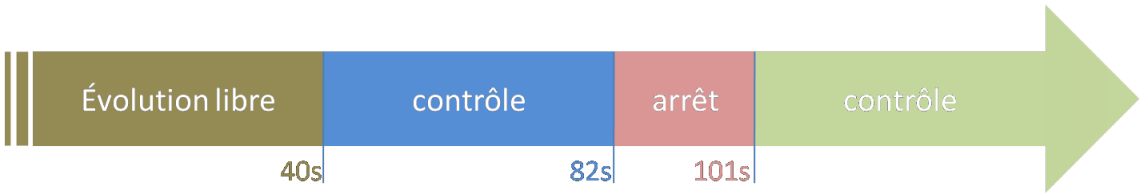


Fig. 7.11 – Les quatre phases de la simulation durant laquelle le contrôleur suspend son activité.

Dans la deuxième simulation, l'angle de gouverne ne change pas immédiatement après que la profondeur désirée a été atteinte.

Il en résulte un premier dépassement important de $3.2m$. On observe par la suite que l'angle des gouvernes varie brusquement d'un extrême à l'autre ($\pm 23^{\circ}$) traduisant les dépassements réguliers que l'on observe sur l'évolution de la profondeur (figure 7.10). La différence de comportement est donc grande puisqu'on observe une période d'oscillations de $7s$ avec une amplitude de $0.4m$ dans le premier cas, contre $26s$ avec une amplitude de $3.6m$ dans le second cas.

Le simulateur est donc bien capable de prendre en compte la fréquence d'échantillonnage des capteurs des véhicules à travers leur fichier de configuration.

7.3.2 Panne de l'ordinateur de bord

Nous avons mis en avant à plusieurs reprises la nécessité de s'assurer du découplage temporel entre la commande et l'évolution du véhicule. L'implémentation de l'architecture de Thetis nous permet de garantir cette propriété ; le but est ici d'en faire la démonstration. Nous simulons donc une mission au cours de laquelle l'ordinateur de bord d'un véhicule ne répond plus pendant un laps de temps.

7.3.2.1 Présentation de la mission

Nous programmons une mission dans laquelle nous impliquons à nouveau un AUV de classe Taipan 300. Nous lui faisons exécuter une mission similaire à la précédente durant laquelle le véhicule est censé atteindre le cap 170°, à une vitesse d'avance $u = 1.5m.s^{-1}$, en se maintenant à une profondeur de $15m$ et en partant d'une profondeur de $10m$.

Nous utilisons le même type de contrôle que précédemment avec la fréquence d'échantillonnage réelle des capteurs. Pour montrer le découplage temporel commande du véhicule / simulation du modèle dynamique, nous commençons par laisser l'engin évoluer librement pendant plusieurs secondes (phase A), puis nous lançons la mission (phase B), avant de simuler une panne de l'ordinateur de bord (phase C) en figeant le contrôle de l'engin. Enfin, nous rétablissons le contrôle (phase D).

7.3.2.2 Résultats

Les résultats sont présentés sur les figures 7.12 / 7.13 / 7.14 / 7.15.

On observe sur ces courbes que lorsque le contrôleur du véhicule cesse de fonctionner, l'engin continue à évoluer dans le simulateur. On différencie deux cas :

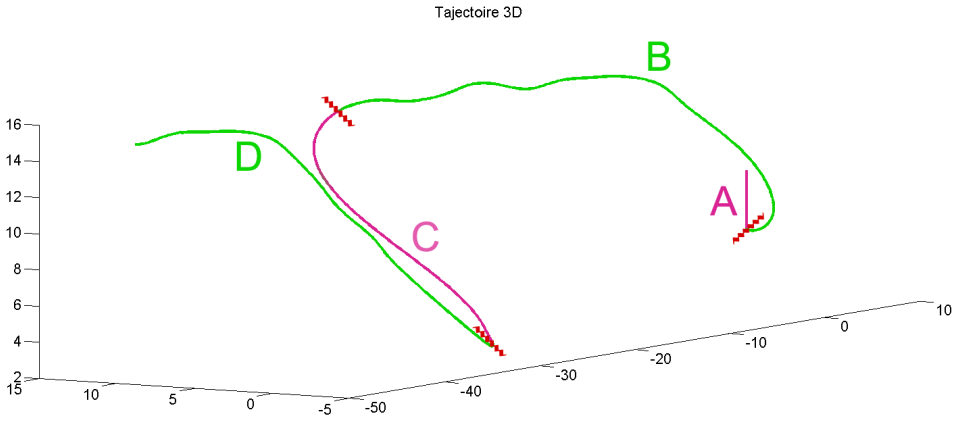


Fig. 7.12 – Trajectographie 3D du véhicule

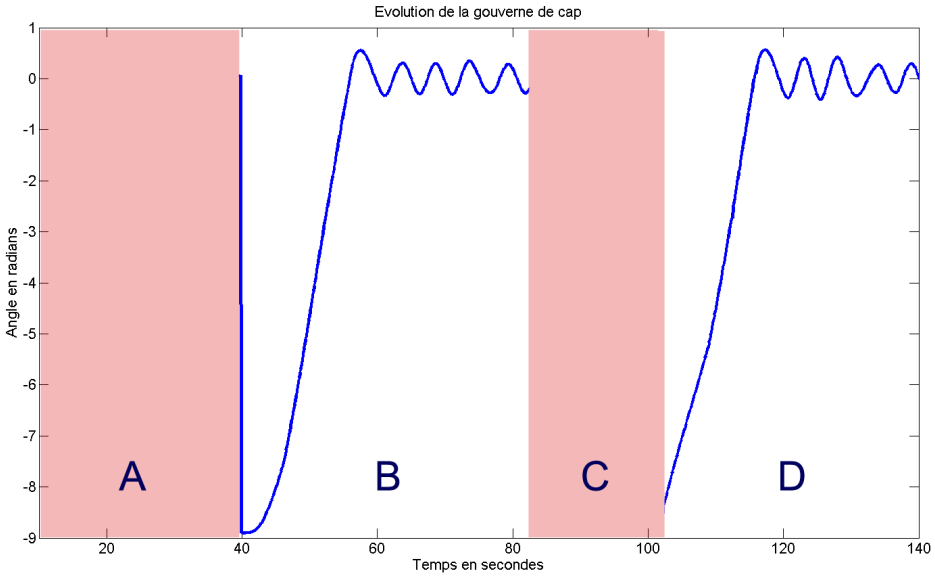


Fig. 7.13 – Evolution de l'angle des gouvernes de cap

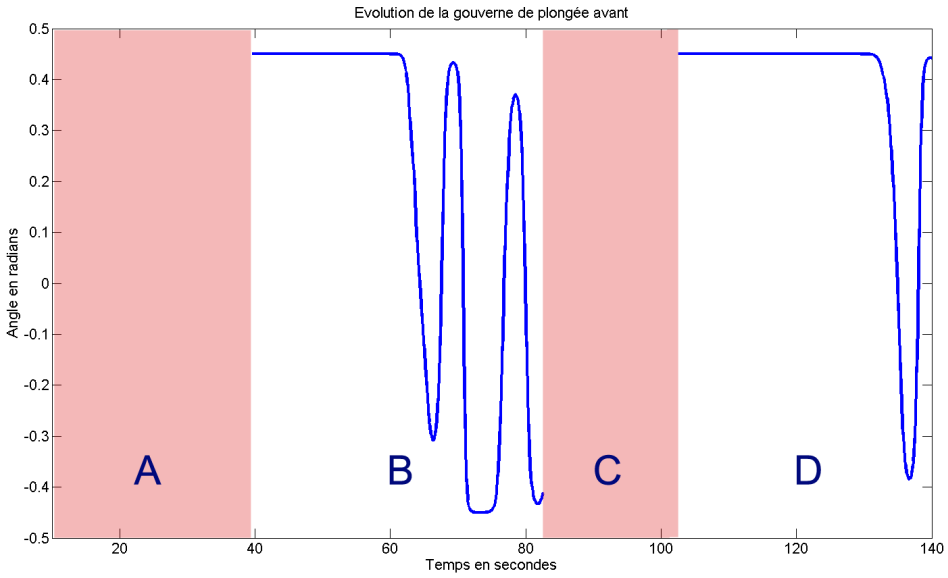


Fig. 7.14 – Evolution de l'angle des gouvernes de plongée

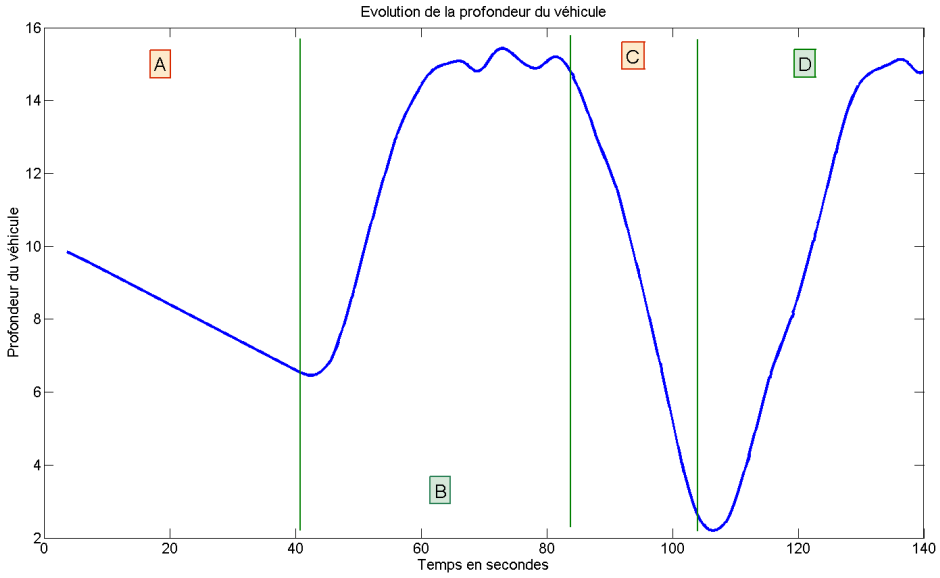


Fig. 7.15 – Evolution de la profondeur du véhicule

- Dans la phase A, le véhicule n'est pas encore sous contrôle. Sa flottabilité étant positive, il remonte donc naturellement vers la surface. La figure 7.15 illustre cette première phase.
- Dans la phase C, en revanche, le véhicule est sous contrôle. Par conséquent, lorsque l'ordinateur de bord cesse de fonctionner, les dernières commandes continuent à être appliquées aux actionneurs. Le véhicule quitte donc son cap et remonte vers la surface (figure 7.12). Il est à noter que dans le cas de Taipan 300, un mécanisme de sécurité a été développé par V. Creuze (Maître de conférence à l'Université de Montpellier) permettant de couper l'énergie aux actionneurs en cas de défaillance prolongée de l'ordinateur de bord. Le véhicule remonte alors en surface naturellement et émet sa position GPS par radio. Ce comportement n'a pas été implémenté dans le modèle du véhicule.

On constate également que l'ordinateur de bord a été capable de reprendre le contrôle de l'engin (en phase B et D) et de poursuivre la mission. Cela pourrait ne pas toujours être le cas suivant l'état du système avant reprise des commandes. Sur les figures 7.13 et 7.14, on voit que le contrôleur n'envoie plus de commande durant les phases A et C.

Le simulateur est donc bien capable de découpler commande du véhicule et évolution de sa dynamique.

7.3.3 Communications sous-marines

Nous avons vu dans le premier chapitre que les rares simulateurs permettant aux véhicules d'échanger des messages ne prenaient pas en compte les contraintes liées à une vraie communication, à savoir retard, limite de portée, interférences, débit limité et comportement du moyen de communication. Nous avons créé un ensemble de modèles permettant de simuler des communications entre les véhicules en tenant compte de ces propriétés. Nous proposons ici une simulation permettant d'en mettre un certain nombre en évidence dans le cadre de la communication sous-marine.

7.3.3.1 Présentation de la mission

La simulation que nous présentons ici implique cinq AUVs de classe Taipan 300. Chacun de ces véhicules sont configurés pour transporter un modem acoustique (modèle ORCA Tapac), leur permettant de communiquer à environ 20 bit/s "certifiés". A l'instant initial, tous les véhicules sont placés suivant l'axe X à des distances variables les uns des autres (voir figure 7.16).

Le but est ici de faire effectuer aux véhicules des changements de cap synchronisés : à un instant donné, un AUV en particulier décide de changer de cap et ordonne à la flottille de faire de même. On choisit en début de mission deux engins (T300_1 et T300_3) qui effectueront ces changements de cap en cours de mission à des instants définis (73s et 75s respectivement).

L'instruction de changement de cap est basé sur un "protocole" basique qui consiste juste à formater le message de la façon suivante : $Nouveau\_CAP\_XXX$ où XXX représente le nouveau cap en degré. La simulation se déroule bien sûr dans un espace 3D, mais tous les AUVs sont situés à une profondeur de $40m$ à $t = 0$ et ont pour consigne de s'y maintenir pour permettre une meilleure appréciation du déroulement de la scène ;

tous les véhicules reçoivent donc une mission identique au départ : suivre le cap 0° à une profondeur de $40m$ et à une vitesse de $1.8m.s^{-1}$.

Afin de pouvoir correctement observer l'ensemble des phénomènes liés à la propagation des messages, nous avons artificiellement divisé la vitesse du son par un facteur dix. Nous aurions pu éloigner davantage les véhicules pour obtenir le même résultat mais les modems que nous employons ne sont pas prévus pour fonctionner à de telles distances.

7.3.3.2 Résultats

Les résultats de la simulation sont présentées figures 7.16 / 7.17 / 7.18. On constate sur la figure 7.17 que le résultat est très éloigné de ce à quoi on pouvait s'attendre :

- Première constatation, tous les véhicules ne changent pas de cap. Les AUVs T300_2 et T300_5 notamment garde un cap nul tout le long de la simulation.
- Parmi les AUVs qui changent effectivement de cap, il ne semble pas y avoir de coordination : certains prennent le cap 120°, d'autres le cap 230°.
- Enfin parmi les AUVs qui prennent le même cap, le changement ne s'effectue pas de façon simultanée.

La figure 7.16 présente un chronogramme sur lequel est reporté le contenu des fichiers (relatif à la communication) résultats de la simulation. On y retrouve l'activité de communication des cinq AUVs : il y figure les messages émis et reçus ainsi que les dates associées à ces événements. Par ailleurs, nous avons mis en évidence les erreurs de communication que nous pouvons détailler :

- *erreur A* : Il s'agit d'une erreur de protocole. En effet, T300_3 a envoyé un message mal formaté qui n'a pas pu être correctement interprété par T300_1. Voici un extrait du fichier résultat (figure 7.19) :
- *erreur B* : Il s'agit d'une erreur liée à une interférence : T300_2 était en train de recevoir un message de T300_1, mais les message de T300_3 survient pendant cette communication avec un niveau de reception correct. Par conséquent il y a une interférence et aucun des deux messages n'est intelligible pour T300_2. Voici un extrait du fichier résultat (figure 7.20) :
  Il est à noter que si un deuxième message arrivait juste après le premier, il ne serait pas non plus entendu car le conflit avec T300_3 serait alors toujours présent : le simulateur est capable de gérer ce type de scénario.
- *erreur C* : Il s'agit là aussi d'une erreur liée à une interférence, mais différente de la précédente. En effet, T300_3 est en train d'émettre lorsque le message émis par T300_1 lui parvient. Son antenne étant réquisitionnée pour émettre un message, il n'est pas en mesure de percevoir entièrement le message de T300_1. Il a uniquement la fin de ce message et ce dernier ne lui est donc pas délivré (du tout). Son message émis, en revanche, est valide et se propage normalement dans le milieu 7.21 :
- *erreur D, E et F* : Les messages arrivent trop atténués pour être perçus par le modem acoustique du véhicule. Par conséquent le simulateur ne lui délivre pas les messages 7.22 :

Finalement seul T300_4 reçoit correctement un message qui lui indique de prendre le cap 120°. Ces erreurs expliquent donc parfaitement la figure 7.17 : on constate qu'effectivement T300_1 et T300_3 changent de cap (120° et 230° respectivement) de façon autonome, que T300_2 et T300_5 ne changent pas de trajectoire et continuent sur un cap nul, et qu'enfin T300_4 change de cap (nouveau cap : 120°).

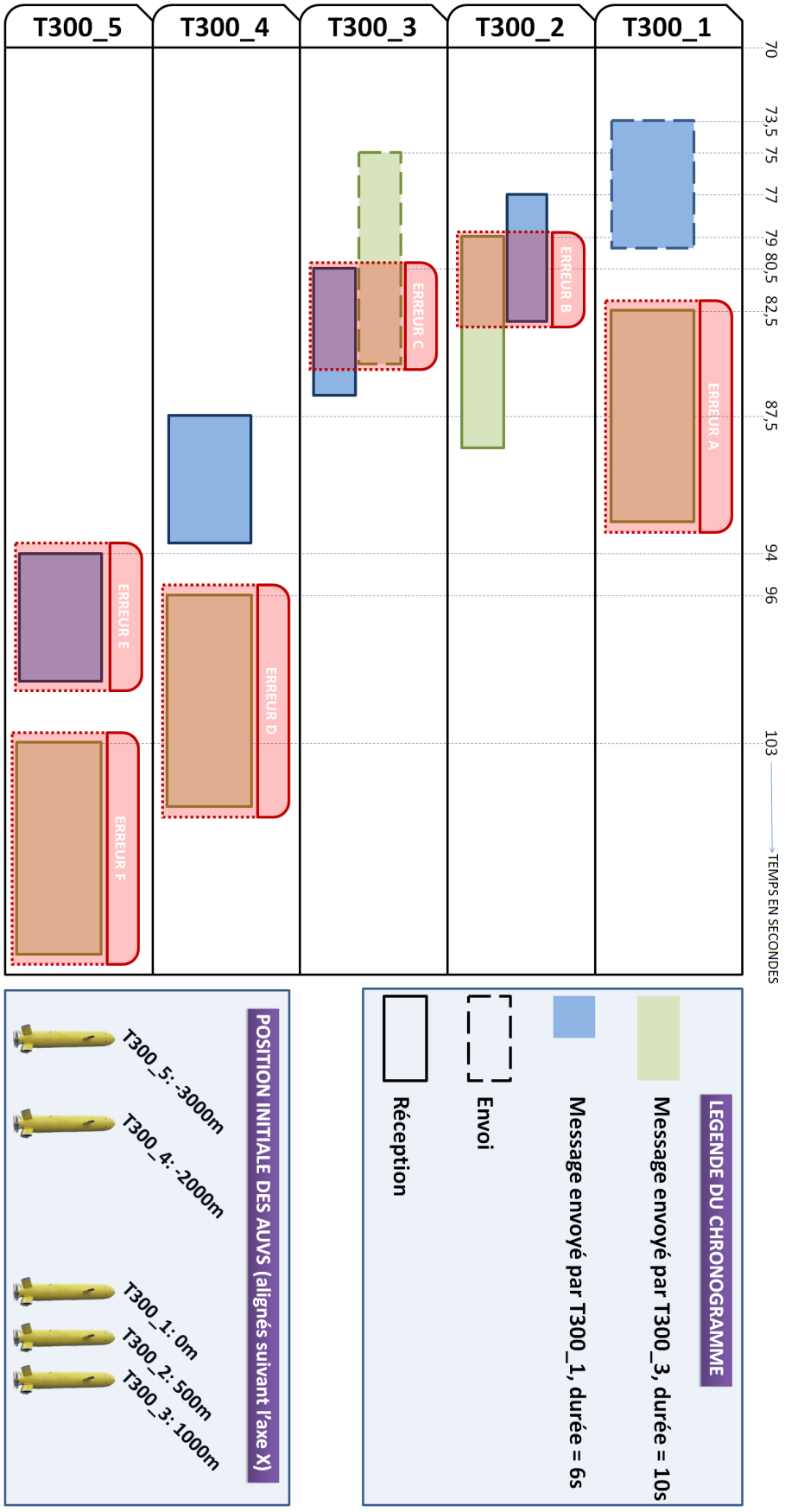


Fig. 7.16 – Chronogramme des échanges entre les véhicules durant la simulation.

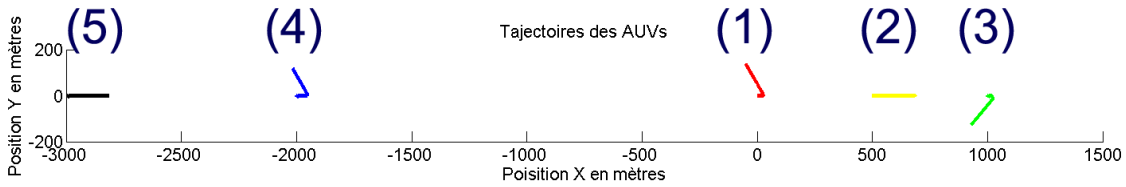


Fig. 7.17 – Représentation de la trajectoires des véhicules dans le plan XY

Cette simulation permet également de mettre en évidence le retard dû aux contraintes liées au moyen de communication et au medium de propagation : la figure 7.18 montre clairement que T300_4 ne change pas immédiatement de cap lorsque T300_1 prend cette décision. En effet il s'écoule environ $14s$ entre la fin de l'émission du message et le changement de cap effectif par T300_4, soit 20s entre le moment où T300_1 change effectivement de cap et le moment où T300_4 fait de même. Cela correspond au temps d'émission ($6s$) et au temps de propagation (environ $2000m$ à parcourir à $174m.s^{-1}$). T300_4 parcourt donc une distance au cap 0° plus grande que celle à laquelle on aurait pu s'attendre.

Enfin un dernier point qui n'a pas été présenté ici est à mentionner. Ce simulateur est capable de gérer les *multi-émissions* : le cas où un modem émet un nouveau message alors qu'une ou plusieurs ondes issues de son antenne sont encore en cours de propagation, est pris en charge. Cette remarque peut paraître aller de soi, mais cela augmente considérablement la complexité du problème car il est nécessaire de gérer dynamiquement plusieurs contextes de façon simultanée ; un contexte est une solution particulière de l'évolution d'une onde émise par un moyen de communication d'un véhicule. Dans ce contexte, les dates de délivrance des messages, les temps de propagation, les atténuations, ... sont en permanence re-calculées. Plusieurs contextes peuvent donc être rattachés à un moyen de communication en particulier et il est nécessaire de les créer et de les détruire de façon dynamique afin de gérer au mieux la mémoire disponible.

A travers cette simulation, nous constatons qu'intégrer un modèle de communication prenant en compte les phénomènes de latence, d'interférences, d'atténuation et de comportement des moyens de communication s'avère être une nécessité dans le cadre de la coordination multi-véhicule. En effet, les simulations réalisées dans des simulateurs ne présentant pas de tels modèles différeront fortement de la réalité (en particulier pour les communications sous-marines) et il n'est alors guère possible de valider des solutions de coordination.

7.4 Conclusion

Dans ce chapitre, nous avons présenté la méthode et les résultats de validation de notre architecture. Nous avons vu que bien que n'étant pas encore temps réel au sens strict du terme, son comportement temporel est satisfaisant dans notre cadre de travail. Bien entendu, n'étant pas temps réel, il n'est pas possible de garantir une réponse en un temps borné ; pour prendre en compte cette possibilité, nous avons temporairement mis en place un mécanisme d'alerte en cas de débordement temporel significatif, nous permettant de décider de la validité de la simulation.

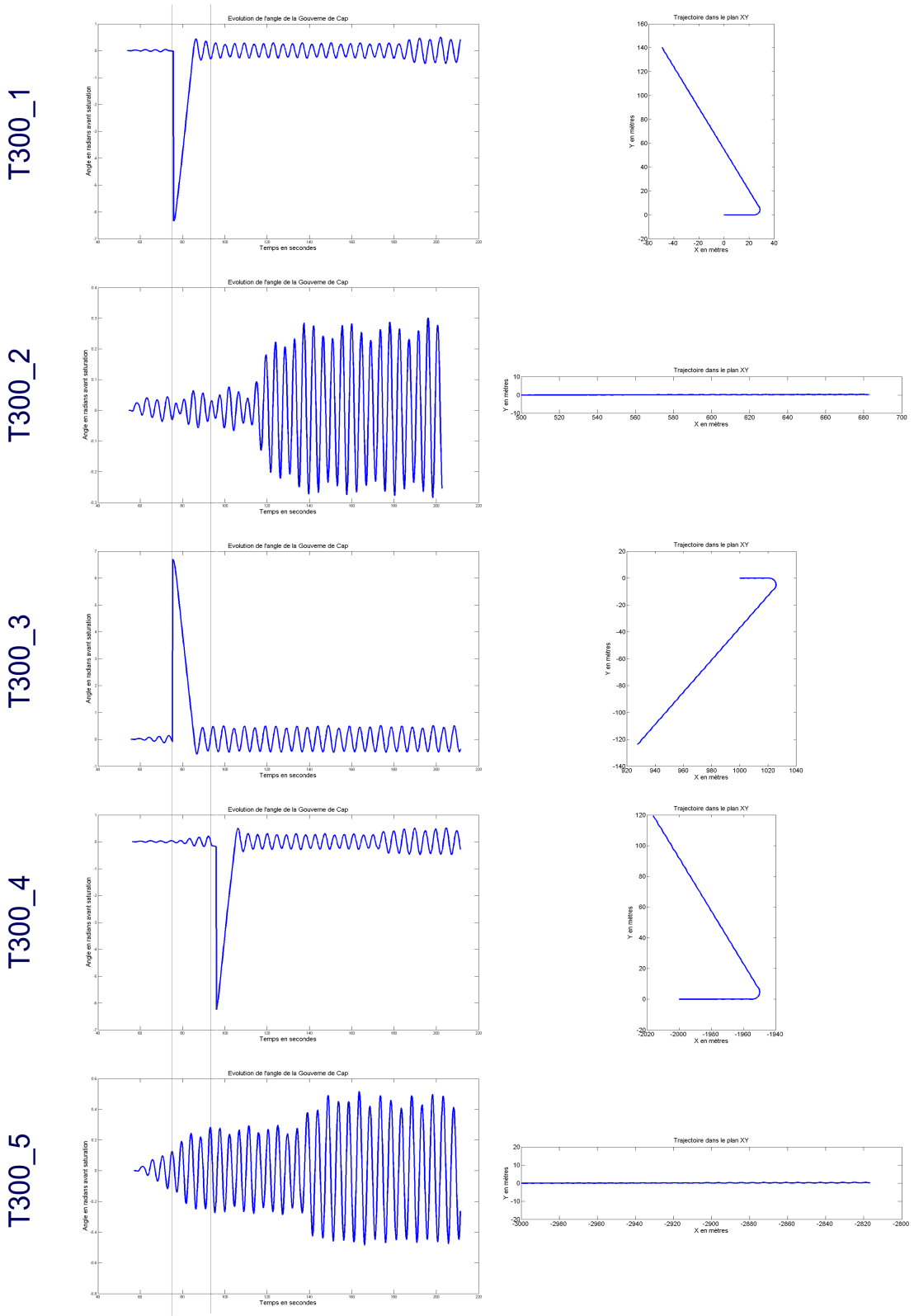


Fig. 7.18 – Evolution de l'angle des gouvernes de cap de chaque véhicule et trajectoire associée dans le plan XY.

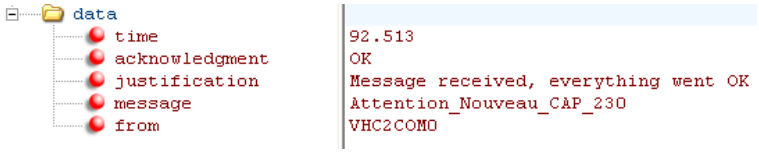


Fig. 7.19 – Extrait du fichier résultat de la simulation [Communications].

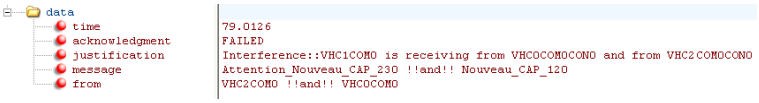


Fig. 7.20 – Extrait du fichier résultat de la simulation [Communications].

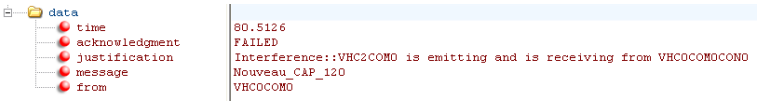


Fig. 7.21 – Extrait du fichier résultat de la simulation [Communications].

Fig. 7.22 – Extrait du fichier résultat de la simulation [Communications].

Cette validation a porté sur le module de base de l'architecture, puis sur le lien unissant les différents simulateurs. Enfin, un rapide calcul nous a montré que le temps de cycle de simulation était bien inférieur à un dixième de la période de la commande, ce qui était bien l'objectif recherché.

La validation des modèles utilisés dans ce simulateur nécessite un travail conséquent qui sort du cadre de cette thèse. En ce qui concerne la dynamique des véhicules, nous avons donc implémenté des modèles connus (Fossen notamment pour les AUVs) et utilisé des paramètres que nous avons affinés sur plusieurs expérimentations réelles. Une validation qualitative est à prévoir lorsque nos véhicules seront à nouveau opérationnels. En ce qui concerne la modélisation des capteurs, nous avons respecté la fréquence d'échantillonnage, la portée, la nature et l'amplitude du bruit. Bien qu'une validation soit nécessaire, celle-ci nous semble pour l'instant un problème annexe.

Enfin, nous avons présenté quelques simulations typiques afin de mettre en exergue les comportements potentiellement divergents des véhicules soumis à différentes perturbations. Cela nous a permis de mettre en avant les propriétés temps réel du simulateur, de

valider le découplage commande / évolution de la dynamique des robots, et enfin de montrer l'impact du modèle de communication sur le comportement d'une flottille d'engins sous-marins.

Chapitre 8

Configuration et Exploitation

8.1 Introduction

Après avoir validé notre simulateur, nous présentons dans ce chapitre quelques exemples concrets de son exploitation. Il est entendu, que le but n'est pas ici de proposer de nouveaux algorithmes ou de nouvelles lois de commande. Nous souhaitons juste mettre en exergue au travers des exemples proposés, les problématiques liées au contexte de la coordination multi-véhicule et pouvant être pleinement abordée avec le système de simulation que nous proposons.

Nous allons donc effectuer une série de missions typiques. Chacune de ces missions a été imaginée pour montrer les aspects d'un problème particulier. En effet, les problématiques liées à la coordination multi-véhicules sont nombreuses et nous avons volontairement limité le nombre de simulations ici. Les simulations que nous présentons ne constituent donc pas une liste exhaustive de ce que peut faire le simulateur ; nous laissons au lecteur le soin d'extrapoler à partir des informations présentées les scenarii envisageables en simulation à l'aide de cet outil.

Ce chapitre propose donc des simulations de complexité croissante pour résoudre au fur et à mesure, les problèmes qui se sont présentés. Toutes ces simulations tournent autour du thème de l'acquisition d'un panache d'eau douce sous-marin (issus d'une source karstique par exemple) par un ou plusieurs véhicules. Il s'agit là d'un exemple qui pourrait bien sûr être remplacé par autre chose (relevé bathymatrique, détection de mines...).

Nous débutons ce chapitre par une section présentant la configuration d'une simulation. Nous montrons rapidement les interfaces que nous avons développées afin de faciliter la prise en main de cet outil. Puis nous exposons le représentation de la source que nous avons créée et qui sera acquise (ou du moins ils vont essayer...) par un robot autonome dans un troisième temps. Le temps d'acquisition étant élevé, nous réalisons la même manipulation mais avec deux véhicules qui débutent cette fois-ci la mission par une phase de recherche. Puis la même mission est effectué en présence d'un faible courant et nous constatons que les véhicules ne sont plus en mesure de produire des données spatiales cohérentes. Le dernier point consiste donc à recourir à un troisième véhicule qui est cette fois en surface. Ce dernier peut alors envoyer aux véhicules sous-marins leur position réelle,

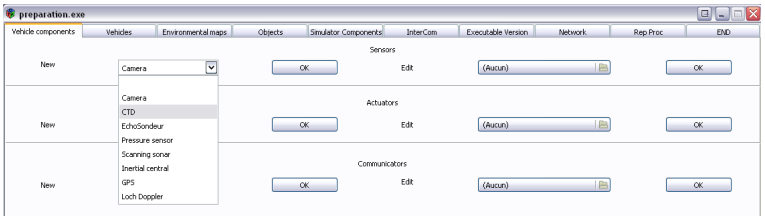


Fig. 8.1 – Interface de création et d'édition des composants véhicules

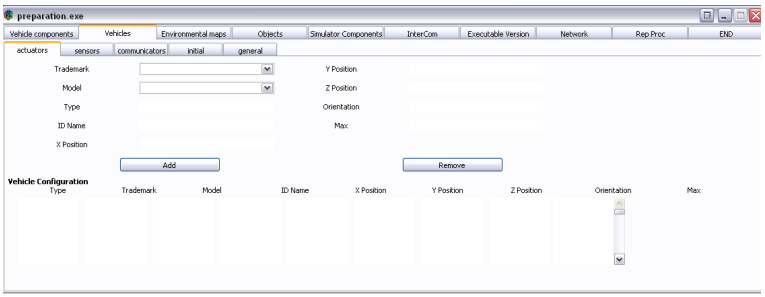


Fig. 8.2 – Interface de création et d'édition des véhicules

grâce à un algorithme réputé connu.

