Rapport sur le mémoire de thèse de

Pierre FILIOL

intitulé

« Hardware Acceleration of Interval Arithmetic in the RISC-V and Application to Mobile Robotics »

en vue de l'obtention du titre de

DOCTEUR de l'ENSTA

Spécialité : Robotique

par Christophe JEGO, professeur des Universités à Bordeaux INP

La thèse de Monsieur Pierre FILIOL s'insère dans le cadre d'une collaboration scientifique entre deux équipes de recherche du Laboratoire des Sciences et Techniques de l'Information, de la Communication et de la Connaissance (Lab-STICC, UMR6285) dont l'ENSTA est l'un des membres fondateurs. L'équipe de recherche ROBEX développe des outils académiques afin de concevoir des algorithmes intelligents permettant à des robots mobiles d'accomplir une mission d'exploration de façon autonome. L'équipe de recherche ARCAD (architectures matérielles et outils de CAO) se focalise, quant à elle, sur les méthodes et les outils pour la conception d'architectures matérielles. Les travaux de thèse portent sur l'utilisation de la méthode d'analyse par intervalles pour générer des estimations garanties d'ensembles de solutions dans des systèmes non linéaires ou des problèmes de minimisation. En effet, dans un contexte applicatif de robotique mobile, cette méthode est couramment employée afin de résoudre des problèmes de localisation, de planification de trajectoire ou de détection de collision. Cependant, l'implémentation de ce type de méthode sur des processeurs embarqués s'avère particulièrement complexe. Il faut notamment surmonter les limitations actuelles des bibliothèques logicielles dans la gestion des différents modes d'arrondi pour les nombres en virgule flottante. Ce travail a nécessité de la part du doctorant d'aborder conjointement deux domaines complémentaires que sont l'arithmétique par intervalles et l'implémentation et le prototypage sur circuit FPGA d'instructions spécifiques pour un processeur à jeu d'instructions.

Le manuscrit de thèse soumis se compose de sept chapitres principaux complétés par une introduction et une conclusion. Il est organisé en deux parties distinctes. La première partie est une partie introductive qui correspond aux chapitres un à trois du manuscrit. Le premier chapitre présente les méthodes d'intervalles basées sur les ensembles telles qu'elles sont utilisées en robotique embarquée, à travers ses concepts mathématiques les plus pertinents tels que l'algèbre des contracteurs, la propagation avant-arrière et l'estimation de l'ensemble des solutions. Puis, le deuxième chapitre se focalise sur les stratégies pertinentes d'accélération matérielle pour l'arithmétique par intervalles. Enfin, une introduction au standard RISC-V,

qui est l'architecture de jeu d'instructions utilisée pour intégrer les opérateurs matériels d'arithmétique par intervalles est proposée dans le troisième chapitre. Quant à la seconde partie, qui constitue le cœur de ce mémoire, elle comprend quatre chapitres. Le quatrième chapitre contient la description d'une méthode de raffinement en arithmétique par intervalles permettant de générer avec précision le complément d'un contracteur impliquant des opérateurs ou fonctions réelles partiellement définis. Les instructions fournies par l'extension personnalisée *xinterval* et leur intégration dans une architecture de processeur RISC-V sont décrites dans le cinquième chapitre. Le sixième chapitre détaille la mise en œuvre des opérateurs matériels associés aux instructions de l'extension *xinterval* et présente les performances obtenues lors de leur implémentation sur circuit FPGA. Enfin, une méthodologie *hardware-in-the-loop* utilisée pour concevoir, valider et estimer les performances de l'extension *xinterval* est présentée dans le septième chapitre.