8.2 Configuration d'une simulation

Comme nous l'avons précédemment évoqué, lancer une simulation nécessite une phase de création des fichiers de configuration. Ces fichiers sont décrits en langage XML ce qui assure une certaine interopérabilité. Néanmoins l'édition de ces fichiers est fastidieuse pour une personne non initiée. Nous avons donc développé au travers de deux stages (V. Geny et J. Adhami) une interface de création et d'édition des fichiers de configuration nécessaires à une simulation. Nous présentons rapidement certains des onglets de ces interfaces. Ainsi nous représentons figure 8.1 l'onglet de l'interface dédiée à l'édition et à la création des composants d'un véhicule. On retrouve donc sur trois lignes les capteurs, les actionneurs et les moyens de communication. Chaque véhicule peut alors être composé à l'aide des fichiers ainsi créés (figure 8.2). On retrouve cinq sous-onglets et de menus déroulant permettant d'accéder facilement aux composants créés. Une fois les véhicules créés une seconde interface (qui sera fusionnée avec la première dans le futur) permet de choisir les entités participantes (véhicules, bouées, cartes) et leur position initiales pour la simulation (figure 8.3). Un second onglet nous permet de déterminer quels sont les exécutables à utiliser pour lancer la simulation. Cela permet de choisir les versions des processus et donc de personnaliser une simulation si nécessaire. Un troisième onglet permet de définir les interactions entre les capteurs et moyens de communications interférents. On peut ainsi définir le sens dans lequel l'interaction à lieu (A perturbe B, mais B ne perturba pas obligatoirement A), et la nature de cette interaction (sensible ou possibilité de communication). D'autres onglets permettent de choisir les ordinateurs et les processeurs sur lesquels seront exécutés les composants du simulateur. Chaque étape peut enregistrée sous forme d'un "fichier projet" évitant ainsi d'avoir à redéfinir plusieurs fois les même configurations. Nous avons choisi de développer cette interface en GTK+ afin

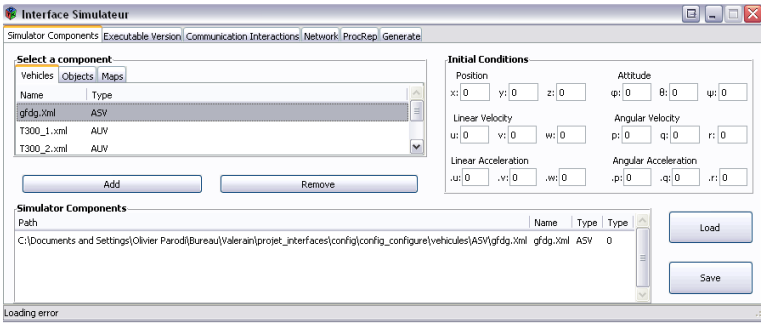


Fig. 8.3 – Interface permettant déterminer et placer les entités participantes

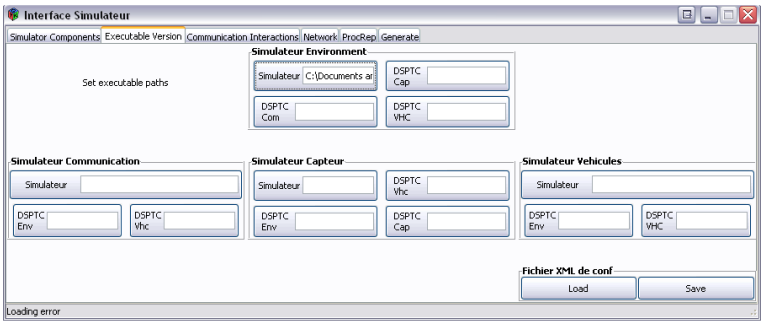


Fig. 8.4 – Interface permettant de choisir les exécutables à distribuer sur le réseau

de garantir l'indépendance de sa plateforme d'exécution (Mac, Windows, Linux).

Il reste encore un travail conséquent à fournir pour rendre cette interface pleinement opérationnelle. En effet, nous ne gérons pas pour l'instant le lancement de la mission. Seul le superviseur en est actuellement capable, et son interface en ligne de commande présente une ergonomie pour le moins limitée.

8.3 Acquisition d'un panache d'eau douce sous-marine

8.3.1 Représentation d'une source d'eau douce sous-marine

Afin de réaliser une mission opérationnelle en simulation, nous avons commencé par établir une représentation d'une source d'eau sous-marine. Le modèle employé ne s'appuie sur aucun fondement mathématique, mais nous avons essayé de tenir compte d'une certaine réalité. Ainsi, pour générer les données (au format netCDF donc), nous spécifions la hauteur du panache, le diamètre du noyau de la source à la base, le diamètre du noyau en surface. Ce noyau est constitué majoritairement d'eau douce, mais notre modèle permet d'augmenter les inclusions d'eau salée dans ce noyau proportionnellement à la distance entre le point considéré et la surface. Outre le noyau, la source est constituée d'une enveloppe périphérique dont le diamètre croît de façon exponentielle. Dans cette zone périphérique, le mélange eau salée/eau douce croît également avec le diamètre de l'enveloppe au point considéré. Des fonctions de bruit gaussien nous permettent d'augmenter le facteur aléatoire de la forme générée. Enfin nous avons considéré un courant déplaçant la source sur l'axe longitudinal. En remontant vers la surface, la source se décale donc proportionnellement au niveau considéré. La figure 8.5 présente le modèle de cette source. A gauche, nous voyons clairement que la source se décale avec la hauteur du panache. On distingue également le noyau (en bleu) avec des inclusions qui augmentent petit à petit. La zone périphérique, quant à elle, est croît plus rapidement que le noyau, ce qui est bien conforme au modèle retenu. La figure en haut à droite présente une vue en transparence de sous la source (on regarde vers la surface donc). On voit clairement au premier plan le 'petit' noyau, puis en transparence le décalage de la source qui s'agrandit. La figure juste en dessous, présente une série de coupe de cette source permettant de mieux la visualiser (attention l'axe de la profondeur est inversé par rapport au sens conventionnel : en robotique sous-marine, on prend généralement le sens croissant de cet axe, dirigé vers le bas).

8.3.2 Acquisition virtuelle de la source d'eau douce par un AUV

Le premier objectif est de réaliser une acquisition virtuelle de cette source par un véhicule. Pour se faire, nous définissons une mission de type "râteaux" à l'AUV. La mission consiste à lui faire une série de changement de cap au bout de temps déterminé (consigne cap - temps) et ce, à différentes profondeur.

Il est nécessaire ici de préciser les provenances des données affichées. Ces données peuvent provenir soit du simulateur de véhicule, soit du véhicule lui-même. Il faut donc bien faire le distinguo entre données mesurées et données réelles.Les premières proviennent du véhicule : elles sont constituées soit des valeurs capteurs brutes, soit sont le fruit d'un calcul effectué par le robot (recallage par exemple). Les secondes proviennent du simulateur et représentent la réalité. Nous qualifions par la suite de virtuelles les données issues du simulateur et de réelles les données issues du véhicule.

L'objet de cette première simulation est donc de présenter les résultat d'une acquisition virtuelle faite par l'AUV sur la source. La figure 8.6 représente les résultats de cette acquisition. En haut à gauche, nos avons tracé l'évolution de la profondeur du véhicule. On retrouve bien les différents paliers auquel il a effectué sa mission. Les trois autres graphiques adjacents présentent l'évolution des vitesses linéaires au cours de la mission. Il

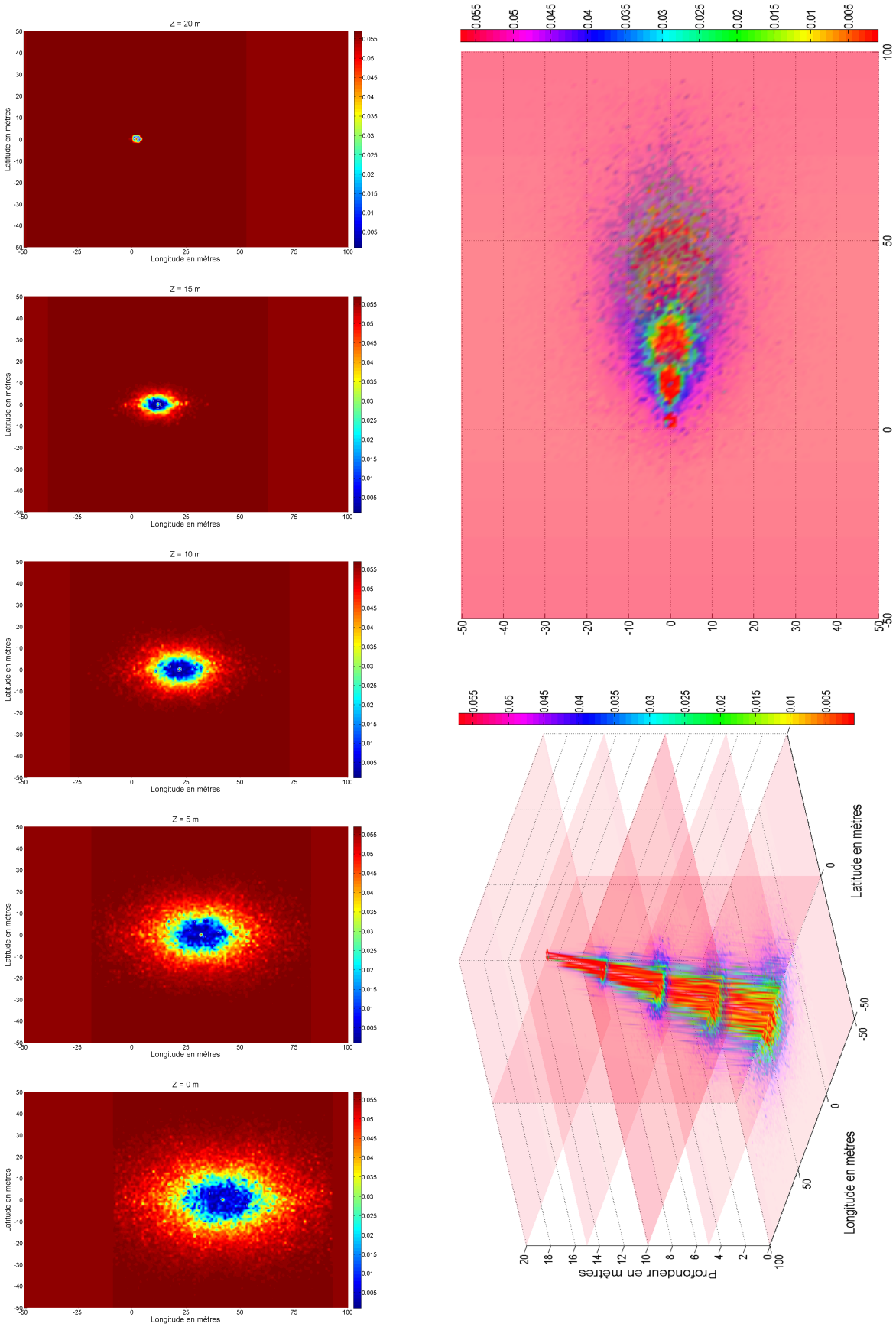


Fig. 8.5 – Représentation de la source d'eau douce sous-marine

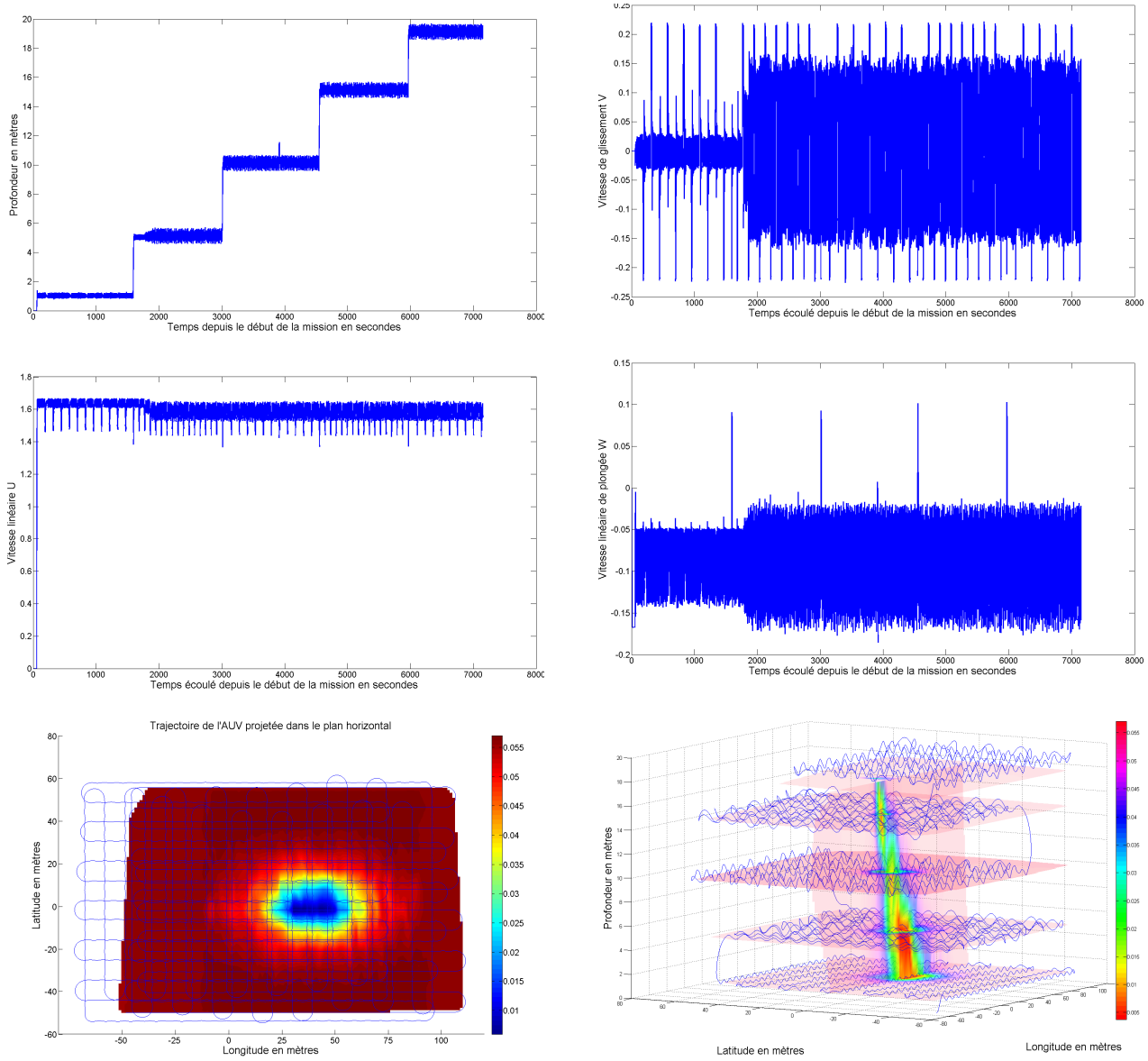


Fig. 8.6 – Evolution du véhicule

est à noter que le robot considère que sa vitesse d'avance linéaire est de $1m/s$ et que ses vitesses de glissement (embardée) et de descente (pilonement) sont nulles. Nous verrons plus tard les conséquences que cette hypothèse a sur les relevés effectués. Les deux figures en dessous présentent quant à elles les déplacement du véhicule par rapport à la source, d'une part dans le plan horizontal et d'autre part dans un volume 3D. Les vues de la source sur cette figure sont donc celles reconstituées à partir des données issues du simulateur.

La figure 8.7 présente quant à elle un comparatif des relevés effectués. La colonne de gauche représente la source d'eau douce interpolée à différents niveau à partir des acquisitions virtuelles (les données (trajectoire réelle + valeurs CTD envoyées au véhicule donc) issues du simulateur).

Le colonne du milieu, quant à elle, représente les mêmes vues mais interpolée à partir des données perçues par le véhicule. On note qu'on observe sensiblement la même chose, à la différence près que la localisation de la source est erronée. Les hypothèses de la

vitesse d'avance constante à $1m/s$ se révèle ici être un facteur d'erreur de géo-localisation important. Il est noter que la forme d'une torpille implique effectivement des vitesses de glissement et de descente très faible (cf figure 8.6). Ceci n'est pas le cas de tous les véhicules ; par conséquent se genre d'erreur pourrait bien impliquer une déformation de la forme observée en plus des translations évoquées ici.

Enfin la troisième colonne représente les résultats d'une interpolation des données acquises par le véhicules pour la même mission, mais en présence de bruit. On distingue donc toujours correctement la forme de la source, approximativement au même emplacement. On arrive même à percevoir à travers le bruit de fond, le sens de déplacement du véhicule pendant l'acquisition. En revanche à partir de la profondeur de $Z = 10m$, la source se décale franchement à l'est. Ceci est dû à une petite expérience que nous avons mené et sur laquelle nous revenons dans la section suivante.

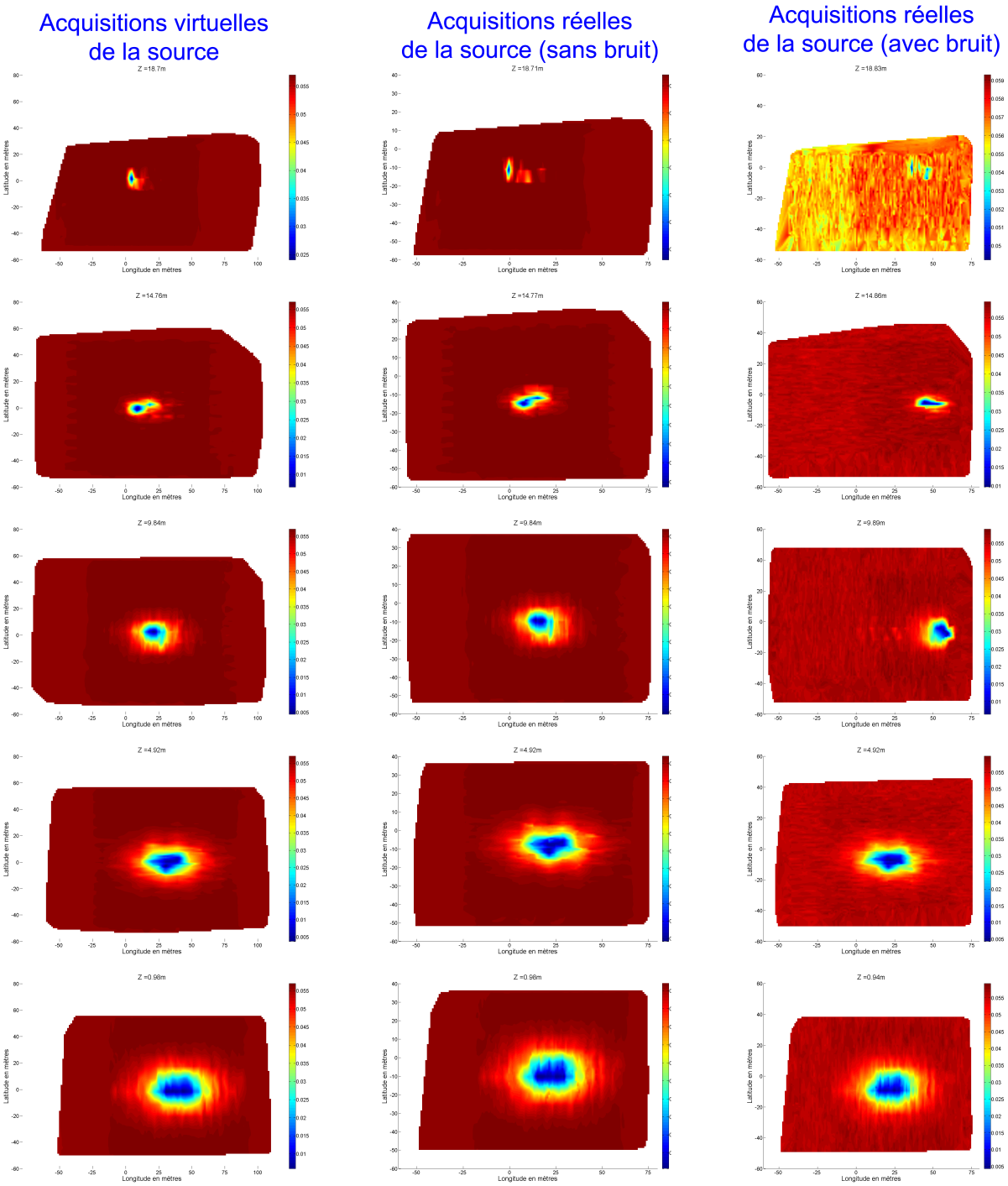


Fig. 8.7 – Acquisitions de la source dans différentes conditions

8.3.3 Acquisition réelle de la source d'eau douce par un AUV

8.3.3.1 Sans bruit capteur

La figure 8.8 présente l'évolution du véhicule mesurée par le véhicule. Cette simulation se fait en l'absence de bruit capteur. On retrouve sur la figure du haut les cinq paliers de travail du véhicule, et la reconstruction de la source perçue semble cohérente à des translations près. La dernière figure présente les déplacements du véhicule à travers la source. L'accomplissement de cette mission ne pose donc pas de problème particulier à ce stade, mise à part le recalage de la centrale inertielle pour le refrènements spatiale des mesures effectuées.

8.3.3.2 Avec bruit capteur

La figure 8.9 présente les données issues du véhicules pour la même mission que précédemment mais en présence de bruit capteur. On retrouve donc bien les "strilles" caractéristiques du bruit du CTD sur les deux projections dans le plan de l'interpolation de la source. La première figure présente l'evolution de la profondeur du véhicule. Un petit détail est à noter ici sur lequel nous allons revenir : sur le troisième palier, on distingue une sorte de 'trou' dans les données. Les deux figures en dessous présentent à gauche, le trajet du véhicule vu par le véhicule, sur la source à $Z = 1m$ de la surface. Vu du véhicule la trajectoire semble parfaite. En revanche lorsqu'on observe la trajectoire de droite, il n'en va pas de même. En effet, nous nous somme livrés ici à une petite expérience permettant de montrer une fois de plus l'importance de l'aspect temps-réel du système de simulation.

La manipulation a consisté à suspendre l'exécution du simulateur de véhicule pendant 40s. Le fait de stopper le simulateur de véhicules, n'empêche en aucun cas au simulateur de capteur de fonctionner. En revanche ce dernier va envoyer les même valeurs capteur aux véhicules pendant les 40s, au bruit près. C'est effectivement ce qu'on distingue sur la courbe de pression relevé par le véhicule : un trou sur le troisième palier avec des valeur bruitées autour de la dernière valeur calculée par le simulateur de véhicule et envoyée au simulateur de capteur. Par conséquent l'AUV continue à évoluer avec des données erronée et calcule donc une position fausse. Puis au bout de 40s le simulateur se remet en route. L'AUV tourne quelques instants plus tard... Par conséquent du point de vue du simulateur le râteau est plus court, puisque l'AUV était virtuellement statique durant sa phase de sommeil. L'interprétation des données est donc faussée si l'on n'est pas en mesure de détecter ce genre d'événement, qui, il est vrai n'est jamais survenu pour l'instant. Néanmoins, notre système ne garantissant pas l'aspect temps-réel, il est nécessaire de se prémunir de mécanismes de détection pour éviter toute interprétation erronée de l'exploitation du simulateur.

Enfin les deux derniers graphiques représentent l'evolution dans le volume vu par le véhicule (le premier) et calculé par le simulateur (le deuxième).

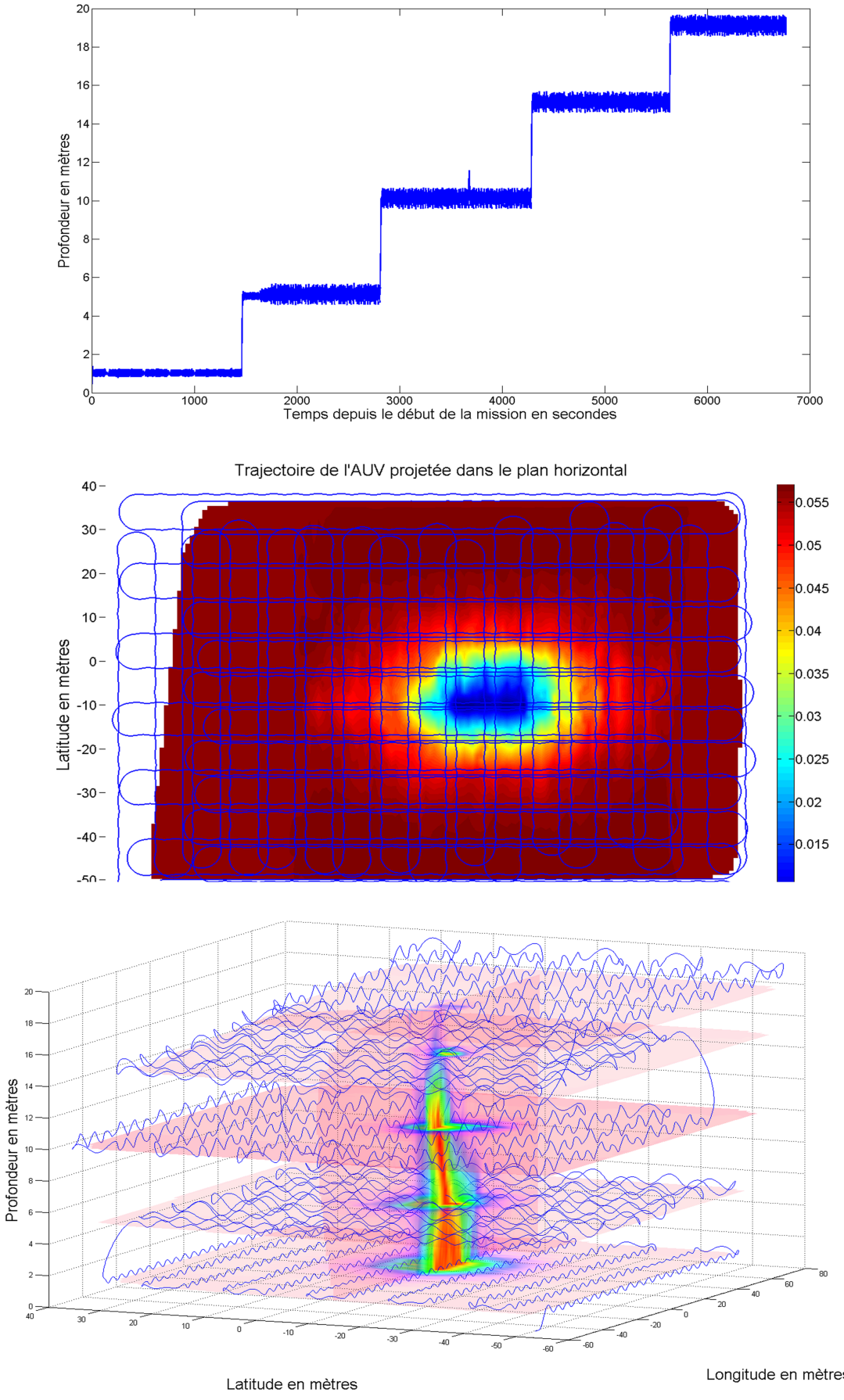


FIG. 8.8 – Déplacement du véhicule réel sans bruit capteur

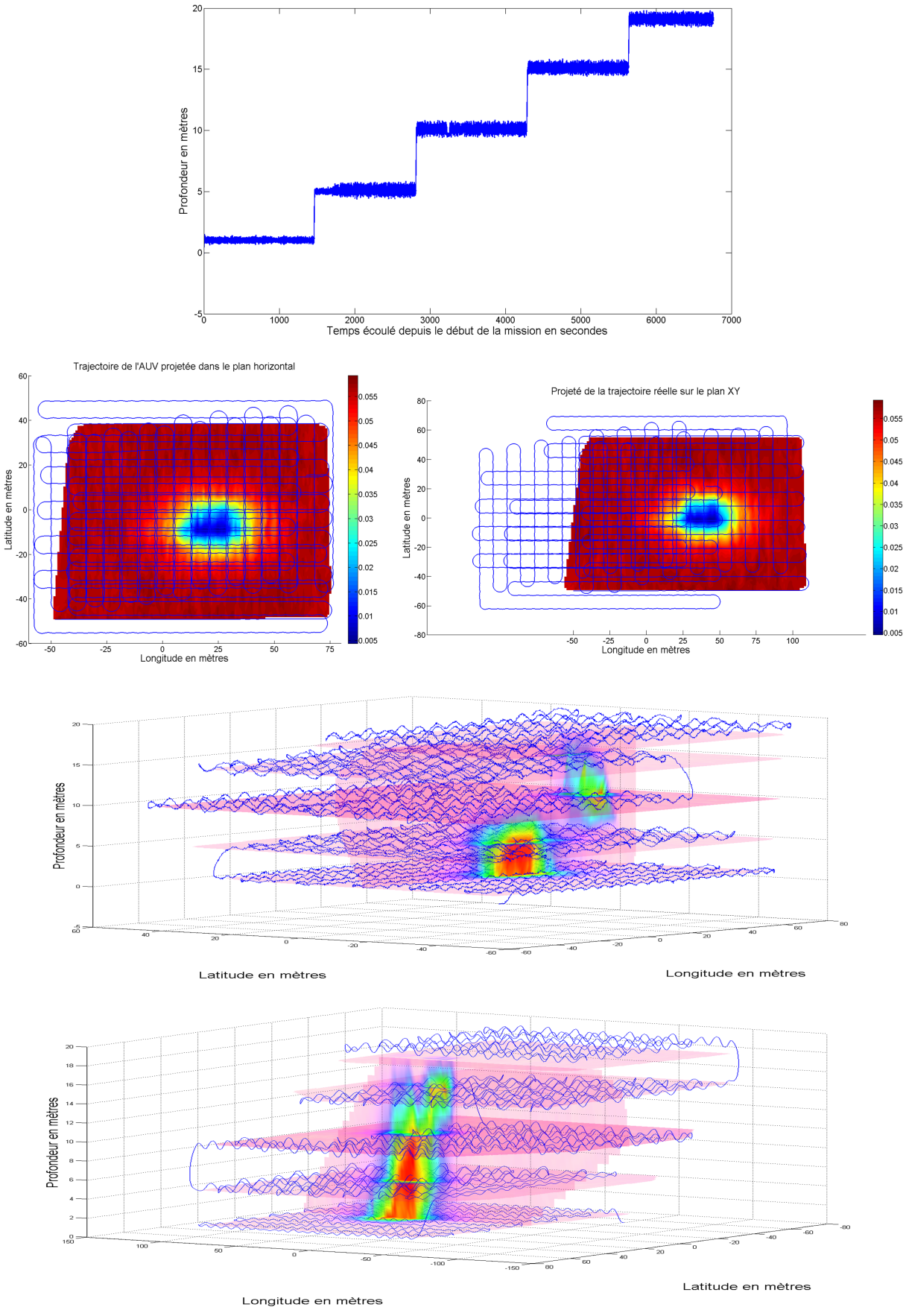


Fig. 8.9 – Déplacement du véhicule réel en tenant compte du bruit capteur

8.4 Interactions entre deux véhicules homogènes

Nous présentons maintenant une simulation dans laquelle interviennent deux véhicules de type AUV. L'objectif est de trouver une source d'eau et le cas échéant, d'envoyer une requête au second véhicule afin d'accélérer le processus de capture. Les deux véhicules sont donc placés à différents endroits de la carte, et font des "zigzag" en essayant de détecter un gradient de conductivité. Lorsqu'une source est détectée, l'AUV détecteur envoie un message indiquant les coordonnées de rendez-vous, le type de mission et les paliers de râteaux à effectuer. En même temps, il fait route vers un waypoint calculé relativement au point de détection afin d'entamer lui aussi une série de râteaux. Il est à noter que le ralliement de ces points se fait en surface pour nous assurer du géo-refrènement correct des échantillons (au moins pendant un certain temps). Nous présentons ici deux missions identiques. La première est effectuée sans courant, alors que dans la seconde nous simulons un courant faible de l'ordre de $0.2m/s$ d'est en ouest.

8.4.1 Sans courant

la figure 8.10 présente l'évolution de la position des véhicules au cours du temps et à différentes étapes clé.

Sur la première figure, on peut distinguer la trajectoire suivi par les deux AUVs et la source telle qu'elle est simulée par l'environnement à la profondeur d'immersion de travail des véhicules. Durant cette étape l'AUV1 survole la source (il faut dire qu'il était bien placé au départ...). Cet image correspond au moment où l'AUV1 émet le message (figure 8.11) et s'apprête à rejoindre un waypoint calculé relativement au point de détection.

Sur la seconde figure, on distingue l'interruption de la mission courante de l'AUV2, qui se dirige maintenant aux coordonnées envoyées par l'AUV1 pour y effectuer une série de râteaux à différentes profondeurs 8.12. L'AUV1 quant à lui a atteint son waypoint. L'interprétation graphique de l'échange du message est représenté figure 8.13.

L'étape trois est l'étape au cours de laquelle l'AUV2 atteint le waypoint spécifié dans le message reçu. Pendant ce temps l'AUV1 a entamé sa première branche de râteaux.

Les étapes quatre et cinq montrent les instants ou l'un des deux AUVs doit changer de palier. Enfin la dernière étape représente l'évolution des positions des véhicules durant toute la simulation.

L'évolution de la profondeur est visible figure 8.14. On y distingue le passage en surface des véhicules pour se rendre à leur waypoint avant de plonger pour effectuer la mission. La troisième figure représente la trajectoire des véhicules vue du simulateur autour du modèle de la source (et non pas de la source acquise).

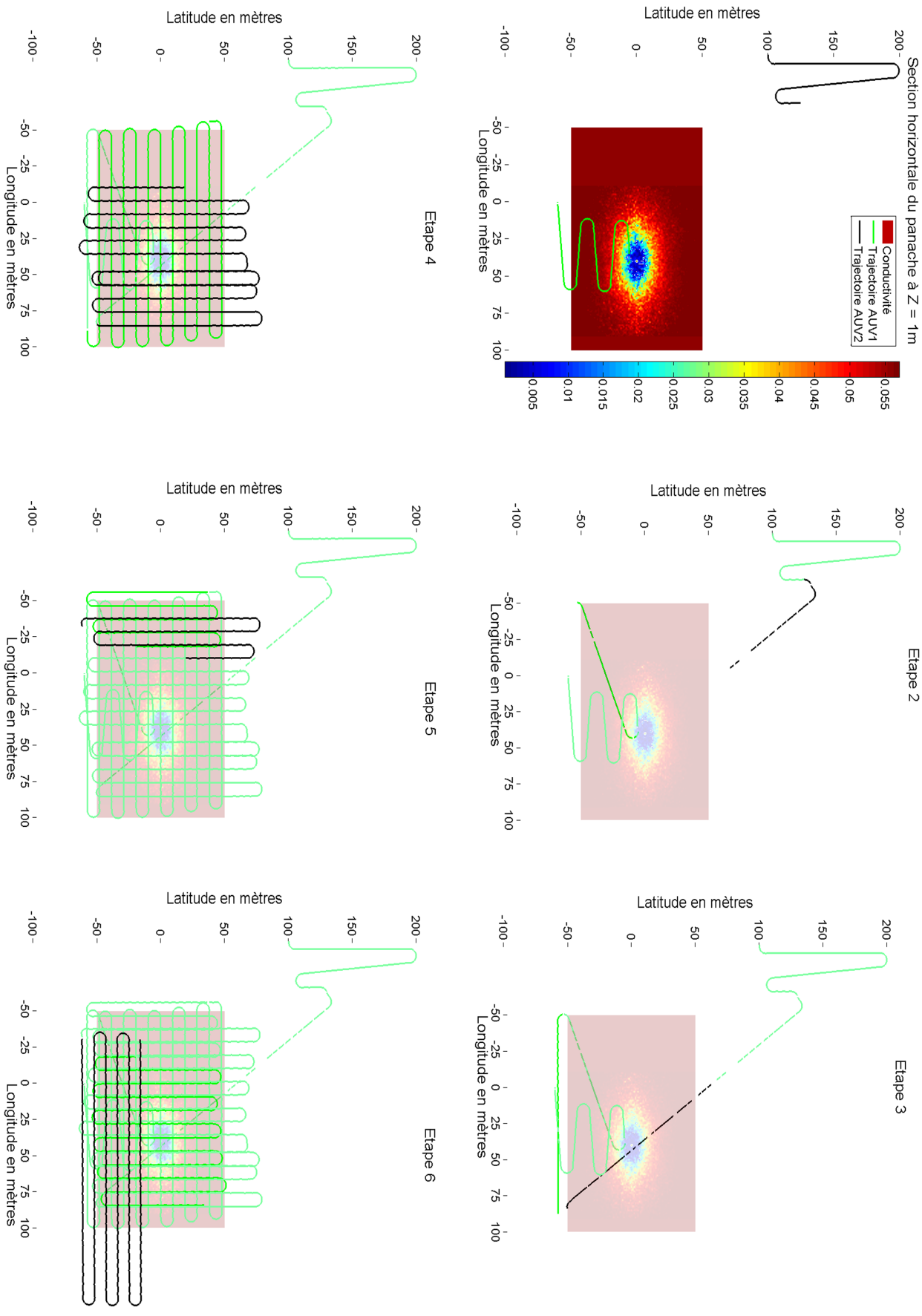


Fig. 8.10 – Acquisitions de la source par deux véhicules sans courant

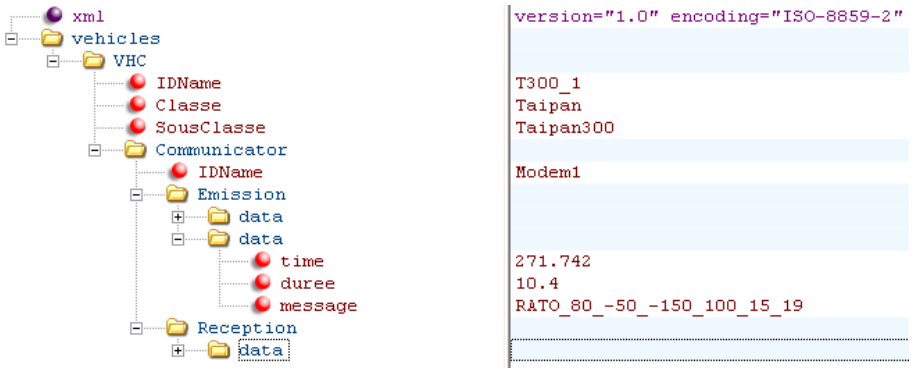


Fig. 8.11 – Extrait du fichier log relatif aux communication de T300_1

Fig. 8.12 – Extrait du fichier log relatif aux communication de T300_2

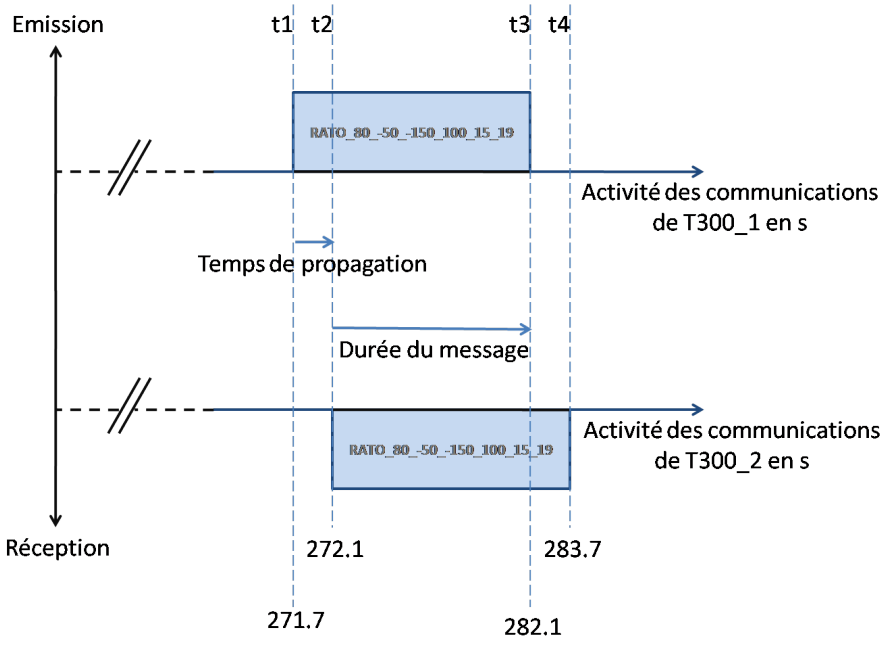
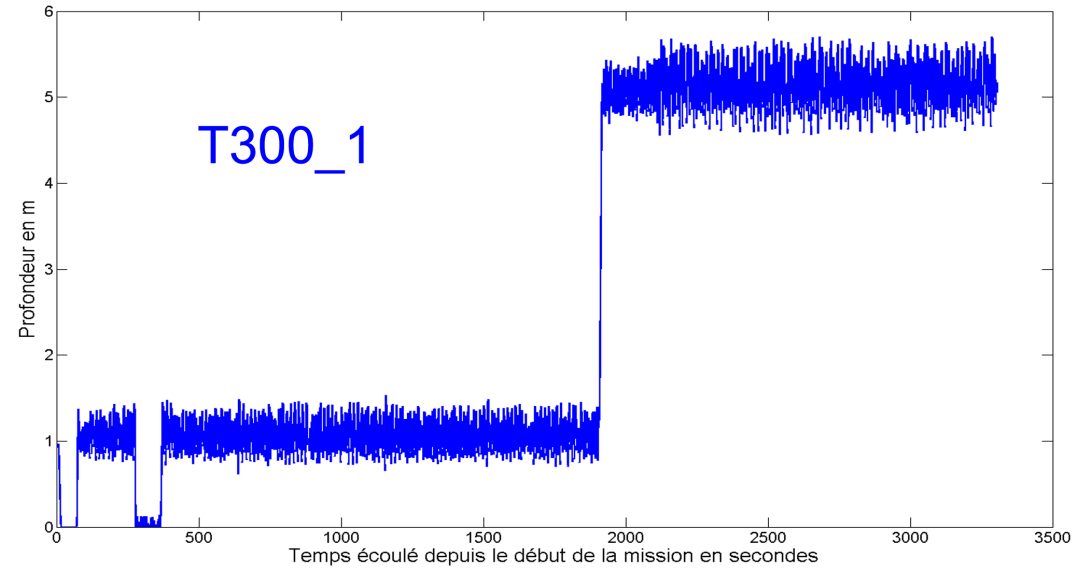
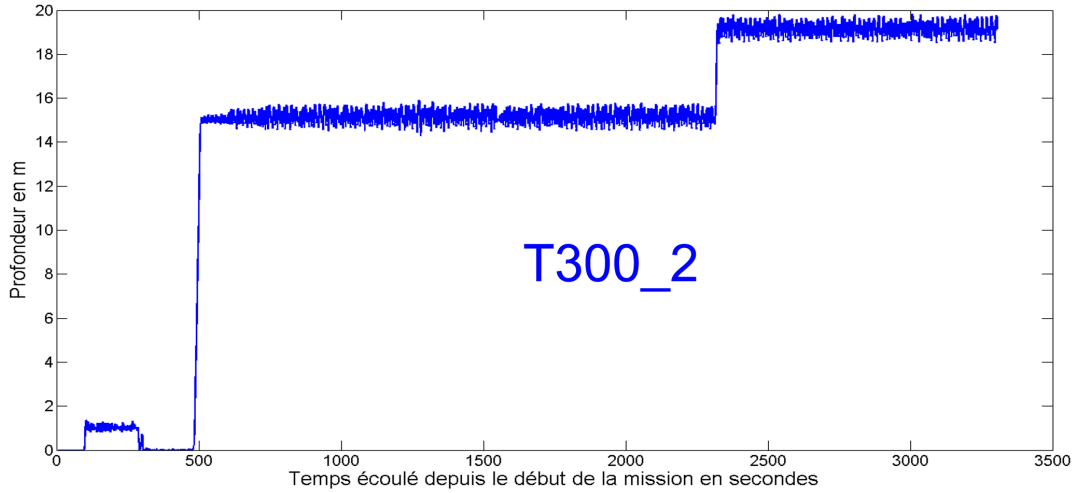


Fig. 8.13 – Diagramme des communications entre les deux véhicules





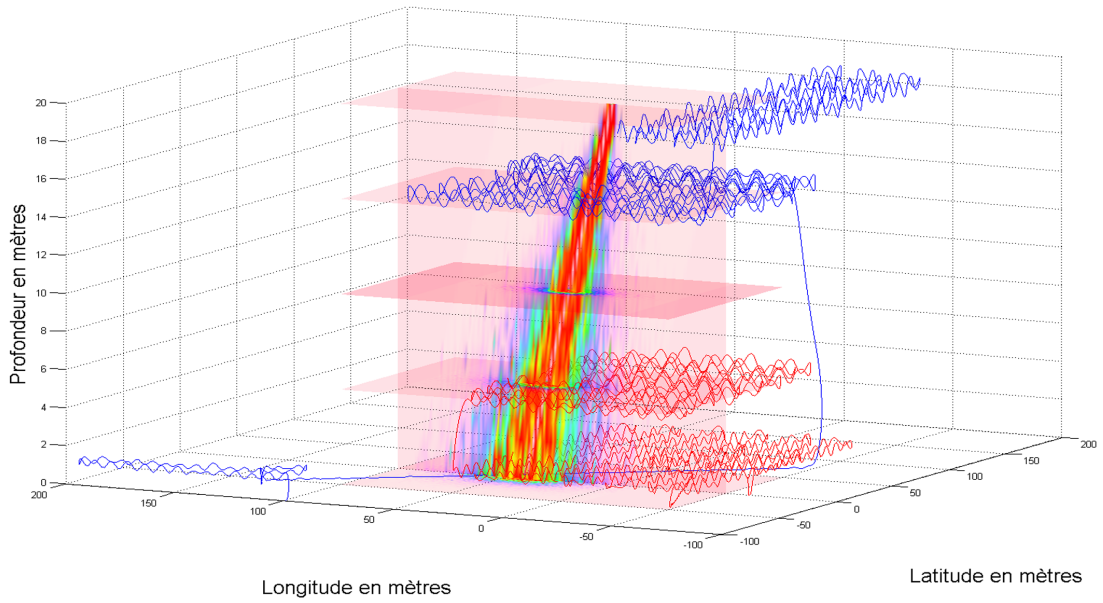


Fig. 8.14 – Evolution des véhicules

8.4.2 Avec courant

Nous effectuons maintenant la même mission mais nous introduisons cette fois-ci un courant est->ouest à $0.2m/s$. Il est à noter que la vitesse de ce courant est relativement faible par rapport à la vitesse déplacement d'un véhicule (environ $1.6m/s$). Pourtant nous allons voir à travers cete exemple que le résultat de la simulation est fondamentalement différent du résultat obtenu précédemment.

La figure 8.16 présente également les différentes étapes d'interactions entre les véhicules. Nous renvoyons le lecteur au paragraphe précédent pour une explication des trois premières étapes.

En revanche nous observons un comportement très différent de celui auquel on aurait pur d'attendre dans les étapes suivantes. En effet, les râteaux de l'AUV2 se sont transformés en série de '8'. Cela est bien sûr dû au courant. On constate également que l'AUV1 qui est colinéaire à la direction du courant se décale petit à petit vers l'est sans modifier la forme de sa trajectoire.

Durant les étapes quatre et cinq en revanche, les '8' de l'AUV2 se transforme en série de 'S' très élargie. Cela s'explique toujours parfaitement par la présence d'un courant.

Sur la figure 8.16, on représente l'évolution de la profondeur des véhicules. On remarque sur la troisième figure que on projette ici les données acquises par le véhicule sur le modèle de la source. En effet, on voit bien que lorsque l'engin se dirige vers son waypoint la ligne s'épaissit largement. Cela est du au fait que le véhicule capte sa position GPS durant cette phase. Un modèle de bruit ayant été implémenté dans le simulateur pour le capteur GPS, on observe bien des 'mini-téléportations' du véhicule le long de sa trajectoire.

Enfin les deux graphiques suivants représentent l'évolution de l'AUV1 dans le plan puis dans un volume 3D dans lequel se trouve la source. On retrouve bien la phase d'acquisition GPS, et le décalage à l'est dû au courant.

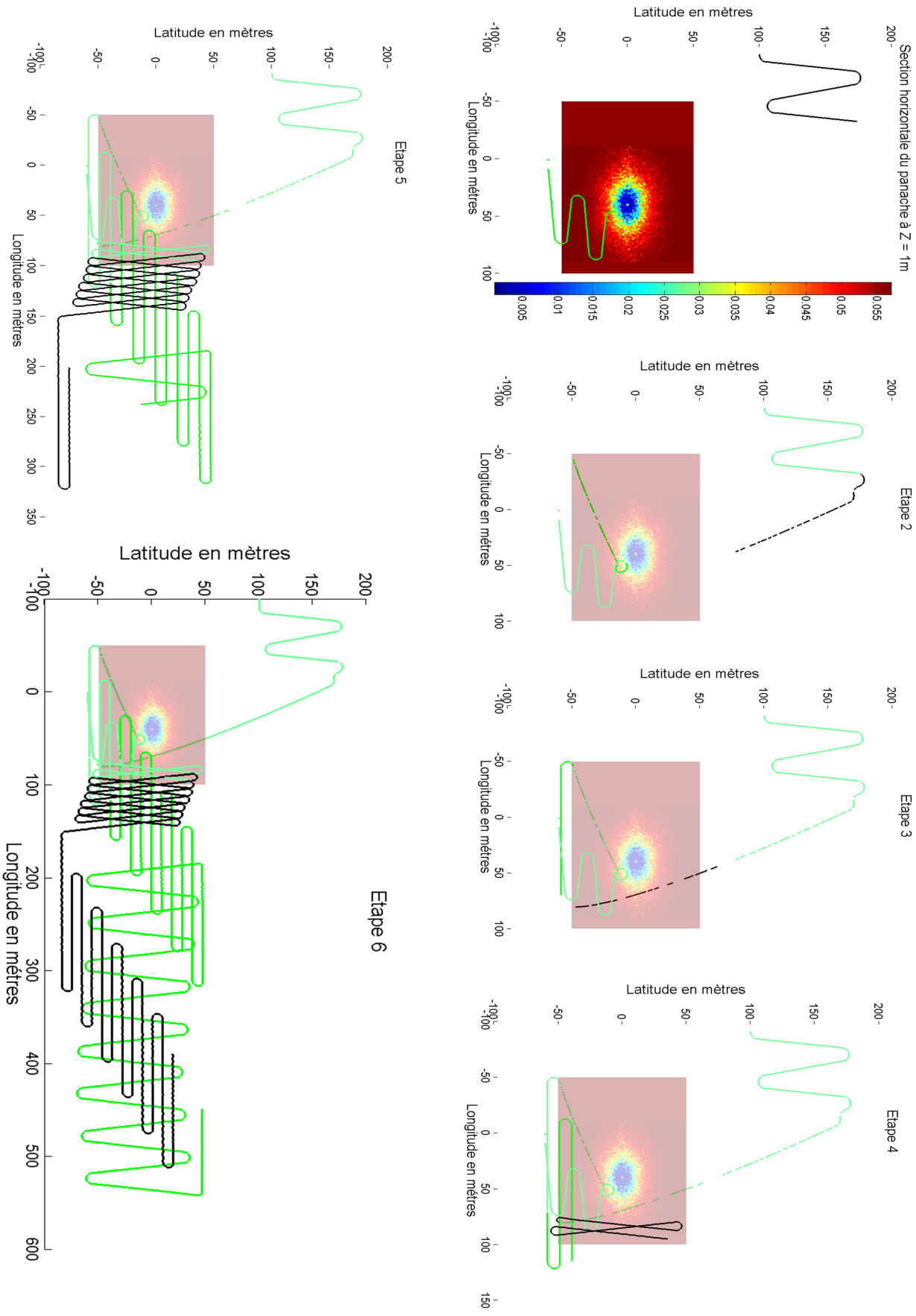


Fig. 8.15 – Acquisitions de la source par deux véhicules en présence de courant

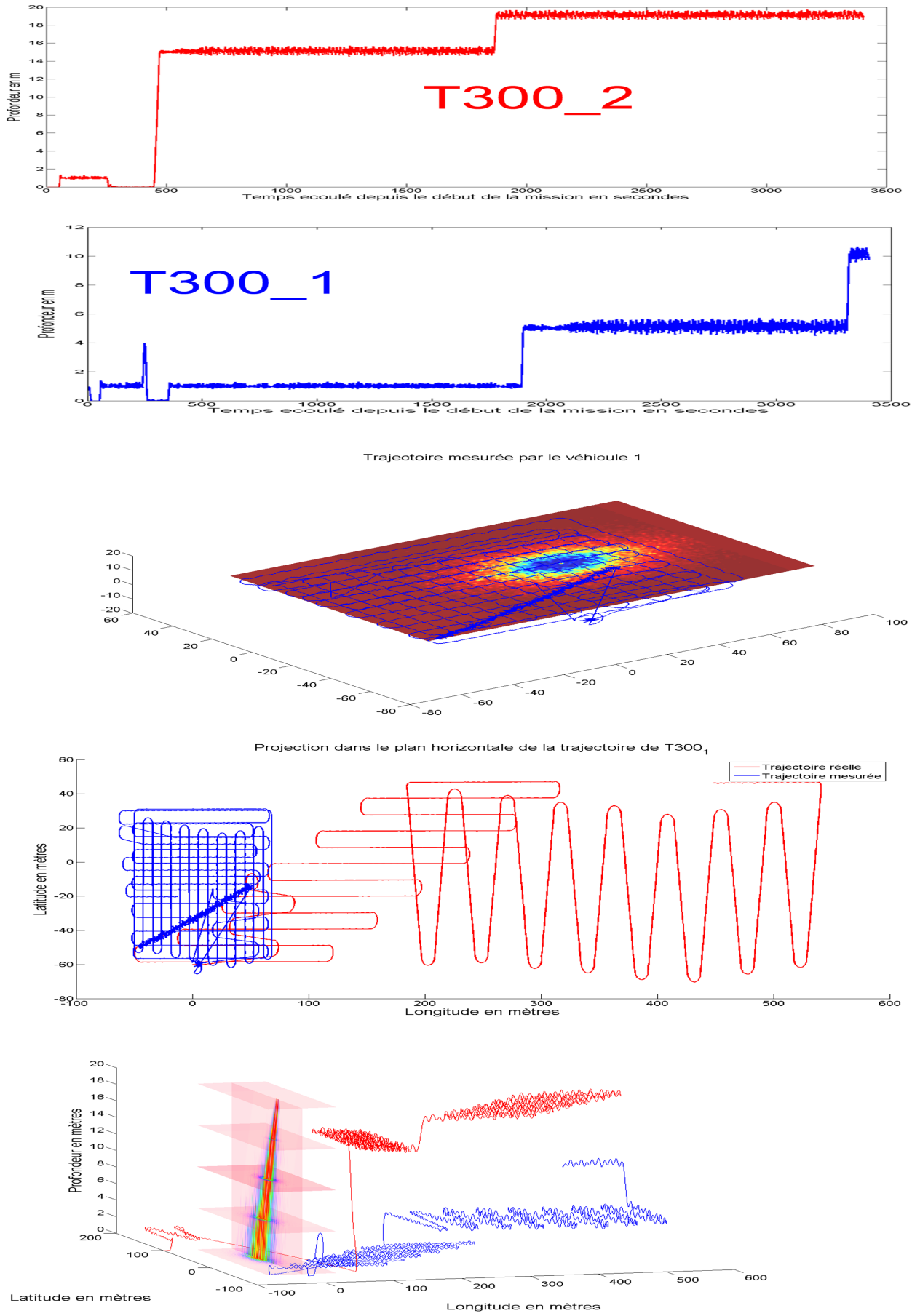


Fig. 8.16 – Evolution des véhicules

8.5 Un exemple de coopération multi-véhicules hétérogènes

Nous avons vu dans l'exemple précédent qu'en présence de courant, même faible, la réussite de la mission est pour le moins compromise. Une des solutions envisageables pour remédier au problème de la mauvaise estimation du vecteur d'état qui en résulte, est de recaler régulièrement la position du véhicule. Il est donc nécessaire de faire régulièrement parvenir au véhicule effectuant le relevé, sa véritable position.

Nous allons mettre en œuvre dans cette simulation, un AUV et un catamaran de surface. On suppose que le catamaran possède un capteur "magique" qui lui fournit la position de l'AUV. Des solutions pour résoudre ce problème existent et nous pourrons tester leur performance ultérieurement. Nous nous attachons plutôt ici à la problématique du recallage avec retard. En effet, le catamaran envoie à intervalle régulier le vecteur d'état (position, vitesse) à deux AUVs (nous n'avons pas représenté le second véhicule ici par soucis de clarté ; nous n'en parlerons donc pas). L'émission de ces vecteurs d'état prend 23.6s (maximum) avec les modems que nous possédons : nous envoyons 12 nombres à 4 digits, ce qui nous permet de faire parvenir une position au mètre près dans un volume de $1000m^3$. Le catamaran ne peut donc pas envoyer la position des véhicules à une période inférieur à environ $25s$.

Nous faisons une parenthèse ici concernant le catamaran. Nous avons implémenté le modèle décrit en B.2 dans le simulateur. Nous sommes donc en mesure de simuler simultanément des AUVs et des ASVs, ce sui confirme l'aspect multi-véhicule de Thetis. De même d'autres modèles (UAV notamment) auraient pu également être implémentés... A l'heure où nous écrivons ces lignes, nous ne sommes pas en possession du véritable contrôleur de Charlie. Nous avons donc utilisé l'architecture de contrôle développée au LIRMM, dans laquelle nous avons implémenté une loi de commande de type sliding-mode pour l'asservissement en cap. La logique de contrôle de l'ASV est donc en tout point comparable à celle de l'AUV. Nous connecterons prochainement le véritable contrôleur de Charlie au système de simulation.

La mission envisagée est identique à la précédente à la différence près que nous avons remplacé les setpoints (cap - temps) par des waypoints (atteinte d'une coordonnée dans l'espace). Bien sûr, les effets du courant sont identiques en l'absence de recallage. Un premier AUV est donc à la recherche d'une source d'eau douce, et émet un message à un second AUV (non représenté) comportant les instructions d'acquisition de la source. Le début de la mission est donc identique à la précédente 8.17. La première figure représente le moment où T300_1 a atteint la source. Sur la figure suivante, il rejoint à $Z = 1m$ (donc première erreur de positionnement car pas de GPS) le premier waypoint (nominalement $[X, Y, Z] = [-50; -50; 1]$, puis quand il "pense" l'avoir atteint commence son relevé (troisième figure). On constate à nouveau ici l'effet du courant : le véhicule se décale petit à petit vers l'est. Nominalement, les râteaux ont lieu au dessus de la source.

Jusqu'à présent, Charlie n'est pas rentré en action. Il dérive lentement dans le courant (haut des trois premières figures). Puis nous lançons sa mission qui consiste à réaliser une trajectoire en carré au dessus de la source (figure centrale). Il se dirige donc vers son premier waypoint $[X, Y, Z] = [-60; -60; 0]$ tout en émettant la positions des deux AUVs à une période variable d'environ $12s$ (le modem de Charlie réémet un vecteur d'état dès que celui-ci est libre ; la position des véhicules influent sur la longueur du message à

transmettre - 3 prend moins de temps que 876 à être émis par exemple). Au bout du temps de propagation ajouté au temps d'émission du message, T300_1 reçoit sa position d'il y a $10s < T_R < 23.6 + \delta ts$. Nous avons implémenté un algorithme de backstepping qui nous permet de recalculer la position réelle actuelle à partir des données acquises antérieures et de la position avec retard envoyée par Charlie. Actuellement, nous considérons que le retard est toujours de $23s$, ce qui est faux : ce retard varie en fonction de la longueur du message et du temps de vol jusqu'à l'AUV. Nous pourrons améliorer cet algorithme plus tard et donc améliorer la précision du positionnement de l'AUV.

Dès que T300_1 a calculé s'est recallé, il s'aperçoit qu'il a dépassé le waypoint courant (en rouge sur la figure centrale). Il fait donc demi-tour pour se diriger vers lui. Puis il l'atteint (figure en bas à gauche - waypint bleu) et passe au suivant (waypoint rouge). Par la suite il effectue de façon classique les râteaux grâce au recalage. sur la dernière figure, nous avons supprimé le début de la trajectoire pour plus de clarté. On distingue la route à suivre (chemin bleu reliant les waypoint), et la trajectoire réelle de l'AUV (en vert). On voit que le courant "fléchi" la trajectoire vers l'est ; on distingue également le moment où l'AUV se recale (inflexion de la trajectoire). Le catamaran quant à lui effectue correctement sa mission, mais est également perturbé par le courant (dans une moindre mesure puisqu'il dispose de ses coordonnées GPS en permanence).

Sur la figure 8.18 nous présentons en détail un plan de trajectoire de l'AUV. On voit la trajectoire poursuivie (les waypoints traqués pour être exact), la trajectoire réelle (simulée) de l'AUV et celle qu'il perçoit. Sur la figure du bas, nous montrons un agrandissement d'une zone de la première figure, où l'on distingue nettement le "saut" de recalage. On constate ici les limites de notre algorithme car la position calculée est relativement éloignée de la position réelle. Il est à noter qu'on ne pourra pas exactement superposer position réelle et position recalée à cause notamment des erreurs de mesure.

Enfin sur la figure 8.19, nous présentons les trajectoires suivies par l'AUV Taipan 300 et l'ASV Charlie, tous deux connectés au simulateur. La mission consiste à donner un waypoint à atteindre à Charlie au travers de son interface de programmation de mission. A réception, Charlie envoie alors ce waypoint à Taipan *via* son modem virtuel (modem simulé, car le véhicule réel - Charlie - n'en possède pas). Taipan le reçoit et se dirige à son tour vers le waypoint désigné par l'opérateur. Cette simple simulation a montré qu'il n'était pas aisé de jouer un scénario faisant intervenir des engins appartenant à des équipes différentes : les problèmes de protocole de communication entre les engins, d'interfaces de programmation de mission et de connectivité au simulateur ont été clairement mis en évidence et un travail sera nécessaire dans le cadre de cette collaboration pour réaliser des expérimentations réelles.