Après un chapitre introductif, le premier chapitre présente l'arithmétique par intervalles comme une méthode numérique permettant de calculer des encadrements garantis de solutions à des problèmes d'ingénierie non linéaires, particulièrement adaptée à la robotique mobile. Dans un premier temps, les fondements théoriques de l'arithmétique par intervalles sont détaillés de manière pédagogique. Ainsi, les intervalles et leurs opérations sont d'abord définis. Puis les vecteurs d'intervalles sont présentés, avant que les contracteurs ne soient introduits via la méthodologie *forward-backward* et l'algorithme HC4R. Enfin, le principe de représentation d'un ensemble de solutions par une collection de sous-intervalles disjoints ou *subpaving* et son obtention à l'aide de l'algorithme SIVIA sont exposés. Puis, une illustration pertinente et concrète de ces concepts à travers un problème de localisation robotique par trilatération est proposée. L'objectif est de démontrer la pertinence d'une méthode par intervalles dans le contexte de la robotique mobile. Dans une dernière section, un état de l'art des solutions logicielles existantes pour l'implémentation de l'arithmétique par intervalles est proposé. Pour ce faire, un panorama du standard IEEE-1788 (1.4.1) et de l'écosystème logiciel existant sont fournis. Ils permettent d'identifier les deux limitations majeures des approches logicielles, à savoir la gestion des différents modes d'arrondi pour les nombres en virgule flottante et l'usage des bibliothèques logicielles existantes. Elles justifient le besoin d'une approche alternative basée sur des accélérations matérielles qui est l'objet des travaux de thèse de Monsieur Pierre FILIOL.

Le deuxième chapitre récapitule les stratégies d'accélération matérielle applicables à l'arithmétique par intervalles, en distinguant deux niveaux d'approche complémentaires. La première, dite accélération au niveau du graphe, exploite le parallélisme inhérent aux algorithmes de type *Branch & Bound* en distribuant l'évaluation des différents nœuds sur des architectures massivement parallèles telles que des GPU. Ainsi, plusieurs travaux de la littérature rapportent pour cette approche des gains de performance allant jusqu'à trois ordres de grandeur par rapport à une implémentation CPU séquentielle. La seconde approche, dite accélération au niveau du contracteur, vise à optimiser l'exécution de l'opérateur d'intervalle lui-même, soit par l'utilisation d'instructions SIMD disponibles sur les processeurs généralistes, soit par la conception d'accélérateurs matériels dédiés ciblant un circuit FPGA. Il s'avère que l'accélération matérielle à l'aide de circuit FPGA est particulièrement efficace en raison de ses capacités de reconfiguration et d'exécuter des traitements parallèles. Par ailleurs, la maîtrise précise que cette approche offre sur les modes d'arrondi conformes à la norme

IEEE-754 est un atout indéniable pour la validité des calculs par intervalles. Un défi à relever concerne cependant la définition des opérateurs en virgule flottante qui jouent un rôle crucial dans l'arithmétique par intervalles. Pour ce faire, le projet FloPoCo est identifié par Monsieur Pierre FILIOL pour faciliter la génération d'opérateurs en virgule flottante correctement arrondis et paramétrables au niveau de la précision. L'approche architecturale retenue est d'implémenter les opérateurs d'arithmétique par intervalles directement sur circuit FPGA. Mais il est nécessaire de pouvoir exploiter ces opérateurs lors de développement logiciel. Pour y parvenir, il est possible d'ajouter de nouvelles instructions à un jeu d'instructions ouvert comme celui du standard RISC-V. Cette conclusion sert de transition naturelle vers le troisième chapitre, qui traite justement du standard RISC-V et de son mécanisme d'extensibilité.

Dans un premier temps, le troisième chapitre rappelle le contexte historique ayant conduit à l'émergence des architectures RISC comme alternative aux architectures de processeurs CISC. Puis, les limitations structurelles des jeux d'instructions commerciaux, qu'ils soient de nature CISC ou RISC, sont analysées. Trois problèmes majeurs sont identifiés : l'accumulation architecturale liée à la compatibilité ascendante, les contraintes de gouvernance et de licences et enfin l'extensibilité très limitée des jeux d'instructions. Face à ces limitations, les principes fondateurs du standard RISC-V sont présentés. L'architecture du standard est détaillée dans un second temps. L'ensemble réduit des instructions de base est rappelé ainsi que le principe entièrement optionnel d'extensions modulaires identifiées par des lettres standardisées. Enfin, le mécanisme d'extension custom prévu dans le standard RISC-V est précisé. Dans le cadre de cette thèse, l'extensibilité naturelle de processeur RISC-V sera mise à profit pour incorporer une extension personnalisée dédiée aux opérateurs d'intervalles, afin d'exécuter des applications orientées robotique. A ce stade du document, nous pouvons souligner la pertinence de la synthèse proposée par Monsieur Pierre FILIOL pour justifier les choix effectués lors du positionnement des travaux de thèse.