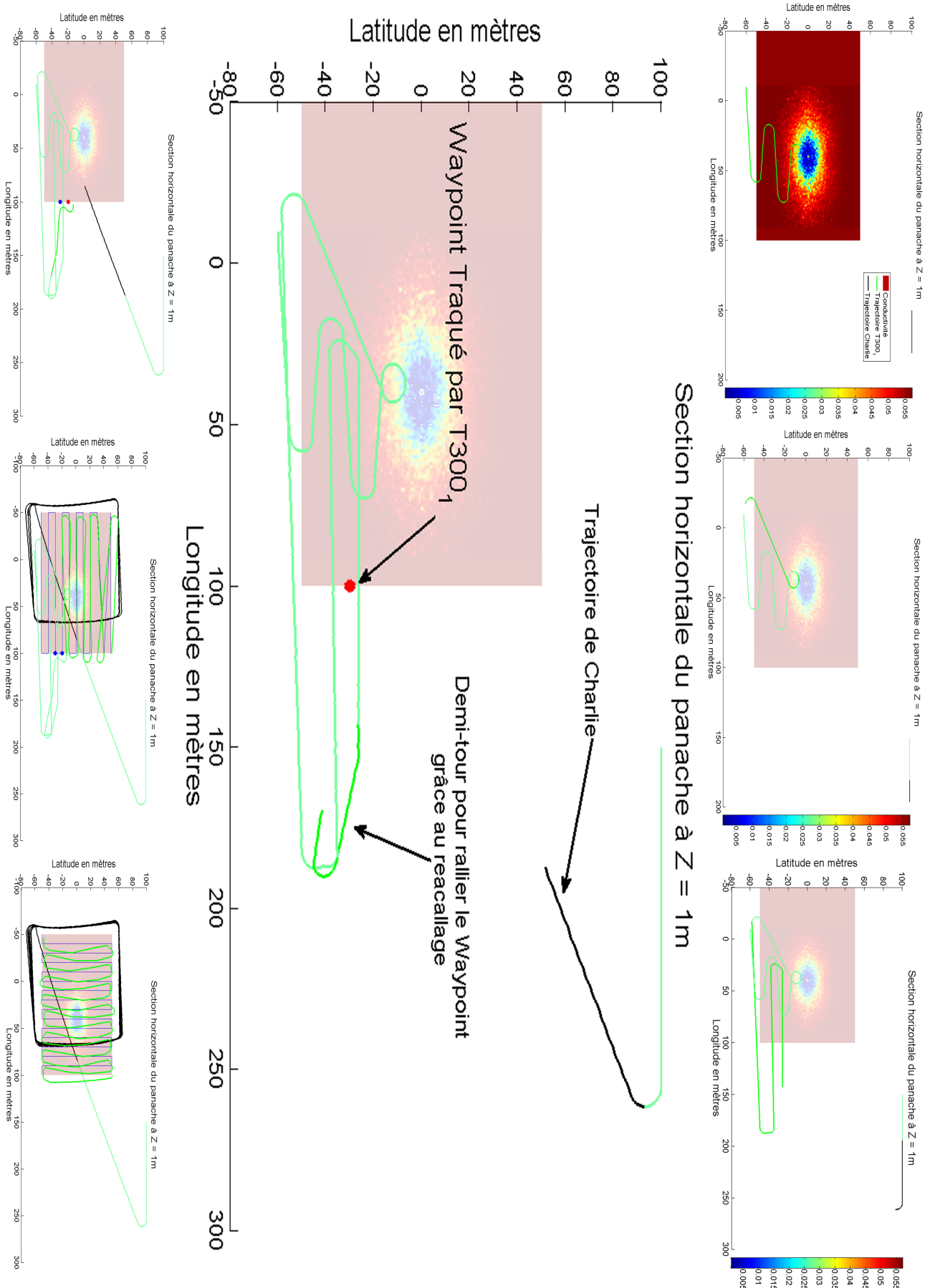


Fig. 8.17 – La mission étape par étape

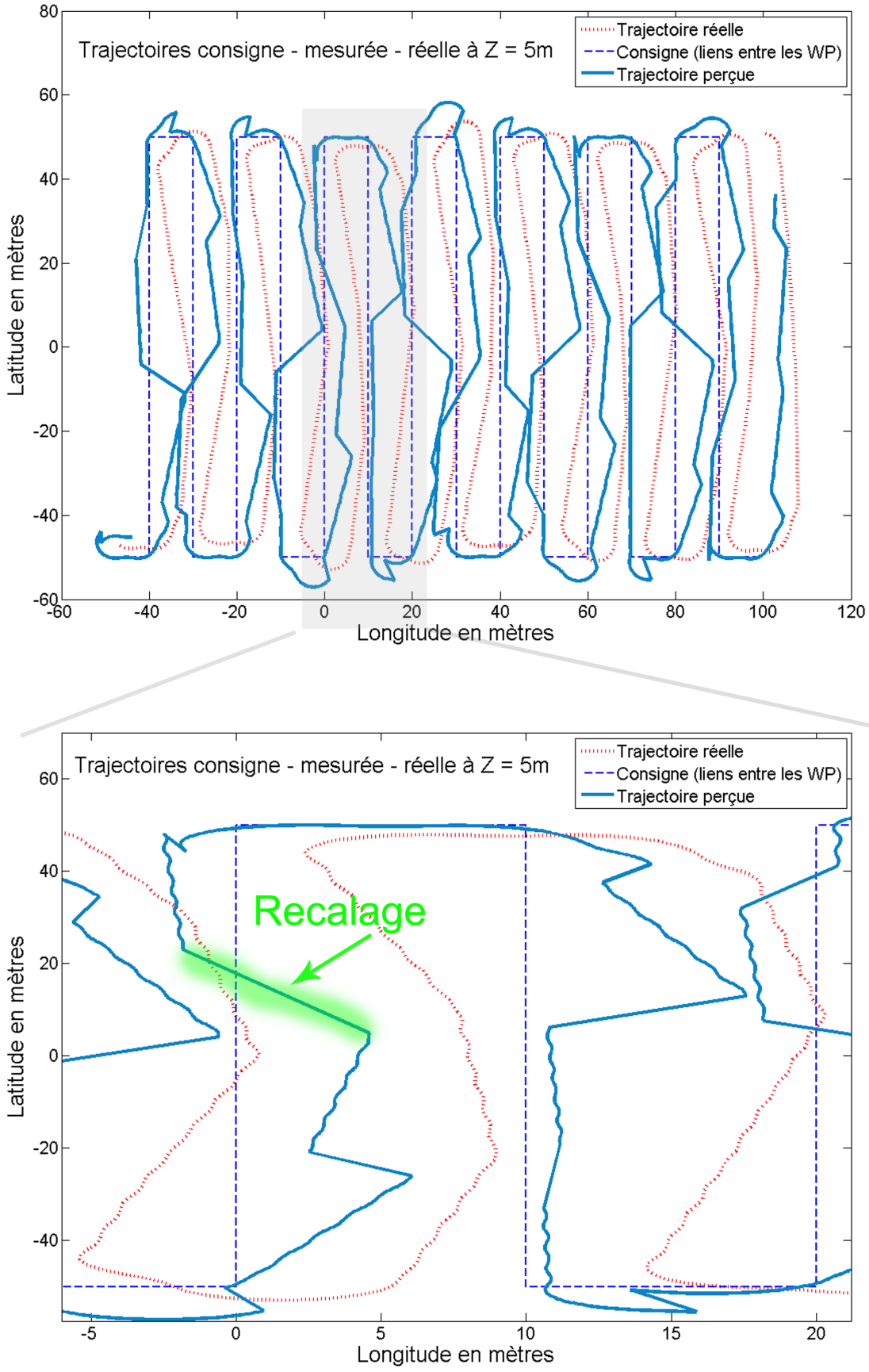


Fig. 8.18 – Recallage du vecteur d'état

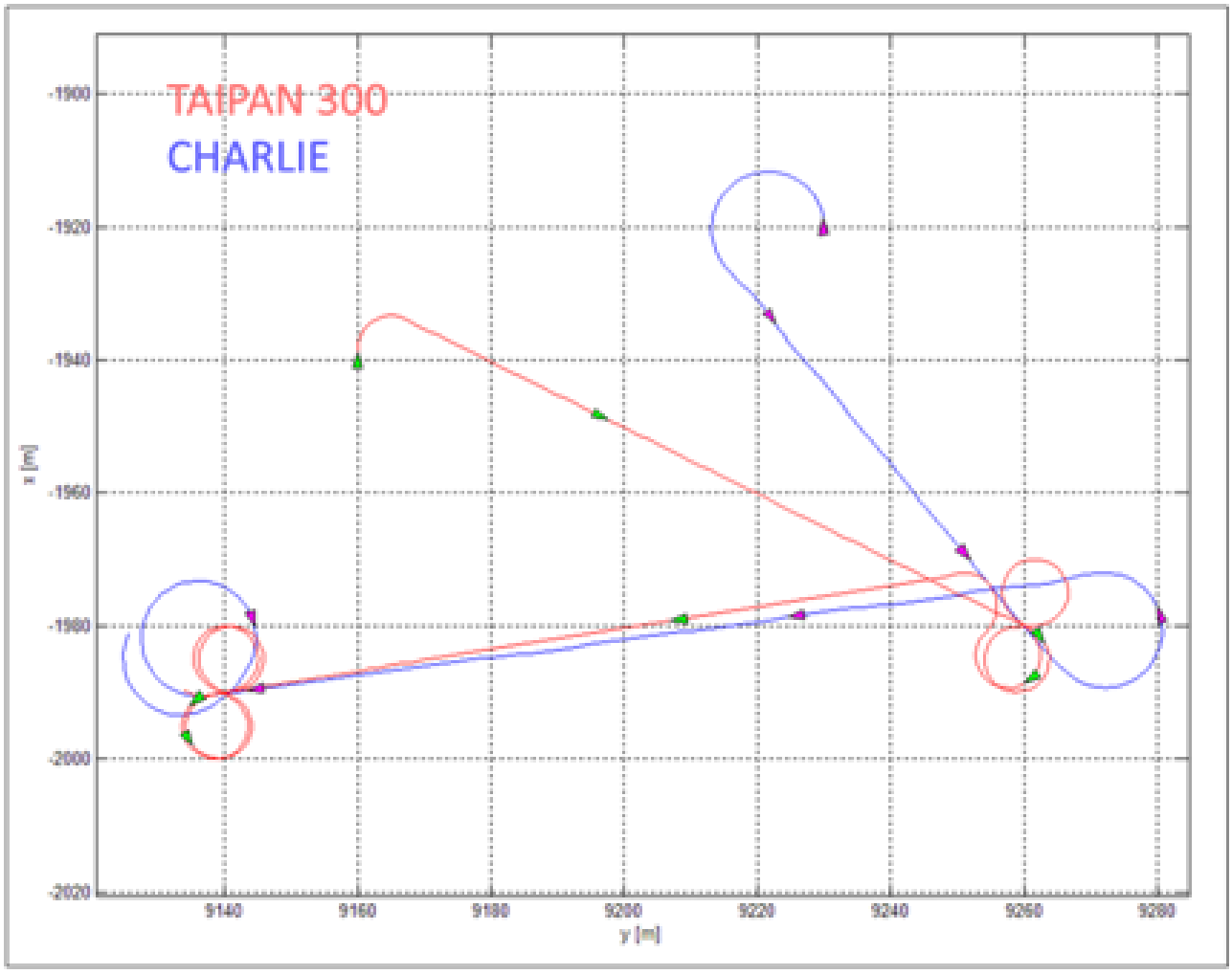


Fig. 8.19 – Trajectoires de Taipan et Charlie, connectés à Thetis.

8.6 Conclusion

Nous avons vu à travers tous ces exemples que le simulateur est maintenant suffisamment mature pour aborder des problématiques variées liées au contexte de la coopération d'engins hétérogènes. Le principal intérêt réside dans le fait de pouvoir à volonté changer les conditions d'expérimentation (bruit capteur, latence de propagation, courants, ...) et stresser les algorithmes pour en connaître les limites de performance.

Outre cet aspect, nous pouvons bien sûr préparer dans le détail une expérimentation réelle. L'implémentation du format netCDF dans notre simulateur permet par exemple d'utiliser des relevés (éventuellement datés, de courants, salinité, température, ...) de précédents essais réels et de vérifier le comportement des engins face à cet environnement. Il est alors possible d'optimiser les algorithmes utilisés (suivi de gradient, suivi de fond..) pour une manipulation en particulier.

Même si nous n'avons pas particulièrement évoqué cet aspect dans ce manuscrit, nous avons fourni un travail conséquent sur l'architecture de contrôle développé au LIRMM afin de la rendre connectable au système de simulation [52]. Ce travail a été l'occasion de développer de nouveaux modules pour gérer la communication, développer de nouveaux algorithmes liés à la coopération de véhicule (algorithmes de backstepping - correction de position passée et courante - pour le recallage avec retard du vecteur d'état des véhicules, détection de gradient de salinité, commande en mode glissant pour l'ASV Charlie ...), et de tester de façon poussée son comportement en l'utilisant pour des missions réelles. Dans ce cadre, le simulateur a pleinement joué son rôle car nous avons pu détecter et traiter un certain nombres de bugs, qui auraient à coup sûr remis en question la réussite des missions.

Il nous est apparu très clair, que l'utilisation d'un tel simulateur se révèle absolument nécessaire pour préparer des missions de coordination même relativement simples. En effet, un long travail a toujours été nécessaire pour présenter les résultats du dernier chapitre de ce manuscrit : la préparation soigneuse de la définition de la mission, les bugs de l'architecture (ou des modules rattachés), la spécification de solutions *ad-hoc* pour une mission particulière et bien sûr le déroulement effectif de la simulation (au sens où prendre en compte tous ces phénomènes induit des comportements très différents de ceux attendus - trajectoires des AUVs en présence de courant, communications non reçues, bruitage des capteurs ...) sont autant d'éléments qui nécessitent une étape de préparation minutieuse. La mission la plus simple aurait nécessité des jours d'expérimentations avec un risque important de perdre ou d'endommager les véhicules.

Conclusions & perspectives

Contexte des travaux

Nous disposons aujourd'hui d'une large gamme de véhicules autonomes permettant de réaliser des tâches complexes dans un milieu potentiellement inaccessible à l'Homme. Le champs d'application de ces robots est potentiellement très vaste et ne cesse de croître. Que ce soit pour l'exploration du milieu, la sécurisation de zones, le relevé des caractéristiques physico-chimiques d'un volume de façon systématique, l'avantage de l'utilisation de robots autonomes est aujourd'hui une certitude.

Ces engins réalisant des missions dans un environnement naturel inconnu et non structuré, sont munis d'une architecture logicielle de contrôle dont le niveau de sophistication est le reflet de la variété des situations dans lesquelles le robot doit pouvoir apporter une réponse appropriée. Il s'agit d'un domaine de recherche actif proposant régulièrement de nouvelles solutions. Le niveau de complexité de ces architectures nécessite une phase de tests et de validation *in situ* avant une mise en œuvre sûre dans un contexte opérationnel. Un simulateur *Hardware-In-Loop* constitue à cet égard une solution intéressante.

Un autre aspect de l'autonomie des véhicules est l'élaboration de lois de commande offrant des garanties de performances. La conception de ces lois se fait généralement en trois phases. Le recours à un modèle mathématique du robot permet de les définir en première approximation. Une seconde phase durant laquelle ces lois sont implémentées dans un simulateur permet de les valider. Pour finir les coefficients de ces lois sont affinés durant des phases d'expérimentations. Dans ce cadre, un simulateur reproduisant avec suffisamment de finesse le comportement dynamique du véhicule constitue une solution avantageuse. La simulation *hybride* permet quant à elle de s'affranchir de la connaissance du modèle du véhicule puisque c'est le véhicule réel qui est utilisé dans un milieu sécurisé et virtuellement personnalisable à volonté. Ce type de simulation correspond à la dernière limite de ce qu'il est possible de faire avant l'expérimentation.

Si nous avons constaté qu'utiliser un seul véhicule est une tâche délicate, le recours à plusieurs engins, potentiellement hétérogènes, présente des difficultés majeures qui font l'objet de nombreux travaux de recherche. Les problématiques liées à la coordination de flottille d'engins sont variées. On pourra citer :

- La conception de loi de commandes permettant aux véhicules de se déplacer sous contraintes (maintenir des liens de communications, garder des configurations spatiales, tenir compte de la cinématique des autres véhicules pour se déplacer, être

capable de prendre une décision locale en présence d'un obstacle garantissant un comportement global cohérent)
- La gestion des communications inter-véhicules (hétérogénéité des moyens de communications, propagation des ondes dans différents media, protocoles de communication multiples, optimisation du routage des données acquises à travers les nœuds du réseau constitué par les vecteurs autonomes)
- La localisation des véhicules (définition d'amers communs, géo-référencement sous-marin, SLAM distribué, localisation par rapport à des modèles de l'environnement préalablement connu, fusion de données en-ligne acquises par les différents capteurs des différents véhicules, relevés multi-échelles)
- Gestion des interférences (interférences des communications acoustiques notablement, ou liées à l'utilisation conjointe de plusieurs capteurs interférents - echo-sondeurs, sonar - sur différents véhicules).

Un simulateur offrant un degré de réalisme suffisant pour prendre en compte ces phénomènes se révèle être un outil indispensable pour concevoir, tester et valider les solutions permettant de traiter de ces problèmes.

Le contexte des travaux présentés ici est donc relatif à plusieurs domaines transversaux complexes. Pour autant les expérimentations de coordination multi-véhicules passeront nécessairement par la fusion de ces travaux appartenant à des domaines de recherche distincts, vraisemblablement au travers d'une plateforme de simulation commune.

Problématiques abordées

La question principale qui a guidé ces travaux a été de se demander quelle était l'étape nécessaire permettant de réaliser des missions de coordination de flottille d'engins hétérogènes. L'objectif de ce travail a été d'apporter une réponse à cette question à travers la proposition d'une architecture de simulateur. Une attention particulière a été portée au cas des véhicules sous-marins qu'il est difficile de superviser durant les expérimentations réelles. Ce problème ne peut être résolu facilement en particulier à cause de la faible bande passante offerte par le milieu aquatique. Les problématiques abordées durant ce travail ont été multiples :

- Comment tester et valider le contrôleur réel d'un robot ? Le problème est de concevoir une solution permettant de tester le logiciel de contrôle du véhicule exécuté par l'ordinateur de bord de l'engin. Dans ce cadre, nous avons donc choisi de développer un simulateur permettant d'effectuer des simulations *online*, *hardware-in-loop* ou *hybrides*. Une des difficultés réside dans le fait de minimiser l'impact de la connexion de ce simulateur sur le comportement du contrôleur afin de garantir la reproductabilité de ce comportement lors de son exploitation réelle [52].
- Comment permettre un travail collaboratif autour du simulateur ? Nous avons largement évoqué au long de ce travail la nécessité de permettre à chacun d'exprimer ces compétences liées à son domaine d'activité. Cet aspect est relié à la notion d'ouverture de la solution proposée et nous avons porté une attention particulière à la génération et publication de la documentation du code et à la généricité de la représentation des participants. Une des difficulté est de faire en sorte que le système de simulation soit clairement divisé en plusieurs entités autonomes regroupant les fonctionnalités par domaine d'activité.

- Quels sont les modèles pertinents à retenir pour réaliser des simulations de ce type ? En effet, les modèles nécessaires pour préparer de façon réaliste une expérimentation sont très nombreux (capteurs, actionneurs, environnements, modèles dynamiques des véhicules, moyens de communication...). Il est nécessaire de déterminer les conséquences que les hypothèses de départ ont sur le domaine de validité. La réponse à ces questions nous permet de définir clairement les conditions d'exploitations dans lesquelles les résultats scientifiques obtenus à partir des simulations sont valides. Une des difficulté est de concevoir des modèles compatibles avec l'aspect temps-réel propre à ce simulateur.
- Quels sont les contraintes liées à la connexion simultanée de plusieurs véhicules au système de simulation ? Préparer des missions de coordination implique de pouvoir être capable de simuler l'évolution de ces robots, gérer les communications entre eux et produire les réponses capteurs dont ils ont besoin pour effectuer la mission. Le problème est de pouvoir garantir que l'ajout de nouveaux robots n'a pas de conséquences sur le comportement temporel de la simulation sans pour autant dégrader la finesse des modèles préalablement déterminée.
- Comment distribuer les calculs sur plusieurs unités de calcul appartenant à une même machine physique et/ou reliées entre elles par un réseau ? En effet, si nous voulons respecter la contrainte d'extensibilité tout en préservant la finesse requise des modèles, il semble inévitable de distribuer les calculs. Une des difficultés est d'échanger les données nécessaires au fonctionnement des processus autonomes sous contraintes de temps et d'intégrité tout en assurant la cohérence temporelle entre les nœuds de la grille de calcul.
- Comment permettre aux véhicules de communiquer entre eux ? Si le but poursuivi est de préparer des missions de coordination de façon réaliste, il n'est pas envisageable de s'affranchir des contraintes liées à la propagation des ondes. Si ce constat semble évident pour les ondes électro-magnétiques, il est d'autant plus vrai pour les ondes acoustiques qui induisent encore davantage de limitations sur les communications et l'occupation du medium de propagation. La difficulté est de définir un modèle compatible avec la notion de temps-réel prenant en compte les effets de l'environnement sur la propagation des ondes, et en tenant comptes des possibles interférences (entre moyens de communications ou avec des capteurs perturbateurs).
- Comment valider temporellement et logiquement le comportement d'une architecture de contrôle dans le cadre de missions multi-véhicules ? La notion de temps-réel est très présente pour répondre à cette question. En effet, il est nécessaire que le système de simulation puisse calculer l'état des entités connectées (environnement, véhicules, ...) et délivrer la bonne réponse, le tout au bon moment. Globalement le problème est donc d'être capable de s'assurer de la capacité du système de simulation à opérer dans des conditions temps-réel à tout instant et pour tout modèle compatible.
- Comment assurer le découplage entre la commande et le système de simulation ? Nous avons largement évoqué cette problématique tout au long de ce manuscrit et clairement indiqué en quoi cet aspect était particulièrement important pour valider les résultats obtenus. Le problème est de trouver le moyen au niveau de l'implémentation de s'assurer de ce point particulier.

Des simulateurs abordant certaines de ces problématiques existent, mais aucun ne permet de résoudre l'ensemble des problèmes évoqués. Ce problème est ce qui a guidé cette thèse : créer et regrouper ces aspects au sein d'une architecture de simulation ; ce regroupement ne va pas de soi, car il s'agit d'un problème fortement contraint. En effet, la difficulté n'est pas de traiter l'une ou l'autre de ces problématiques, mais réside dans le fait d'être en mesure d'aborder l'ensembles de ces points identifiés comme critiques pour le contexte qui est le notre.

Résumé des propositions

Notre travail traite de l'architecture d'un système de simulation dédié à la préparation d'expérimentations de flottille de véhicules autonomes hétérogènes [55]. La proposition que nous avons formulée est une architecture logicielle structurée sur une entité de base appelée module. Ce module est à la base de la construction de notre simulateur et nous assure certaines propriétés fondamentales comme le découplage temporel entre les différents blocs composant le simulateur et les véhicules. La construction du système proposé répond entre autres, aux critères de découplage temporel, d'extensibilité et de modularité que nous nous sommes fixés au départ. Il est en outre basé sur quatre simulateurs réalisant des tâches différentes : ainsi un simulateur de la dynamique des véhicules, un simulateur de capteurs, un simulateur de moyens de communication et un simulateur d'environnement sont exécutés sur des unités différentes pouvant être reliées entre elles sur un réseau dédié (dédié, afin de s'assurer de l'aspect temps-réel des échanges inter-processus). Ce simulateur a été créé en respectant un certain nombre de contraintes que nous avions identifiées comme étant critiques au départ. Il s'agit principalement de la possibilité de pouvoir distribuer notre application afin d'être en mesure de faire de la simulation temps-réel multi-véhicules sans avoir à dégrader les modèles utilisés. Un superviseur a été développé pour contrôler le déroulement de la séquence de lancement d'une simulation et pour surveiller le déroulement de son exécution. L'approche proposée se distingue des simulateurs existants sur différents points :

- Thetis est le seul simulateur à prendre en compte tous les aspects critiques liés au contexte de la préparation d'expérimentations multi-véhicules.
- Nous avons utilisé une approche fonctionnelle, là où les autres ont privilégié une approche par composant. Ce choix nous permet de scinder le simulateur par spécialité et offre un avantage certain eut égard à la pluridisciplinarité des intervenants potentiels
- Ce simulateur offre la possibilité de tester l'ensemble des aspects de la préparation d'une mission multi-véhicules et ne se focalise pas sur un point particulier (commande ou communication ou environnement...)
- L'architecture proposée évite d'avoir à dupliquer le code de l'application sur tous les nœuds du simulateur. La maintenance s'en trouve donc facilitée.

Une seconde proposition porte sur la définition et le développement de modèles permettant la communication entre les véhicules, au sens large du terme. En effet, par communication, nous entendons ici la propagation des ondes dans le milieu [71] mais également le comportement logico-temporel des moyens de communications. La contrainte forte qui a dirigé nos choix, a été de faire en sorte que les modèles et mécanismes développés pour l'occasion restent compatibles temps-réel. Cela excluait donc (au moins dans un premier

temps) les algorithmes de "jeté de rayons" gourmands en temps de calcul. Nous avons donc mis au point un "moteur de rendu des messages" permettant de tenir compte des temps de latence, du bruit, de la bande passante, des interférences (définissables par l'utilisateur) entre les moyens de communications ou avec des capteurs interférents et enfin des multi-émissions d'un moyen de communication (émission d'une onde alors que les n précédentes sont en cours de propagation dans l'environnement). Ce "moteur de rendu des messages" apporte un réalisme indéniable à la problématique de la coordination de véhicules, et au delà à la mise au point de protocoles de communication permettant de relier les différentes entités (bouées, véhicules, humains, ...) au travers des différents milieux (air, eau). Thetis est à l'heure actuelle le seul simulateur disposant de cette fonctionnalité en temps réel [55].

Une troisième proposition concerne une façon générique de décrire les véhicules mobiles. En effet, un robot mobile peut toujours être considéré comme un ensemble d'actionneurs, de moyens de communication et de capteurs. Par ailleurs, placé dans un contexte de simulation, nous avons ajouté à cette "vue" différentes propriétés permettant de pleinement qualifier un véhicule donné. Cette approche nous permet de décrire un engin par ajout de "liens" vers des fichiers de description réutilisables par tous. Un fichier de configuration d'un robot est donc un annuaire de liens vers des fichiers décrivant les composants de ce robot.

Enfin, nous avons proposé une méthode permettant de classifier un simulateur donné. En effet, nous avons constaté que la variété des simulateurs était pour le moins étendue et qu'il était relativement difficile de se faire une idée précise des contextes dans lesquels chacun d'eux peut-être utilisé. Nous avons donc proposé une grille d'évaluation qui permet d'observer tout nouveau simulateur par rapport à un référentiel variable fixable par une communauté en particulier.

Perspectives

Les perspectives à ce travail sont nombreuses et peuvent être regroupées en quatre volets :

- Les améliorations de l'architecture de simulation :
  - Un premier point concerne la migration du système proposé vers le temps-réel. En effet, nous avons constaté que même si le comportement du simulateur était bien celui d'un système temps-réel, nous ne pouvions en aucun cas en apporter la garantie. Même si ce système n'est pas critique au sens où une défaillance ne remet pas en cause l'intégrité du matériel, il est toujours pénible de devoir annuler une simulation de plusieurs heures parce que nous avons détecté une latence pendant un instant. Le choix que nous avons fait de développer le simulateur en n'utilisant pas les primitives temps-réel offertes par le système hôte, ne se justifie plus dans la mesure où le projet est arrivé à un stade suffisamment mature pour intégrer ces contraintes.
  - Une seconde amélioration porte sur la modularité et donc par extension sur la distributivité du système. En effet, dans l'état actuel des choses, le simulateur est constitué de 16 processus, mais il semble intéressant de pouvoir en augmenter le nombre afin de distribuer davantage l'application. Maintenant que les fondations de l'architecture sont posées, nous envisageons de séparer les modèles des proces-

sus de simulation actuels. Ainsi chaque modèle serait un processus à part entière exploitant le module de base développé et amélioré (approche davantage orientée composant) pour l'occasion.
  - Enfin l'interopérabilité de ce type de système est un élément clé. En effet, des normes dédiées à la communication entre simulateurs existent (HLA, et DIST notamment), et il semble opportun de s'y intéresser afin d'améliorer l'interopérabilité du simulateur proposé.
- L'amélioration des modèles au sein de l'architecture de simulation :
  - La problématique de la vision en robotique constitue un domaine de recherche très actif. Dès lors, il nous semble important de finir de développer l'afficheur 3D pour incorporer des modèles d'appareil de prise de vue et ainsi étendre le domaine d'utilisation de ce simulateur. Il sera alors possible de valider des algorithmes développés au sein de notre équipe (estimation du vecteur vitesse d'un véhicule sous-marin à partir de séquences vidéo notamment) de recherche.
  - L'implémentation de capteurs de type sonar est également une voie présentant un intérêt certain. En effet, nombre d'algorithmes utilisent ce type de technologie dans des domaines très variés (SLAM, cartographie de champs de mine, relevés bathymétriques ...) et couplés à tous les phénomènes pris en compte dans le simulateur, ouvre la voie à de nouvelles recherches. Pour se faire, nous envisageons d'avoir recours aux nouvelles technologies : en effet, les algorithmes permettant la simulation de ce type de capteur sont très consommateurs de temps de calcul et cela rend inopérant dans la pratique la simulation temps-réel de ces capteurs. Mais de nouvelles technologies sont disponibles depuis quelques mois (le langage CUDA (Compute Unified Device Architecture) de Nvidia qui permet de lancer des opérations en parallèles sur 292 processeurs de flux notamment) et commencent à susciter l'intérêt des chercheurs ayant besoin de moyens de calcul important.
  - La détermination des paramètres hydrodynamiques de nos nouveaux véhicules constitue également un aspect essentiel des travaux à effectuer. Pour cela nous envisageons d'embarquer le simulateur à bord des véhicules sur un ordinateur de bord dédié. Cette approche sera facilitée par l'arrivée des nouveaux processeur Intel multi-coeurs (Larabee). Nous pourrions ainsi envisager une estimation en ligne des paramètres des modèles que nous utilisons afin d'en augmenter la précision.
  - La plupart des robots communicant par ondes électro-magnétiques, l'évolution du "moteur de rendu des messages" actuellement en service est une étape qui nous semble importante. Si cela n'a pas encore été fait, il s'agit principalement d'une question de temps et d'intérêt dans l'immédiat. Cela nous permettra d'aborder le contexte de la coordination de vecteurs hétérogènes reliés par un réseau mixte (aquatique/aérien) et utilisant des moyens de communications hétérogènes (WIFI, radio, modem acoustique, bluetooth...). Peu de travaux existent dans le domaine et il s'agit là d'une voie de recherche riche en nouveautés.
  - Les modèles des actionneurs animant les effecteurs ne sont pas pris en compte pour l'instant. Pourtant il s'agit là d'un aspect important influant largement sur le comportement du véhicule : une commande sliding-mode sur un actionneur possédant une dynamique faible impliquera une comportement du véhicule actionné très différent d'un qui utiliserait des actionneurs à dynamique élevée. Un travail a été réalisé dans ce sens et sera prochainement implémenté.

- L'amélioration de ce qui gravite autour du simulateur :
  - Nous avons largement évoqué la nécessaire implication d'acteurs ayant des compétences différentes autour de ce projet. Afin de faciliter cette coopération, nous envisageons de développer un site internet regroupant la documentation et les ressources disponibles (modèles, fichiers de description ...). Cela permettra de thésauriser le travail de chacun et de le rendre disponible pour tous.
  - Dans ce contexte, une recherche active de coopérations entre des laboratoires différents spécialisés dans des domaines qui leur sont propres doit être envisagée.
  - Améliorer les outils existants constitue également un objectif à court terme. En effet, nous avons commencé à développer des interfaces de configuration de simulation mais l'ergonomie et les fonctionnalités peuvent encore être largement améliorées. Par ailleurs, nous envisageons de créer ces interfaces sous forme de page WEB afin de rendre accessible à tous les outils de simulations mais aussi de s'affranchir du problème de plateforme (Win32, Mac, Linux) pour ces interfaces de configuration.
- Le dernier volet concerne naturellement l'utilisation du simulateur dans le cadre de la coordination de flottille
  - Continuer à travailler de concert avec l'équipe de M. Caccia afin de tester, préparer et valider les lois de commande développés conjointement par les deux équipes (path following notamment).
  - En 2009, il est prévu de connecter les UGV développés au LIRMM par R. Zapata au simulateur afin de réaliser des simulations hybrides. En effet, ces engins terrestres sont plus faciles à utiliser que des véhicules sous-marins et constituent par ailleurs une plate-forme différente de celles mises en œuvre jusqu'à présent. Il est dès lors intéressant de montrer que Thetis est un simulateur pouvant s'adapter facilement aux solutions retenues par chacun.
  - Nous souhaitons concevoir et valider effectivement de nouvelles lois de commande liées à la coordination de flottille de véhicules. En effet, les problématiques soulevées sont vastes et cet outil nous permettra à coup sûr d'évaluer les solutions théoriques proposées mais pas nécessairement testées, dans des conditions opérationnelles proches de la réalité. Nous serons alors en mesure de formuler de nouvelles propositions aussi bien en terme de lois de commande, de stratégie de coordination, de protocoles et routage des communications inter-véhicules que d'algorithmes de traitement des données en ligne innovants.

Bibliographie

[1] PAP : Poisson Auto-propulsé. http ://sauvmer.free.fr/marine/pap.html , accédé le 26/10/08.

[2] SPIV : Sonar de l'avant Propulsé à Immersion Variable. http ://www.netmarine.net/bat/cm/mission-cmt.htm , accédé le 26/10/08.

[3] R.P. Stokey, A. Roup, C. von Alt, B. Allen, N. Forrester, T. Austin, R. Goldsborough, M. Purcell, F. Jaffre, G. Packard, and A. Kukulya. Development of the remus 600 autonomous underwater vehicle. *OCEANS, 2005. Proceedings of MTS/IEEE*, pages 1301–1304 Vol. 2, 2005.

[4] M. Sibenac, W.J. Kirkwood, R. McEwen, F. Shane, R. Henthorn, D. Gashler, and H. Thomas. Modular auv for routine deep water science operations. *Oceans '02 MTS/IEEE*, pages 167–172 vol.1, Oct. 2002.

[5] Spiewak, J., Jouvencel, B., Fraisse, P. New design of auv for shallow water applications : H160. *Proceedings of the International Offshore and Polar Engineering Conference*, 2006.

[6] Spiewak J-M. *Contribution à la Coordination de Flottille de Véhicules Sous-marins Autonomes*. PhD thesis, Université de Montpellier II, 2007.

[7] O. Parodi, L. Lapierre, B. Jouvencel. Optimised gait for anguilliform system. *Caractérisation du Milieu Marin, Brest CMM 06*, October 2006.

[8] O. Parodi, L. Lapierre, and B. Jouvencel. Optimized gait generation for anguilliform motion. *OCEANS 2006 - Asia Pacific*, pages 1–8, May 2006.

[9] S. Lacroix and G. Le Besnerais. Issues in cooperative air/ground robotic systems. In *13th International Symposium on Robotics Research, Hiroshima (Japan)*, 2007.

[10] S. Lacroix, S. Joyeux, T. Lemaire, S. Bosch, P. Fabiani, C. Tessier, O. Bonnet, D. Dufourd, and E. Moline. Projet acrobate : Algorithmes pour la coopération entre robots terrestres et aériens. In *Quatrièmes journées du programme Robea, Paris (France)*, March 2006.

[11] Maczka, D. K., Stilwell, D. J. Experiments in distributed navigation for auv platoons. *IEEE/MTS OCEANS Alberta BC, Canada*, 2007.

[12] E. Fiorelli, N.E. Leonard, P. Bhatta, D. Paley, R. Bachmayer, and D.M. Fratantoni. Multi-auv control and adaptive sampling in monterey bay. *Autonomous Underwater Vehicles, 2004 IEEE/OES*, pages 134–147, June 2004.

[13] P. Ridao, E. Batlle, D. Ribas, and M. Carreras. Neptune : A hil simulator for multiple uuvs. *OCEANS '04. MTTS/IEEE TECHNO-OCEAN '04*, 1 :524–531, Nov. 2004.

[14] Prasanna Sridhar and Mo Jamshidi. Distributed architecture for modeling and simulation of autonomous multi-agent multi-physics systems. *Autonomous Control Engineering (ACE).*

[15] T. Perez, ØN. Smogeli, T.I. Fossen, A.J. Sørensen. An overview of the marine systems simulator (mss) : A simulink toolbox for marine control systems. *Modeling, Identification and Control*, 27, 4 :259–275(17), October 2006.

[16] M. Carreras, J. Batlle, P. Ridao, G.N. Roberts. An overview on behaviour-based methods for auv control. *IFAC Conference on Manoeuvring and Control of Marine Crafts*, 2000.

[17] Javier Antich, Alberto Ortiz. Experimental evaluation of the control architecture for an underwater cable. *IFAC Conference on Manoeuvring and Control of Marine Crafts*, 2003.

[18] C.J. Cannell, D.J. Stilwell, and J.A. Austin. A simulation tool to support the development of adaptive sampling algorithms for multiple autonomous underwater vehicles. *Autonomous Underwater Vehicles, 2004 IEEE/OES*, pages 127–133, June 2004.

[19] ao Borges de Sousa Jo and Aleks Göllü. A simulation environment for the coordinated operation of multiple autonomous underwater vehicles. In *WSC '97 : Proceedings of the 29th conference on Winter simulation*, pages 1169–1175. IEEE Computer Society, 1997.

[20] G. Conte and A. Serrani. Modelling and simulation of underwater vehicles. *Computer-Aided Control System Design, 1996., Proceedings of the 1996 IEEE International Symposium on*, pages 62–67, Sep 1996.

[21] R. Bono, M. Caccia, and G. Veruggio. Simulation and control of an unmanned underwater vehicle. *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*, pages 1573–1578 vol.2, May 1995.

[22] Willi Hornfeld. Deepc, the german auv development project.

[23] E. Kobayashi, T. Aoki, T. Maeda, K. Hirokawa, T. Ichikawa, T. Saitou, S. Miyamoto, S. Iwasaki, and H. Kobayashi. Development of an autonomous underwater vehicle maneuvering simulator. *OCEANS, 2001. MTS/IEEE Conference and Exhibition*, 1 :361–368 vol.1, 2001.