Une première contribution, à savoir une méthode rigoureuse pour construire des contracteurs associés au complément d'un ensemble dans le cas où les fonctions impliquées dans la contrainte ne sont que partiellement définies est l'objet du quatrième chapitre. En effet, il s'agit d'une limitation des solveurs d'intervalles existants qui produisent des approximations intérieures incorrectes. Pour remédier à cette limitation, une nouvelle arithmétique dite arithmétique réelle totale est proposée. Elle repose sur l'ajout d'un élément particulier dans le cas où des fonctions partielles sont impliquées. Cette proposition s'inspire de la norme IEEE-754 pour les nombres à virgule flottante et en particulier de la valeur NaN (*Not a Number*). L'objectif est d'identifier les points hors du domaine de définition d'une fonction partielle et de construire une extension totale rigoureuse. Cette extension est ensuite élevée au niveau des intervalles, au travers de la notion d'intervalle total. Une arithmétique cohérente est établie, couvrant les opérateurs binaires élémentaires ainsi que les fonctions partielles courantes telles que la racine carrée ou le logarithme. Sur cette base, des contracteurs totaux sont formellement définis et leurs algorithmes de calcul sont explicitement dérivés pour les contraintes binaires et ternaires. La méthode *forward-backward* classique est ainsi généralisée à des intervalles totaux. Cela permet de construire de manière systématique le contracteur du complément d'un ensemble défini par des fonctions partielles, en substituant les images cibles par leurs compléments. La validité de l'approche est illustrée sur deux cas de

test concrets, dont un système dynamique inspiré de l'application de Hénon, démontrant que les pavages obtenus sont désormais exacts là où les solveurs classiques produisaient des classifications erronées. Il est souligné que cette contribution a été valorisée par une publication scientifique dans le journal *Acta Cybernetica*.

La définition de l'extension dénommée *xinterval* au jeu d'instruction RISC-V pour l'arithmétique par intervalles est l'objet du cinquième chapitre. Dans un premier temps, la représentation des intervalles est abordée. Le choix a été de coder les intervalles à l'aide des registres 64 bits de l'extension double-précision D du jeu d'instructions RISC-V. La structure sur 64 bits est divisée en deux moitiés de 32 bits, chacune accueillant une borne de l'intervalle, auxquelles s'ajoutent deux signaux d'un bit. Les 31 bits restants de chaque moitié sont utilisés pour encoder les bornes dans un format virgule flottante custom dénommé *Float31*. Dans ce format, un bit d'exposant est sacrifié car jugé sans impact significatif pour les applications de robotique ciblées. L'ensemble des instructions de l'extension adopte le format R associé à l'opcode *custom-0*, garantissant la conformité avec la spécification du standard RISC-V. Dans la suite du cinquième chapitre, deux groupes d'instructions sont décrits. La première porte sur les opérations de base sur les intervalles. Quant au second, il modélise des opérations arithmétiques, des fonctions algébriques et des fonctions transcendantes adaptées au contexte applicatif. Au final, l'extension *xinterval* est constituée de 21 instructions. Il s'agit d'une contribution originale et cohérente qui définit rigoureusement les instructions associées aux primitives mathématiques pour les contracteurs.

Dans la continuité de la définition de l'extension *xinterval*, différents opérateurs matériels ont été conçus par Monsieur Pierre FILIOL pour permettre l'exécution des 21 instructions préalablement définies. Ces opérateurs sont présentés dans le sixième chapitre. Pour ce faire, une architecture minimaliste appelée *xinterval core* est proposée afin de permettre l'évaluation des différents opérateurs. Il est à noter qu'une attention particulière a concerné la définition d'une interface générique conforme au format R du jeu d'instructions RISC-V pour tous les opérateurs matériels. Par ailleurs, comme souligné dans le manuscrit de thèse, ce travail a été grandement facilité par l'utilisation de la bibliothèque FloPoCo pour la génération des opérateurs au format virgule flottante *Float31*. Les onze opérateurs matériels décrits ont été synthétisés en ciblant le circuit FPGA Xilinx XC7Z020. Les résultats de synthèse, obtenus pour une fréquence d'horloge de 100 MHz, montrent que les opérateurs sont entièrement pipeline, avec des latences allant de 1 cycle à 42 cycles. Comme attendu, les opérateurs trigonométriques sont les opérateurs matériels les plus complexes car ils nécessitent 42 cycles et plus de 60 % des ressources FPGA allouées au total.