[24] H. Robinson and A. Keary. Remote control of unmanned undersea vehicles. *International UUV Symposium*, April 2000.

[25] Tobias Bielohlawek. Subsim - an autonomous underwater vehicle simulation system. Technical report, Arbeitsgruppe Robotersysteme Fachbereich Informatik Universitat Kaiserslautern, April 2006.

[26] Gracanin D., Matijasevic M., Valavanis K.P. Virtual environment testbed for underwater robotics applications. *ICAR'97, 8th International Conference on Robotics and Automation*, pages 793–797, Juillet 1997.

[27] D. Gracanin, M. Matijasevic, N. Tsourveloudis, and K.P. Valavanis. Virtual reality simulation testbed for underwater environments. *OCEANS '98 Conference Proceedings*, pages 1392–1396 vol.3, Sep-1 Oct 1998.

[28] S. Phoha, E.M. Peluso, and R.L. Culver. A high-fidelity ocean sampling mobile network (samon) simulator testbed for evaluating intelligent control of unmanned underwater vehicles. *Oceanic Engineering, IEEE Journal of*, 26(4) :646–653, Oct 2001.

[29] J. S. Albus. System description ans design architecture for multiple autonomous underwater vehicles. Technical report, National institute of Standards and Technology, Gaithersburg, MD, 1988. 1251.

[30] M.J. Zyda, R.B. McGhee, S. Kwak, D.B. Nordman, R.C. Rogers, and D. Marco. Three-dimensional visualization of mission planning and control for the nps autonomous underwater vehicle. *Oceanic Engineering, IEEE Journal of*, 15(3) :217–221, Jul 1990.

[31] R. Hillson and C. Jones. Evolution of an auv mission simulation testbed. *Unmanned Untethered Submersible Technology, 1989. Proceedings of the 6th International Symposium on*, pages 525–535, Jun 1989.

[32] Donald P. Brutzman. *A Virtual World for an Autonomous Underwater Vehicle*. PhD thesis, Naval Postgraduate school, Monterey, California, 1998.

[33] P. Ridao, J. Battle, J. Amat, and M. Carreras. A distributed environment for virtual and/or real experiments for underwater robots. *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, 4 :3250–3255, 2001.

[34] Pere Ridao, Joan Batlle, and Marc Carreras. An underwater autonomous agent. from simulation to experimentation.

[35] Chappell S.G., Komerska R.J. An environment for high-level multiple auv simulation and communication. *Proceeding of the Collaborating and Leveraging Outer Space and Undersea Technologies (CLOUT) NOAA/NASA Workshop*, August 2000.

[36] R.J. Komerska and S.G. Chappell. A simulation environment for testing and evaluating multiple cooperating solar-powered auvs. *OCEANS 2006*, pages 1–6, Sept. 2006.

[37] Feijun Song, P.E. An, and A. Folleco. Modeling and simulation of autonomous underwater vehicles : Design and implementation. *Oceanic Engineering, IEEE Journal of*, 28(2) :283–296, April 2003.

[38] Feijun Song, A. Folleco, and E. An. High fidelity hardware-in-the-loop simulation development for an autonomous underwater vehicle. *OCEANS, 2001. MTS/IEEE Conference and Exhibition*, 1 :444–449 vol.1, 2001.

[39] D. Suriano and C. Moriconi. A distributed simulator for the development of the unmanned underwater vehicles control software. *13th IASTED International Conference, Robotics and Applications*, August 2007.

[40] S.K. Choi, S.A. Menor, and J. Yuh. Distributed virtual environment collaborative simulator for underwater robots. *Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on*, 2 :861–866 vol.2, 2000.

[41] S.K. Choi and J. Yuh. A virtual collaborative world simulator for underwater robots using multidimensional synthetic environment. *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, 1 :926–931 vol.1, 2001.

[42] F. Devie and J. Lemaire. A flexible hardware in the loop simulator for a long range autonomous underwater vehicle. *OCEANS '98 Conference Proceedings*, pages 1359–1363 vol.3, Sep-1 Oct 1998.

[43] D.M. Lane, G.J. Falconer, G.W. Randall, N.D. Duffy, J.T. Herd, P. Chernett, J. Hunter, M. Colley, J. Standeven, V. Callaghan, J. Smith, J. Evans, A. Woods, J. Penrose, G.A. Whittaker, D. Smith, and I. Edwards. Mixing simulations and real subsystems for subsea robot development. specification and development of the core simulation engine. *OCEANS '98 Conference Proceedings*, pages 1382–1386 vol.3, Sep-1 Oct 1998.

[44] Y. Kuroda, K. Aramaki, and T. Ura. Auv test using real/virtual synthetic world. *Autonomous Underwater Vehicle Technology, 1996. AUV '96., Proceedings of the 1996 Symposium on*, pages 365–372, Jun 1996.

[45] S. Lacroix, T. Lemaire, and C. Berger. More vision for SLAM. In *IEEE International Conference on Robotics and Automation, Roma (Italy), workshop on Unifying Perspectives in Computational and Robot Vision*, April 2007.

[46] S. Lacroix, M. Devy, J. Sola, and T. Lemaire. Modélisation 3d par vision pour la robotique mobile : Approches de cartographie et localisation simultanées. *Revue de la société Française de Photogrammétrie et de Télédétection*, (180) :26–40, 2006.

[47] Y. Kuroda, T. Ura, and K. Aramaki. Vehicle control architecture for operating multiple vehicles. *Autonomous Underwater Vehicle Technology, 1994. AUV '94., Proceedings of the 1994 Symposium on*, pages 323–329, Jul 1994.

[48] Chappell, Steve, Rick J. Komerska, D. Richard Blidberg, Christiane N. Duarte, Gerald R. Martel, Denise M. Crimmins, Michel A. Beliard, Robert Nitzel, James C. Jalbert, and Radim Bartos. Recent field experience with multiple cooperating solar powered auvs. *the Fifteenth International Symposium on Unmanned Untethered Submersible Technology*, August 2007.

[49] G. Bruzzone, R. Bono, M. Caccia, and G. Veruggio. A simulation environment for unmanned underwater vehicles development. *OCEANS, 2001. MTS/IEEE Conference and Exhibition*, 2 :1066–1072, 2001.

[50] E.A. Carlson, P.-P. Beaujean, and E. An. Simulating communication during multiple auv operations. *Autonomous Underwater Vehicles, 2004 IEEE/OES*, pages 76–82, June 2004.

[51] CONNECT : CONtrol of NEtworked Cooperative sysTems. http ://www.lag.ensieg.inpg.fr/connect/index.php , accédé le 09/11/08.

[52] O. Parodi, A. El Jalaoui, and D. Andreu. Connectivity of thetis, a distributed hybrid simulator, with a mixed control architecture. *Autonomic and Autonomous Systems, 2008. ICAS 2008. Fourth International Conference on*, pages 130–135, March 2008.

[53] Abdellah El Jalaoui, David Andreu, and Bruno Jouvencel. Contextual management of tasks and instrumentation within an auv control software architecture. *Intelligent*

Robots and Systems, 2006 IEEE/RSJ International Conference on, pages 3761–3766, Oct. 2006.

[54] El Jalaoui, A. and Andreu, D., Jouvencel, B. A control architecture for contextual tasks management : Application to the auv taipan. *Oceans 2005 - Europe*, 2 :752–757 Vol. 2, June 2005.

[55] O. Parodi, L. Lapierre, B. Jouvencel. Thetis : A real-time multi-vehicles hybrid simulator for heterogeneous vehicles. *IROS08 : International Conference on Intelligent RObots and Systems, Workshop on robot simulators : Available Software, Scientific Applications and Future Trends*, September 2008.

[56] Doxygen : A Source Code Documentation Generator Tool. http ://www.stack.nl/ dimitri/doxygen/ , accédé le 04/10/08.

[57] Abdellah El Jalaoui. *Gestion Contextuelle de Tâches pour le Contrôle d'un Véhicule Sous-marin Autonome*. PhD thesis, Université de Montpellier II, 2007.

[58] RTnet : Hard Real-Time Networking for Real-Time Linux. http ://www.rts.uni-hannover.de/rtnet/index.html , accédé le 05/10/08.

[59] Putty : Telnet and SSH client. http ://www.chiark.greenend.org.uk/ sgtatham/putty/download.html , accédé le 05/10/08.

[60] UML : L'UML en français. http ://uml.free.fr/ , accédé le 07/10/08.

[61] TinyXML : Documentation. http ://www.grinninglizard.com/tinyxmldocs/index.html , accédé le 07/10/08.

[62] Geodise : Gris Enabled Optimisation and Dessign Search for Engineering. http ://www.geodise.org/toolboxes/generic/xml_toolbox.htm , accédé le 07/10/08.

[63] Thor I. Fossen. *Guidance and Control of Ocean Vehicles*. John Wiley & Sons Ltd, 1994.

[64] Michel Aucher. Dynamique des sous-marins. Technical Report 55, Sciences et Techniques de l'Armement, 1981.

[65] Timothy Prestero. Verification of a six degree of freedom simulation model for the remus autonomous underwater vehicle, 2001.

[66] Aristide Simon Santos. *Contribution à la Conception des Sous-marins Autonomes : Architecture des Actionneurs, Architecture des Capteurs, et Commandes Référencées Capteurs*. PhD thesis, Ecole Nationale Supérieure des Mines de Paris, 1995.

[67] Caccia M., Bibuli M., Bono R., Bruzzone G. Basic navigation, guidance and control of an unmanned surface vehicle. *Springer*, Jul 2008.

[68] M. Caccia, G. Bruzzone, and R. Bono. Modelling and identification of the charlie2005 asc. *Mediterranean Conference on Control and Automation*, pages 1–6, 2006.

[69] M. Perrier, L. Brignone, M. Drogou. Communication constraints and requirements for operating multiple unmanned marine vehicles (mumvs). *Proceedings of the Sixteenth (2007) International Offshore and Polar Engineering Conference*, page 1053, 2007.

[70] Olivier Le Calvé. Le son dans la mer.

[71] O. Parodi, V. Creuze, B. Jouvencel. Communications with thetis, a real time multi-vehicles hybrid simulator. *ISOPE'08 : International Society of Offshore and Polar Engineers*, July 2008.

[72] Perrine Fleury. *Sources sous-marines et Aquifères Karistiques Côtiers Méditerranéennes. Fonctionnement et Caractérisation.* Université Paris VI, 2005.

[73] Open Dynamics Engine. http ://www.ode.org/ , accédé le 30/09/08.

[74] Ahcene Bouzoualegh. *Etude et Proposition d'un Réseau Local Acoustique Aquatique.* PhD thesis, Mémoire de Thèse soutenue à l'Université de Toulouse II, 2006.

[75] Ethem Sozer and Milica Stojanovic. Simulation and rapid prototyping environment for auv networks.

[76] K.L. Hall. What is hardware-in-the-loop simulation ? *The Society for Computer Simulation Conferece*, pages 62–65, 1987.

[77] P. Ridao, D. Ribas, E. Batlle, and E. Hernandez. Simulation of physical agents. an application to underwater robots. In *V Workshop on Physical Agents*, Girona, SP, march 2004.

[78] L. Lapierre, V. Creuze, B. Jouvencel. Robust diving control of an auv. *Manoeuver and Control of Marine Craft (MCMC'06)*, 2006.

[79] M. Caccia, R. Bono, G. Bruzzone, E. Spirandelli, G. Veruggio, A.M. Stortini, and G. Capodaglio. Sampling sea surfaces with sesamo : an autonomous craft for the study of sea-air interactions. *Robotics & Automation Magazine, IEEE*, 12(3) :95–105, Sept. 2005.

[80] P.K. Paim, B. Jouvencel, and L. Lapierre. A reactive control approach for pipeline inspection with an auv. *OCEANS, 2005. Proceedings of MTS/IEEE*, pages 201–206 Vol. 1, 2005.

[81] M. Caccia, G. Indiveri, and G. Veruggio. Modeling and identification of open-frame variable configuration unmanned underwater vehicles. *Oceanic Engineering, IEEE Journal of*, 25(2) :227–240, Apr 2000.

[82] SNAME. Nomenclature for treating the motion of a submerged body through a fluid. Technical and research bulletin no. 1-5, The Society of Naval Architects and Marine Engineers, 1950.

[83] W. Khalil and E. Dombre. *Modélisation, Identification et Commande des Robots.* Hermés Science, 1999.

[84] A. Quazi and W. Konrad. Underwater acoustic communications. *Communications Magazine, IEEE*, 20(2) :24–30, Mar 1982.

[85] M. Stojanovic, J. Catipovic, and J. G. Proakis. Adaptive multichannel combining and equalization for underwater acoustic communications. *Acoustical Society of America Journal*, 94 :1621–1631, sep 1993.

[86] A. Kaya and S. Yauchi. An acoustic communication system for subsea robot. *OCEANS '89. Proceedings*, 3 :765–770, Sep 1989.

[87] M. Stojanovic. Recent advances in high-speed underwater acoustic communications. *IEEE Journal of Oceanic Engineering*, 21(2) :125–136, Apr 1996.

[88] M. Stojanovic. Underwater acoustic communication. *Entry in Encyclopedia of Electrical and Electronics Engineering, John G. Webster, Ed., John Wiley and Sons*, 22 :688–698, Apr 1999.

[89] L. Berkhovskikh, Y. Laysanov. *Fundamentals of Ocean Acoustics.* N.Y : Springer, 1982.

[90] V.V. Varadan, V.K. V aradan. Scattering matrix for elastic waves. iii. application to spheroids. *Journal of Acoustical Society of America*, 65 :896–905, 1979.

[91] P.C. Waterman. Matrix ttheory of elastic wave scattering. *Journal of the Acoustical Society of America*, 60(3) :567–580, 1976.

[92] Hamson R.M. The modelling of ambient noise due to shipping and wind sources in complex environments. *Applied Acoustics*, 51 :251–287(37), July 1997.

[93] D. Ross. *Mechanics of Underwater Noise*. Pergamon press, 1976.

[94] D. Gaucher. *Etude des Potentialités de la Tomographie Acoustique Océanique Passive*. PhD thesis, Mémoire de Thèse soutenue à l'Université de Bretagne Occidentale, 2005.

[95] Richardson. *Marine Mammals and Noise*. Academic press, 1995.

[96] G.M. Wenz. Acoustic ambiant noise in the ocean : Spectra and source. *Journal of American Acoustic Society*, 34 :1936–1956, 1962.

[97] H. Medwin. Speed of sound in water for realistic parameters. *Journal of American Acoustic Society*, 58 :1318, 1975.

[98] C.C. Leroy. Development of simple equations for accurate and more realisitic calculation of the speed of sound in sea water. *Journal of American Acoustic Society*, 42 :216, 1969.

[99] K.V. Mackenzie. Nine-term equaion for sound speed in the oceans. *Journal of American Acoustic Society*, 70 :807, 1981.

[100] VMware Server : technologie de virtualisation. http ://www.vmware.com/fr/products/server/ , accédé le 06/10/08.

[101] Thetis : Developement website. http ://www.lirmm.fr/taipan/thetis , accédé le 07/10/08.

[102] R. Zapata and P. Lepinay. Collision avoidance and bottom following of a torpedo-like auv. *OCEANS '96. MTS/IEEE. 'Prospects for the 21st Century'. Conference Proceedings*, 2 :571–575 vol.2, Sep 1996.

[103] L. Lapierre, D. Soetanto, and A. Pascoal. Nonlinear path following with applications to the control of autonomous underwater vehicles. *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, 2 :1256–1261 Vol.2, Dec. 2003.

[104] V. Creuze, O. Parodi. Utilisation de la diffraction acoustique pour la détection de fond. *Revue Traitement du Signal*, 25 :3–12, 2008.

Annexe A

Présentation des véhicules

A.1 Taipan 2 ou H160

Aussi appelé *H160* sous sa dénomination commerciale, ce véhicule a été développé conjointement par la société Eca-Hytec et par le LIRMM (figure A.1). Il s'agit d'un véhicule de type torpille de petite taille et de faible coût dédié aux applications en eaux peu profondes (max 160m)[5]. Ce véhicule de 1,8m et de 20cm de diamètre pèse 50Kg. Grâce à sa petite taille, les essais en mer demandent une logistique réduite au minimum à deux personnes et un bateau à moteur. Ce prototype dispose de trois heures d'autonomie à une vitesse de trois nœuds. Sa flottabilité positive lui permet de remonter à la surface une fois la mission effectuée. Cet AUV possède un propulseur à l'arrière, une paire de gouvernes de cap et deux paires de gouvernes de plongées situées à l'avant et à l'arrière du véhicule. Par ailleurs, ce robot est équipé d'une batterie de type NiMH (Nickel Métal Hybride) de 48V/16Ah pour alimenter son moteur à courant continu de 230W et ses quatres servo-moteurs de 30NCm pour le contrôle des gouvernes. La capacité de la torpille à plonger de la surface tout en maintenant son angle de tangage quasi nul, grâce à sa paire de gouvernes de plongée avant, constitue la principale caractéristique de cet engin [78]. L'équipement scientifique de Taipan II lui permet d'être employé pour de nombreuses applications sous-marines. Nous citerons notamment l'inspection de pipeline, l'étude de sources d'eau douce (sous-marine), la cartographie du fond marin.

Sur Taipan II, le sonar latéral est composé de deux barres longitudinales. Chacun de ces barreaux insonifie un secteur perpendiculaire à l'appareil et dirigé vers le bas. L'acquisition de ce capteur nous renseigne sur les altitudes relatives de chaque point de la zone insonifiée. La juxtaposition de plusieurs lignes acquises durant le déplacement de l'engin représente alors une cartographie du fond à l'aplomb duquel le véhicule s'est déplacé. Avec un traitement d'image approprié, il est possible de détecter différents objets dont un pipeline dans le cas qui nous intéresse. L'"image" reconstituée à partir des données de ce capteur permettent d'une part de longer le pipeline étudié mais également d'analyser sa surface à la recherche d'éventuelles fissures ou autres anomalies visibles à la surface. L'inspection de pipeline est une des principales applications de Taipan II.

Un capteur CTD, fixé sur la partie supérieure du véhicule, permet de mesurer trois

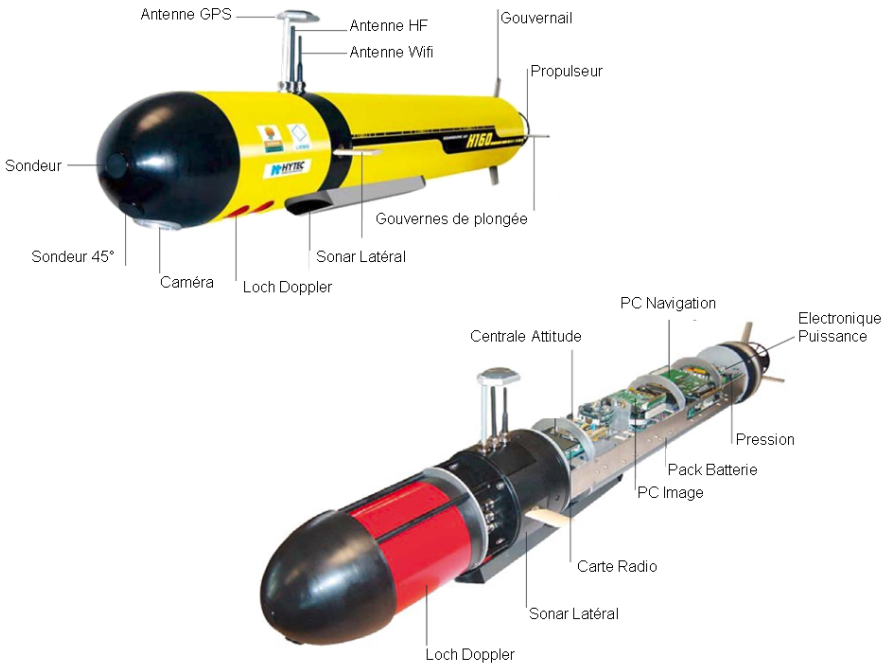


Fig. A.1 – Véhicule sous-marin autonome Taipan 2 ou H160

grandeurs physiques dont la conductivité électrique de l'eau environnante. Cette conductivité est une fonction de la concentration de l'eau en sel. Ce type d'acquisition sera utilisé pour l'analyse des résurgences d'eau douce sous-marine. En effet l'eau douce ayant une faible concentration en sel par rapport à l'eau de mer, en relevant la salinité d'un volume d'eau donné, nous pouvons reconstruire le profil d'une source sous-marine. Ce profil n'est autre que la surface de transition de la salinité de l'eau de valeurs hautes aux valeurs basses.

La caméra embarquée permet d'acquérir des images du fond, mais aussi d'estimer le vecteur vitesse de l'engin par analyse du flot optique. L'ensemble de ces capteurs sont représentés sur la figure A.1 et détaillés dans le tableau A.2.

A.2 Taipan 300

Le véhicule Taipan 300 est plus petit que Taipan II (voir figure A.3). Avec un poids de 27 kg cet engin présente l'avantage d'être facilement manipulable. Le transport vers les sites d'expérimentation ne nécessite pas un matériel particulier et le véhicule peut être lancé depuis la rive sans avoir recours à un bateau. Ses autres caractéristiques sont :

- longueur : 172 cm,
- diamètre : 15 cm,
- Profondeur maximum : 300 m,
- Vitesse moyenne : 2 $m.s^{-1}(3,9noeuds)$,
- Autonomie : 2 heures.
- Batterie : acide/plomb

Du fait de sa petite taille, Taipan 300 ne peut embarquer certains capteurs comme le Loch Doppler (utile pour connaître la vitesse de déplacement) ou le Sonar Latéral. Une

Tab. A.1 – Instrumentation de l'AUV H160

Communication			
Nom	Utilisation	Données brutes	Données traitées
Radio	Chargement de la mission, monitoring/téléopération en surface		
WiFi	Connexion à distance sur le PC embarqué (en surface et à courte distance)		
Modem acoustique	Monitoring en plongée et communication inter-véhicule (flottille)		
Pinger de secours	Balise de localisation		
Capteurs			
GPS	Localisation géoréférencée		
CTD	Mesure physico-chimique (caractérisation de sources d'eau douce)	Conductivité (mS.cm^{-1}), Température (°C), Pression (Pa).	Salinité ($g.l^{-1}$) obtenue à partir de la conductivité, température (°C), profondeur (m) obtenue à partir de la pression
Sondeur Acoustique 1 Sondeur Acoustique 2 Sondeur Acoustique 3	Suivi de fond et évitement d'obstacle	Temps de vol de l'onde acoustique	
Caméra vidéo	Acquisition d'image	Suivi d'amer	
Sonar à effet Doppler (Loch Doppler)	estimation des déplacements	vitesses (u,v,w) (m.s^{-1}) dans le repère local et attitude (ϕ, θ, ψ) ($rad.s^{-1}$) vitesses par rapport au fond ou vitesses par rapport à l'eau environnante	Attitude (ϕ, θ, ψ) (rad), vitesse ($\dot{x}, \dot{y}, \dot{z}$) ($m.s^{-1}$) obtenues à partir de (u,v,w et ψ) Position estimée de l'engin ($x, y, z, \phi, \theta, \psi$)
Sonar latéral	Cartographie, detection de pipeline, detection de mines	2 lignes perpendiculaires à l'engin constituées de mesures d'intensité de l'onde réfléchie (valeur entre 0 et 80 db) par le fond	
Centrale d'attitude et gyros.	Attitude et vitesse angulaire	Attitude (ϕ, θ, ψ) (rad), vitesse et accélérations angulaires (p,q,r) ($rad.s^{-1}$) et ($\dot{p}, \dot{q}, \dot{r}$) ($rad.s^{-2}$)	Attitude ϕ, θ, ψ (rad), vitesses et accélérations angulaires p,q,r ($rad.s^{-1}$) et ($\dot{p}, \dot{q}, \dot{r}$) ($rad.s^{-2}$)
Capteur de pression (6 bars)	Calcul de la profondeur (0-60m)	Pression (Pa)	profondeur (m)
Capteur de pression (16 bars)	Calcul de la profondeur (0-160m)	Pression (Pa)	profondeur (m)
Actionneurs			
Gouverne de cap (up)			
Gouverne de cap (down)			
Gouverne avant			
Gouverne arrière			
propulseur			

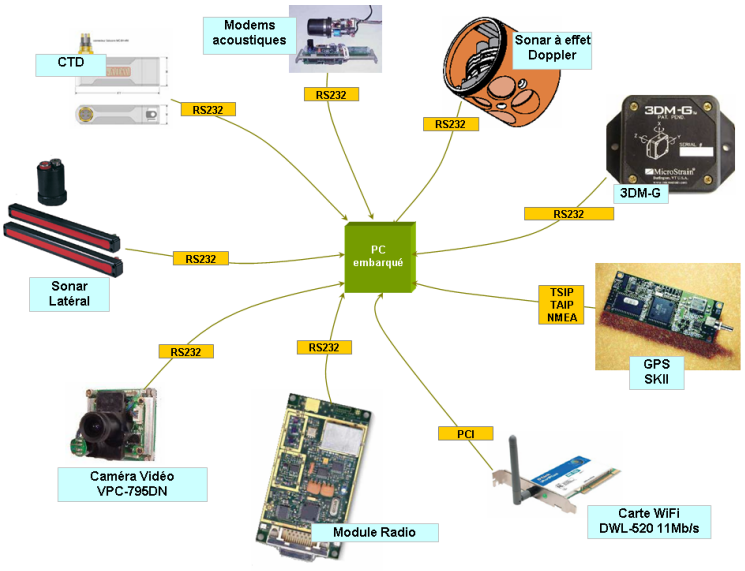


Fig. A.2 – Capteurs et moyens de communication de Taipan 2

Fig. A.3 – Taipan 300 devant le lac du Salagou

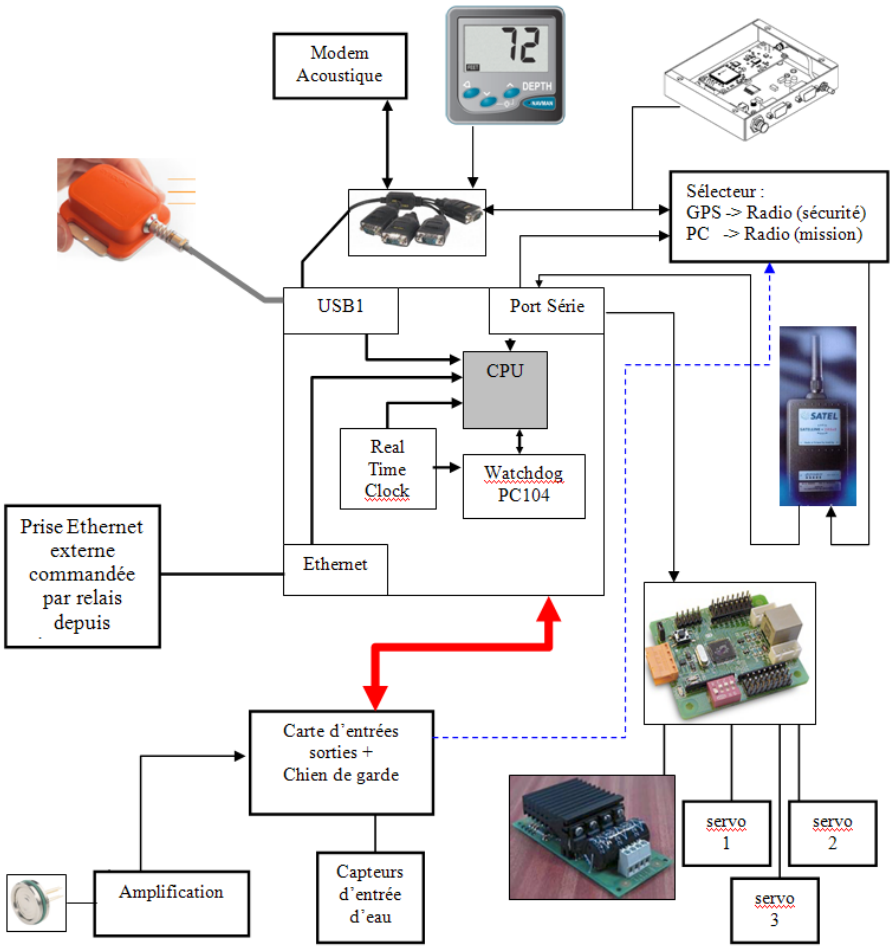


Fig. A.4 – Architecture matérielle informatique de Taipan 300 et connexions avec l'instrumentation

section (ou tronçon) du véhicule située avant la tête permet de recevoir soit une caméra vidéo soit un capteur CTD, soit un modem acoustique. Il est donc nécessaire de mettre en place un de ces capteurs suivant le type de mission à réaliser. Le tableau A.2 répertorie l'instrumentation présente actuellement sur ce véhicule. Taipan 300 est muni d'une carte mère embarquée de petite dimension (PC104) munie d'un processeur Intel Celeron 400 Mhz. Cette carte permet de recevoir comme unité de stockage, un disque dur IDE et une carte compact flash. Elle dispose en outre de deux ports USB et deux ports série qui vont nous servir à connecter l'instrumentation de bord.

Le premier port série est relié au module radio. Ce module radio établit une connexion avec un autre module identique relié au port série d'un ordinateur utilisateur. Ces modules sont configurés de façon à ce que la liaison radio ainsi établie entre les ports série des deux ordinateurs soit équivalente à un simple câble série croisé. Les communications radio étant impossible sous l'eau, ces modules sont exploités dans le cadre de téléopérations en surface.

Une carte électronique qui implémente un watchdog matériel est connectée sur le second port série de l'ordinateur embarqué. Ce watchdog requiert d'être stimulé à intervalles de temps réguliers de 1 seconde à défaut de quoi, il coupera automatiquement la puissance d'alimentation des moteurs du véhicule et connectera directement le GPS à la radio. Ainsi en cas de panne logicielle, les moteurs se mettront à l'arrêt et l'engin, fera naturellement

Tab. A.2 – Instrumentation de l'AUV Taipan 300

Communication			
Nom	Utilisation	Données brutes	Données traitées
Radio	Chargement de la mission, monitoring/téléopération en surface		
WiFi	Connexion à distance sur le PC embarqué (en surface et à courte distance)		
Modem acoustique	Monitoring en plongée et communication inter-véhicule (flottille)		
Pinger de secours	Balise de localisation		
Capteurs			
GPS	Localisation géoréférencée		
CTD	Mesure physico-chimique (caractérisation de sources d'eau douce)	Conductivité (mS.cm^{-1}), Température (°C), Pression (Pa).	Salinité obtenue à partir de la conductivité et température (°C), profondeur (z) (m) obtenue à partir de la pression
Sondeur Acoustique	Suivi de fond et évitement d'obstacle	Temps de vol de l'onde acoustique	
Caméra vidéo	Acquisition d'image et détermination de la vitesse par rapport au fond		
Centrale inertielle	Calcul de l'estime pour une vitesse de croisière connue	Attitude (ϕ, θ, ψ) (rad), vitesse et accélérations angulaires (p,q,r) $(rad.s^{-1})$ et $(\dot{p}, \dot{q}, \dot{r})$ $(rad.s^{-2})$	Attitude ϕ, θ, ψ (rad), vitesses et accélérations angulaires p,q,r $(rad.s^{-1})$ et $(\dot{p}, \dot{q}, \dot{r})$ $(rad.s^{-2})$
Capteur de pression (1 bars)	Calcul de la profondeur (0-10m)	Pression (Pa)	profondeur (m)
Capteur de pression (10 bars)	Calcul de la profondeur (0-100m)	Pression (Pa)	profondeur (m)
Capteur entrée d'eau	détecter une fuite		
Actionneurs			
Gouverne de cap (up)			
Gouverne de cap (down)			
Gouverne avant			
Gouverne arrière			
propulseur			

surface du fait de sa flottabilité positive. Une fois à la surface l'opérateur pourra capter le signal radio qui contient la position du véhicule mesurée par le GPS.

Un capteur d'entrée d'eau surveille l'humidité à l'intérieur du véhicule et près des connexions étanches. Ce capteur est relié au PC 104 via une carte entrée/sortie analogique et signale la présence d'eau à l'intérieur de la coque. La réaction à une entrée d'eau sera purement logicielle (gestion de la remontée en surface).

Un dongle (ou clef) Wifi est connectée sur un des port usb et permet à la torpille de créer un réseau Ad-hoc. Ce réseau nous permettra de transférer des fichiers depuis un ordinateur vers le véhicule. Ce matériel sera utilisé pour les expérimentations en laboratoire ou sur site avant la mise à l'eau (voir figure A.4).

Taipan 300 embarque très peu de matériel scientifique. La seule application envisageable aujourd'hui pour ce véhicule est la cartographie de source sous-marine d'eau douce. Ce véhicule a été déployé pour ce type de mission dans le cadre du projet Meditate, dans la baie de Gokova en Turquie durant l'été 2007.

A.3 Charlie

Le véhicule autonome de surface, appelé Charlie, est un petit catamaran A.5 qui a d'abord été développé et exploité durant la XIXe expédition italienne en Antarctique. Cet USV mesure 2.40m de long pour 1.70 m de large et pèse environ 300 Kg. Sa propulsion est assurée par deux moteurs à courant continu de 300 watts chacun (sous 48V). Un moteur brushless se charge quant à lui d'actionner une paire de safrans connecté entre eux, positionnés à l'arrière du système de propulsion. En ce qui concerne la navigation, ce véhicule est muni d'un GPS et d'un compas indiquant le nord véritable. Des batteries au plomb de 12V et 100Ah couplées à des panneaux solaires (32W) fournissent l'énergie du bord. L'ordinateur de bord est également un PC-104 sur lequel est exécuté Linux et qui supporte les liaisons ethernet et série. Il est à noter que dans sa nouvelle version, cet engin est équipé d'une caméra vidéo capable de retransmettre les images à terre par liaison sans fil WIFI. Un des objectifs de ce projet est d'évaluer des solutions dans le cadre de la navigation autonome, du guidage et du contrôle le long des côtes ou dans des zones portuaires. Une mission typique a par exemple consisté à recueillir des échantillons d'eau de surface en Antarctique afin de tester la présence de polluants et d'évaluer leur concentration [79]. Ce véhicule peut également être exploité dans le cadre de la surveillance anti-intrusion sous-marine des zones côtières ou des ports et faire du relevé bathymétrique en présence de trafic maritime. Par ailleurs une coopération avec le LIRMM est en cours afin d'évaluer les performances d'algorithmes permettant de faire du suivi de chemin et de l'évitement d'obstacles en coopération avec les AUVs Taipan [80]. Le tableau A.3 répertorie l'instrumentation présente actuellement sur ce véhicule.

Fig. A.5 – L'USV du CNR : le catamaran Charlie dans le port de Gênes en phase de test

Tab. A.3 – Instrumentation de l'ASV Charlie

Communication			
Nom	Utilisation	Données brutes	Données traitées
WiFi	Connexion à distance sur le PC embarqué pour envoyer la mission et obtenir au sol des images de la caméra embarquée en temps réel		
Modem acoustique	Monitoring en plongée et communication inter-véhicule (flottille)		
Capteurs			
GPS	Localisation géoréférencée		
Caméra vidéo	Acquisition d'images		
Compas	Permet d'obtenir le nord véritable grâce au couplage avec le GPS		
Actionneurs			
Gouvernail			
Propulseur tribord			
Propulseur babord			

Annexe B

Modélisation des véhicules

B.1 Taipan 2 & Taipan 300

B.1.1 Conventions

B.1.1.1 Référentiels

La modélisation nécessite la définition des référentiels par rapport auxquels on décrit l'évolution de l'engin, comme le montre la figure B.1.

On définit d'abord un repère absolu $R_0(O, X_0, Y_0, Z_0)$, avec :

- X_0 axe longitudinal confondu avec le Nord géographique,
- Y_0 axe transversal orienté vers l'Est,
- Z_0 axe normal dirigé vers le bas.

Un second repère $R_v(C, X_v, Y_v, Z_v)$, lié au véhicule permet d'exprimer les vitesses de l'engin. Les principaux axes d'inertie du véhicule coïncident avec les axes du repère :

- X_v axe longitudinal orienté de l'arrière vers l'avant de l'engin,
- Y_v axe transversal orienté vers tribord,
- Z_v axe normal dirigé du haut vers le bas.

Le choix du point d'origine C de ce repère est stratégique. La SNAME(Society of Naval Architects and Marine Engineers) propose une méthode pour choisir son emplacement en fonction des caractéristiques géométriques de l'engin [82].

Par exemple, si l'engin comporte des plans de symétrie, le point d'origine C appartient à l'intersection de ces plans de symétrie. Si le centre de gravité ou de flottabilité du véhicule appartient à cette intersection, le point d'origine est confondu avec l'un de ces deux points.

B.1.1.2 Vocabulaire

- Le *centre de gravité* d'un corps correspond au barycentre des particules qui composent le corps en question ; chaque particule étant pondérée par son poids propre. On l'appelle aussi *centre de masse*.
- Le *centre de carène* est défini comme étant le centre de volume d'une carène (partie immergée de la coque d'un bateau), qui correspond au centre de gravité du vo-

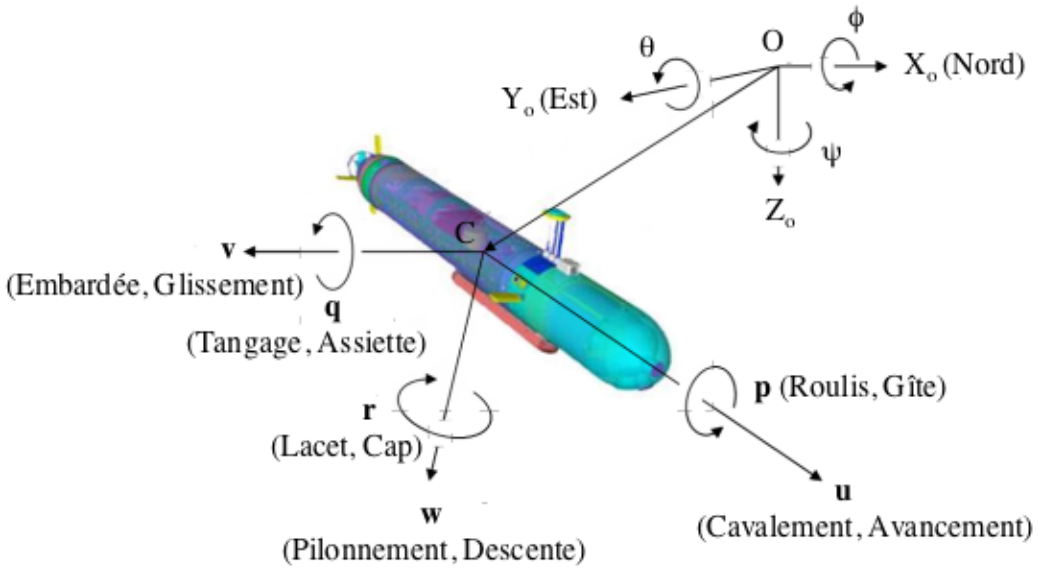


Fig. B.1 – Repères fixe et inertiel, Variables d'état

lume d'eau déplacée. On l'appelle aussi *centre de volume*. On peut également parler de *centre de poussée* en relation avec la Poussée d'Archimède. Par abus de langage, on dit également *centre de flottabilité* en référence au terme anglais "*centre of buoyancy*".

B.1.1.3 Dénomination des variables

Pour décrire l'attitude (position et orientation) de l'engin, on utilisera la notation suivante issue de la norme établie par la [82]. L'origine C du repère R_v est confondue avec le centre de gravité du véhicule. Pour définir la position de l'engin, le point C est défini dans le repère absolu R_0 par ses coordonnées cartésiennes :

$$\boldsymbol{\eta}_1 = \begin{pmatrix} x, & y, & z \end{pmatrix}^T$$

L'orientation de l'engin, définie dans le repère absolu, est exprimée par :

$$\boldsymbol{\eta}_2 = \begin{pmatrix} \phi, & \theta, & \psi \end{pmatrix}^T$$

où ϕ, θ et ψ représentent respectivement les angles de roulis, de tangage et de lacet (cf. figure B.2).

Le vecteur position général s'exprime par :

$$\boldsymbol{\eta} = \begin{pmatrix} \boldsymbol{\eta}_1, & \boldsymbol{\eta}_2 \end{pmatrix}^T \tag{B.1}$$

Pour définir le vecteur vitesse $\boldsymbol{\nu}$ de l'engin exprimé dans R_v, on adoptera la notation suivante :

$$\boldsymbol{\nu}_1 = \begin{pmatrix} u, & v, & w \end{pmatrix}^T$$

où u, v et w représentent respectivement les vitesses linéaires d'avancement, de glissement et de descente, et :

$$\boldsymbol{\nu}_2 = \begin{pmatrix} p, & q, & r \end{pmatrix}^T$$

où p, q et r représentent respectivement les vitesses angulaires de roulis, de tangage et de lacet.

Le vecteur vitesse global s'écrit alors :

$$\boldsymbol{\nu} = \begin{pmatrix} \boldsymbol{\nu}_1, & \boldsymbol{\nu}_2 \end{pmatrix}^T \tag{B.2}$$

B.1.2 Cinématique

B.1.2.1 Les angles d'Euler

Les "angles d'Euler" utilisés dans ce rapport correspondent au systeme dit R.T.L. en robotique, pour Roulis, Tangage, Lacet (ϕ, θ et ψ), décrits sur la figure B.2.

D'autres descriptions peuvent être utilisées, comme par exemple, les cosinus directeurs, les paramètres d'Euler ou encore les quaternions. L'inconvénient majeur de la représentation par les angles d'Euler réside dans l'existence d'une singularité pour un angle de tangage $\theta = \frac{\pi}{2} \pm k\pi$. Une description par les quaternions permet d'éviter cette singularité. Toutefois, dans notre cas, cette singularité correspond à une situation extrême que l'engin, par hypothèse, n'atteindra jamais.

B.1.2.2 Transformation des vitesses linéaires

La trajectoire du véhicule dans le référentiel inertiel lié a la Terre est obtenue par la relation cinématique suivante :

$$\dot{\boldsymbol{\eta}}_1 = \boldsymbol{J}_{C_1}(\boldsymbol{\eta}_2)\boldsymbol{\nu}_1 \tag{B.3}$$

où $\boldsymbol{J}_{C_1}(\boldsymbol{\eta}_2)$ est la matrice de rotation de $R(x, y, z)$ à $R_v(X_v, Y_v, Z_v)$, c'est une matrice de déterminant unité ayant pour inverse sa transposée.

$$\boldsymbol{J}_{C_1}(\boldsymbol{\eta}_2) = \begin{bmatrix} \cos\theta\cos\psi & \sin\theta\sin\phi\cos\psi - \sin\psi\cos\phi & \sin\theta\cos\phi\cos\psi + \sin\psi\sin\phi \\ \cos\theta\sin\psi & \sin\theta\sin\phi\sin\psi + \cos\phi\cos\psi & \sin\theta\cos\phi\sin\psi - \cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{bmatrix} \tag{B.4}$$

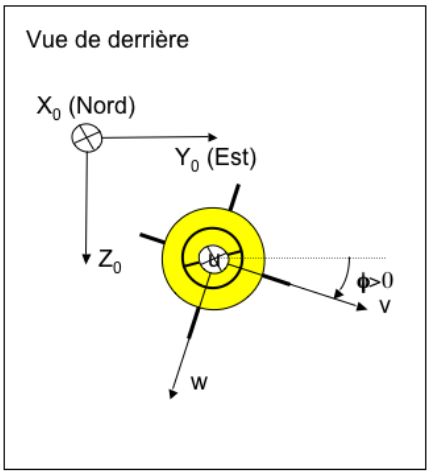
B.1.2.3 Transformation des vitesses angulaires

Les vitesses angulaires dans les différents repères considérés sont liées par la relation :

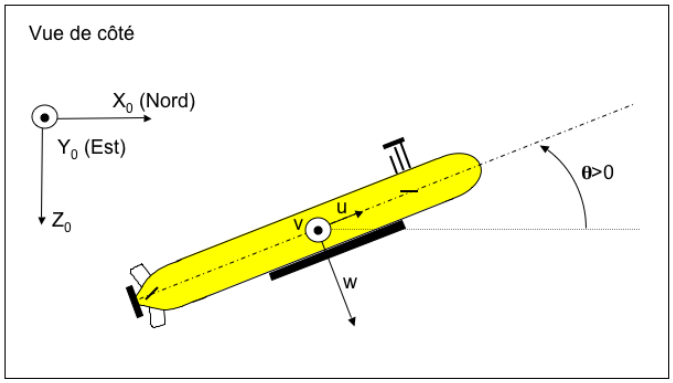
$$\dot{\boldsymbol{\eta}}_2 = \boldsymbol{J}_{C_2}(\boldsymbol{\eta}_2)\boldsymbol{\nu}_2 \tag{B.5}$$

où $\boldsymbol{J}_{C_2}(\boldsymbol{\eta}_2)$ est la matrice :

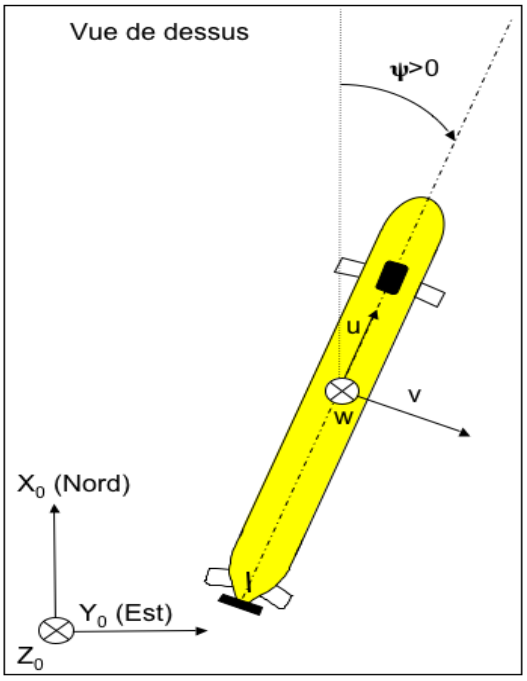
$$\boldsymbol{J}_{C_2}(\boldsymbol{\eta}_2) = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix}, \quad \theta \neq \frac{\pi}{2} \pm k\pi \tag{B.6}$$



(a) Angle de Roulis



(b) Angle de Tangage



(c) Angle de Lacet

Fig. B.2 – Les angles d'Euler

B.1.2.4 Relation générale de la cinématique

D'une façon générale, la relation cinématique, basée sur les équations (B.3) et (B.5) est :

$$\dot{\boldsymbol{\eta}} = \boldsymbol{J}_C(\boldsymbol{\eta}_2)\boldsymbol{\nu} = \begin{bmatrix} \dot{\boldsymbol{\eta}}_1 \\ \dot{\boldsymbol{\eta}}_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{J}_{C_1}(\boldsymbol{\eta}_2) & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \boldsymbol{J}_{C_2}(\boldsymbol{\eta}_2) \end{bmatrix} \begin{bmatrix} \boldsymbol{\nu}_1 \\ \boldsymbol{\nu}_2 \end{bmatrix} \tag{B.7}$$

Si le lecteur désire plus de détails concernant l'obtention des matrices de transformation $\boldsymbol{J}_{C_1}(\boldsymbol{\eta}_2)$ et $\boldsymbol{J}_{C_2}(\boldsymbol{\eta}_2)$ (3 rotations successives pour passer de R_0 a R_v), il pourra se référer à [83] ou encore à [63].

B.1.3 Équations de la dynamique

La dynamique d'un véhicule sous-marin consiste dans l'étude des mouvements engendrés par les effets de certaines actions de contrôle comme l'orientation des gouvernes, ou extérieures comme l'action de la houle lorsqu'il navigue au voisinage de la surface, ou encore les courants marins.

B.1.3.1 Dynamique d'un corps rigide

Les équations générales du mouvement d'un solide indéformable à 6 ddl traduisent les mouvements de translation et de rotation de ce solide. Elles sont basées sur le formalisme de Newton et Lagrange et elles sont établies en adoptant les conventions de [82]. Soit G le centre de gravité du véhicule de coordonnées $\overrightarrow{\boldsymbol{CG}} = [x_G, y_G, z_G]^T$ dans le repère véhicule, m la masse du véhicule, et $\boldsymbol{\Gamma}_1 = [X, Y, Z]^T$ et $\boldsymbol{\Gamma}_2 = [K, M, N]^T$ respectivement les forces et les moments qui s'appliquent sur le véhicule.
On obtient deux séries d'équations : **Equation des forces**
La dynamique en translation d'un corps rigide se traduit sous la forme suivante :

$$m\left[\dot{\boldsymbol{\nu}}_1 + \boldsymbol{\nu}_2 \wedge \boldsymbol{\nu}_1 + \dot{\boldsymbol{\nu}}_2 \wedge \overrightarrow{\boldsymbol{CG}} + \boldsymbol{\nu}_2 \wedge \left(\dot{\boldsymbol{\nu}}_2 \wedge \overrightarrow{\boldsymbol{CG}}\right)\right] = \boldsymbol{\Gamma}_1 \tag{B.8}$$

Equation des moments
Soit $\boldsymbol{I}_0$ la matrice d'inertie du véhicule définie par :

$$\boldsymbol{I}_0 = \begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}$$

avec, les I_{ii} étant les moments d'inertie et les I_{ij} les produits d'inertie.

La dynamique en rotation d'un corps rigide s'écrit :

$$\boldsymbol{I}_0\dot{\boldsymbol{\nu}}_2 + \boldsymbol{\nu}_2 \wedge (\boldsymbol{I}_0\boldsymbol{\nu}_2) + m\overrightarrow{\boldsymbol{CG}} \wedge (\dot{\boldsymbol{\nu}}_1 + \boldsymbol{\nu}_2 \wedge \boldsymbol{\nu}_1) = \boldsymbol{\Gamma}_2 \tag{B.9}$$

Synthèse

Si on développe les équations (B.8) et (B.9), on obtient :

$$\begin{aligned}
m\left[\dot{u} - vr + wq - x_g\left(q^2 + r^2\right) + y_g\left(pq - \dot{r}\right) + z_g\left(pr + \dot{q}\right)\right] &= X \\
m\left[\dot{v} - wp + ur - y_g\left(r^2 + p^2\right) + z_g\left(qr - \dot{p}\right) + x_g\left(qp + \dot{r}\right)\right] &= Y \\
m\left[\dot{w} - uq + vp - z_g\left(p^2 + q^2\right) + x_g\left(rp - \dot{q}\right) + y_g\left(rq + \dot{p}\right)\right] &= Z \\
I_{xx}\dot{p} + \left(I_{zz} - I_{yy}\right)qr - \left(\dot{r} + pq\right)I_{xz} + \left(r^2 - q^2\right)I_{yz} + \left(pr - \dot{q}\right)I_{xy} & \\
+m\left[y_g\left(\dot{w} - uq + vp\right) - z_g\left(\dot{v} - wp + ur\right)\right] &= K \\
I_{yy}\dot{q} + \left(I_{xx} - I_{zz}\right)rp - \left(\dot{p} + qr\right)I_{xy} + \left(p^2 - r^2\right)I_{zx} + \left(qp - \dot{r}\right)I_{yz} & \\
+m\left[z_g\left(\dot{u} - vr + wq\right) - x_g\left(\dot{w} - uq + vp\right)\right] &= M \\
I_{zz}\dot{r} + \left(I_{yy} - I_{xx}\right)pq - \left(\dot{q} + rp\right)I_{yz} + \left(q^2 - p^2\right)I_{xy} + \left(rq - \dot{p}\right)I_{zx} & \\
+m\left[x_g\left(\dot{v} - wp + ur\right) - y_g\left(\dot{u} - vr + wq\right)\right] &= N
\end{aligned} \tag{B.10}$$

Le système d'équations (B.10) peut se mettre sous forme matricielle et devient :

$$\boldsymbol{M}_{rb}\dot{\boldsymbol{\nu}} + \boldsymbol{C}_{rb}(\boldsymbol{\nu})\boldsymbol{\nu} = \boldsymbol{\Gamma} \tag{B.11}$$

où,

- $\boldsymbol{M}_{rb}$ est la matrice d'inertie du système, définie positive :

$$\boldsymbol{M}_{rb} = \begin{bmatrix}
m & 0 & 0 & 0 & mz_G & -my_G \\
0 & m & 0 & -mz_G & 0 & mx_G \\
0 & 0 & m & my_G & -mx_G & 0 \\
0 & -mz_G & my_G & I_{xx} & -I_{xy} & -I_{xz} \\
mz_G & 0 & -mx_G & -I_{yx} & I_{yy} & -I_{yz} \\
-my_G & mx_G & 0 & -I_{zx} & -I_{zy} & I_{zz}
\end{bmatrix} \tag{B.12}$$

- $\boldsymbol{C}_{rb}$ est la matrice des forces de Coriolis et centrifuges, anti-symétrique :

$$\boldsymbol{C}_{rb} = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0 \\
-m(y_G q + z_G r) & m(y_G p + w) & m(z_G p - v) \\
m(x_G q - w) & -m(z_G r + x_G p) & m(z_G q + u) \\
m(x_G r + v) & m(y_G r - u) & -m(x_G p + y_G q)
\end{bmatrix} \tag{B.13}$$

$$\begin{bmatrix}
m(y_G q + z_G r) & -m(x_G q - w) & -m(x_G r + v) \\
-m(y_G p + w) & m(z_G r + x_G p) & -m(y_G r - u) \\
-m(z_G p - v) & -m(z_G q + u) & m(x_G p + y_G q) \\
0 & -I_{yz}q - I_{xz}p + I_{zz}r & I_{yz}r + I_{xy}p - I_{yy}q \\
I_{yz}q + I_{xz}p - I_{zz}r & 0 & -I_{xz}r - I_{xy}q + I_{xx}p \\
-I_{yz}r - I_{xy}p + I_{yy}q & I_{xz}r + I_{xy}q - I_{xx}p & 0
\end{bmatrix}$$

- $\boldsymbol{\Gamma} = [\boldsymbol{\Gamma}_1, \boldsymbol{\Gamma}_2]^t$ est le vecteur des forces et moments qui s'appliquent sur le véhicule, qui peut être décomposé de la façon suivante :

$$\boldsymbol{\Gamma} = \boldsymbol{\Gamma}_h + \boldsymbol{\Gamma}_g + \boldsymbol{\Gamma}_u + \boldsymbol{\Gamma}_p$$

où,

$\boldsymbol{\Gamma}_h$ regroupe les forces et moments hydrodynamiques,

$\boldsymbol{\Gamma}_g$ est le vecteur des forces et moments dus à l'action de la gravité et de la flottabilité,

$\boldsymbol{\Gamma}_u$ est le vecteur des forces et moments générés par les actionneurs du véhicule. Nous considérons qu'il est déterminé par l'addition des effets de chacun des actionneurs du véhicule,

$\boldsymbol{\Gamma}_p$ regroupe les forces et moments résultant des perturbations dues à l'environnement (courants marins, houle, ...).

B.1.3.2 Efforts hydrodynamiques

Ils agissent sur tout corps immergé en mouvement relatif dans un fluide visqueux, et peuvent être classés comme suit :

1. Les forces et moments dus à l'inertie et à la masse d'eau ajoutée,
2. Les forces dues aux frottements visqueux du fluide sur le corps, qui correspondent aux efforts de portance et de traînée.

La principale difficulté réside dans leur connaissance et leur formulation. En effet, ces efforts ne peuvent pas être obtenus de manière analytique.

Inertie et masse d'eau ajoutée

Du point de vue physique, tout corps mobile en eau libre provoque un déplacement d'une certaine quantité de cette eau. Le bilan des efforts dus à l'inertie et à la masse d'eau ajoutée peut se mettre sous la forme :

$$\boldsymbol{\Gamma}_{aj} = -(\boldsymbol{M}_a\dot{\boldsymbol{\nu}} + \boldsymbol{C}_a(\boldsymbol{\nu})\boldsymbol{\nu}) \tag{B.14}$$

où,

- $\boldsymbol{M}_a$ est la matrice d'inertie d'eau ajoutée, définie positive et peut se mettre la forme suivante :

$$\boldsymbol{M}_a = -\begin{bmatrix} X_{\dot{u}} & X_{\dot{v}} & X_{\dot{w}} & X_{\dot{p}} & X_{\dot{q}} & X_{\dot{r}} \\ Y_{\dot{u}} & Y_{\dot{v}} & Y_{\dot{w}} & Y_{\dot{p}} & Y_{\dot{q}} & Y_{\dot{r}} \\ Z_{\dot{u}} & Z_{\dot{v}} & Z_{\dot{w}} & Z_{\dot{p}} & Z_{\dot{q}} & Z_{\dot{r}} \\ K_{\dot{u}} & K_{\dot{v}} & K_{\dot{w}} & K_{\dot{p}} & K_{\dot{q}} & K_{\dot{r}} \\ M_{\dot{u}} & M_{\dot{v}} & M_{\dot{w}} & M_{\dot{p}} & M_{\dot{q}} & M_{\dot{r}} \\ N_{\dot{u}} & N_{\dot{v}} & N_{\dot{w}} & N_{\dot{p}} & N_{\dot{q}} & N_{\dot{r}} \end{bmatrix} \tag{B.15}$$

  Une propriété essentielle de cette matrice est : $M_{ij} = M_{ji}$. Nous pouvons ajouter aussi que, par convention, tous les coefficients sont négatifs.
- $\boldsymbol{C}_a$ est la matrice des forces de Coriolis et des forces centrifuges hydrodynamiques. Elle est de la forme :

$$\boldsymbol{C}_a = \begin{bmatrix} 0 & 0 & 0 & 0 & -a_3 & a_2 \\ 0 & 0 & 0 & a_3 & 0 & -a_1 \\ 0 & 0 & 0 & -a_2 & a_1 & 0 \\ 0 & -a_3 & a_2 & 0 & -b_3 & b_2 \\ a_3 & 0 & -a_1 & b_3 & 0 & -b_1 \\ -a_2 & a_1 & 0 & -b_2 & b_1 & 0 \end{bmatrix} \tag{B.16}$$

avec :

$$
\begin{aligned}
a_1 &= X_{\dot{u}}u + X_{\dot{v}}v + X_{\dot{w}}w + X_{\dot{p}}p + X_{\dot{q}}q + X_{\dot{r}}r \\
a_2 &= Y_{\dot{u}}u + Y_{\dot{v}}v + Y_{\dot{w}}w + Y_{\dot{p}}p + Y_{\dot{q}}q + Y_{\dot{r}}r \\
a_3 &= Z_{\dot{u}}u + Z_{\dot{v}}v + Z_{\dot{w}}w + Z_{\dot{p}}p + Z_{\dot{q}}q + Z_{\dot{r}}r \\
b_1 &= K_{\dot{u}}u + K_{\dot{v}}v + K_{\dot{w}}w + K_{\dot{p}}p + K_{\dot{q}}q + K_{\dot{r}}r \\
b_2 &= M_{\dot{u}}u + M_{\dot{v}}v + M_{\dot{w}}w + M_{\dot{p}}p + M_{\dot{q}}q + M_{\dot{r}}r \\
b_3 &= N_{\dot{u}}u + N_{\dot{v}}v + N_{\dot{w}}w + N_{\dot{p}}p + N_{\dot{q}}q + N_{\dot{r}}r
\end{aligned}
$$

Portance et Traînée

Ce sont des forces qui s'exercent sur tout corps en incidence par rapport à un fluide visqueux en écoulement. L'angle d'incidence ε est défini sur la figure B.3.

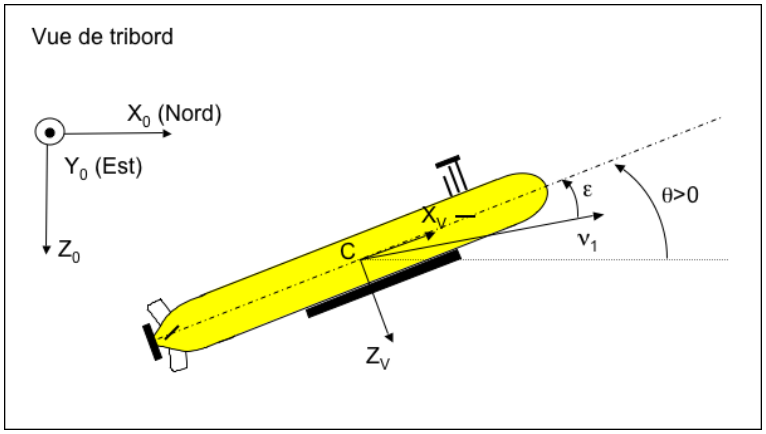


Fig. B.3 – Définition de l'angle d'incidence ε

Les efforts de portance et de traînée sont deux composantes des efforts de résistance de l'eau aux mouvements de l'engin. Les termes portance et traînée sont plutôt utilisés pour les actionneurs type ailerons ou gouvernes, et lorsqu'il s'agit du corps principal du véhicule hors actionneurs, on parlera alors des efforts d'amortissement. Le bilan des efforts dus aux amortissements peut se mettre sous la forme :

$$\boldsymbol{\Gamma}_{am} = (\boldsymbol{D}_p + \boldsymbol{D}_t(\boldsymbol{\nu})\boldsymbol{\nu}) \tag{B.17}$$

où,

- $\boldsymbol{D}_p$ est la matrice d'amortissement linéaire, définie négative et constante :

$$\boldsymbol{D}_p = -\begin{bmatrix} X_u & X_v & X_w & X_p & X_q & X_r \\ Y_u & Y_v & Y_w & Y_p & Y_q & Y_r \\ Z_u & Z_v & Z_w & Z_p & Z_q & Z_r \\ K_u & K_v & K_w & K_p & K_q & K_r \\ M_u & M_v & M_w & M_p & M_q & M_r \\ N_u & N_v & N_w & N_p & N_q & N_r \end{bmatrix} \tag{B.18}$$

- $\boldsymbol{D}_t$ est la matrice d'amortissement non linéaire, définie négative et non constante. L'effort résultant a une expression quadratique :

$$\boldsymbol{D}_t = -\begin{bmatrix} |\nu|^T D_X \\ |\nu|^T D_Y \\ |\nu|^T D_Z \\ |\nu|^T D_K \\ |\nu|^T D_M \\ |\nu|^T D_N \end{bmatrix} \tag{B.19}$$

avec

$$\boldsymbol{D}_j = -\begin{bmatrix} j_{uu} & j_{uv} & j_{uw} & j_{up} & j_{uq} & j_{ur} \\ j_{vu} & j_{vv} & j_{vw} & j_{vp} & j_{vq} & j_{vr} \\ j_{wu} & j_{wv} & j_{ww} & j_{wp} & j_{wq} & j_{wr} \\ j_{pu} & j_{pv} & j_{pw} & j_{pp} & j_{pq} & j_{pr} \\ j_{qu} & j_{qv} & j_{qw} & j_{qp} & j_{qq} & j_{qr} \\ j_{ru} & j_{rv} & j_{rw} & j_{rp} & j_{rq} & j_{rr} \end{bmatrix}$$

où $j = X, Y, Z, K, M, N$.

B.1.3.3 Gravité et Flottabilité

A l'arrêt, l'engin est uniquement soumis à son poids et à sa flottabilité. Ces efforts dépendent des caractéristiques de l'engin et des propriétés du milieu aquatique.

Soit ρ la masse volumique de l'eau de mer, qui dépend de la salinité, de la pression et de la température. Soient Δ le volume de la torpille et F le centre de flottabilité du véhicule de coordonnées $\overrightarrow{\boldsymbol{CF}} = [x_F, y_F, z_F]^t$ dans le repère véhicule. Soit g l'accélération de la pesanteur.

La poussée d'Archimede est la force qui s'exerce sur toute partie immergée d'un corps. Elle est égale à la force opposée au poids du volume de fluide déplacé $B = -\rho\Delta g$. Le poids du véhicule est égal à $W = mg$.

Le vecteur des forces hydrostatiques $\boldsymbol{\Gamma}_g$ s'écrit alors :

$$\boldsymbol{\Gamma}_g = g\begin{bmatrix} -(m-\rho\Delta)\sin\theta \\ (m-\rho\Delta)\cos\theta\sin\phi \\ (m-\rho\Delta)\cos\theta\cos\phi \\ (y_G m - y_F\rho\Delta)\cos\theta\cos\phi - (z_G m - z_F\rho\Delta)\cos\theta\sin\phi \\ -(z_G m - z_F\rho\Delta)\sin\theta - (x_G m - x_F\rho\Delta)\cos\theta\cos\phi \\ (x_G m - x_F\rho\Delta)\cos\theta\sin\phi + (y_G m - y_F\rho\Delta)\sin\theta \end{bmatrix} \tag{B.20}$$

B.1.3.4 Actionneurs hydrodynamiques

On peut répartir les actionneurs en deux groupes :

1. Ceux qui agissent par modification des caractéristiques d'un écoulement existant (surfaces mobiles : ailerons ou gouvernes),
2. Ceux qui génèrent l'écoulement de fluide qui peut être initialement au repos (propulseurs).

Concernant les propulseurs, il en existe une grande variété, mais nous nous intéressons uniquement au propulseur du Taipan 2 et 300 : un propulseur à hélice.

Surfaces mobiles

Toute surface en incidence par rapport a un fluide visqueux en écoulement est soumise

à une force de portance F_P perpendiculaire à cette surface et une force de traînée F_T parallèle à celle-ci (figure B.4).

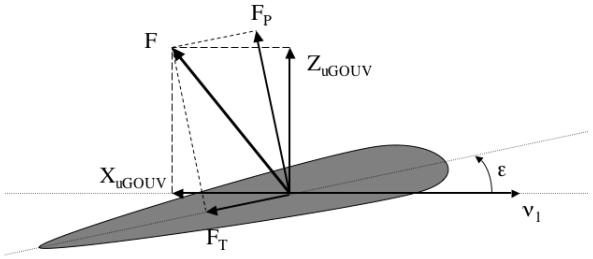


Fig. B.4 – Vue en coupe de la gouverne

On considère une aile mobile dont l'axe de rotation se trouve à une distance d_a de l'origine du repère véhicule. Avec l'hypothèse que la gouverne est fixée sur une surface perpendiculaire, les coefficients de portance et de traînée rapportés aux axes de l'aile sont définis par [64] :

$$C_{Z_s} = \frac{2\pi\lambda_{s_e}\left[1-3\left(\frac{e_s}{c_s}\right)^2\right]}{\sqrt{\left(\frac{\lambda_{s_e}^2}{\cos^2\gamma_s}+2\right)}\cos\gamma_s+2.6}(\varepsilon+\delta)+2.1(\varepsilon+\delta)^3 \quad \text{(B.21)}$$

$$C_{X_s} = 0.01-0.7\lambda_{s_e}(\varepsilon+\delta)^2 \quad \text{(B.22)}$$

où,

- λ_{s_e} est l'allongement effectif de l'aile : $\lambda_{s_e} = 2\lambda_s = \dfrac{b_s}{c_s}$, avec b_s l'envergure (longueur) de l'aile et c_s la corde (largeur) de l'aile,
- e_s/c_s désigne l'épaisseur relative de l'aile,
- γ_s est la flèche de l'aile, supposée nulle dans cette étude,
- ε représente l'angle d'incidence du corps principal du véhicule,
- δ caractérise le braquage de l'aile, et est l'élément de commande.