Une difficulté majeure de ce travail de thèse fût la validation et l'analyse des atouts et des limitations de l'extension *xinterval*. Cela a impliqué la mise en place d'une approche méthodologique telle que décrite dans le septième chapitre. Ainsi, une plateforme de test a été développée, combinant un simulateur de jeu d'instructions RISC-V avec un simulateur événementiel de robot écrit en langage Rust. De plus, une bibliothèque de contracteurs en langage C a été développée en deux versions : *libctc_noacc* reposant sur les instructions RISC-V standard et *libctc_acc* reposant sur les instructions *xinterval*. L'objectif était de faciliter l'évaluation de l'impact de l'extension *xinterval*. Lors des expérimentations,

l'architecture *xinterval core* a été interfacée avec le simulateur de jeu d'instructions RISC-V au travers de différentes couches d'abstraction :

- Simulation : les instructions sont décrites fonctionnellement à l'aides du langage C ;
- Co-simulation (GHDL) : les opérateurs matériels décrits à l'aide du langage VHDL sont évalués via le simulateur open-source GHD ;
- Prototypage sur circuit FPGA : l'architecture *xinterval core* est synthétisée et implémentée sur un circuit FPGA qui est interfacé à un simulateur RISC-V via une interface série de type UART ;
- Prototypage au sein d'un SoC Zynq: l'architecture *xinterval core* est intégrée comme module périphérique sur la partie logique programmable d'un SoC Zynq. Le simulateur de jeu d'instructions RISC-V s'exécute sur le processeur ARM embarqué au sein du SoC Zynq.

Les expérimentations ont été menées sur quatre benchmarks d'inspiration robotique (*ring*, *waves*, *donut* et *vortex*) de complexité croissante. Les résultats obtenus lors du prototypage sur le SoC Zynq montrent des accélérations allant de ×3,2 pour le benchmark le plus simple à ×8,2 pour les plus complexes. Ce gain croissant s'explique par le fait que les benchmarks les plus complexes font appel à des opérations particulièrement coûteuses à exécuter en logiciel. Comme indiqué en conclusion de ce chapitre, les gains paraissent limités vis-à-vis de la capacité d'accélération attendue. Cependant, de nombreuses optimisations peuvent encore être investiguées. Selon moi, la principale limitation des travaux réalisés réside dans le fait que l'architecture *xinterval core* n'a finalement pas été intégrée au sein d'une architecture de processeur RISC-V.

Sur la forme, le manuscrit en langue anglaise est très bien écrit et correspond à une synthèse scientifique d'un travail de thèse. La structuration du document permet d'identifier les nombreuses contributions qui ont été nécessaires pour mener à bien l'étude. Au niveau de la valorisation scientifique de ce travail de thèse, Monsieur Pierre FILIOL a été productif. Il est l'auteur principal de deux publications dans la revue international *Acta Cybernetica*. Par ailleurs, il est auteur de trois publications dans des conférences internationales (NEWCAS, ICECS et SCAN). Enfin, il a présenté ses travaux de thèse dans trois *workshops* internationaux avec comité de sélection mais sans actes publiés.

En conclusion, le travail de Monsieur Pierre FILIOL est de très bon niveau et original. Les travaux de recherche contiennent des contributions particulièrement intéressantes pour le domaine de la robotique mobile et en particulier l'exécution efficace sur des processeurs embarqués d'algorithmes basés sur l'arithmétique des intervalles, Pour toutes ces raisons, j'émets un avis très favorable pour que Monsieur Pierre FILIOL puisse soutenir oralement son mémoire de thèse en vue d'obtenir le titre de DOCTEUR de l'ENSTA dans la spécialité « Robotique ».

Fait à Talence, le 24 Mars 2026 Christophe JEGO