En se rapportant aux axes du repère véhicule, nous obtenons les efforts de portance et de traînée (figure B.5) :

$$\begin{aligned} X_{u_{GOUV}} &= -0.5\rho S_s V_0^2(C_{Z_s}\sin\delta + C_{X_s}\cos\delta) \\ Z_{u_{GOUV}} &= -0.5\rho S_s V_0^2(C_{Z_s}\cos\delta - C_{X_s}\sin\delta) \\ M_{u_{GOUV}} &= Z_{u_{GOUV}}(0.2c_s\cos\delta - d_a) + X_{u_{GOUV}}(0.2c_s\sin\delta) \end{aligned} \quad \text{(B.23)}$$

avec,

- S_s la surface de l'aile définie par $S_s = b_s c_s$,
- V_0 le module de l'écoulement du fluide autour de l'aile,
- $0.2c_s$ la distance du bord d'attaque de l'aile au point d'application des forces hydrodynamiques.

Propulseur

Des pales inclinées formant une hélice sont fixées sur un arbre en rotation et permettent de générer une force de poussée dans l'axe de rotation de l'arbre. Une approximation de

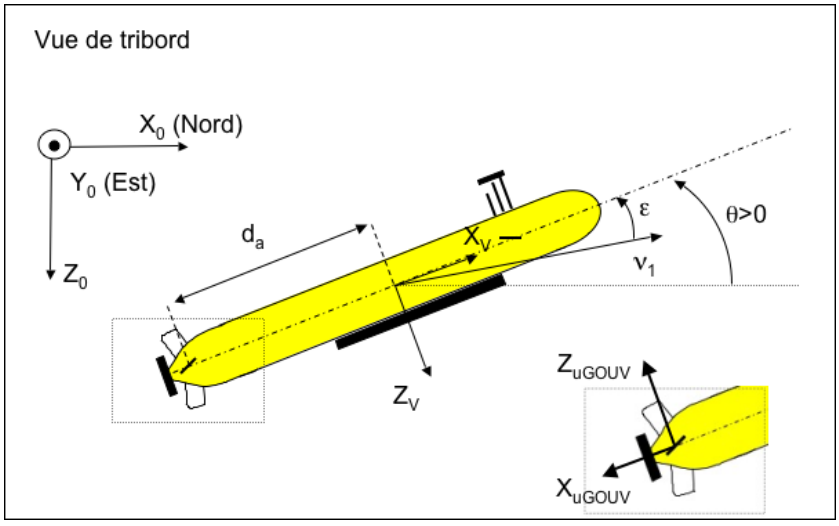


Fig. B.5 – Forces de portance $Z_{u_{GOUV}}$ et de traînée $X_{u_{GOUV}}$ rapportées au repère véhicule

la poussée T_p et du couple résistant Q générés dans le cas d'un propulseur à une hélice est [64] :

$$\begin{array}{rcccl} X_{u_{PROP}} & = & T_p & = & \rho D_p^4 K_T(J_0)|n_p|n_p \\ K_{u_{PROP}} & = & Q & = & \rho D_p^5 K_Q(J_0)|n_p|n_p \end{array} \tag{B.24}$$

où,

- n_p est la vitesse de rotation de l'hélice,
- ρ est la masse volumique de l'eau de mer,
- D_p est le diamètre de l'hélice,
- J_0 est le coefficient d'avancement du propulseur dans l'eau, il est défini par :

$$J_0 = \frac{V_a}{n_p D_p}$$

  avec, V_a la vitesse moyenne de l'eau autour de l'hélice, définie par : $V_a = (1 - w_a)V_0$, où V_0 est la composante axiale de la vitesse de l'eau en amont du propulseur, et w_a est un coefficient compris dans $[0.1\,;0.4]$ caractérisant le sillage du véhicule.
- K_T est le coefficient de poussée qui est égal à : $K_T = Ct_0 + Ct_1 J_0 + Ct_2 J_0^2 + Ct_2 J_0^3 + Ct_3 J_0^3$, ou les constantes Ct_i sont données dans le tableau (B.1),
- K_Q est le coefficient de couple égal à : $K_Q = Cq_0 + Cq_1 J_0 + Cq_2 J_0^2 + Cq_2 J_0^3 + Cq_3 J_0^3$, où les constantes Cq_i sont données dans le tableau (B.1).

B.1.3.5 Perturbations

L'environnement sous-marin introduit des effets perturbateurs de nature :

- *Non-additive*, par la modification des coefficients hydrodynamiques liés au milieu marin. Le principal coefficient hydrodynamique pouvant introduire des perturbations importantes est *la masse volumique de l'eau de mer*, ou de maniere équivalente *sa densité*.
- *Additive*, par l'action d'un mouvement ou d'une force supplémentaire sur la dynamique initiale du véhicule. C'est le cas *des courants marins, de la houle*, et dans une moindre mesure pour les déplacements proches de la surface, *le vent*.

Tab. B.1 – Coefficients de poussée et de couple utilisés pour le propulseur de H160. Valables hors cavitation.

i	Ct_i	Cq_i
0	0.50539	0.090271
1	-0.088971	-0.013470
2	-0.29960	-0.023529
3	0.046836	-0.0020050

Densité de l'eau de mer

La masse volumique de l'eau de mer, milieu dans lequel évolue la torpille, est un facteur qui intervient dans la détermination des coefficients hydrodynamiques du véhicule. Il s'agit ici de présenter la plage de variation de cette perturbation à travers la variable équivalente qu'est la densité.

La densité de l'eau de mer ϑ dépend de la salinité S_e, de la température T_e et de la pression P_e au point considéré. Un véhicule sous-marin, même parfaitement équilibré, évolue dans un milieu où la densité peut croître ou décroître légèrement selon le gradient de salinité ou de température.

En conclusion, nous retiendrons que les coefficients hydrodynamiques d'un sous-marin évoluent selon la localisation géographique de l'engin (pôle Nord ou mer Méditerranée par exemple) et sa profondeur.

Courants marins

Les courants marins sont les résultats d'un certain nombre de facteurs incluant :

- les gradients de température et de densité de l'eau de mer,
- la marée,
- la rotation de la Terre,
- les effets dus a l'activité solaire,
- les vents.

Ils sont aussi influencés par la proximité des côtes et la topographie du fond marin. [63] propose, pour les besoins de simulation, un modèle simplifié des courants marins.

Vagues

Une des raisons courantes de l'apparition des vagues est le vent. En effet, une vague est une oscillation de surface créée par une différence de pression ou de vitesse de fluide. Fossen présente dans [63] un état de l'art sur les modèles de vagues existants.

B.1.3.6 Relation générale de la dynamique

La modélisation dynamique d'un véhicule sous-marin de type torpille, dans le repère R_v, conduit aux équations générales suivantes :

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} = \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{\Gamma}_g + \boldsymbol{\Gamma}_p + \boldsymbol{\Gamma}_u \tag{B.25}$$

avec,

- $\boldsymbol{M}$ est la matrice d'inertie, symétrique et définie positive. Elle est égale à :

$$\boldsymbol{M} = \boldsymbol{M}_{rb} + \boldsymbol{M}_a$$

où, $\boldsymbol{M}_{rb}$ et $\boldsymbol{M}_a$ sont respectivement déterminées par les relations B.15 et B.12.
- $\boldsymbol{C}(\boldsymbol{\nu})$ est le vecteur de forces de Coriolis et d'eau ajoutée, défini par :

$$\boldsymbol{C}(\boldsymbol{\nu}) = \boldsymbol{C}_{rb}(\boldsymbol{\nu}) + \boldsymbol{C}_a(\boldsymbol{\nu})$$

$\boldsymbol{C}_{rb}$ et $\boldsymbol{C}_a$ étant les matrices respectivement déterminées par les relations B.13 et B.16.
- $\boldsymbol{D}(\boldsymbol{\nu})$ est la matrice des coefficients d'amortissement. Elle est égale à :

$$\boldsymbol{D}(\boldsymbol{\nu}) = \boldsymbol{D}_p(\boldsymbol{\nu}) + \boldsymbol{D}_t(\boldsymbol{\nu})$$

$\boldsymbol{D}_p$ et $\boldsymbol{D}_t$ étant respectivement les matrices d'amortissement linéaire et non linéaire déterminées par les relations B.18 et B.19.
- $\boldsymbol{\gamma}_g$ est le vecteur des forces et moments dus à l'action de la gravité et de la poussée d'Archimède, défini en B.20.
- $\boldsymbol{\gamma}_p$ regroupe les perturbations environnementales tels que les effets dus aux courants, à la houle...
- $\boldsymbol{\gamma}_u$ est le vecteur des forces et moment générés par les actionneurs du véhicule.

B.2 Charlie

B.2.1 Conventions

Le repère attaché à la Terre est appelé $< e >$. On y exprime la position et l'orientation du véhicule $[x, y, \psi]$. Le repère attaché au véhicule est appelé $< b >$. La vitesse d'avancement, la vitesse de glissement ($[u, v]$ pour les vitesses absolues ou $[u_r, v_r]$ pour les vitesses relatives à l'eau), le taux de giration r et les forces et moments $[XYN]$ sont représentés dans ce repère. La figure B.6 représente l'USV Charlie, les repères mentionnés, la vitesse absolue et relative, la position des actionneurs et enfin l'angle des gouvernails.

B.2.2 Cinématique

En notant $[\dot{x}_C \dot{y}_C]^T$ le courant, la vitesse absolue et la vitesse par rapport au fluide dans le repère lié au véhicule est exprimé par :

$$\begin{cases} u = u_r + \dot{x}_C \cos(\psi) + \dot{y}_C \sin(\psi) \\ v = v_r - \dot{x}_C \sin(\psi) + \dot{y}_C \cos(\psi) \end{cases} \tag{B.26}$$

Dans le repère $< e >$ lié à la Terre, la cinématique du véhicule devient :

$$\begin{cases} \dot{x} = u_r \cos(\psi) - v_r \sin(\psi) + \dot{x}_C \\ \dot{y} = u_r \sin(\psi) + v_r \cos(\psi) + \dot{y}_C \\ \dot{\psi} = r \end{cases} \tag{B.27}$$

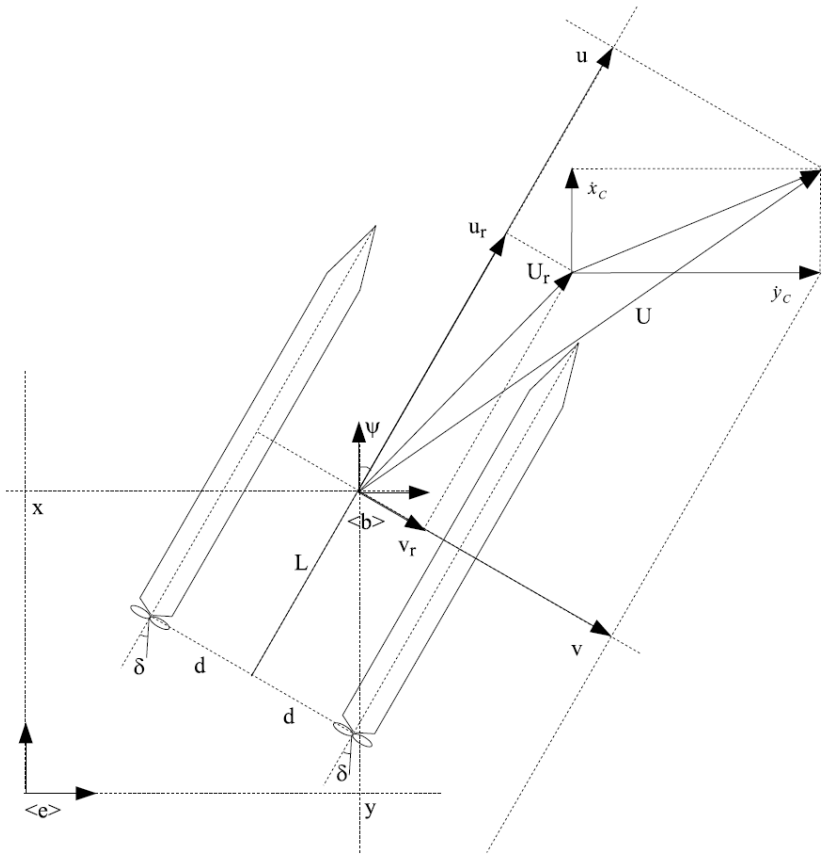


Fig. B.6 – Appellation des variables utilisées dans le modèle de Charlie

B.2.3 Équations de la dynamique

Un modèle théorique de l'hydrodynamie du véhicule a été explicité dans [68]. Ce modèle a été simplifié en se basant sur des hypothèses raisonnables tout en préservant la cohérence et la qualité des paramètres estimés. L'impossibilité de mesurer la vitesse de glissement par rapport au fluide et d'estimer les termes de couplage entre la vitesse d'avancement et le cap avec les capteurs disponibles, a conduit les auteurs a négliger ces termes. Les équations de la dynamiques sont donc réduites à :

$$\begin{cases} \tilde{m}_u \dot{u}_r = \tilde{k}_u u_r + \tilde{k}_{u_r^2} u_r^2 + \tilde{k}_{\bar{n}^2\delta^2} \bar{n}^2 \delta^2 + \bar{n}^2 \\ \tilde{I}_r \dot{r} = \tilde{k}_r r + \tilde{k}_{r|r|} r|r| + \tilde{k}_{\bar{n}^2} \bar{n}^2 + \bar{n}^2 \delta \end{cases} \tag{B.28}$$

avec δ l'angle du gouvernail, $\tilde{m}_u$ et $\tilde{I}_r$ les termes estimés d'inertie, $\tilde{k}_u, \tilde{k}_{u_r^2}, \tilde{k}_r, \tilde{k}_{r|r|}$ les termes de traînée.
n représente la vitesse de rotation du propulseur. Le $\bar{n}$ utilisé dans ces équations représente la vitesse de rotation normalisée, exprimée en volts. Pour ce faire, le véhicule est équipé d'un contrôleur-amplificateur assurant une relation de proportionnalité entre la tension appliquée au moteur et la vitesse de rotation du propulseur.
Le terme $\tilde{k}_{\bar{n}^2\delta^2}$ représente la résistance qu'oppose le gouvernail et $\tilde{k}_{\bar{n}^2}$ permet de prendre en compte les asymétries longitudinales du véhicule.

Annexe C

Propagation des ondes sonores

C.1 Propagation

Le champ de recherche des communications aquatiques sans fil a rapidement gagné de l'intérêt ces dernières années. D'un usage strictement militaire, elles ont envahi petit à petit le monde industriel. Ce type de communication a probablement vu le jour pendant la seconde guerre mondiale, pour des applications militaires. On citera par exemple un téléphone acoustique développé en 1945 permettant de communiquer avec des sous marins [84]. Avec l'apparition récente des DSP (Digital Signal Processor), il a été possible de faire du traitement du signal et donc de compresser les données à transmettre. Dans [85], des données télémétriques ont ainsi été transmises sur près de 200km. De nos jours, militaires et industriels souhaitent pouvoir communiquer avec des véhicules autonomes ou des capteurs en temps réel mis en réseau [74]. Les scénarios de communication dans lesquels les systèmes de communication modernes opèrent sont en fait des réseaux aquatiques partageant des données dont les nœuds peuvent être fixes ou mobiles (embarqués à bord de véhicules autonomes). Il sera alors possible à l 'utilisateur final d'accéder à ce réseau par un lien radio basé sur une station de surface.
Notre intérêt particulier pour le domaine acoustique nous a conduit à commencer par développer un modèle de propagation acoustique. L'objectif principal de ce modèle est donc de permettre de tester à un bas niveau l'implémentation de nouveaux protocoles liés à notre problématique. Une étude similaire devra être entreprise pour la propagation des ondes électromagnétiques.

C.1.1 Spectre des fréquences acoustiques

Comme nous l'avons évoqué, les systèmes de communications *UWA (UnderWater Acoustic)* sont fortement affectés par leur milieu de propagation. Cet affaiblissement dépend de la distance et de la fréquence de transmission du signal acoustique. Ainsi, il est nécessaire d'adapter cette fréquence en fonction de la distance de transmission visée : alors qu'il est possible d'utiliser une porteuse à quelques centaines de KHz sur une distance de quelques mètres, cette fréquence chute à quelques KHz, lorsqu'on l'on vise une distance

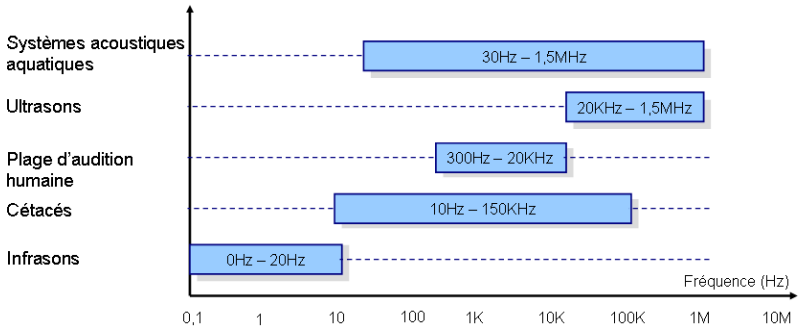


Fig. C.1 – Occupation du spectre acoustique

de quelques dizaines de Km. Dans [74], une organisation du spectre des fréquences acoustiques est proposée. Ces différentes bandes de fréquence sont représentées sur la figure C.1.

C.1.2 Qualification d'un canal acoustique

La propagation d'un son sous l'eau est caractérisé par l'atténuation, le bruit, la réverbération et la variabilité temporelle et spatiale (déplacements des utilisateurs, propriétés intrinsèques du canal UWA). L'atténuation et le bruit ambiant sont les principaux facteurs permettant de déterminer la bande passante disponible et le rapport signal sur bruit (*SNR : Sound to Noise Ration*). Les trajets multiples de l'onde influencent quant à eux la conception et le traitement des signaux, imposant fréquemment de sévères limitations aux performances du système.

C.1.2.1 Bande passante

La bande passante peut être définie comme étant la quantité d'informations que peut véhiculer un canal de communication, déterminée par la nature du canal en question, ainsi que les technologies de transmission mises en œuvre grâce aux équipements situés à chaque extrémité. Elle se mesure en bit par seconde. Dans le cas de système *UWA*, les ressources larges bandes sont grandement limitées. En effet, comme nous l'avons mentionné auparavant, la bande passante d'un canal aquatique dépend de la distance et de la gamme de fréquences employées. De ce fait, la largeur de bande passante exploitable pour les communications *UWA*, est limitée à environ 1MHz [86] sur une courte distance. Dans [87], l'auteur présente un tableau explicitant les bandes passantes utilisées par différents appareils de communication acoustique, ainsi que les débits atteints.
Cette limitation de la bande passante est due à un ensemble de phénomènes liés au milieu de propagation que nous allons détailler.

C.1.2.2 Multi-trajets

En plus de la bande passante très limitée, les systèmes de communications acoustiques doivent faire face à un phénomène appelé *Multi-trajets*. Ce phénomène est dû notamment aux réflexions du signal lors de sa propagation dans l'eau, aussi bien sur des obstacles que sur des "couches frontières" d'eau aux propriétés différentes (salinité, température). Un

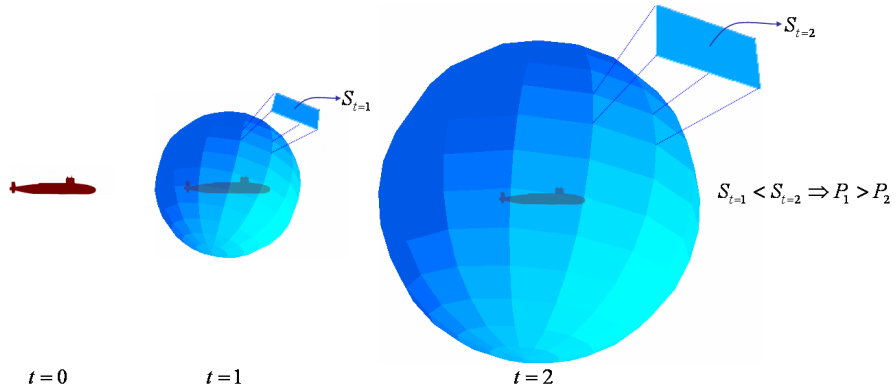


Fig. C.2 – Dispersion d'une onde acoustique : la pression acoustique diminue lorsque le rayon de la sphère de propagation augmente

deuxième phénomène favorise l'apparition de ces multi-trajets : il s'agit de la réfraction des rayons acoustiques due à la vitesse de propagation du son qui varie (suivant la pression, salinité, température), et dépendant de l'angle d'incidence des émissions.
La propagation multi-trajets du signal à travers un canal acoustique, dont les caractéristiques peuvent varier avec le temps, est également liée à la configuration du lien de communication : alors que les canaux verticaux sont peu propices aux multi-trajets, les canaux horizontaux peuvent y être très sensibles. La dégradation du signal due à ce phénomène se fait surtout sentir lors des communications horizontales à moyenne et longue portée. Elle se traduit par ce que l'on appelle des Interférences Entre Symboles (*IES* ou *ISI* en anglais pour InterSymbol Interference), qui correspondent en fait à un phénomène de recouvrement temporel, où la pulsation correspondant à chaque symbole transmis va recouvrir les symboles précédents. Limiter ce phénomène est sans nul doute considéré comme étant le défi le plus difficile à relever dans les systèmes de communication *UWA*. Diverses techniques basées sur la cohérence de phase sont à l'étude pour contrer ce phénomène. M. Stojanovic en donne un bon aperçu dans [88].

C.1.2.3 Atténuation - Distance

Quel que soit le milieu dans lequel se propage une onde acoustique, celle-ci subit une atténuation qui dépend principalement de la distance parcourue par l'onde et de sa fréquence. Cette atténuation est en fait due à 3 principaux phénomènes : l'absorption, qui traduit le fait qu'une partie de l'énergie sonore est convertie en chaleur le long du trajet parcouru, la diffusion (phénomène par lequel un faisceau est dévié dans de multiples directions), et enfin la dispersion qui représente l'étalement de l'onde acoustique lors de sa propagation.

- atténuation par *dispersion* : lorsqu'une onde sonore s'éloigne de son point d'émission, la surface de celle-ci augmente avec le carré de la distance au point d'émission. La pression acoustique par unité de surface tend donc à diminuer et ce, indépendamment de la fréquence utilisée. La figure C.2 illustre ce phénomène. L'étalement de l'onde acoustique peut être isotropique (c'est-à-dire que l'onde est rayonnée dans toutes les directions $\Leftrightarrow$ étalement sphérique), ou cylindrique (c'est-à-dire que l'onde est rayonnée suivant une direction privilégiée). L'expérience montre que la forme de

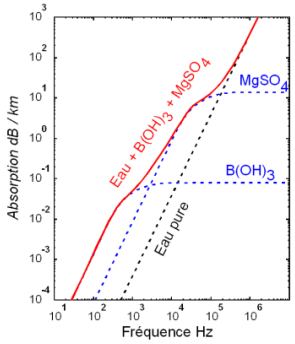


Fig. C.3 – Absorption de l'énergie acoustique à différentes fréquences dans de l'eau de mer pour une température et une pression données

l'étalement de l'onde se situe quelque part entre ces deux rayonnements : il s'agit de l'étalement pratique. Pour un étalement sphérique en milieu isotrope, une simple relation faisant intervenir 2 positions de distance R_1 et R_2 à la source ainsi que les puissances P_1 et P_2 associées nous est fourni par la relation C.1 :

$$\frac{P_2}{P_1} = \left(\frac{R_2}{R_1}\right)^2 \tag{C.1}$$

L'équation C.2, nous fournit quant à elle l'atténuation due à la dispersion (TL pour *Transmission Loss*) en dB en fonction de la forme de l'onde (sphérique, cylindrique, ...)

$$TL_{dispersion} = 10\log(d^{\gamma}) \tag{C.2}$$

avec
- d est la distance de propagation de l'onde acoustique
- γ est le facteur d'étalement d'énergie ($\gamma = 1$ pour un étalement cylindrique, $\gamma = 2$ pour un étalement sphérique)
- $TL_{dispersion}$ est l'atténuation par dispersion en dB.

– atténuation par *absorption* : elle est due à la conversion d'énergie acoustique en chaleur et énergie chimique ; elle dépend de la fréquence du signal. Dans [70], O. Le Calvé explique que cette absorption est principalement due à la viscosité de l'eau aux hautes fréquences, alors qu'aux basses fréquences, elle est due à la relaxation de l'acide borique $B(OH)_3$ et du sulfate de magnésium $MgSO_4$ (voir figure C.3. Le processus de relaxation consiste en l'absorption d'énergie par changement de structure moléculaire au passage de l'onde acoustique et conduit à la dissociation des molécules et hydratation des ions. L'absorption augmente donc avec la fréquence du signal acoustique et la distance de transmission. Une formule C.3 permettant d'approximer l'absorption des ondes acoustiques dans l'eau de mer peut être trouvée dans [89] :

$$TL_{absorption} = \left(\frac{0.11f^2}{1+f^2}\right) + \left(\frac{44f^2}{4100+f^2}\right) + 2.75\times10^{-4}f^2 + 0.003 \tag{C.3}$$

où $TL_{absorption}$ est le coefficient d'absorption en fonction de la fréquence f en dB/Km. Marsh et Schulkin ont établi dans [89] une formule empirique (C.4) permettant de

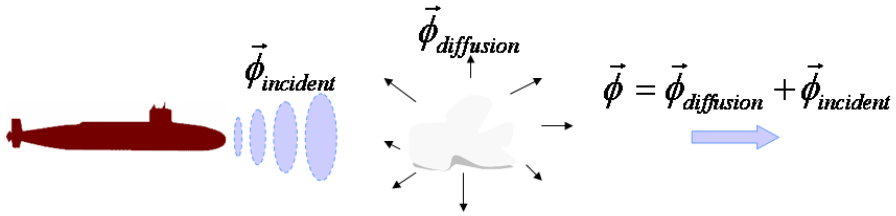


Fig. C.4 – Diffusion simple sur un obstacle

calculer ce même coefficient d'absorption faisant intervenir la salinité, la pression, et la fréquence de relaxation (en fonction de la température).

$$TL_{absorption} = 8.68 \times 10^3 \left(\frac{2.34.10^{-6}.S.f_T.f^2}{f_T^2 + f^2} + \frac{3.38.10^{-6}.f^2}{f_T} \right) \left(1 - 6.54.10^{-4}.P\right) \tag{C.4}$$

avec
- S, la salinité en $\%_0$
- P la pression hydrostatique en Kg/cm^2
- 3KHz$< f <$500KHz la fréquence en KHz
- $f_T = 21.9.10^{6-\frac{1520}{T+273}}$ la fréquence de relaxation en KHz où T est la température en $^\circ$C

– atténuation par diffusion : elle est due à la déviation et à la dispersion spatiale d'une partie de l'onde sonore incidente sur un obstacle. Varadan et Waterman traitent dans [90] et [91] de la diffusion des ondes longitudinales et transversales sur tout le domaine fréquentiel pour une grande variété d'obstacles de différents types, natures et géométries). Le phénomène de diffusion étant négligeable par rapport à celui de l'absorption nous négligerons cet aspect de l'atténuation. Le principe de la diffusion est repris sur la figure C.4.

C.1.2.4 Bruit

Le bruit est omniprésent dans l'océan ; qu'il soit d'origine naturelle (hydrodynamique, sismologique, géologique...), animale (cétacé notamment), ou humaine (sous marins, navires...), ce bruit exhibe toute une gamme de fréquences qui sont fortement liées à la fréquentation du site.
En effet, l'environnement côtier est très souvent perturbé par les diverses activités humaines. Le niveau de bruit y est donc beaucoup plus élevé qu'au fond des océans par exemple. Les navires représentent la composante principale du bruit acoustique ambiant dans les fréquences allant de 50Hz à 300Hz [92]. Ce bruit est généré par les machines, les ventilateurs, le déferlement des vagues sur sa coque ou bien encore la rotation de l'hélice qui est de loin le bruit le plus important. D. Ross a établi une modélisation du spectre de bruit des navires, d'après des mesures acoustiques effectuées durant la seconde guerre mondiale [93]. Ce spectre est définit en fonction de la vitesse du navire et de sa longueur. Sur le diagramme de directivité (figure C.5), on peut voir le bruit dû à la cavitation de l'hélice.
Les cétacés quant à eux produisent des sons dans une bande de fréquence se situant entre

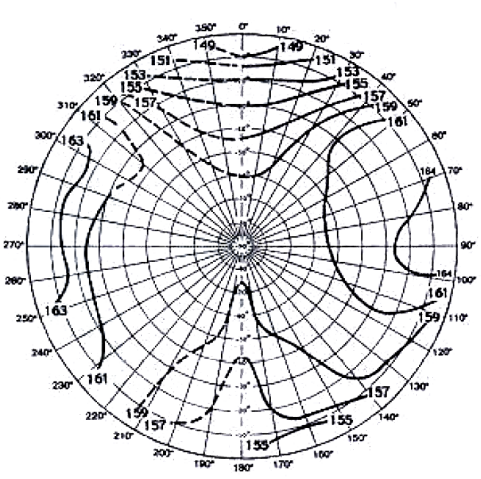


Fig. C.5 – Diagramme de rayonnement d'un navire pour la bande 340-360Hz

Animal	Son produit	Bande de fréquence (Hz)	Niveau sonore (dB réf 1µPa à 1m)
Baleine à bosse	Chant	120 – 4000	145 – 190
Baleine Rorqual Commun	Mugissement	30 – 750	155 – 165
Grand Dauphin	Sifflement	3500 – 14500	125 – 173
	Clics	110KHz – 130 KHz	218 – 228
Cachalot	Clics	2KHz – 16 KHz	160 – 210
Globicéphale	Clics	2000 – 14000	180

Fig. C.6 – Fréquences utilisées par différents cétacés

10Hz et 150KHz. Ces sons, souvent modulés en fréquence, peuvent être de différentes natures (clics, bourdons, sifflements, cris...) suivant que l'animal cherche à se localiser ou à communiquer. Dans [94], D.Gaucher propose un tableau élaboré d'après [95] qui reprend les fréquences acoustiques utilisées par différentes espèces (voir tableau C.6).

Il existe également un bruit de surface lié à l'activité du vent. En effet, le vent affecte l'état de la mer, et si il souffle suffisamment, des vagues peuvent se former. Leur déferlement emprisonne des bulles d'air qui gênèrent un bruit uniforme lorsqu'elle éclatent.

Pour conclure, le bruit océanique est principalement constitué du bruit émis par l'activité humaine, l'agitation de surface créée par le vent, la pluie, les animaux et enfin le bruit thermique. Le niveau de bruit diminue globalement avec l'augmentation de la fréquence. Wenz [96] propose un modèle (équation C.5) qui nous permet d'approximer le niveau moyen de chaque composante du bruit. Nous reprenons sur la figure C.7, la plage de fréquence occupée par les différentes sources du modèle de Wenz ainsi que l'intensité acoustique typique.

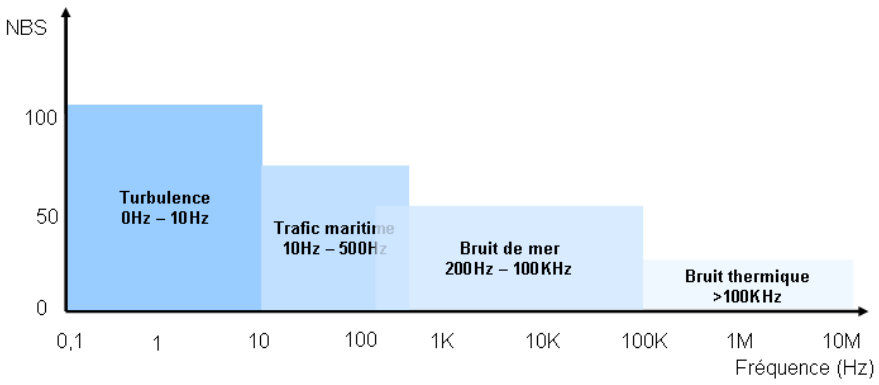


Fig. C.7 – Niveau de bruit isotrope provenant de différentes sources

$$\begin{cases} f < 10Hz : NBS_{Turbulence} = 107 - 30\log(f) \\ 10Hz < f < 500Hz : NBS_{Trafic} = 76 - 20\left(\log\left(\frac{f}{30}\right)\right)^2 + 5(I_s - 4) \\ 200Hz < f < 1KHz : NBS_{Bruitdemerlow} = 44 + \sqrt{21\nu_{kt}} + 17(3 - \log f)(\log f - 2) \\ 1KHz < f < 100KHz : NBS_{Bruitdemerhigh} = 95 + \sqrt{21\nu_{kt}} - 17\log f \\ f > 100KHz : NBS_{Bruitthermique} = -75 + 20\log f \end{cases} \tag{C.5}$$

où

- f est la fréquence centrale de la bande considérée
- I_s est un indice du traffic (1 : faible 7 : fort)
- ν_{kt} la vitesse du vent en nœuds.

Il est à noter que le niveau du bruit NBS donné ici est le niveau de bruit isotrope spectral exprimé en dB/μP/$\sqrt{Hz}$. En effet, la puissance du bruit est exprimée comme une densité énergétique, donc pour des bande de fréquences de largeur 1Hz. Par conséquent le niveau de bruit NB (exprimé en dB) doit être rapporté à la largeur de bande W (exprimé en Hz) du système récepteur utilisé :

$$NB = NBS + 10log(W) \tag{C.6}$$

En outre, la relation de Shannon C.7 nous fournit le débit maximal D possible d'un canal UWA de largeur B en fonction de la puissance du signal P_s et de la puissance du bruit P_b.

$$D = B\log\left(1 + \frac{P_s}{P_b}\right) \tag{C.7}$$

C.1.2.5 Vitesse de propagation

La vitesse du son dans l'océan est une variable importante puisqu'elle entre en jeu dans nombres d'équations décrivant les phénomènes de propagation aquatiques.
Cette vitesse varie en fonction des saisons, c'est-à-dire en fonction de la température de l'eau. Elle dépend également de la salinité et de la profondeur. Même si ces variations sont assez faibles (+/- 3%) et restent bornées entre $1450m.s^{-1}$ et $1540m.s^{-1}$, elle entraînent

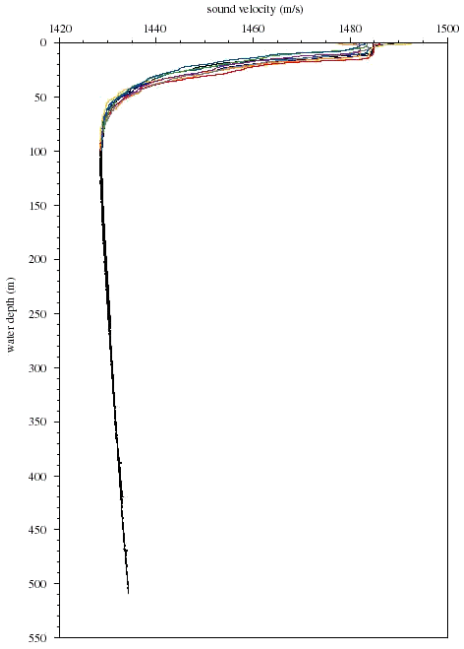


Fig. C.8 – Profil de la vitesse du con collecté pendant une campagne de mesure sur le lac Tahoe (Crédits : U.S. Geological Survey)

toutefois une dégradation des performances dans les systèmes où la connaissance de cette variable est primordiale (comme par exemple les systèmes SONAR). De plus cette vitesse reste assez faible comparée à celle des ondes radio en espace libre aérien (facteur de 1/200000) ; il est donc nécessaire de tenir compte du retard engendré pour certaines applications comme pour faire de la commande de coordination de flottille, lorsque les AUVs sont éloignés les uns des autres, ou plus généralement pour les systèmes ayant une dynamique élevée et une large occupation spatiale.
Il existe principalement deux méthodes permettant de déterminer la vitesse des ondes acoustique en milieu aquatique.

- La première est une mesure directe réalisée à l'aide d'un appareil appelé velocimètre. Ce matériel est plongé depuis la surface et mesure la variation de la vitesse du son dans toute la colonne d'eau. La figure C.8 illustre les relevés obtenus grâce à ce genre d'équipement. Si la vitesse de propagation du son dans l'eau varie peu dans le cadre des communications horizontales (peu de variations de pression, température, salinité), il n'en est pas de même pour les communications se propageant sur une colonne d'eau verticale. En effet, ce profil se divise en 3 couches distinctes successives que l'on représente sur la figure C.9 :
  - *Couche mixte* : la profondeur de cette couche relativement homogène varie en fonction du lieu, de la saison, et des conditions météo qui sont autant de critères responsables de la variation de température en son sein. A l'intérieur de cette zone, la célérité des ondes acoustiques augmente lentement avec la profondeur (due à l'augmentation de la pression)
  - *Couche thermocline* : il s'agit d'une couche de transition entre la couche mixte et couche profonde. Dans cette couche la température décroît rapidement avec la profondeur, entraînant la chute de la vitesse de propagation qui n'est pas com-

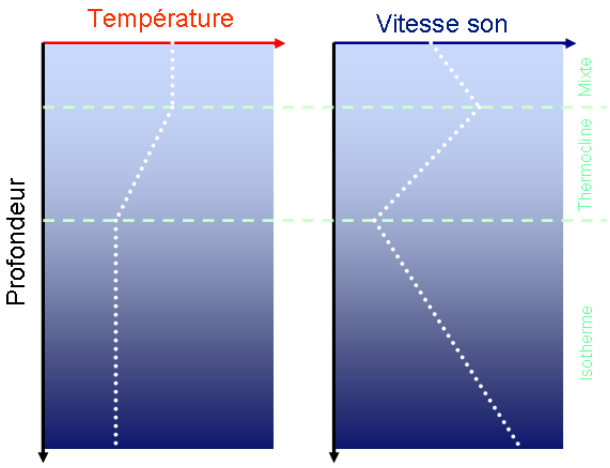


Fig. C.9 – Modèle de variation de la vitesse du son dans l'océan en fonction de l'évolution de la température et de la pression

pensée par l'augmentation de la pression.
- *Couche Isotherme* : Dans les plus basses couches de l'océan, la température ne varie pratiquement plus. Seule l'augmentation de la pression fait augmenter la vitesse de propagation du son.

– La seconde, fait appel à un calcul empirique donné par l'équation (C.8), exposée dans [97] permet d'obtenir une valeur approchée pour c :

$$c = 1449.2 + 4.6T - 5.5\times10^{-2}T^2 + 2.9\times10^{-4}T^3 + (1.34 - 10^{-2}T)(S-35) + 1.6\times10^{-2}D \quad \text{(C.8)}$$

où
- c est la vitesse de propagation du son exprimée en $m.s^{-1}$
- D est la profondeur en m avec $0 \leq D \leq 1000$
- S est la salinité en $‰$avec $0^{\circ}C \leq S \leq 45^{\circ}C$
- T est la température en $^{\circ}C$ avec $0 \leq T \leq 35$

Cette formule reste valable dans les limites données ci-dessus. Il existe néanmoins d'autres formules ([98] et plus récemment [99]) permettant d'approximer la vitesse du son hors de ces plages.

C.1.2.6 Latence de propagation

Le phénomène de latence est directement lié à la faible célérité des ondes acoustiques. En effet, on appelle latence le temps que va mettre le signal pour traverser un canal acoustique reliant l'émetteur et le récepteur. Ainsi une onde acoustique mettra au minimum 0.67s pour parcourir une distance d'un kilomètre, là où une onde électromagnétique mettra à peine 3.3×10^{-5}s. De plus, il ne faut pas perdre de vue qu'une onde ne se propage pas en ligne droite dans le cadre des communications horizontales. Elle se propage plutôt par rebonds successifs sur les interfaces délimitant les milieux (surface/fond, thermocline...). Cela augmente donc considérablement la distance à parcourir par l'onde et donc le temps de latence. A l'inverse, les ondes acoustiques se propagent plutôt en ligne droite dans le plan vertical.

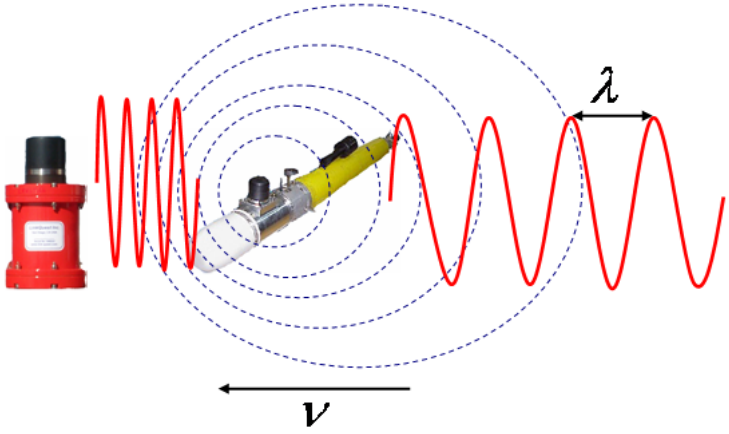


Fig. C.10 – Illustration de l'effet Doppler : la fréquence acoustique des ondes se propageant à l'avant de l'AUV est augmentée alors que celles se propageant à l'arrière subissent une diminution de fréquence

C.1.2.7 Effet Doppler

Un dernier effet peut être pris en compte dans le cadre des communications aquatiques. Lorsque l'émetteur et/ou le récepteur sont en mouvement l'un par rapport à l'autre, le signal reçu est décalé en fréquence : il s'agit de l'effet Doppler qui est fonction de la vitesse de déplacement de l'émetteur par rapport au récepteur et de la vitesse de propagation d'une onde acoustique dans l'eau (voir figure C.10). Cette variation de fréquence nous est donnée par l'équation (C.9) qui est une approximation pour les signaux en bande étroite. Si l'on a pour habitude de négliger cet effet lorsque nous utilisons les ondes radio, il n'en va pas de même pour les ondes acoustiques. En effet le rapport entre la vitesse d'un AUV par exemple et celle du son tombe à 10^{-3} environ. Ce rapport peut alors affecter les transmissions à haut débit.

$$f = f_0 \times \left(1 \pm \frac{\nu}{c} \cos\beta\right) \tag{C.9}$$

où

- f est la fréquence du signal à la réception
- f_0 est la fréquence du signal à l'émission
- ν vitesse relative de mouvement entre l'émetteur et le récepteur
- c la célérité du son dans l'eau
- β l'angle d'incidence du signal acoustique

C.1.3 Propagation des ondes acoustiques

La propagation du son dans un milieu élastique peut être décrite mathématiquement à l'aide de *l'équation du son*(C.10).

$$\triangle\phi - \frac{1}{c^2}\frac{\partial^2\phi}{\partial t^2} = 0 \tag{C.10}$$

où c représente la vitesse du son dans l'eau (pouvant varier dans l'espace), et où ϕ désigne n'importe quelle onde vibratoire. Plusieurs approches déterministes permettent d'appro-

cher une solution à cette équation.

- *Théorie des rayons*
  Cette théorie part du postulat qu'il existe une ligne de front le long de laquelle la phase de la solution est constante d'une part, et d'autre part qu'il existe des rayons qui décrivent l'endroit vers lequel le rayon émis à la source va être propagé. Comme en optique géométrique, l'utilisation des rayons acoustiques est assez intuitive mais présente certaines limitations inhérentes au modèle. En effet, le modèle manque rapidement de précision lorsque le rayon de courbure des rayons ou la pression change rapidement sur une distance d'une fois la longueur d'onde ; de ce fait, l'utilisation de cette théorie est plutôt réservée aux hautes fréquences. De plus, la non prise en compte des phénomènes de diffraction ne permet pas de déterminer l'intensité sonore dans les zones d'ombre acoustique.
- *Théorie des modes*
  Rechercher des solutions de l'équation du son sous forme de modes consiste à déterminer dans le faisceau de rayons issus de la source, ceux dont l'énergie va interférer de manière constructive après la réflexions à la surface et sur le fond. En effet, aux frontières du milieu, les rayons vont subir des déphasages dont il faut tenir compte : les énergies de deux rayons en opposition de phase s'annihilent.
  Les rayons interférant constructivement sont définis par leurs sites initiaux, et sont baptisés *modes*.
- *Approximation parabolique*
  Cette dernière approximation fait appel a une fonction solution de Hankel (solutions linéairement indépendantes de l'équation de Bessel). Il s'agit d'une fonction radiale dont le développement asymptotique tend vers une fonction décroissante, typique d'une propagation en onde cylindrique. Cela débouche sur une équation dite *parabolique* qui est résolue en connaissant la valeur de l'intensité sonore à une distance donnée de la source, qui constitue une position spatiale d'initialisation des calculs. Les valeurs, solutions de l'équation, s'obtiennent alors grâce à l'équation de propagation, par itérations successives.

Chacune de ces théories est bien entendu valable sur un domaine bien particulier. En effet, alors que la *théorie des rayons* s'applique pour le calcul de la trajectoire d'un rayon haute fréquence se propageant dans une couche stratifiée à gradient de célérité constant et obéissant à la loi de Snell (équation C.11), la *théorie des modes* sera plutôt employée dans un milieu petit fond et stratifié pour prendre en compte l'influence des frontières ; enfin l'approximation parabolique est préférentiellement utilisée pour les milieux stratifiés ou variables et reste valable jusqu'à des fréquences relativement élevées. Les différentes plages de mise en œuvre de ces théories est résumées sur le figure C.11. Un des résultats importants résultant de la théorie des rayons, est la loi de Snell qui décrit la réfraction des rayons sonores dans un milieu dans lequel la vitesse du son est variable. Comme nous l'avons vu précédemment, cette variation de célérité est due principalement aux variations de pression, température, salinité. Par conséquent, la propagation des ondes acoustiques dans l'eau sont soumises à de multiples réflexions et réfractions au fur et à mesure que les caractéristiques du milieu varient au cours de leur propagation [89]. La loi de Snell énonce donc que dans un milieu constitué de couches à vitesse du son constante, l'angle entre le rayon réfléchi et le rayon transmis est lié à la vitesse du son propre à ces couches.

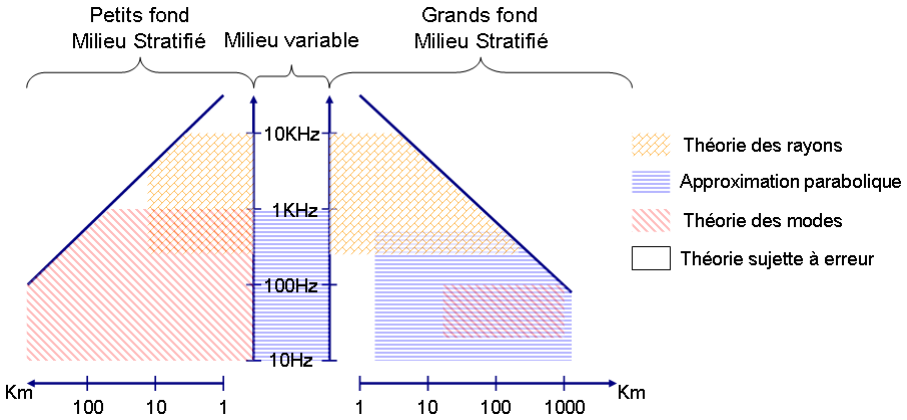


Fig. C.11 – domaines d'utilisation des différentes théories

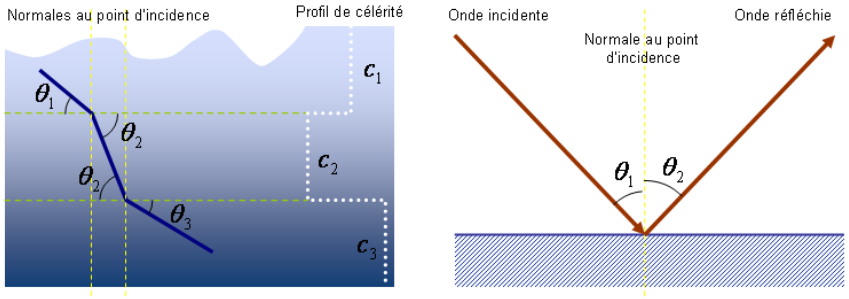


Fig. C.12 – Réfraction et reflexion dans un milieu stratifié

La figure C.12 illustre cette relation donnée par l'équation C.11.

$$\begin{cases} \dfrac{\cos\theta_1}{c_1} = \dfrac{\cos\theta_2}{c_2} = \dfrac{\cos\theta_3}{c_3} = \ldots = constante \\ \theta_{incident} = \theta_{reflechi} \end{cases} \tag{C.11}$$

Les ondes incidentes à la surface sont presque intégralement réfléchies (grande différence d'indice entre les 2 milieux) ; on parle alors de miroir acoustique. Ce phénomène peut également se rencontrer au niveau des interfaces séparant 2 couches adjacentes. En effet, sous certaines incidence, l'onde émise peut être totalement réfléchie pour peu que la différence de célérité du son entre ces 2 couches soit importante. Le rayon acoustique suit alors une trajectoire qui dépend du profil de vitesse de propagation, de la profondeur d'immersion de la source acoustique et enfin de l'angle d'émission, créant ainsi des zones d'ombre acoustique qui ne sont pas insonifiées.
Pour conclure, ces variations de vitesse des rayons sonores peuvent créer :

- des zones non insonifiées : les zones d'ombre où les sous marins ont la possibilité de se cacher
- des zones canalisant le son : ce sont les chenaux acoustiques qu'ils soient de surface ou profonds, sont repérables par les minima de célérité des profils bathycélérimétriques (vers la surface et vers la zone des 400/600m) ; ce sont des canaux naturels de

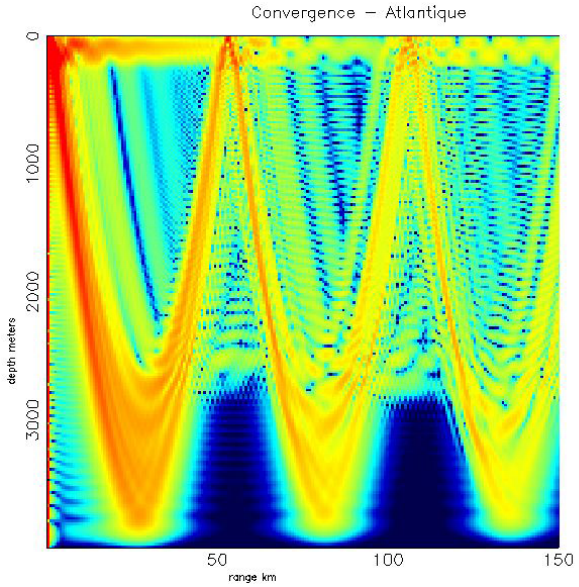


Fig. C.13 – Qualification du canal de propagation - méthode des modes

communication sous-marine
- des zones de convergence : les rayons vont se concentrer, après avoir parcouru 40 à 60 Km dans des chenaux de convergence beaucoup moins rectilignes que les chenaux classiques (de surface ou profonds).

Tout ceci est illustré sur le relevé de propagation acoustique effectué par le sonar d'un bâtiment de surface (C.13) : on peut voir l'effet du chenal de surface, la convergence à 60 puis 120 Km et enfin en bleu, les zones d'ombre acoustique.

C.2 Ondes sonores

C.2.1 Production de sons

Différents technologies peuvent être utilisées pour produire des sons à l'aide de transducteurs :

- La *magnétostriction*
  Considérons un barreau d'une substance ferromagnétique quelconque (fer, nickel, ...) de longueur l. Soumettons ce barreau à un champ magnétique parallèlement à son axe ; si l'on applique une force à ses extrémités, créant une variation de longueur δl, on obtient une variation de l'induction magnétique B fonction de la nature du matériau et de la force appliquée. C'est l'effet magnétostrictif direct.
  Ce phénomène est réversible : si on applique un champ magnétique à ce même barreau, sa longueur variera d'une longueur δl. L'amplitude de cette variation dépendra alors de l'intensité du champ magnétique et le sens de la nature du matériau. C'est l'effet magnétostrictif inverse, utilisé pour la production de sons (voir figure C.14)
  Les matériaux les plus utilisés sont le nickel et ses alliages tel que l'Invar (64% Fe + 36% Ni), le Permalloy (54,7% Fe + 45% Ni + 0,3% Ni) le Permendur (49,7% Fe + 50% Co) ainsi que certaines ferrites.

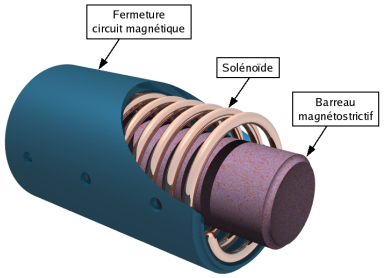


Fig. C.14 – Transducteur magnétostrictif - Crédits : Wikipédia

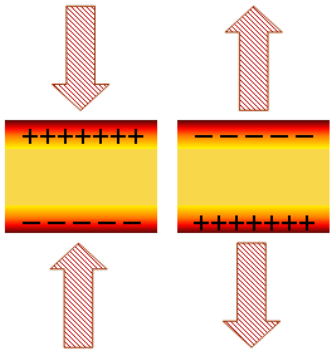


Fig. C.15 – Des charges apparaissent lorsqu'un matériau piézoélectrique est soumis à une action mécanique

– L'*électrostriction*
Cet effet est très général puisqu'il se caractérise de la façon suivante : si on applique un champ électrique à un milieu quelconque (liquides, solides, gazeux), il subira une déformation proportionnelle au carré du champ appliqué. En acoustique sous marine, ce sont surtout les matériaux dits *ferroélectriques* et *piézoélectriques* auxquels nous nous intéressons.
En effet, on qualifie de ferroélectrique tout corps cristallin doué d'une polarisation électrique spontanée ou rémanente. Ces corps ont un comportement tout à fait comparable à celui des ferromagnétiques en magnétostatique. Tous les corps ferroélectriques sont aussi des piézoélectriques, ce qui n'implique pas que l'inverse soit toujours vrai (ex : le quartz). Un cristal est dit *piézoélectrique* lorsqu'une contrainte appliquée dans une direction fait apparaître une polarisation électrique proportionnelle à la force (phénomène réciproque utilisé pour la production de sons). Cet effet est représenté sur la figure C.15. Plusieurs cristaux possèdent ce genre de propriété, au nombre desquels le quartz, mais aussi le sel de Seignette (un tartrate double de sodium et de potassium), le phosphate monoamonique etc...
Les céramiques ferroélectriques, quant à elles, ne peuvent être obtenues qu'en monocristaux et doivent être taillées par la suite. Elles présentent des performances supérieures à celles des cristaux piézoélectriques et sont à base de titane de baryum et de titanozirconate de plomb. Pour les céramiques à hautes performances, des terres rares comme le niobium ou le lanthane peuvent entrer dans la composition.

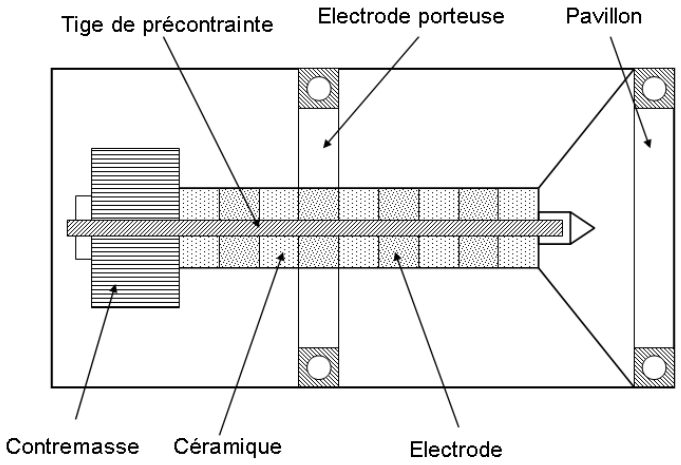


Fig. C.16 – Transducteur Tonpilz

Pour permettre de diffuser un son dans l'eau il est nécessaire d'avoir recours à un *projecteur*, qui peut être représenté sous une forme simple sur la figure C.16. Le projecteur est constitué d'un pavillon, lui même couplé à l'eau, et qui sert d'adaptateur d'impédance entre les céramiques et le milieu. Bien entendu, ses dimensions et le matériau utilisé pour la construction sont fonction de l'utilisation qu'on lui réserve (bande passante souhaitée, fréquence d'utilisation...). On trouve à l'intérieur de ce projecteur, une série de céramiques ferroélectriques mises en parallèle par des électrodes qui amènent le courant électrique. La masse arrière quant à elle sert surtout à régler la largeur de bande du projecteur et réduire sa longueur. Enfin, la tige de précontrainte assure la rigidité de l'ensemble en les maintenant en compression. Une application demandant une forte puissance d'émission dans une direction privilégiée, fera appel à un groupement de ces éléments. On obtient alors une *antenne*. Il existe toute une gamme de projecteurs mettant en œuvre différentes technologies que nous n'exposerons pas ici.

C.2.2 Réception de sons

Recevoir un son consiste à capter le signal acoustique dans le milieu de propagation. Comme nous l'avons évoqué précédemment, cette opération se fait en ayant recours à un *hydrophone*. Un hydrophone peut être considéré comme un projecteur fonctionnant à l'inverse. La conception mécanique est donc peu différente de celle des projecteurs ; la principale différence réside dans le fait qu'on recueille une tension électrique que l'on doit amplifier et mesurer, même si elle est très faible. C'est donc la sensibilité de l'hydrophone qui va jouer un rôle prépondérant dans la réception des signaux acoustiques. D'autre part, un hydrophone doit être en mesure de capter un large gamme de fréquences et donc d'avoir une réponse plate dans cette gamme. La construction d'un hydrophone requiert donc de rejeter la fréquence de résonance vers les hautes fréquences, ce qui explique le faible volume de ces appareils. Ici encore, il existe toute une gamme d'hydrophone, utilisant des cristaux, des parties sensibles, des joints de couplage... différents. Sur la figure C.17, on peut voir un hydrophone de la marque *Reson*.

Fig. C.17 – Hydrophone TC4013 de la marque Reson

C.2.3 Directivité des antennes

Comme nous l'avons vu précédemment, les antennes sont constituées de plusieurs projecteurs et/ou hydrophones suivant qu'elles ont pour mission d'émettre ou recevoir des son dans l'eau. Un groupement d'hydrophones présente 2 avantages principaux. Tout d'abord, l'utilisation de plusieurs éléments augmente la sensibilité de l'appareil puisqu'ils produiront une tension plus élevée à la réception d'un signal (mise en série des éléments). D'autre part, des propriétés directionnelles, permettant de déterminer la direction d'une onde acoustique incidente, émergent. Cela permet donc d'améliorer le rapport signal sur bruit car ils sont plus sensibles au signal dans la direction vers laquelle ils sont pointés qu'au bruit isotrope ambiant.
Les hydrophones peuvent être disposés d'un grand nombre de façons différentes (en ligne, en carré, en sphère, en cylindre, épouser la forme de la coque d'un bateau ...), permettant ainsi une grande diversité des caractéristiques de l'antenne suivant l'application visée. L'utilisation de toutes ces dispositions et types d'hydrophones ont pour corollaire d'obtenir des diagrammes de directivité très différents. En effet, ces diagrammes traduisent les variations spatiales du niveau sonore émis par un projecteur ou la variation de sensibilité d'un récepteur. Plusieurs paramètres caractérisant la directivité sont représentés sur la figure C.18.

- Les *lobes* : il s'agit de la portion du diagramme de directivité entre deux minima successifs
- Les *trous* : ce sont les minima prononcés de la courbe de directivité
- Le *lobe principal* : c'est le lobe dont le maximum est supérieur aux autres lobes dits *secondaires*. il peut y avoir plusieurs lobes principaux
- Les *premiers secondaires* : ce sont ceux dont le maximum est immédiatement inférieur à celui du lobe principal.
- *Largeur du lobe principal* : c'est l'écart angulaire mesuré entre 2 directions pour lesquelles l'amplitude du lobe principal est inférieur à 3dB.

Plus les dimensions de l'antenne sont grandes devant la longueur d'onde, plus nombreux seront les lobes secondaires et plus étroit sera le lobe principal.

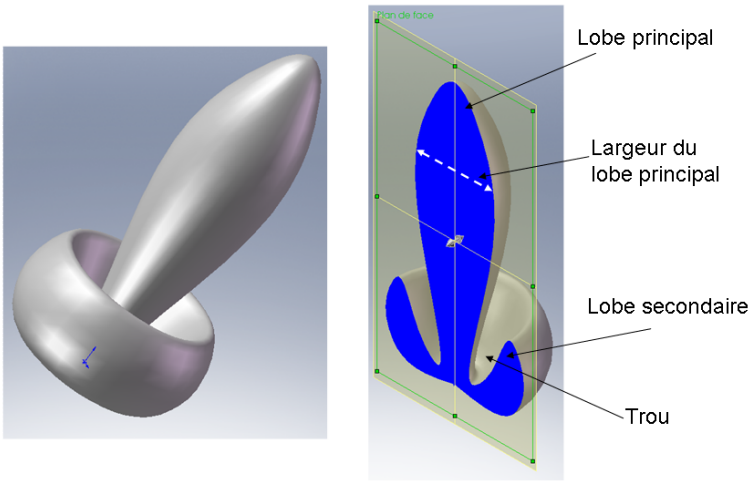


Fig. C.18 – Représentation schématique du diagramme de rayonnement d'une antenne acoustique

Thèse de Olivier Parodi

Simulation hybride pour la coordination de véhicules hétérogènes au sein d'une flottille

Si l'utilisation d'un véhicule autonome présente un intérêt certain, l'utilisation simultanée de plusieurs robots, potentiellement différents permet d'améliorer la qualité des acquisitions et la rapidité avec lesquelles elles sont effectuées. Le déploiement de plate-formes multi-capteurs sous-entend donc qu'il est nécessaire de mettre en œuvre une commande et une stratégie de commande coordonnée de la formation. Les difficultés de localisation et de communication rendent d'emblée la tâche difficile. Dès lors, il n'est plus concevable de mettre en œuvre une flottille d'engins autonomes sans avoir testé au préalable la faisabilité de la mission, le comportement logico-temporel du contrôleur des engins, l'efficacité des algorithmes employés et le bon fonctionnement de tous les sous-systèmes. Ceci est d'autant plus vrai lorsque les engins sont hétérogènes, proviennent d'organisations ou d'institutions différentes et sont rassemblés pour l'accomplissement d'une mission commune.

La complexité des architectures de contrôle d'une part et les difficultés soulevées par le choix de stratégies de contrôle multi-véhicules d'autre part, rendent nécessaires la création de nouveaux outils de simulation permettant de tester et valider lois de commande et architectures de contrôle tout en détectant les inconsistences préliminaires des scenarios envisagés. L'objet de cette thèse est donc l'étude d'un outil de simulation collaboratif appelé THETIS.

Il s'agit d'un simulateur conçu avant tout pour aborder les problèmes liés au contexte de la flottille. Il est multi-véhicules hétérogènes puisqu'il permet de simuler par exemple, un scenario dans lequel un AUV (Autonomous Underwater Vehicle) et un ASV (Autonomous Surface Vehicle) interviennent simultanément. Les véhicules peuvent communiquer entre eux au sein de la simulation et les contraintes liées au milieu de propagation (interférences, bande passante, atténuation...) d'une part et à l'utilisation de matériel spécifique (temps de réveil, conflit émission/réception...) d'autre part sont prises en compte. L'architecture du simulateur est ouverte pour faciliter l'intégration et la mise à disposition pour tous, du travail de modélisation des différentes équipes possédant des compétences propres, tout en favorisant la réutilisabilité et la modularité de ces modèles. La capacité du système proposé à réaliser des simulations Hardware-In-The-Loop permet de tester et valider le comportement temporel du contrôleur. Par ailleurs ce simulateur est distribué afin de pouvoir étendre dynamiquement la puissance de calcul nécessitée par l'augmentation du nombre de véhicules et/ou la complexification des modèles, tout en respectant les contraintes temps-réel et le découplage temporel entre la commande et l'évolution des modèles dynamiques.

THETIS est donc un des seuls outils à l'heure actuelle répondant aux contraintes liées au contexte de la simulation de robots marins en flottille. Nous présentons des tests préliminaires mettant en œuvre un AUV de classe Taipan (développée au LIRMM en France) d'une part et un ASV Charlie (développé par l'ISSIA en Italie) d'autre part qui possèdent des architectures de contrôle différentes, et démontrons ainsi la faisabilité et la validité de notre approche.

Mots clés : robotique sous-marine, multi-véhicule coordination, AUV, ASV, architecture de simulateur Hadrware-in-The-Loop, simulation de communications.

Hybrid simulation for the coordination of heterogeneous vehicles within a flotilla

If using a vehicle presents a demonstrated interest, the use of multiple robots potentially heterogeneous should improve the samples quality and the velocity with witch they are acquired. The deployment of multi-sensor platforms makes the use of a flotilla coordinate control strategy necessary. The positioning and communicating difficulties immediately make the task fussy. Therefore, it is no more conceivable to deploy a flotilla without having first tested the feasibility of the mission, the logical and temporal behavior of the robot's controller, the employed algorithms' effectiveness and the functioning of all subsystems. This is especially true when the vehicles are heterogeneous, come from different organizations or institutions and are assembled aiming the fulfillment of a common mission.

The complexity of control architectures on the one hand and the difficulties raised by the choice of control strategies for multi-vehicle scenarios in the other hand, require the creation of new simulation tools in order to test and validate control laws and control architecture while detecting preliminary inconsistencies within the scenarios. The purpose of this thesis is the study of a collaborative simulation tool called Thetis.

This is a simulator uppermost designed to address issues related to the context of robots cooperation. It is able to cope with heterogeneous multi-vehicle sceanrios within the robots can communicate with each other taking into account the propagation constraints (interferences, bandwidth, attenuation...) and the natural behavior of used communication devices (wake up time, conflict between reception/emission...). The architecture of this simulator is open to facilitate the integration and dissemination for all of the modeling work while promoting reusability and modularity of these models. The capacity of the proposed system to deal with Hardware-in-The-Loop simulations allows to test and validate the logical and temporal behavior of the controllers. This simulator is also distributed on several calculating units linked on a dedicated network, to be able to extend dynamically the computing power. This feature is required to increasing the number of vehicles and / or the complexity of the models while respecting real-time constraint and temporal decoupling between the controller and the simulator.

Thetis is currently one of the few available tools able to cope with the constraints related to the context of marine robots in flotilla. Preliminary tests involving one AUV (Autonomous Underwater Vehicle) Taipan (developed at LIRMM in France) and one USV (Unmanned Surface Vehicle) Charlie (developed at ISSIA in Italy) on which are implemented two different control architecture are presented, while demonstrating the feasibility and validity of our approach.

Keywords : underwater robotics, multi-vehicle coordination, AUV, ASV, Architecture of Hardware-in-The-Loop simulator, simulation of communications